How to Design a PCB

Schematic capture, Circuit simulation,
pcb design and 3D modellng.

AutoTRAX Design Express (DEX)

Integrated design of schematics and PCBs.

---

by Iliya Kovac

This manual will give you complete and in-depth coverage on desiging electronic system.
You can create 1 or more schematic diagrams and organize them hierarchically.
A PCB is automatically associated with your schematics and provide true project integration of your design.

How to Design a PCB

Printed: August 2023 in (whereever you are located)

Publisher

...enter name...

Managing Editor

...enter name...

Technical Editors

...enter name...
...enter name...

Cover Designer

...enter name...

Team Coordinator

...enter name...

Production

...enter name...

Special thanks to:

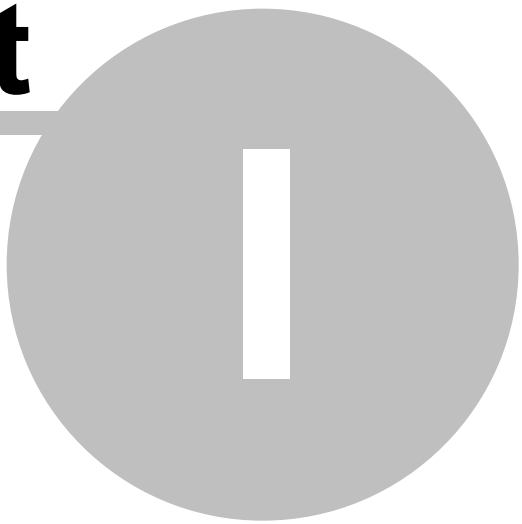
All the people who contributed to this document, in particular, I want to thank all the users of AutoTRAX over the years who can contributed ideas and given feedback.

Table of Contents

Foreword

This manual describes in detail how to use AutoTRAX Design Express.

Part



1 Designing a PCB with AutoTRAX DEX

Published Tuesday, August 8, 2023 at 9:05:34 PM GMT

AutoTRAX PCB Design Express (DEX), is a fully integrated electronic schematic capture program combined with an easy to use PCB Designer.

It has all the features you expect and need to rapidly and easily take your design from conception through to production. Its in-built hierarchical project manager lets you perform both top-down and bottom-up design and reuse design components and sub-systems.

Starting with its easy to use Schematic Capture mode, you can drag previously created parts onto your design sheets and rapidly and reliably connected the terminals together using wires, buses and off-page connectors. AutoTRAX DEX ensures that your design remains correct throughout with no ‘dangling wires’ or design rule violations.

You can also create your own parts either in-place on your design sheets or using the integrated Part Creator. Again, design integrity is maintained throughout using the AutoTRAX DEX design rule checker.

When you are satisfied with your design, you can then quickly proceed to produce your PCB board without leaving AutoTRAX DEX.

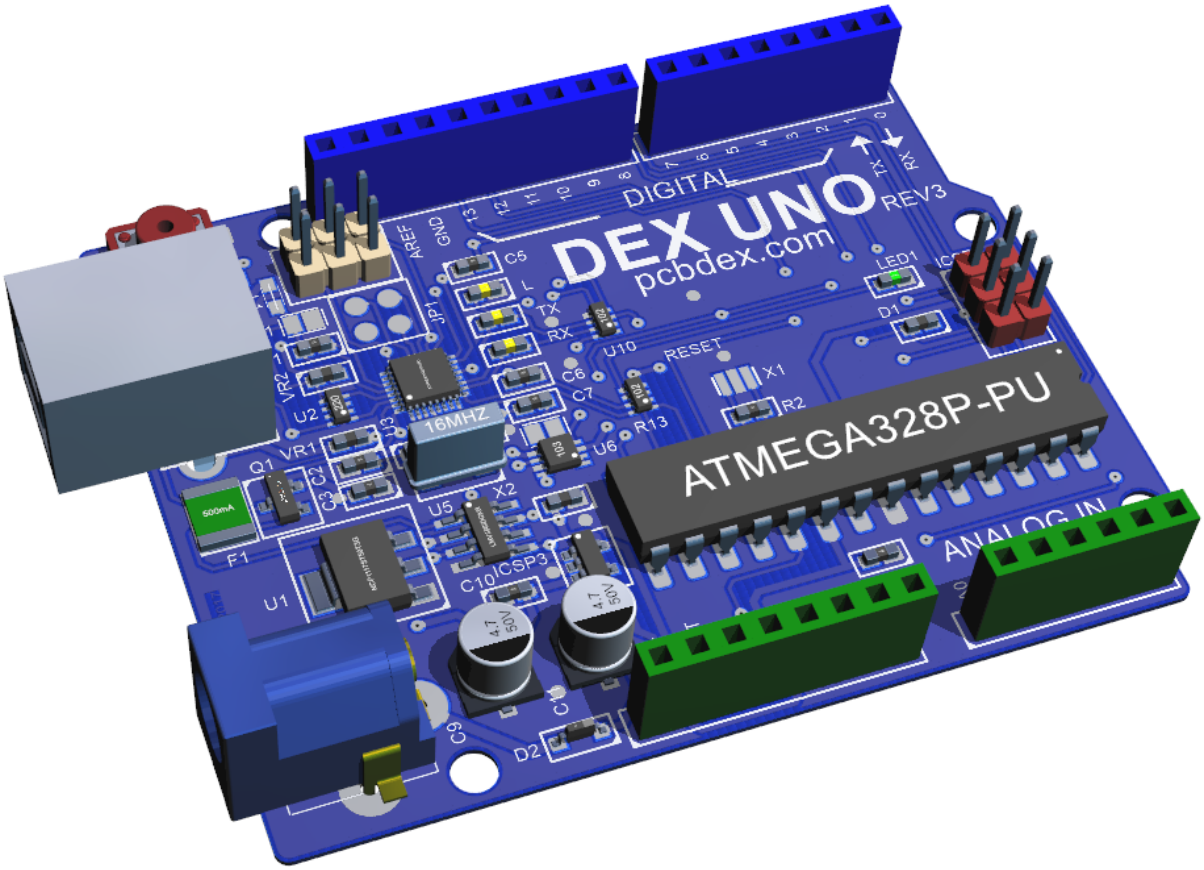
AutoTRAX DEX will take your hierarchical design and place the components, with or without your assistance. Next AutoTRAX DEX can either auto-route your board or you can use a combination of automatic and manual routing to quickly and reliably complete all electrical wiring. The design of AutoTRAX DEX's internal data ensures full electrical integrity between your schematic and PCB designs; there are no surprises with lost or missing PCB tracks or wires, or those ‘extra’ wires.

Now that you have your design finished and routed you can then produce all Computer Aided Manufacturing files you will need to produce the board, drill the holes, cut its profile, order parts using the Bill Of Materials, and place your parts using the pick and place files. These files can be place together and forwarded to your favourite board manufacturers, electronically via email.

In only a short time you will see the fruits of your labour and will be pleased at the time and money you have saved.

Find out more...

See a quick overview of the stunning features in AutoTRAX DEX



Sample 3D Project

1.1 Getting Started

AutoTRAX PCB Design Express (DEX) : Powerful PCB design made easy

AutoTRAX DEX is a powerful integrated Electronic Design Suite for Electronic Engineers. It has all the features you expect and need to rapidly and easily take your design from conception through to production. Its in-built hierarchical project manager lets you perform both top-down and bottom-up design and reuse design components and sub-systems. Schematic capture and PCB layout has never been easier.

AutoTRAX DEX is the industry's only unified electronic design software which gives you an unmatched ability to design and build current and future generation of electronic products.

This is in sharp contrast to the previous generation of software which have separate data files for schematics and PCBs – even different applications with different user interfaces for schematic design and PCB design.

No Limits

Unlike software from other EDA developers AutoTRAX DEX is not stymied by the limited thinking of the software developers of the last century.

Why do they limits their boards? Who know?

There are absolutely no limits* to your design in AutoTRAX DEX

- Unlimited board size
- Unlimited number of layers (More than any PCB manufacturer can make)
- Unlimited number of parts
- Unlimited number of pads
- Unlimited number of nets
- No time limits. The software is yours to keep. You are not even limited to using it on only one machine or having to mess about with floating licenses.

As AutoTRAX DEX has only 1 license type***- unlimited-*** you get it all. No immoral Trojan horses to trap you into paying more. Read more...

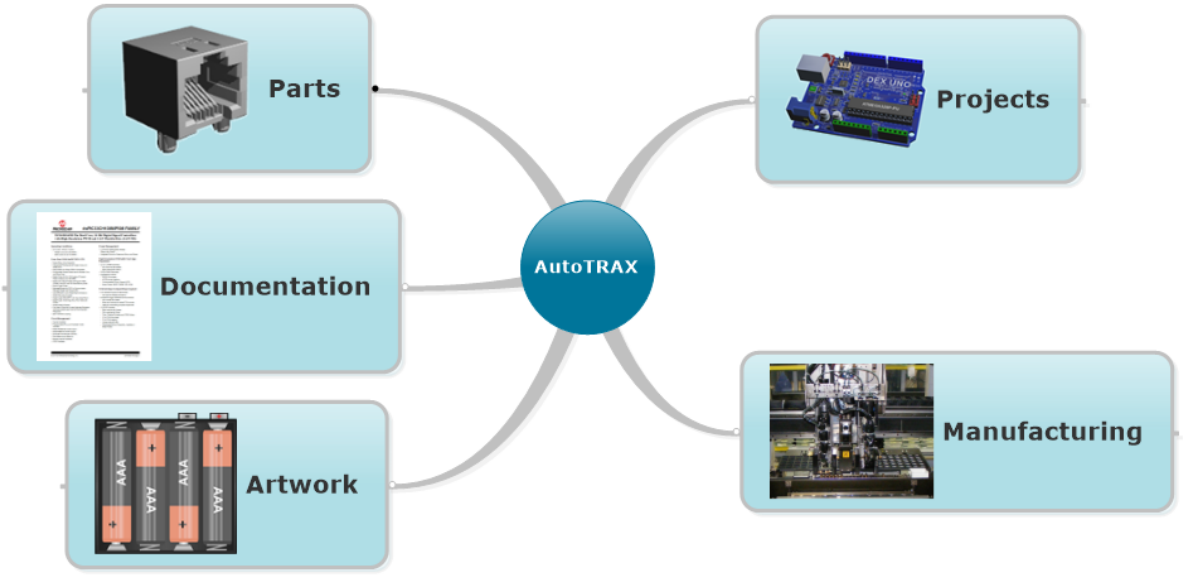
Unified Platform

AutoTRAX DEX is the industry's only unified electronic design software which gives you an unmatched ability to design and build current and future generation of electronic products. Schematic's and PCBs are tightly integrated together; any change in the schematic is fully updated in the PCB and similarly any change in PCB is reflected back in the schematic.

Traditional concepts of an electronics design system haven't changed much, at least for most EDA vendors. This can be apparent in the way they struggle to meet even the current challenges. The fragmented approach of trying to glue separate solutions together eventually breaks down as more challenges are thrown at them. It's just unsustainable and difficult to innovate on top of inconsistent design applications that rely on constantly unmatched data exchanges. That's not how AutoTRAX DEX is designed.

AutoTRAX DEX has been designed from the ground up as a powerful, single-application electronics development environment that contains all the advanced design tools you'll need to capture and simulate hardware, design PCBs, collaborate with MCAD, manage your ECAD data and documentation, as well as manage your designs from concept to production. It's all there in a single, unified design platform.

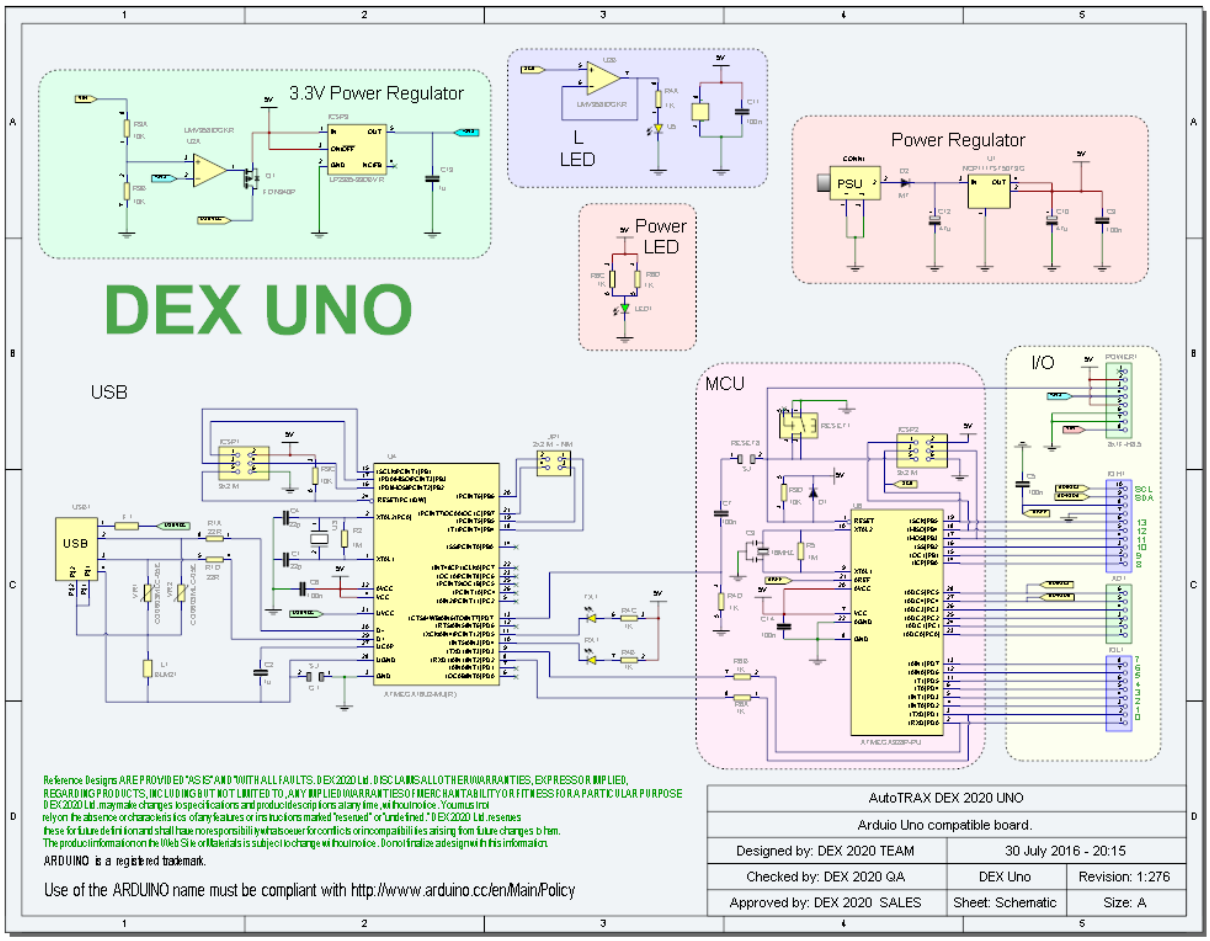
AutoTRAX DEX also unifies your design data, from component models to sub-assemblies. Unlike conventional design systems based on a collection of linked tool applications, the AutoTRAX DEX unified platform allows each design application to access and modify a single, centralized model of the design data. That's one



AutoTRAX Data Flow

Schematics

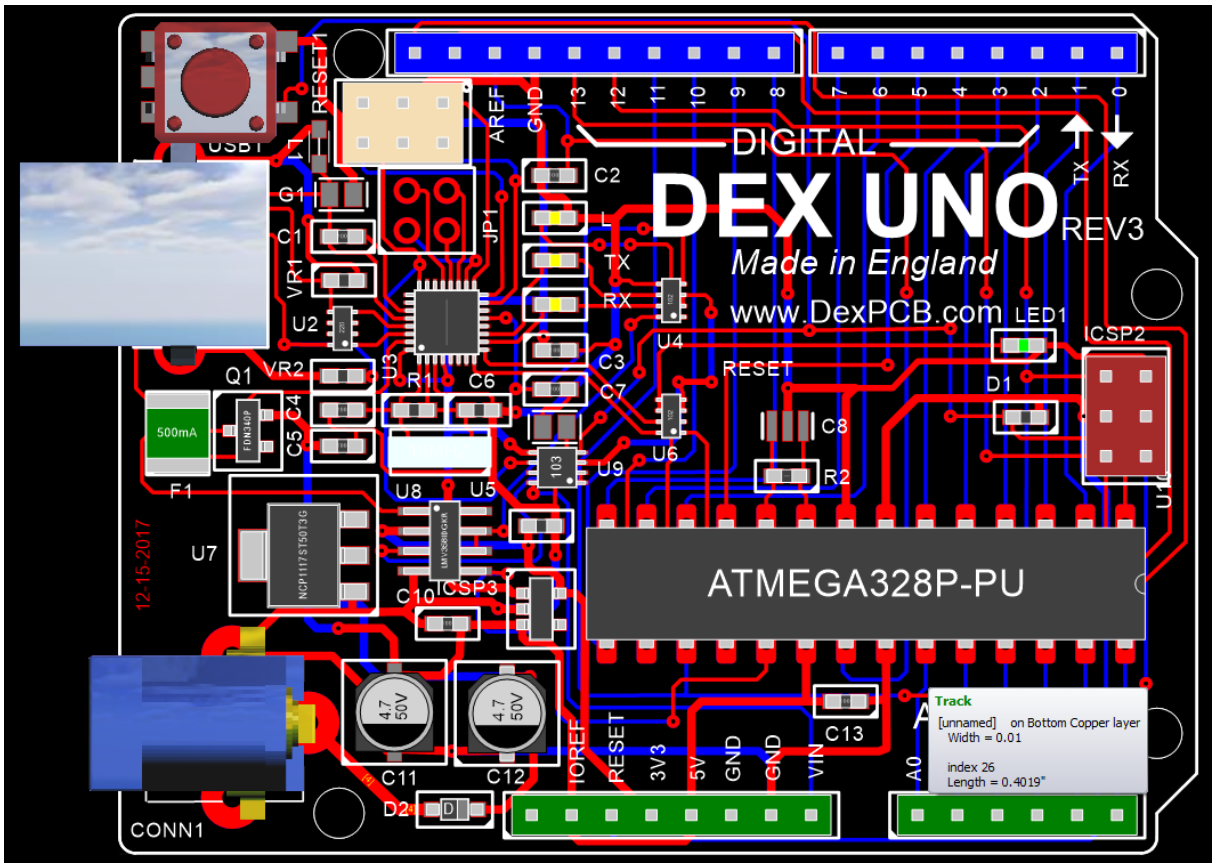
AutoTRAX DEX ensures that your design remains correct throughout, with no dangling wires or PCB design rule violations. You can also create your own parts either in-place on your design sheets or using the integrated Part Creator. Again, design integrity is maintained throughout using AutoTRAX DEX's design rule checker.



The AutoTRAX DEX UNO Schematic

PCB Design

When you are satisfied with your design, you can then quickly proceed to produce your finished and populated PCB board without leaving AutoTRAX DEX. The AutoTRAX DEX's PCB Designer will take your hierarchical design and place the components, with or without your assistance. Next AutoTRAX DEX can either auto-route your board or you can use a combination of automatic and manual routing to quickly and reliably complete all electrical wiring. The design of AutoTRAX DEX's internal data ensures you maintain full electrical integrity with your schematic design.



The AutoTRAX DEX UNO PCB

Manufacturing

Now that you have your design finished and routed you can produce all Computer Aided Manufacturing files you will need to produce the board, drill the holes, cut its profile, order parts using the Bill Of Materials, and place your parts using the pick and place files. These files can be placed together and forwarded to your favorite board manufacturers, electronically via email.

The AutoTRAX DEX UNO PCB in 3D

Have Gerber, Will Travel!

Unlike other programs, AutoTRAX DEX is not locked to a PCB manufacturer where if you use their program you must use their services to produce your boards, and pay the price their demand!

With AutoTRAX DEX Gerber files you can shop around to get the best price and service for your PCBs.

Be in control!

Have XML, Will Travel!

Again, unlike other programs, AutoTRAX DEX does not lock you in to using AutoTRAX DEX now and forever. The design file is totally self contained and is in the industry standard XML file format. This is human readable and you can easily write your own programs to use the data in your design.

This is in sharp contrast to other programs which have their own secret file format. This locks you into using their program now and forever.

If the company goes under then so could your archive of designs.

1.1.1 Limitations

The good news: There are no limits except those imposed by your computer's memory size.

Ther Good News

These are the limitation with AutoTRAX DEX (Well really there are no practical limitations)

Number of schematic sheets	Unlimted *
Number of PCB Layers	Unlimted *
Size of PCB	Unlimted *
Number of Parts	Unlimted *
Number of Terminals	Unlimted *
Size of Schematics	Unlimted *
Units	IEEE Floating Point

*** Limited only by memory size**

1.1.2 Free but not free or “I’m off to Cancun”

Free but not free or “I’m off to Cancun”

Free but not free or “I’m off to Cancun”

Lets Get Started

Cancun

You wake up and the sun is shining. Yep, it's a great day. "I fancy a bit of that electronic designing that I keep hearing about" you proclaim as you drink your first coffee of the day. So you power up the laptop, click on Google and wow loads of PCB Design Software, "I'll have a bit of that" you announce to the world, but only the cat is in the room. The cat gives you a funny look.

You download the freebie feeling well pleased with your mastery of the internet and the bundles of cash you have just saved. You make a mental note to "book vacation in Cancun".

You start up the software and off you go, Edison "eat your heart out".

Your first Gizmo is a success.

A Good Idea

A couple of months pass by. Now “let’s turn Gizmo into Super Gizmo” you say with a big smile. So off you go, the freebie is up and running and you are well into the new design then bang, smash, wallop a dialog pops up saying you have reached the freebie limit and it’s time to pay up. You yelp in pain. The Super Gizmo must be done; you’ve spent another 2 weeks on it. Out comes the plastic, you tap in a few digits ($500) then you are up and running.

The following week Super Gizmo is let loose on the world.

A couple of months later, you wake up with the cat licking your face, you thought it was the girl you meet last night but she dumped you as you both were leaving the bar. A bright idea enters your head “let’s turn Super Gizmo into Mega Gizmo”. So off you go, the not so free freebie is up and running and you are well into the new design then bang, smash, wallop that darned dialog pops up again saying you have reached another limit and it’s time to pay up. You yelp in pain. The Super Gizmo must be done; you’ve spent another 2 weeks on it. So you look for alternatives, you see AutoTRAX DEX; it’s real cheap and no limits., “I’ll have a bit of that” you yell. Download AutoTRAX DEX but what’s this? You can’t move your design from the not so free freebie to AutoTRAX DEX, the not so free freebie developers keep their file format secret. Nobody else can read it. You have been well and truly screwed.

The Cat's Not Happy

Out comes the plastic, you tap in a few digits ($1000) then you are up and running. You make a mental note “cancel Cancun”. The cat hisses at you, turns round and legs it for the door.

The following week Mega Gizmo is let loose on the world and you wonder what to do during your vacation at home.

Getting Busy

You have just been stung.

In the business it's called A Trojan Horse.

A Trojan Horse

Beware Greeks (and software developers) bearing gifts

The term "Trojan horse" comes from a story in Greek mythology in which the Greeks gifted a giant wooden horse to the city of Troy during the Trojan War. The Trojans brought the horse into the city, not realizing that it was filled with Greek soldiers. At night, the soldiers emerged from the horse, opening the city gates and allowing the Greek army to invade and conquer Troy.

1.1.3 Installing DEX

System Requirements

Downloading AutoTRAX DEX

Firewalls

Where are the AutoTRAX DEX files?

Authorizing Your Copy

Authorizing Your Copy Machine Without Internet Acccess

Getting Your Key Online

Your Account Settings

Removing Your License

Manually Removing AutoTRAX DEX

Manually Removing AutoTRAX DEX

Windows Vista and Windows 7/8/10/11 are fully supported (32 bit and 64 bit)

What is Microsoft .NET

Microsoft .NET (pronounced "dot net") is a software framework that can be used for building and running applications on Windows, macOS, Linux, and more. It was first released by Microsoft in 2002 and has gone through multiple iterations and improvements since then.

The .NET framework includes a large class library known as the Framework Class Library (FCL), and it supports several programming languages including C#, Visual Basic .NET (VB.NET), and F#. With .NET, you can build many types of applications including desktop applications, web applications, web services, and even games.

Here are some key components of .NET:

Common Language Runtime (CLR): This is the execution engine for .NET applications. It provides services like memory management, garbage collection, security, and exception handling.

Framework Class Library (FCL): This is a collection of reusable classes, interfaces, and value types that .NET applications can use. It covers a wide range of functionalities, such as file reading and writing, database interaction, XML document manipulation, and more.

ASP.NET: A framework for building web applications, services, and dynamic websites.

Windows Forms: A set of classes for building graphical user interfaces (GUIs) for Windows desktop applications.

Windows Presentation Foundation (WPF): A tool for creating rich user interfaces for desktop applications.

Entity Framework (EF): An Object-Relational Mapping (ORM) framework for ADO.NET, which provides a higher-level, object-centric model for data access.

Language Integrated Query (LINQ): A component that adds native data querying capabilities to .NET languages.

AutoTRAX DEX will automatically prompt you to install the correct Microsoft .NET 4 if it not already installed.

1.1.3.1 System Requirements

Windows 8 or later.

You can use a 32 bit version or a 64 bit version. A 64 bit version is recommended.

DISC DRIVE

A decent size disc with ample space is recommended. At least 500Mb of free space is recommended.

A Solid State Drive (SSD) is recommended for faster start-up.

CPU

The faster the better. We suggest an I7.

RAM

The more the better. On a 32 bit version O/S you are limited to 3GB usable (the extra 1GB is reserved for your video card). I recommend at least 2GB on Windows XP and 3GB on Windows 7 and later. Windows Vista is not recommended.

VIDEO CARD

You must have at least a 24 bit color video card. All modern video cards support 24/32 bit color.

MONITOR

Here the bigger, the better. The Monitor is your view into your design. I recommend at least 1920 pixels horizontal.

AutoTRAX DEX will work with multiple monitors and you can drag parts of AutoTRAX DEX onto different monitors. AutoTRAX DEX remembers the position of all its windows are restored them the next time you run AutoTRAX DEX.

All modern video cards support at least two monitor outputs, and many support even more.

MOUSE

You definitely a 3 button mouse with a central thumb wheel. The thumb wheel is use you zoom in and out and pressing down the thumbwheel/middle mouse button pans the view.

1.1.3.2 Microsoft Windows and 4K Monitors

Microsoft Windows has long supported 4K resolution, and this is certainly true as of the latest versions of Windows as of my knowledge cutoff in September 2021.

To utilize 4K resolution on a monitor, you need:

- **A 4K Monitor:** This is a monitor that supports a resolution of **3840 x 2160 pixels**, which is considered Ultra High Definition (UHD). Some monitors even support 4096 x 2160 pixels, which is the standard for 4K in digital cinema.
- **A Graphics Card that Supports 4K:** Your computer's graphics card needs to be able to output at a 4K resolution. As 4K has become more standard, this is a common feature on most modern graphics cards. Make sure to verify that your graphics card can support this resolution.

- **Appropriate Cables:** You'll need an HDMI (version 1.4 or higher), DisplayPort (version 1.2 or higher), or USB-C cable that supports 4K resolution. The specific cable will depend on what ports are available on your monitor and graphics card.
- **4K Content:** Having a 4K monitor won't do you much good if you don't have content that is rendered in 4K. This includes games, videos, photos, etc.
- **Software Support:** Your operating system and software should support 4K as well. As of my knowledge cutoff in 2021, the latest versions of Windows do support 4K.

Setting your Windows PC to use 4K resolution is usually as simple as going into your display settings (right-click on the desktop, then select "Display settings") and then choosing the 4K resolution from the drop-down menu.

However, be aware that not all software and websites are optimized for 4K, and may appear small or blurry on a high-resolution display. Some apps or websites might not scale properly, leading to smaller text or images. Windows does offer some built-in tools for scaling up the size of text and other items on the screen to compensate for the increased resolution. You can adjust these settings in the same "Display settings" menu.

Moreover, running your PC at a higher resolution can demand more from your computer's graphics card, which might result in increased heat or noise from your computer's fans. This isn't usually a problem with modern hardware, but it's something to keep in mind.

By 2023, most of these issues have likely been ironed out, as 4K has become more common. However, without specific information about the state of technology in 2023, I can't provide a more precise answer.

HDMI Cables

An HDMI (High Definition Multimedia Interface) cable is a type of digital connection used to transmit high-quality video and audio signals. These cables are widely used in home theater systems, gaming consoles, and many other consumer electronics because they can handle high-definition (HD) video and multichannel audio over a single cable.

Here are a few more details about HDMI cables:

- **Quality:** HDMI cables are designed to handle video resolutions of 1080p and beyond, including advanced display technologies such as 4K and 3D.
- **Audio:** They can transmit multi-channel audio data, supporting all standard and high-definition consumer electronics video formats and up to 8 channels of digital audio.
- **Variants:** As of my knowledge cut-off in September 2021, HDMI cables come in different versions, like HDMI 1.4, 2.0, 2.1 etc., each with increased capabilities. HDMI 2.1, for example, supports up to 10K resolution and dynamic HDR.

- **CEC feature:** HDMI also includes a feature called CEC (Consumer Electronics Control) that allows the user to command and control up to 15 CEC-enabled devices.
- **ARC and eARC:** HDMI cables also have features called ARC (Audio Return Channel) and eARC (enhanced Audio Return Channel) which simplify the process of sending audio from a television to a home theater system or soundbar.
- **Ethernet:** Some HDMI cables have an Ethernet channel that enables high-speed, bi-directional communication. This means devices connected by the HDMI cable can share an internet connection.

It's also important to note that the HDMI interface is backward compatible with the single-link Digital Visual Interface (DVI) used on many older devices, so with the right adapter, you can still use an HDMI cable with a device that only has a DVI output.

DisplayPort Cables

DisplayPort is a digital display interface primarily used to connect a video source (like a computer or gaming console) to a display device, such as a computer monitor or a television. Like HDMI, DisplayPort can also carry audio, USB, and other forms of data.

Here are some key aspects of DisplayPort cables:

- **Resolution and Refresh Rates:** As of my last training cut-off in September 2021, DisplayPort 1.4 can support resolutions up to 8K (7680 × 4320) at 60Hz, or 4K (3840 x 2160) at 240Hz, making it a suitable option for high-resolution displays and demanding games.
- **Multi-Stream Transport (MST):** This feature of DisplayPort allows you to use multiple monitors through a single DisplayPort connection by "daisy-chaining" them together, a feature not generally supported by HDMI.
- **Adaptive Sync:** DisplayPort supports adaptive sync technology (like AMD's FreeSync and NVIDIA's G-Sync), which matches the display refresh rate to the frame rate of the graphics card. This reduces issues like screen tearing in games.
- **Different Connectors:** DisplayPort connectors come in two sizes, standard (often just called DisplayPort) and DisplayPort Mini (or Mini DisplayPort). The Mini DisplayPort was commonly used on older Apple devices but has largely been replaced by USB-C or Thunderbolt.
- **Backward Compatibility:** DisplayPort is backward compatible with VGA, DVI, and HDMI, but you'll need the appropriate adapter.

It's also important to note that while HDMI and DisplayPort may look similar at first glance, they were developed by different industry consortiums for slightly different market segments (consumer electronics vs. personal computer), and they differ in

terms of sound and video specifications, supported technologies, and physical connectors.

USB-C Video Cables

USB-C, also known as USB Type-C, is a type of USB (Universal Serial Bus) connector that's reversible, meaning you can plug it in either way. USB-C cables and ports are becoming increasingly common on various types of devices, including laptops, smartphones, and tablets.

Apart from data transfer and charging, USB-C can also support video output if the device supports a protocol called USB-C Alt Mode. This feature repurposes some of the wires in a USB-C 3.1 cable for direct device-to-host transmission of alternate data protocols. The four high-speed lanes, coupled with supporting power and data code, enable this cable to carry significantly more data.

Here are some key aspects of USB-C video cables:

- **DisplayPort Over USB-C:** This is a version of Alt Mode. If a device supports DisplayPort Alt Mode, a USB-C to DisplayPort cable (or a USB-C to USB-C cable if both devices have USB-C ports) can carry a DisplayPort signal.
- **HDMI Over USB-C:** Similarly, HDMI Alt Mode for USB-C allows HDMI-enabled source devices to utilize a USB-C cable to directly connect to HDMI-enabled displays.
- **Thunderbolt 3 and 4:** Some USB-C ports also support Thunderbolt 3 or 4, which are standards that allow for very high data transfer rates and video output capabilities. A single Thunderbolt 3 port and cable can transmit DisplayPort and HDMI video signals, data for external storage drives, and power for device charging.
- **Power Delivery:** USB-C also supports USB Power Delivery (USB PD), a fast charging standard that can deliver up to 100W of power. This means you could potentially use the same USB-C cable to charge your laptop, output video to a monitor, and transfer files to an external drive.
- **Dongles and Adapters:** Since USB-C is compatible with so many different protocols, you can use dongles and adapters to connect to a wide variety of devices. For instance, if your laptop has a USB-C port but your monitor has an HDMI port, you can use a USB-C to HDMI dongle to connect the two.

Please note that not all USB-C ports, cables, or devices support all these features. The exact capabilities can vary depending on the specific device and how its manufacturer has chosen to implement the USB-C standard. Always check your device's specifications to see what is supported.

1.1.3.3 Monitor Text Scaling

AutoTRAX supports screen text scaling.

Microsoft Windows includes a feature that allows you to scale the size of text and other items on your screen. This can be particularly useful if you're using a high-resolution display, such as a 4K monitor, where items might otherwise appear very small.

Windows 10

Here's how you can adjust these settings in Windows 10:

1. Right-click on an empty spot on your desktop and choose "Display settings".
2. Under "Scale and layout", you'll find a drop-down menu labelled "Change the size of text, apps, and other items". By default, this might be set to something like 100% or 150%, but you can select a higher percentage to increase the size of items on your screen.
3. You can also click on "Advanced scaling settings" for more options. Here, you can enter a custom scaling size between 100% and 500%. You can also enable a feature that lets Windows try to fix apps so they're not blurry.

After making these changes, you might need to sign out of your account and sign back in, or possibly even restart your computer, for the changes to take effect.

Windows 11

In Windows 11, the steps are similar:

1. Right-click on an empty spot on your desktop and choose "Display settings".
2. Under "Scale", you'll see a similar drop-down menu where you can select the percentage of scaling.
3. If you click on "Advanced scaling", you can manually enter a custom scaling size.

Remember, these changes will impact all displays if you're using multiple monitors unless you specifically set each monitor's scaling individually. Also, not all software behaves perfectly with scaled text and other items – you might notice that some apps or elements look blurry or pixelated after changing these settings. Windows has gotten better about this over the years, but it can still be a problem with some software.

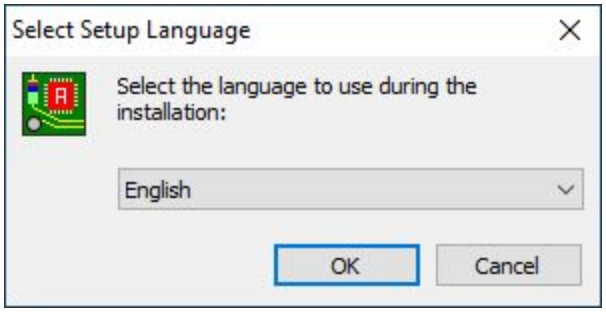
1.1.3.4 Downloading DEX

You can download AutoTRAX DEX from https://pcbdex.com/Download

When you have downloaded AutoTRAX DEX run the installer program/

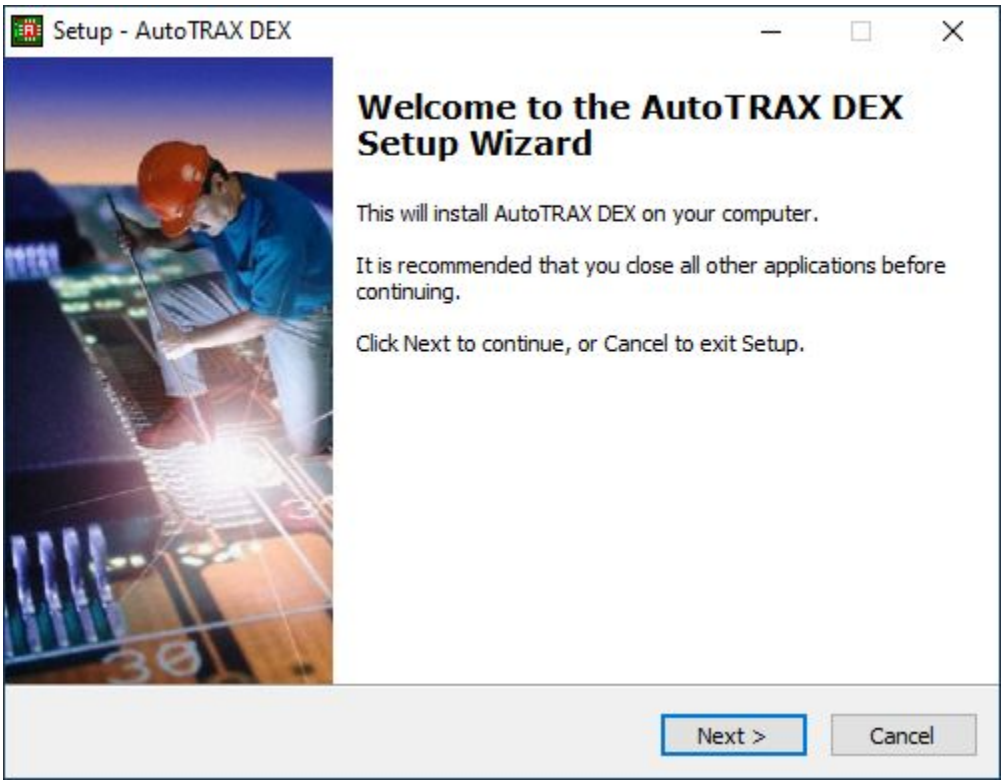
You will first see the select setup language dialog.

Click here see where AutoTRAX DEX installs files.

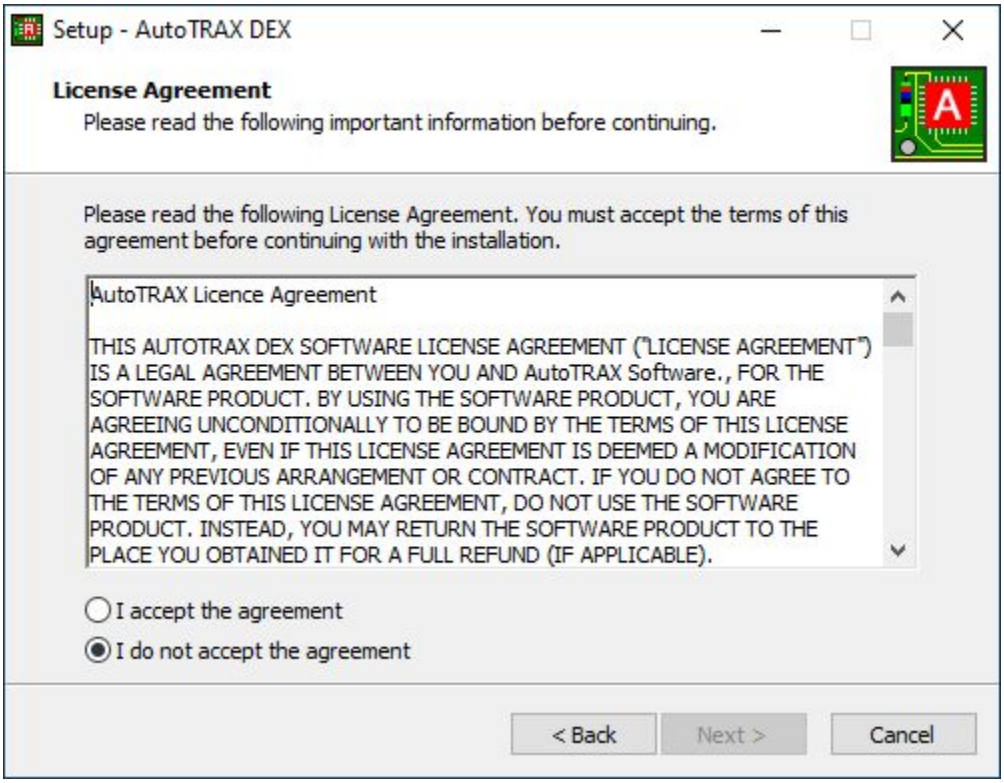


Select your language from the drop down list and click on the OK button to proceed. At any time you can click on the Cancel button to stop the installation and no files will be added to your machine.

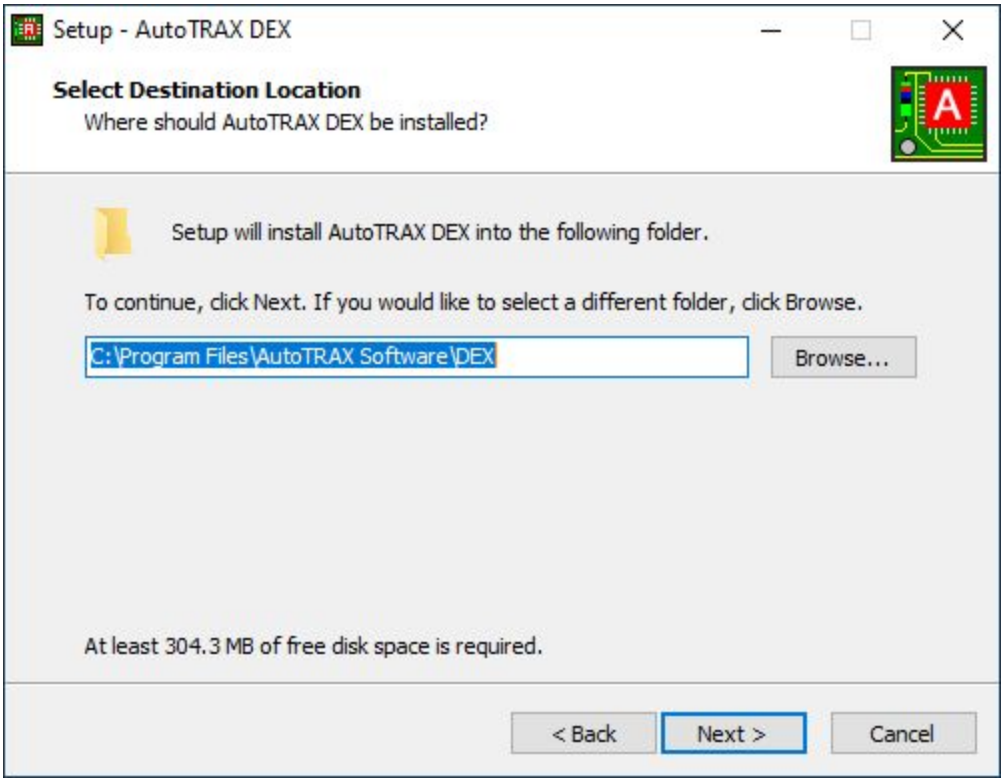
You will see the welcome screen, click on the OK button to proceed.



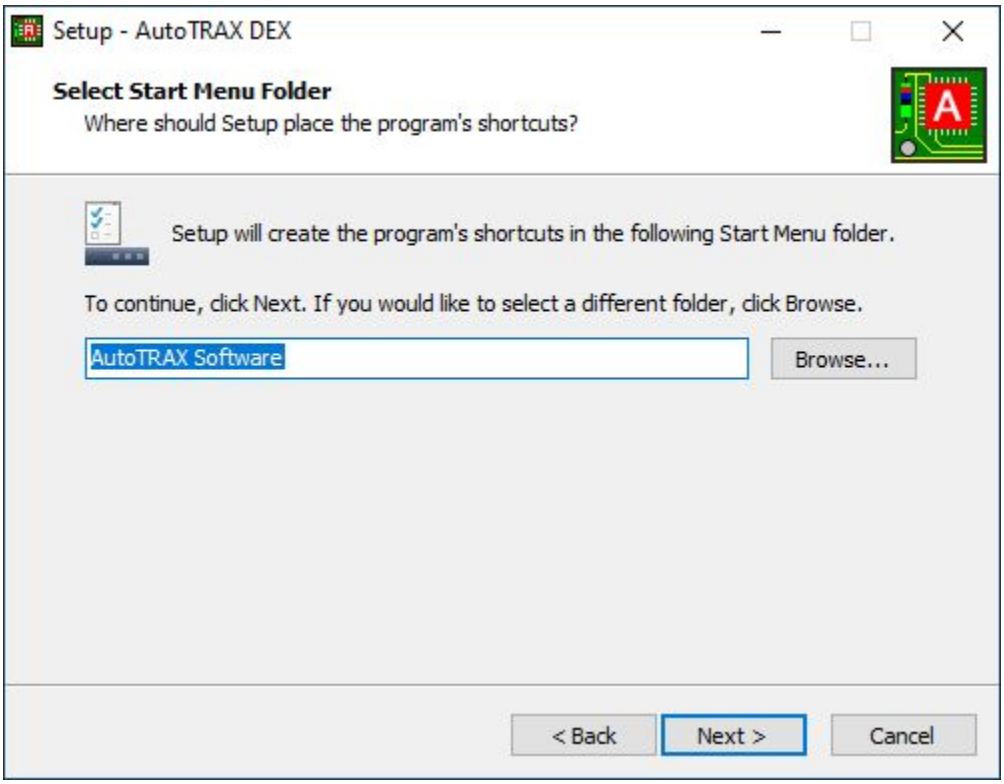
Now you will see the license agreement. You must accept this to proceed. Click on I accept the agreement and then click on the Next button to proceed. Click here to see the full license agreement



Now you get to select where to place AutoTRAX DEX. You can use the default or add your own. Again click the Next button to continue.

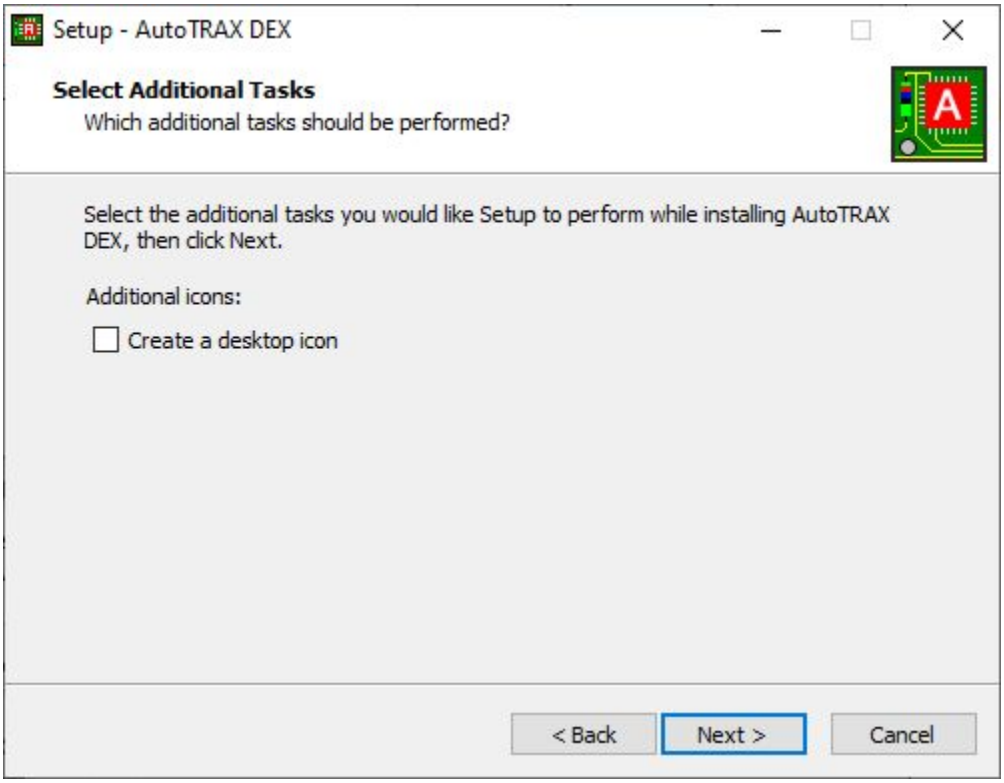


Now you get to name the start menu folder. You can use the default or add your own. Again click the Next button to continue.

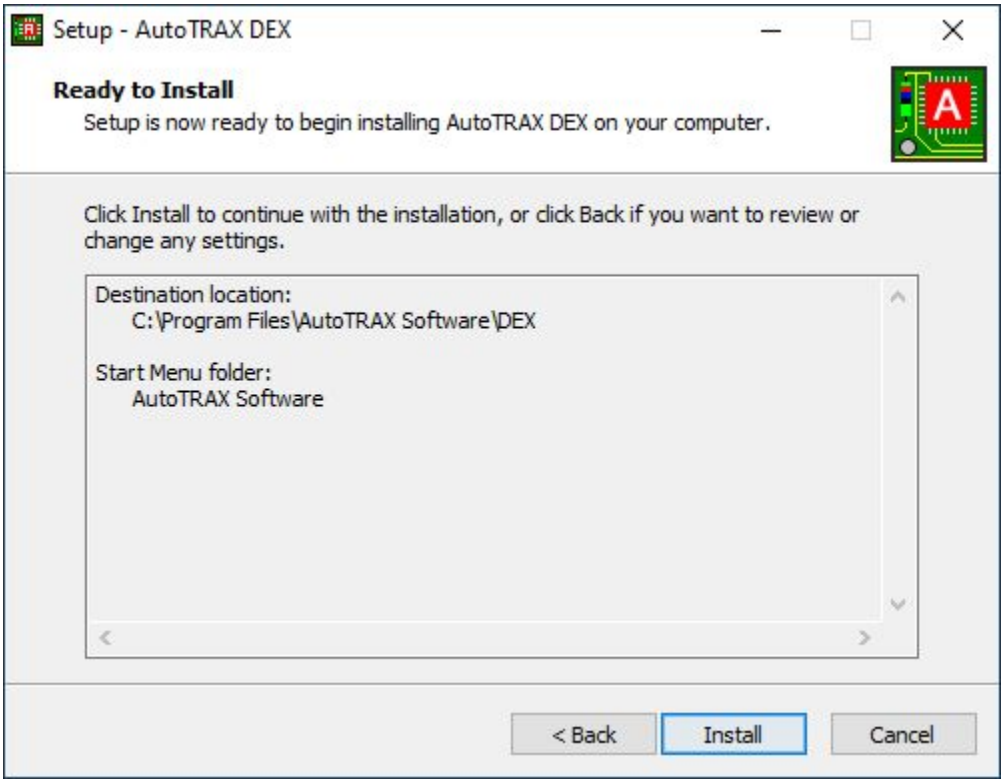


You can optionally add a desktop icon.

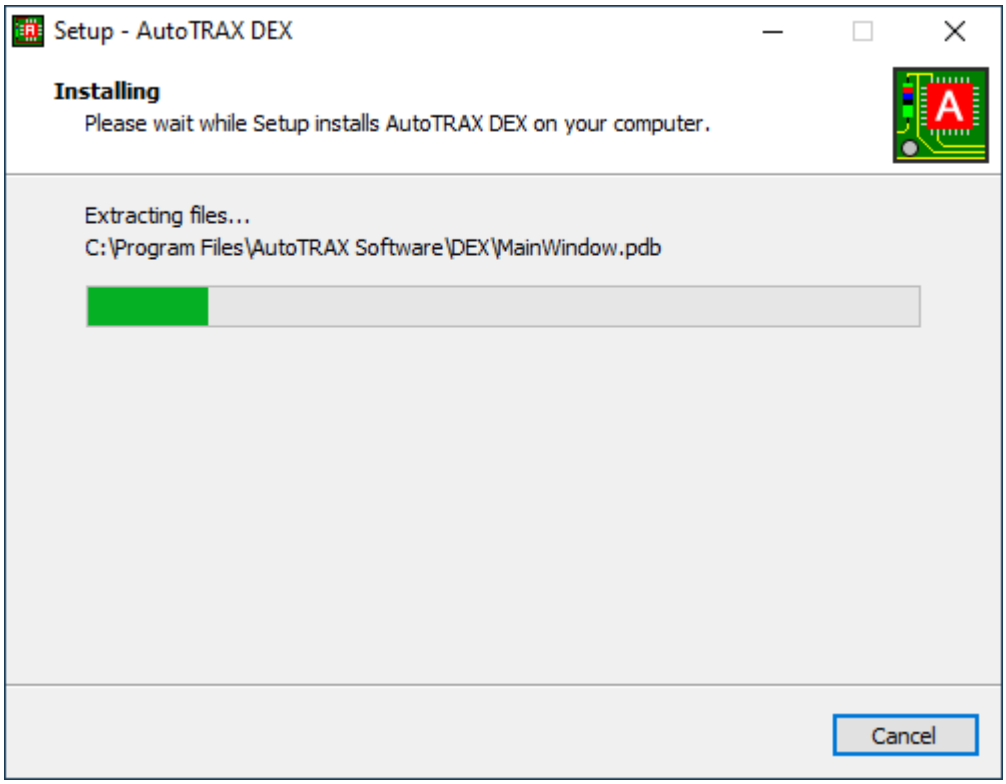
Click the Next button to continue.



Finally you are ready to install. Check the installation options and click the Next button to continue.



During installation the following progress dialog is shown. Click on the Cancel button if you no longer wish to install AutoTRAX DEX.



AutoTRAX DEX will start when the install is complete.

1.1.3.5 Firewalls

Sometime an internet Firewall will block AutoTRAX DEX from accessing the internet.

Windows Firewall is a built-in security feature of Microsoft Windows operating systems that helps protect your computer from unauthorized access over a network. It monitors incoming and outgoing network traffic based on a set of rules and filters that you can configure.

By default, Windows Firewall blocks incoming connections to your computer from the internet and allows outbound connections from your computer to the internet. You can customize the firewall settings to allow or block specific programs, ports, and protocols.

Windows Firewall also includes advanced security features such as network address translation (NAT) and internet protocol security (IPSec). NAT allows multiple devices on a network to share a single public IP address, while IPSec provides encryption and authentication for network traffic.

Overall, Windows Firewall is an important security feature that helps protect your computer and network from potential security threats.

AutoTRAX DEX accesses the internet to:

- Optionally find out if there is a new version available. You can always turn this off.
- Optionally download new versions. You cancel a download if AutoTRAX DEX detects there is a new version. You can also turn off update checks at this point. However I recommend you download new versions as they often contain essential bug fixes.
- Download the latest part libraries.
- Obtain a software license for your machine.
- Allow you to view and/or update your account details.

How to allow a program to communicate through Windows Firewall

A firewall can either be software-based or hardware-based and is used to help keep a network secure. Its primary objective is to control the incoming and outgoing network traffic by analyzing the data packets and determining whether it should be allowed through or not, based on a predetermined rule set. A network firewall provides a security perimeter between a local area network (LAN) and external networks like the internet. Computers on the LAN side of the network are considered to be trusted and secure, while computers on the external side of the firewall are considered to be untrusted and may pose a risk to computers on the trusted side of the firewall. The firewall is utilized to restrict communication between the trusted and untrusted networks.

By default, most programs are blocked by Windows Firewall, to help make your computer more secure. To work properly, some programs might require you to allow them to communicate through the firewall. Here's how to do that:

1. Open Windows Firewall by clicking the Start button, clicking Control Panel, clicking Security, and then clicking Windows Firewall.
2. In the left pane, click Allow a program through Windows Firewall. If you are prompted for an administrator password or confirmation, type the password or provide confirmation.
3. Select the check box next to the program you want to allow, and then click OK.

Warning

Before allowing any program through the firewall, make sure you understand the risks involved.

Can't Get Your Software Key?

You can **enter the software key manually.** To get a software key you can go to https://pcbdex.com/Account/SoftwareKey

Firewalls

A computer firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules. It establishes a barrier between a trusted internal network and untrusted external networks, such as the Internet.

Firewalls are critical for controlling traffic allowed to enter and leave the network, which adds an extra layer of security by blocking unauthorized users from accessing the network, and by preventing malicious applications or viruses from damaging the network or the devices connected to it.

Firewalls can be hardware, software, or a combination of both. Here's a bit about each type:

- **Hardware Firewalls:** These are standalone devices that are positioned between your network and the gateway (where your network connects to the internet). They're useful for protecting multiple computers on a network, making them common in businesses and schools.
- **Software Firewalls:** These are installed directly onto a computer and control network traffic in and out of that single machine. They can be customized to a greater extent than hardware firewalls, allowing you to control the network settings for each application on the computer.

There are different types of firewalls, including:

- **Packet-Filtering Firewalls:** The most basic type of firewall. They inspect packets of data coming into the network and block or allow them based on the source, destination, and type of data.
- **Stateful Inspection Firewalls:** These monitor the state of active connections and use this information to determine which network packets to allow through.
- **Proxy Firewalls:** These act as intermediaries for requests from one network to another, effectively masking the true network addresses.
- **Next-Generation Firewalls (NGFWs):** These combine traditional firewall features with other network device filtering functionalities, such as an intrusion prevention system (IPS) and application control.
- **Threat-Focused NGFWs:** These extend NGFW capabilities beyond identifying applications and blocking malware to also include detecting and remediation of attacks that have either slipped through or originated from inside the network.

Firewalls are a fundamental part of any network security strategy, protecting internal networks from threats on the wider internet. They work alongside other security measures like antivirus software, intrusion detection systems, and encryption tools to maintain the security of a network.

1.1.3.6 Where are the DEX files?

AutoTRAX DEX is installed in the following directories:

Roaming Data Files

C:\Users**XXXX**\AppData\Roaming\AutoTRAX DEX Software\AutoTRAX DEX

Machine Specific Files

C:\Users**XXXX**\AppData\Local\AutoTRAX DEX Software\AutoTRAX DEX

The Parts Library

The default location of the parts library is:

C:\Users***XXXX***\AppData\Roaming\AutoTRAX DEX Software\AutoTRAX DEX\Library

*Where **XXXX** is your account name.*

The Windows Registry

The Windows Registry is a hierarchical database that stores configuration settings and options on Microsoft Windows operating systems. It contains settings for low-level operating system components and for applications running on the platform that have opted to use the registry. The kernel, device drivers, services, SAM, user interface and third party applications can all make use of the registry. The registry also provides a means to access counters for profiling system performance.

AutoTRAX DEX does not use the registry or set entries in the Registry.

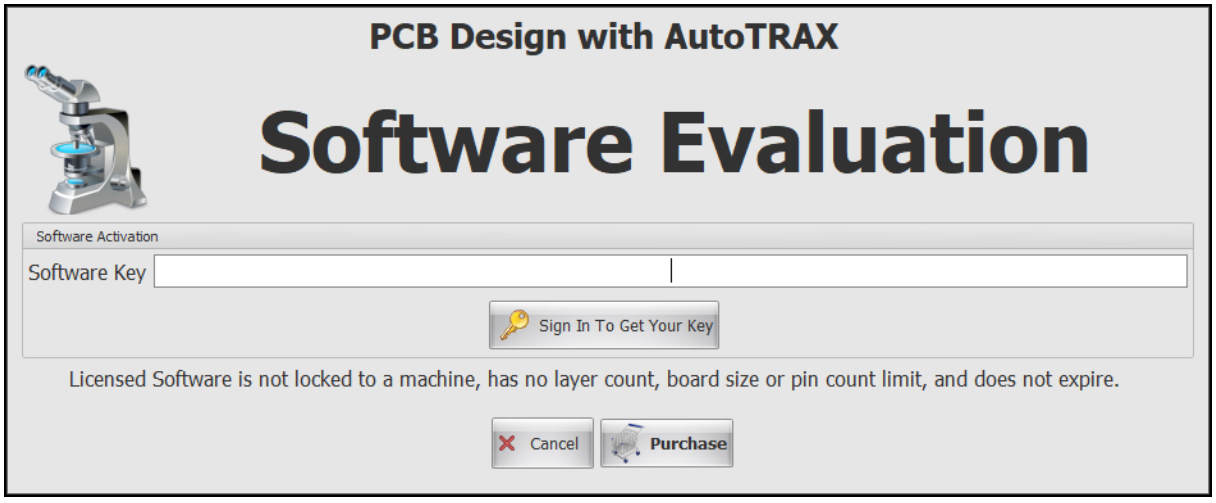
Configuration files are in C:\Users***XXXX***\AppData\Roaming\AutoTRAX DEX Software\AutoTRAX DEX\Settings

1.1.3.7 Authorizing Your Copy

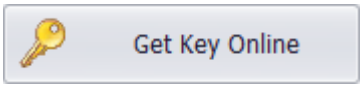
When you purchase AutoTRAX DEX you will receive a signon id and password. This will allow you to automatically authorize your copy of AutoTRAX DEX on the machine you are using.

Demo Expired

When you start AutoTRAX DEX and it has not been authorized, if the demo period has expired you will see the following dialog.



Getting you Software Key Automatically

Click 

to get your key online. You will then see the following dialog. Enter you Sign in ID and your password to authorize AutoTRAX DEX on your

machine. This will automatically retrieve the Software Key from the AutoTRAX DEX web site. You do **NOT** need the software key; AutoTRAX DEX sends it automatically to the website to get your key. **All you need is you Sign In ID and your password**. You will have been sent this automatically when you purchased AutoTRAX DEX. Check your email. If you can't find it, try your spam folder.

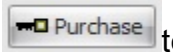
Sometimes your Firewall may prevent AutoTRAX DEX from contacting the AutoTRAX DEX web site, in this case you will have to enter the software key manually.

You can **enter the software key manually.** To get a software key you can go to https://pcbdex.com/Account/SoftwareKey

Demo Not Expired

If the demo period of your copy of AutoTRAX DEX has not expired you can authorize your copy click the 

button in the Home→Account menu.

If you do not have a valid license you will see the sign on dialog box below. If you have purchased a software license you will have received a sign in id and password by email. If you do not have it please check your spam box. If you still cannot find it then please contact us via email using this link. If you have not purchased a license the click on the purchase button 

to obtain a license

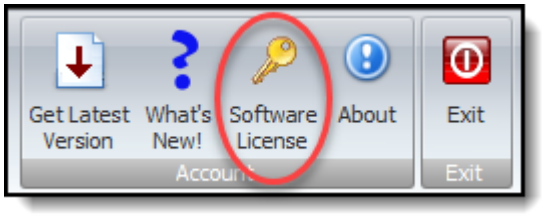
If you cannot sign on and you know you have a valid license, your firewall may be block access for AutoTRAX DEX to the internet. Click to find out how to configure your firewall.

Click here for more details about your account.

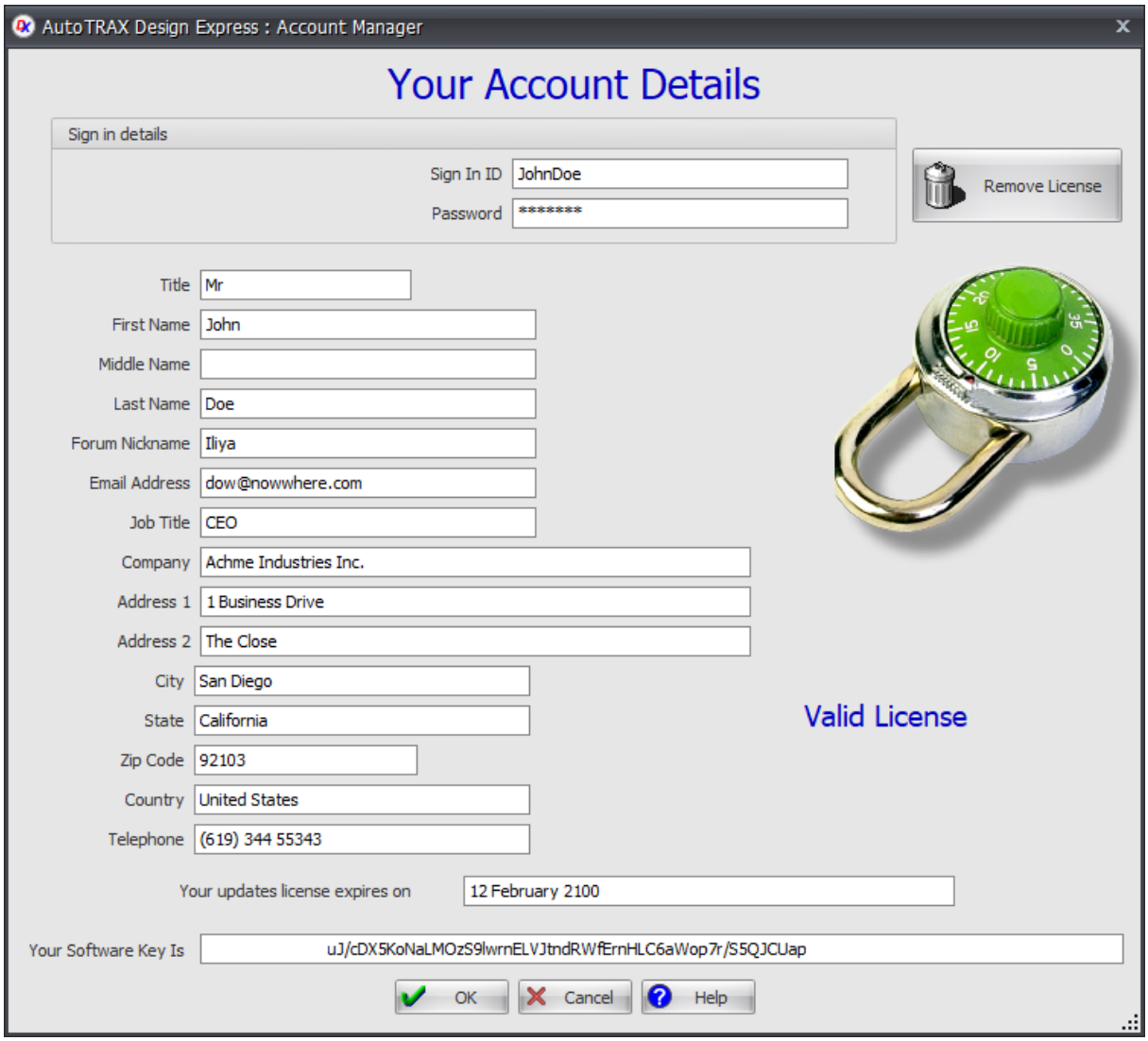
1.1.3.8 Authorizing Your Copy Machine Without Internet Acccess

To check your software license or enter a software key click the Software License button 

in the Home→Account menu.

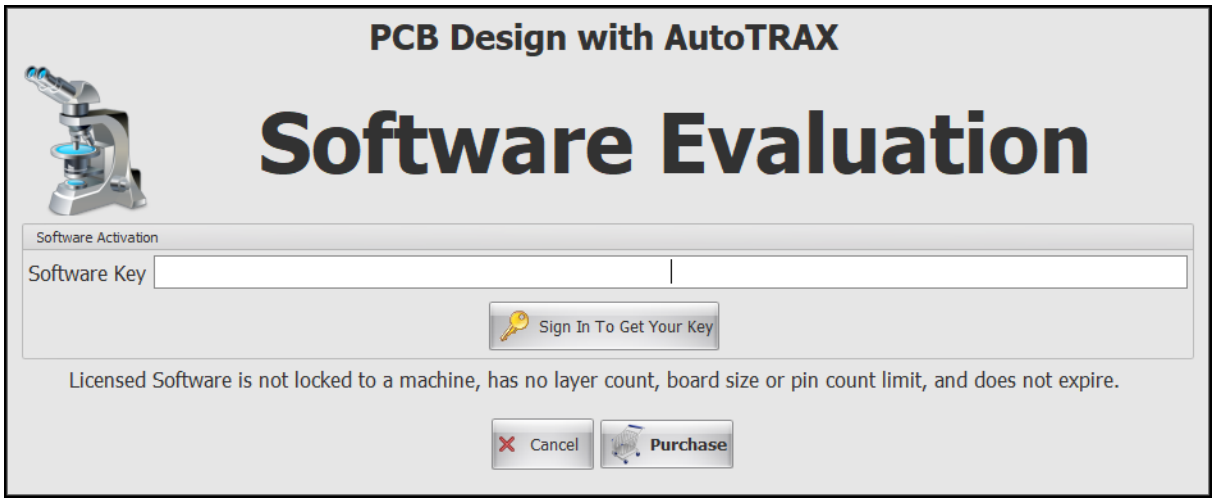


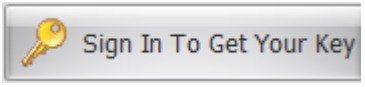
If you have a valid license the account details dialog shown below will be displayed.



If it does not have a valid license then you will see the dialog box below.

When you start AutoTRAX DEX and it has not been authorized, if the demo period has expired you will see the dialog box below.

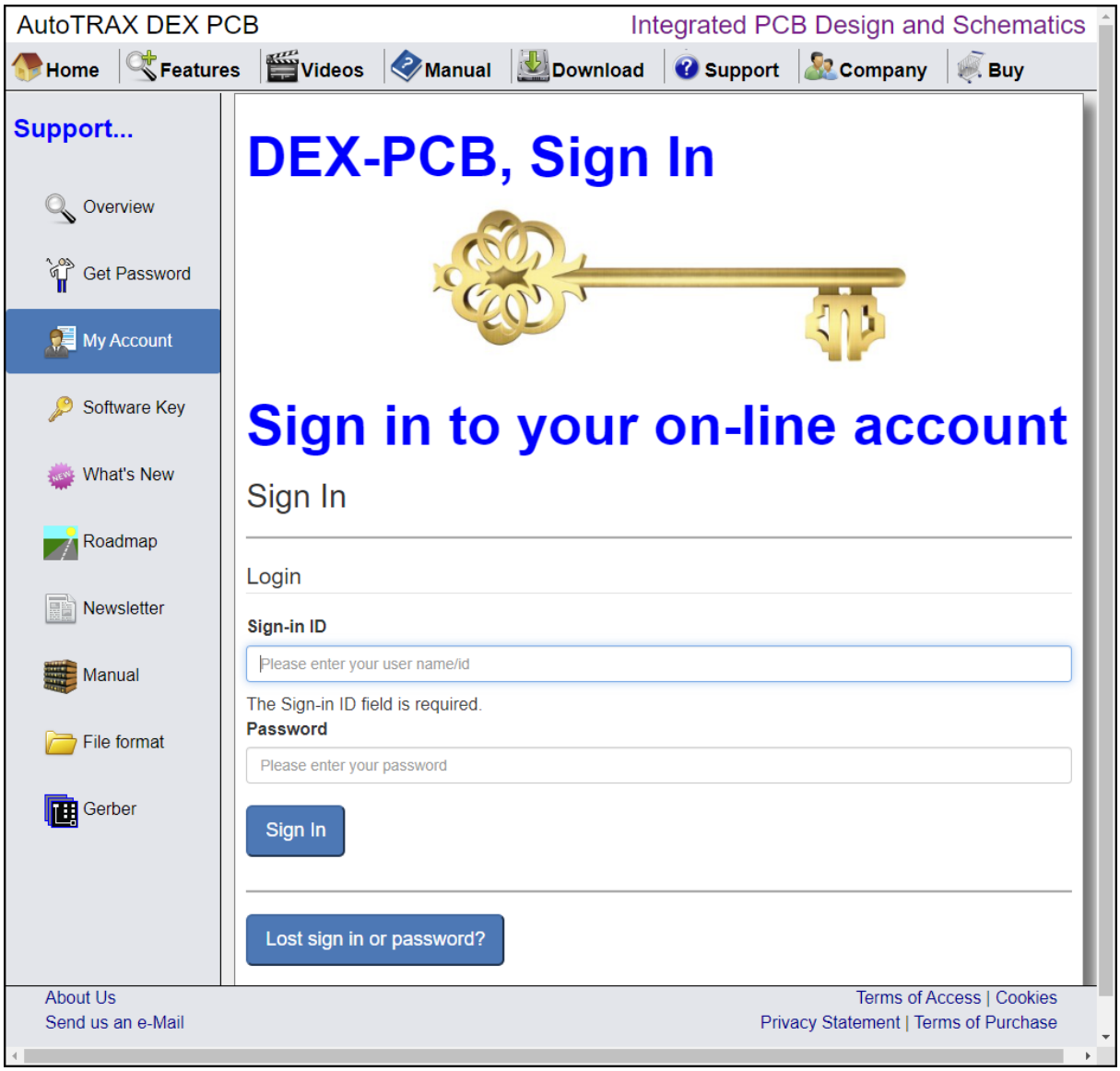


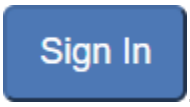
Enter the Software key or click 

1.1.3.9 Getting Your Key Online

You get you key online go to https://pcbdex.com/Account/SoftwareKey

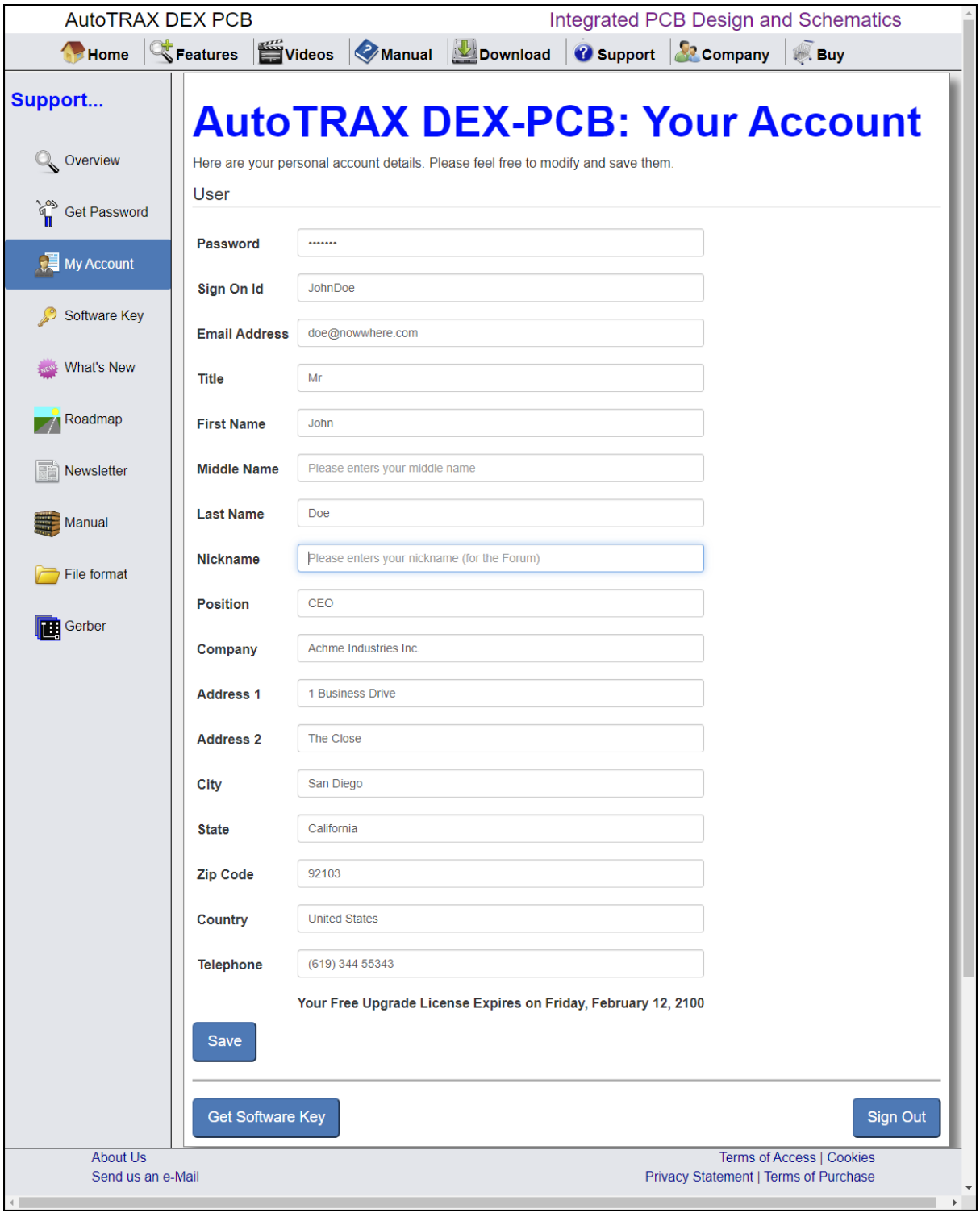
If this is the first time you will need to sign in. You will see the sign in dialog below.

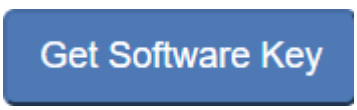


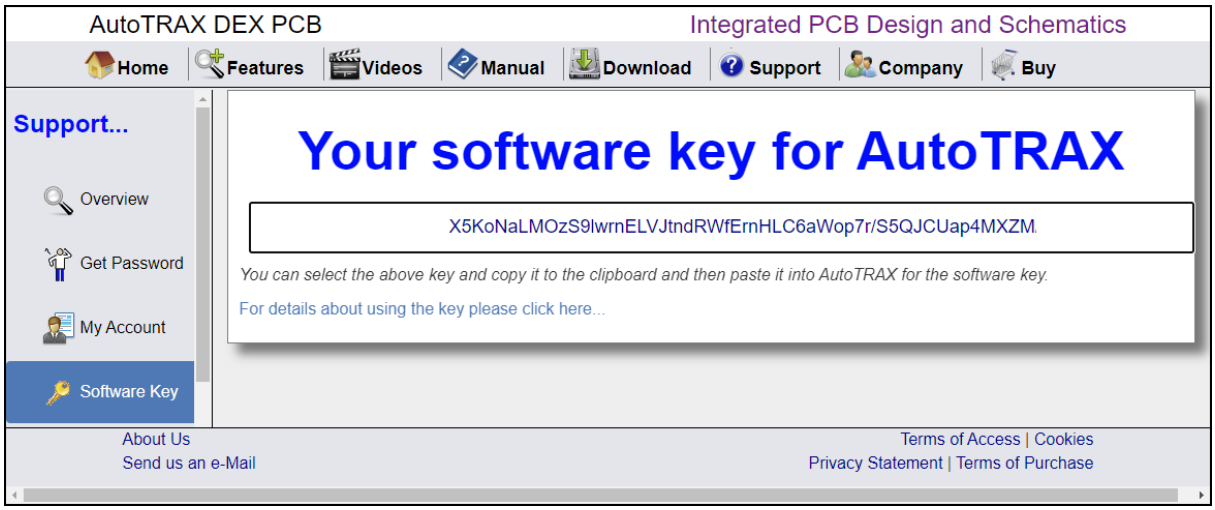
Enter your sign in ID and password and click 

. You will have been sent this automatically when you purchased AutoTRAX DEX. Check your email. If you can't find it, try your spam folder.

If you have entered a valid sign in ID and password your will see your account details.



Click the  button and you will see the Software Key page below.



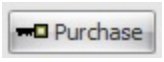
1.1.3.10 Your Account Settings

To change your account details click the Software License button 

in the Home→Account menu.

If you have a valid license the account details dialog shown below will be displayed.

The Sign On Dialog Box

If you do not have a valid license you will see the sign-on dialog box below. If you have purchased a software license you will have received a sign in id and password by email. If you do not have it please check your spam box. If you still cannot find it then please contact us via email using this link. If you have not purchased a license the click on the 

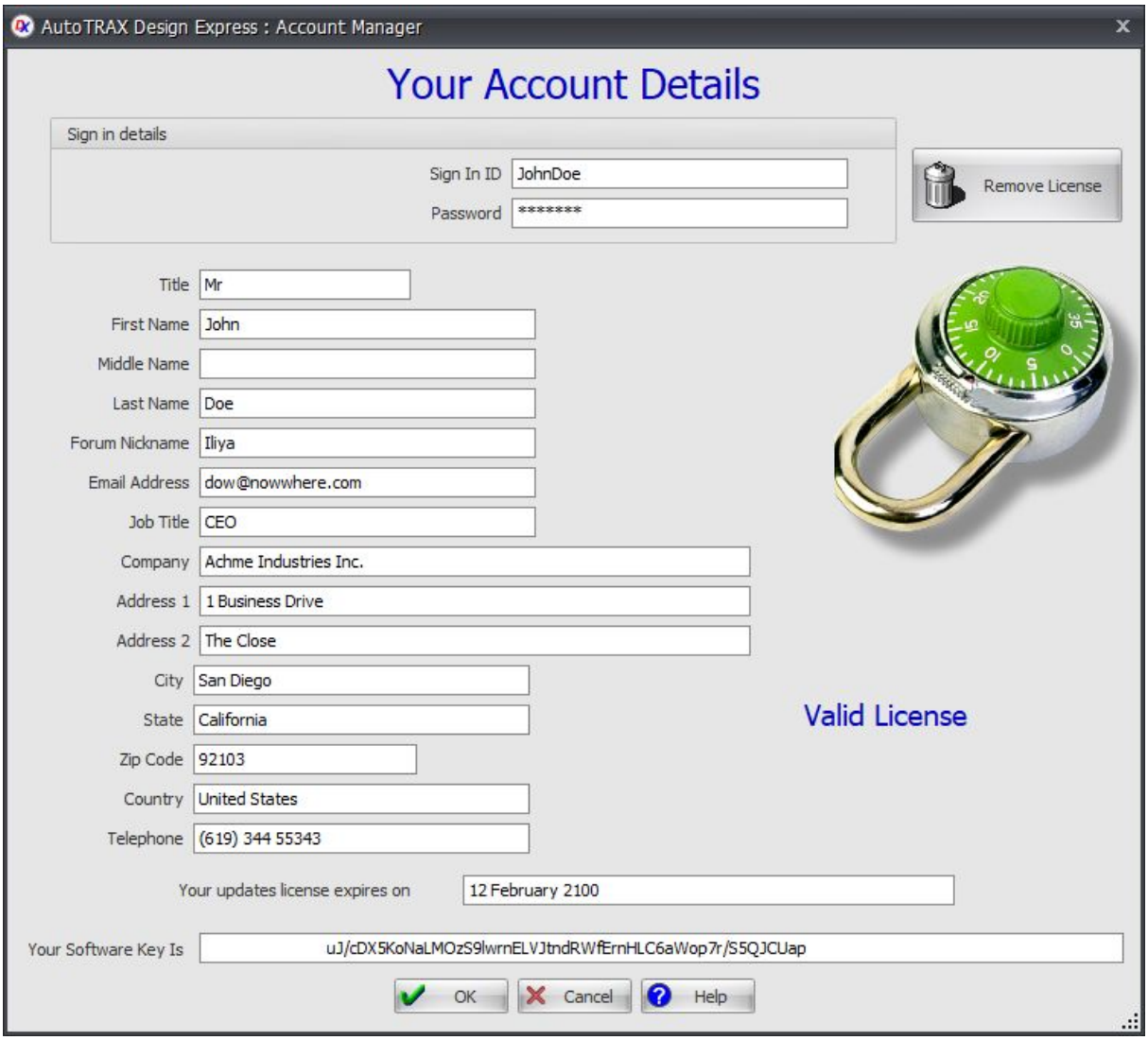
button to obtain a license

Change your details and click the OK button to save your changes. Click Cancel to close the dialog box without saving any changes.

Removing the AutoTRAX DEX software license

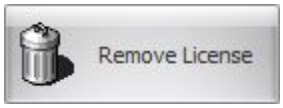
If you click on the 

button, the software license will be removed from the machine. You can easily reauthorize your copy by signing in again.

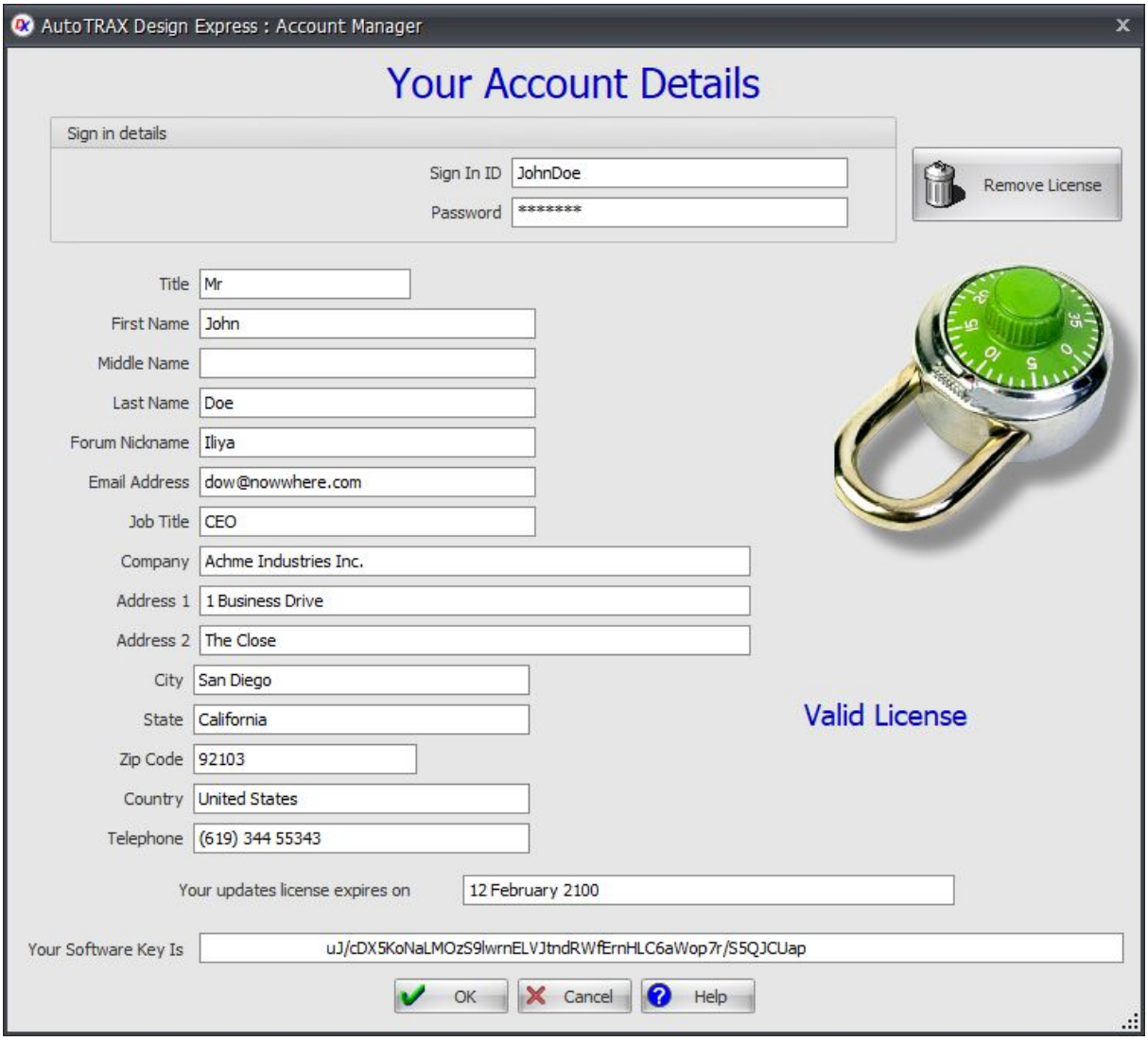


1.1.3.11 Removing Your License

To remove the software license for AutoTRAX DEX on a machine, use the account dialog. See Your Account Settings

Click the 

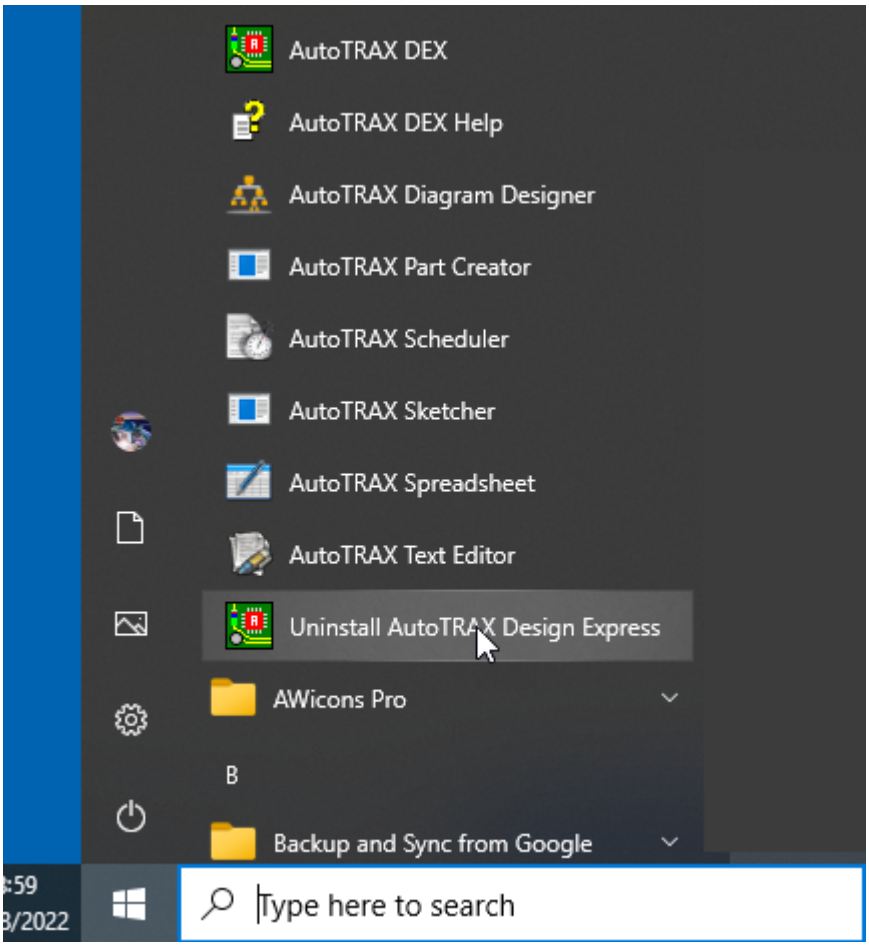
button.



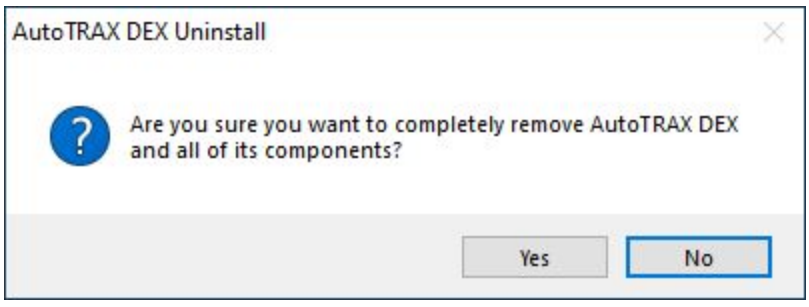
Your Account Details Dialog

1.1.3.12 Removing DEX

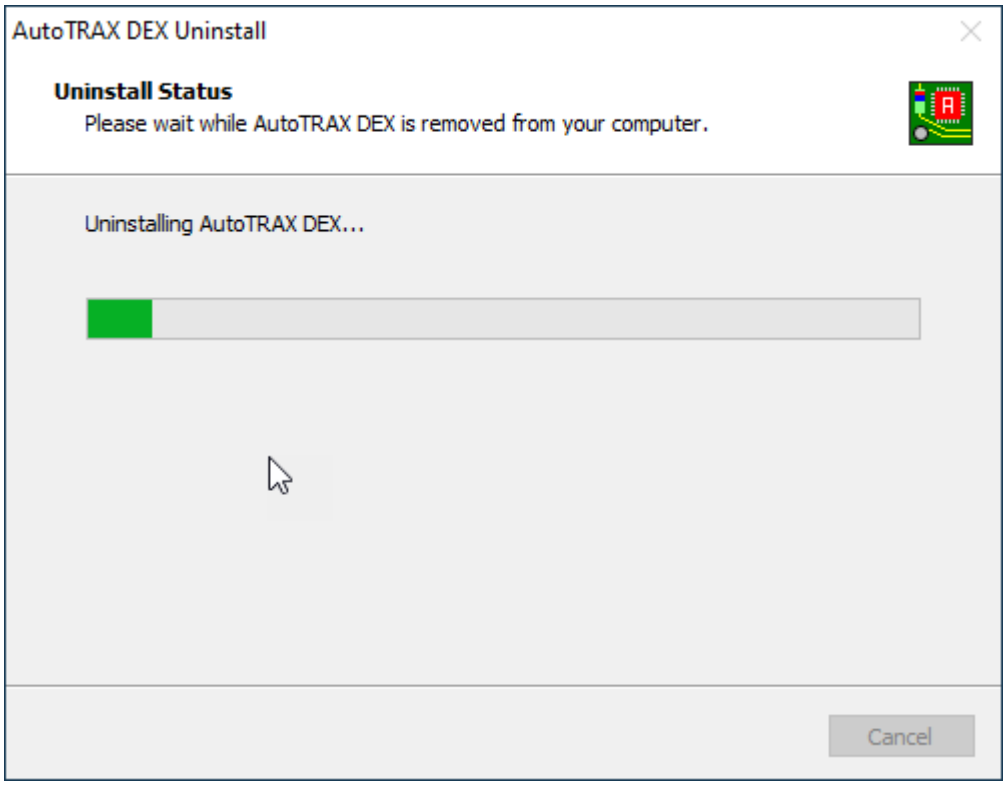
To uninstall AutoTRAX DEX Select Uninstall AutoTRAX DEX Design Express from the Application menu.



The following dialog box will appear. Click Yes to remove AutoTRAX DEX, Click No to cancel the removal.



The following progress dialog box will appear.



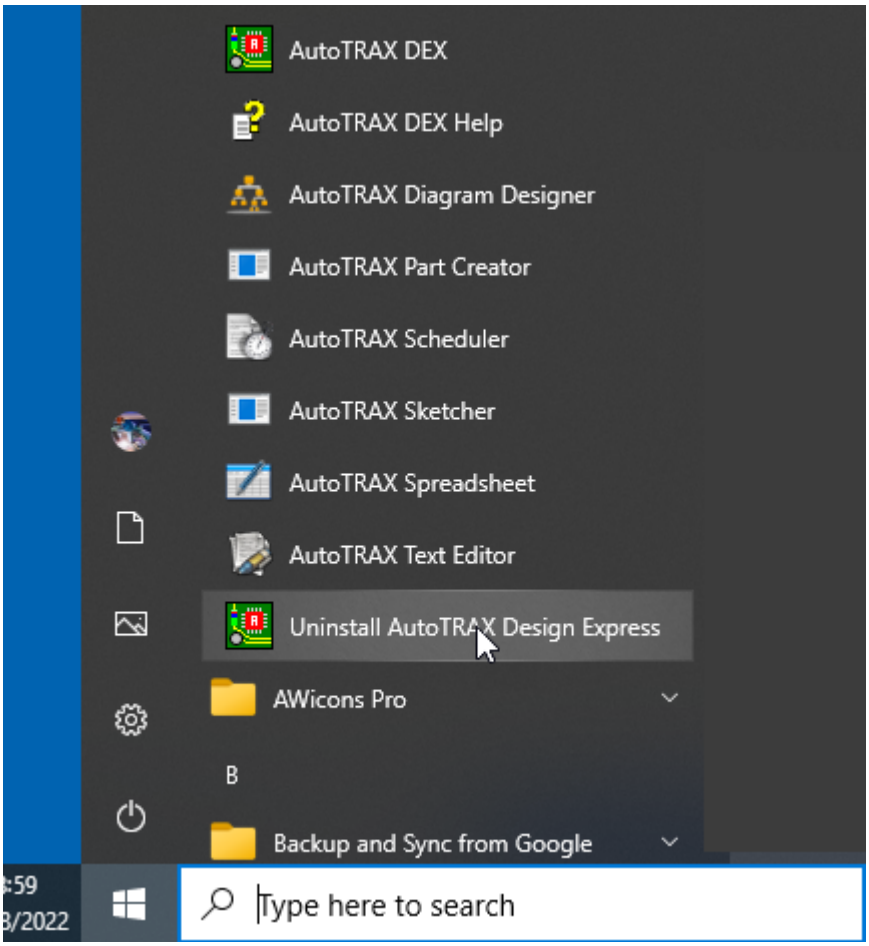
When complete the following dialog will appear. Click the OK button to close it.

1.1.3.13 Manually Removing DEX

To manually remove AutoTRAX DEX, you need to remove:

1. **The installed files.**
2. **The desktop icons.** Select them and press the delete button.
3. The AutoTRAX DEX entry in the Application Start menu. See below. ***right-click*** on the AutoTRAX DEX icon and select **Delete**.

(Note: AutoTRAX DEX does not add any entries to the Registry).



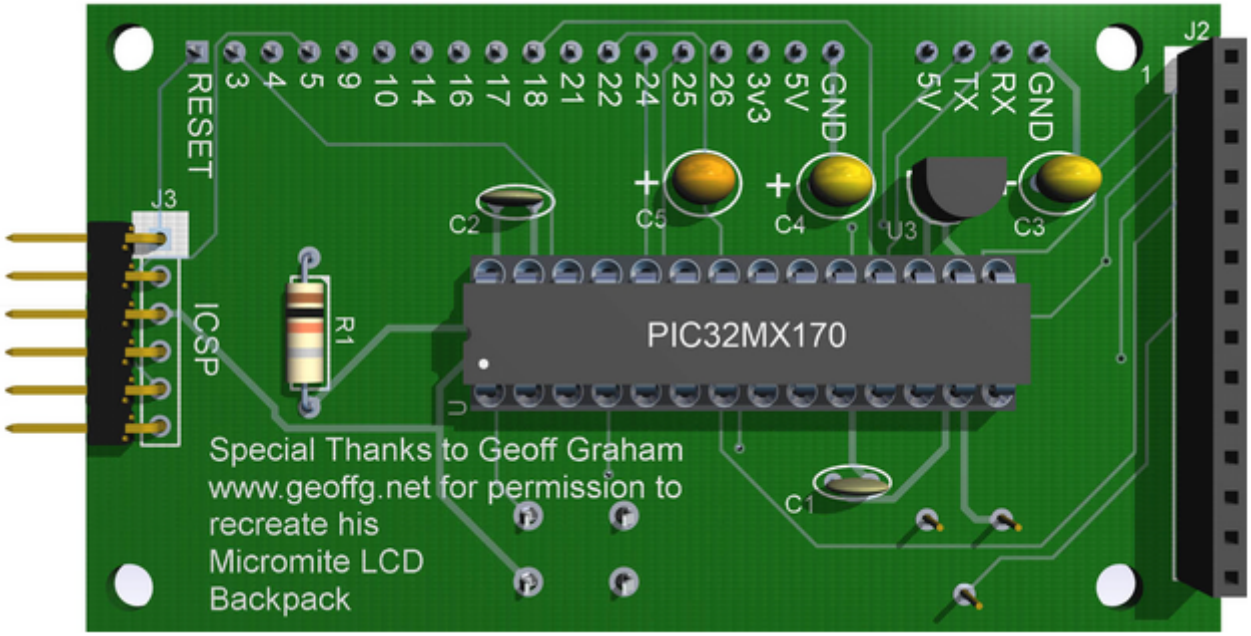
1.1.4 Quick Start

For those of you that are keen to use AutoTRAX DEX here is a quick start guide.

- AutoTRAX DEX does not have separate schematic, part and PCB editors. There is just one AutoTRAX DEX editor that does it all.
- AutoTRAX DEX integrates schematics with the PCB in a single project file.
- AutoTRAX DEX integrates symbols with a single footprint in a single part file. A multi-symbol part with have several schematics with each schematic representing one of the sibling symbols.
- There is a very comprehensive undo/redo facility that will always get you out of trouble and thus allows you to experiment.

- Use a 3 button mouse with a mouse wheel. Use the middle button to pan and the mouse wheel to zoom in/out.
- ***Right-click*** on any object in a viewport to see a context sensitive menu.
- There are comprehensive tooltips that appear when you mouse over dialog controls and objects in schematics and on the PCB.
- Press F1 for help and search for topics using the search menu. Use keyboards like you would with Google.
- View the tutorial videos. They really will help you. Take a look at the video tutorials.
- Take a look at the Sample Projects.

Click on the picture below to see an independent tutorial on PCB design using AutoTRAX DEX..



How to build a PCB design using AutoTRAX DEX

by

Mick Gulovsen

1.1.5 Using This Manual

Table of Contents

Use the **Content tab** on the left to show the table of contents. Then click the + symbol next to the chapter names to open them.

Link

Click on the chapter/page names to view the page.

Finding Specific Information

Use the **Search tab** on the left. Enter a few key words and click the **List Topics** button. If the number of topics found in the query is too large, you can narrow the search by adding more keywords.

Saving/Removing and Viewing Your Favorite Pages

To **add a favorite page** to a list of favorite pages select Favorites tab on the left and click the Add button.

To **view the favorite page c**lick on a page name in the favorites tab.

You can **remove favorites** by selecting the name in the tab and clicking on the Remove Button.

The Very Basics

To use this book, and indeed to use a computer, you need to know a few basics. You should be familiar with these terms and concepts:

- **Clicking**. This book gives you three kinds of instructions that require you to use your computer's mouse or track pad. To click means to point the arrow cursor at something on the screen and then—without moving the cursor at all—to press and release the clicker button on the mouse (or laptop track pad). A right-click is the same thing using the right mouse button. (On a Mac, press Control as you click if you don't have a right mouse button.)
- To **double-click** means to click twice in rapid succession, again without moving the cursor at all. And to drag means to move the cursor while pressing the button. When you're told to c-click something on the Mac, or Ctrl-click something on a PC, you click while pressing the c or Ctrl key (both of which are near the space bar).
- **Menus**. The menus are the words at the top of your screen or window: File, Edit, and so on. Click one to make a list of commands appear; as though they're written on a window shade you've just pulled down. This book assumes that you know how to open a program, surf the Web, and download files. You should know how to use the Start menu (Windows) or the Dock or a menu (Mac), as well as the Control Panel (Windows) or System Preferences (Mac OS X).
- **Keyboard shortcuts.** Every time you take your hand off the keyboard to move the mouse, you lose time and potentially disrupt your creative flow. That's why many experienced computer fans use keystroke combinations instead of menu commands wherever possible. When you see a shortcut like Ctrl+S (c-S) (which

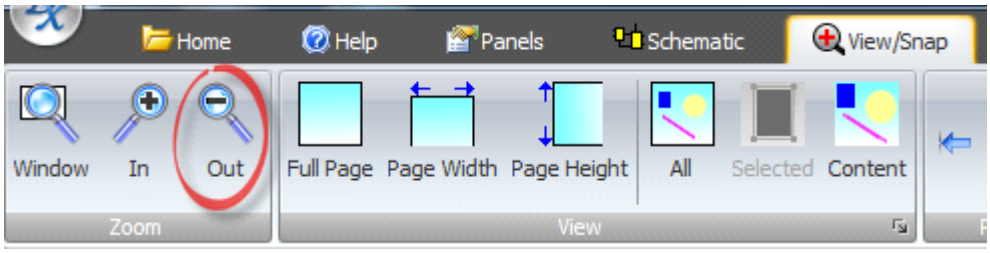
saves changes to the current document), it's telling you to hold down the Ctrl or c key, and, while it's down, type the letter S, and then release both keys.

About→These→Arrows

Throughout this book, and throughout the Missing Manual series, you'll find sentences like this one: "Open the System→Library→Fonts folder." That's shorthand for a much longer instruction that directs you to open three nested folders in sequence, like this: "On your hard drive, you'll find a folder called System. Open that. Inside the System folder window is a folder called Library; double-click it to open it. Inside that folder is yet another folder one called Fonts. Double-click to open it, too."

Similarly, this kind of arrow shorthand helps to simplify the business of choosing commands in menus, as shown with **View/Snap→Zoom→In**

Ribbon Menu



1.1.6 Project, Parts and Artwork

With AutoTRAX DEX you can produce 3 different types of work in different file types.

Project

A project is a self-contained file that contains all the schematics, text documents and the PCB to make a PCB. It relies on no other file. Project files have the file extension **.project**

A project consists of one or more schematics that define the abstract logical design. A project also contains a single PCB which defines the physical Printed Circuit Board that you are designing.

Read more...

Parts

A part is a self-contained file that contains all the symbols and the footprint/land pattern for a part. Like a project file, it relies on no other file. Part files have the file extension **.part**

A part is an electronic device that you will add to a project to construct your design.

Read more...

Artwork

An artwork is yet another self-contained file. This only contains graphics or artwork that you can include in projects and parts by dragging them from the library, Windows File Explorer or the desktop onto a design sheet. Artwork files have the file extension **.artwork**

Read more...

File Associations

When AutoTRAX DEX is installed the following file extensions are associated with AutoTRAX DEX and Windows will open then with AutoTRAX DEX unless you specify another file. To change the file association for a file ***right-click*** on the file name and select **'Open With...'**

Read more...

1.1.7 About AutoTRAX PCB Design Express (DEX)

In keeping with the Open Design philosophy of AutoTRAX DEX the following details on the development of AutoTRAX DEX are provided.

Source Code (7 August, 2023)

DEX source code (not including 3rd party code)

940,986 lines of code (6,558,169 words) in 3915 files which is equivalent to 23,524 pages.

Help Manual (7 August, 2023)

This help manual has:

- 1215 topics.
- 250,186 words.
- 23,879 paragraphs.
- 3113 images.
- 164 videos.

Development

C#

AutoTRAX DEX is mainly written in C# .NET 4 and will run on Windows Vista, Windows 7 and Windows 8., Windows 10 Windows 11.

C# (pronounced "C sharp") is a modern, general-purpose programming language developed by Microsoft as part of its .NET platform. C# was designed by Anders Hejlsberg and his team and was first released in 2000. It's syntactically similar to Java and C++, and it's particularly well-suited for building Windows desktop applications and games, but it's also used to develop web and mobile applications.

Key features and characteristics of C# include:

Object-Oriented: C# is fundamentally an object-oriented language, meaning it supports the concepts of encapsulation, inheritance, and polymorphism.

Type-Safe: The C# language is type-safe. It checks the type of an object at compile-time and doesn't allow type conversions that are unsafe.

Automatic Garbage Collection: In C#, you don't need to explicitly allocate and deallocate memory—this is managed automatically by the garbage collector.

Interoperability: C# has good interoperability, which means it can process and use code libraries that were written in other languages.

Versatility: C# can be used to create a wide variety of applications, including Windows client applications, Windows Store apps, backend systems, cloud-based services, enterprise software, and even video games (using the Unity game engine).

Modern Features: C# has many modern language features, such as indexers, delegates, events, and LINQ (Language Integrated Query), that make it a powerful and flexible language for all kinds of software development.

Part of the .NET Framework: Being a part of the .NET framework, C# comes with a large standard library that supports a variety of common programming tasks, such as string manipulation, data collection, database connectivity, and more.

 It will also run on Apple Macs.

Microsoft CLI

When AutoTRAX DEX is installed it is compiled on the target machine for the target processor. AutoTRAX DEX is therefore a 64 bit application on a 64 bit O/S and a 32 bit application on a 32 bit O/S. The code is also optimized for the target processor

version. This is impossible to do in C++ applications. There is only one installer executable.

The Common Language Infrastructure (CLI) is a specification developed by Microsoft that describes the executable code and runtime environment for running applications written in a variety of high-level programming languages. It's a key part of Microsoft's .NET platform.

The CLI includes a bytecode language, known as Common Intermediate Language (CIL, but sometimes also referred to as MSIL, for Microsoft Intermediate Language). When you compile .NET code (from any .NET language, like C#, VB.NET, F# etc.), it is compiled into this CIL.

The Just-In-Time (JIT) compiler is a part of the CLI and it's responsible for converting the CIL code into machine code that can be executed by the processor in the computer running the application. This compilation happens at runtime (i.e., "just in time" for execution) which is where the term "Just-In-Time Compilation" comes from.

Here's how the process typically works:

1. You write code in a high-level .NET language, such as C#.
2. When you compile your code, the .NET compiler converts it into CIL code. This is a form of bytecode — it's lower-level than C# code, but it's not yet machine code.
3. When you run the program, the .NET runtime loads your CIL code.
4. Just before each piece of code is executed, the JIT compiler in the .NET runtime converts the CIL code into machine code.
5. The machine code is then executed.

One of the advantages of JIT compilation is that, because the final compilation happens on the end user's machine, the compiler knows the exact configuration of the machine and can optimize the machine code for that specific environment.

.NET also supports Ahead-of-Time (AOT) compilation, in which CIL code is converted into machine code in advance, before the application is run. This can provide performance benefits and is commonly used in scenarios such as iOS development where JIT compilation is not allowed. However, JIT remains a core part of the .NET platform. AutoTRAX uses converts to CIL code into machine code on installation.

nGen

Ngen.exe, which stands for Native Image Generator, is a tool that creates native images from managed code assemblies and installs them into the native image cache on the local computer. The runtime can use native images from the cache instead of using the just-in-time (JIT) compiler to compile the original assembly.

Here's why AutoTRAX uses Ngen.exe:

Performance: Applications can load faster because they're pre-compiled to native code, reducing the work done by the JIT compiler during application startup.

Shared Code: If you have assemblies that are shared between applications, using Ngen can improve performance because the native images can also be shared, so each application doesn't need to have its own JIT-compiled code.

Security: Because the assemblies are precompiled, there's less opportunity for malicious code to inject itself at runtime.

However, using Ngen.exe does have some disadvantages:

Disk Space: Native images are often larger than the original assemblies, so they consume more disk space.

Versioning: If you update an assembly, you have to regenerate the native image.

Initial Compilation: There is an overhead to compile the assemblies to native code in the first place.

Ngen.exe is used by the AutoTRAX installer.

Winforms

For the User Interface, AutoTRAX uses WinForms.

Windows Forms (WinForms) is a Graphical User Interface (GUI) class library included in Microsoft's .NET Framework. Developers use it to create Windows desktop applications. It's one of the .NET Framework's two main GUI libraries, the other being Windows Presentation Foundation (WPF).

WinForms was introduced with the .NET Framework in 2002 and has been widely used for desktop applications in the Windows environment. It provides a set of classes for creating windows, controls, and components. These classes include everything from basic elements, such as buttons, checkboxes, and text boxes, to more complex elements like dialog boxes, menus, and toolbars.

3D and Open GL

AutoTRAX uses OpenGL for 3D.

OpenGL, which stands for Open Graphics Library, is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector

graphics. The API is typically used to interact with a Graphics Processing Unit (GPU), to achieve hardware-accelerated rendering.

Since OpenGL is a low-level, high-performance library, it can create high-quality graphics in real-time, which makes it particularly useful in the field of computer games, simulations, and modeling applications. It provides functions to draw geometric primitives (like points, lines, and polygons), transformations of graphical objects, texture mapping, and many others.

OpenGL is platform-independent and served as the foundation for several other APIs and libraries, like WebGL (for web applications) and OpenGL ES (a simplified version for mobile and embedded systems).

Previous Versions

The previous version of AutoTRAX DEX was AutoTRAX DEX EDA. This was written in C++ and MFC.

MFC stands for Microsoft Foundation Class Library. It is a collection of classes (coded in C++), provided by Microsoft to make it easier to develop desktop applications for the Windows operating system. Essentially, MFC provides an Object-Oriented interface to the Windows API, which is normally programmed in C.

A version of AutoTRAX DEX was developed using C++ and QT but this was abandoned in favor of .NET.

Both AutoTRAX DEX and AutoTRAX DEX EDA have been used, under license, to develop application for other software companies.

Qt (often pronounced as "cute") is a free and open-source widget toolkit for creating graphical user interfaces as well as cross-platform applications that run on various software and hardware platforms such as Linux, Windows, macOS, Android or embedded systems with little or no change in the underlying codebase while still being a native application with native capabilities and speed.

Qt was first developed by Trolltech, a Norwegian software company. Nokia later acquired Trolltech, and then the Qt project was distributed under the terms of the GNU Lesser General Public License (LGPL), among others.

Website

The AutoTRAX DEX website was written in C# with ASP.NET MVC 5. It is an HTML 5 site and uses SQLServer, WebGL, SVG and JavaScript with j Query.

ASP.NET MVC 5 is a framework for building scalable, standards-based web applications using well-established design patterns and the power of ASP.NET and the .NET Framework.

Visual Studio

AutoTRAX DEX is developed with Visual Studio Ultimate. Design methodology is Scrum Agile.

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services, and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store, and Microsoft Silverlight.

Key Features of Microsoft Visual Studio include:

Code Editor: The code editor supports syntax highlighting and code completion using IntelliSense for not only variables, functions, and methods but also language constructs like loops and queries.

Debugger: Visual Studio includes a debugger that works both as a source-level debugger and as a machine-level debugger. It works with both managed code and native code and can be used for debugging applications written in any language supported by Visual Studio.

Designer: Visual Studio includes a host of visual designers to aid in the development of applications. These include form designers for building UI, web designer for building web pages and applications, class designer for visualizing and editing the structure of classes, and many others.

Integrated Tools: Visual Studio comes integrated with many tools to aid in software development. These include a form designer for building GUI applications, web designer, class designer, and database schema designer, among others.

Support for multiple languages: Visual Studio supports multiple programming languages by allowing the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists.

Extensions: Visual Studio also supports the creation and use of plug-ins, which enhance the functionality of the IDE or provide integrations with other software or services. This makes Visual Studio a highly extensible platform.

SqLite

AutoTRAX DEX uses SQLite. SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. SQLite is the most used database engine in the world. SQLite is built into all mobile phones and most computers and comes bundled inside countless other applications that people use every day.

SQLite is a software library that provides a relational database management system (RDBMS). The distinguishing feature of SQLite is that it's embedded: instead of running as a separate process with its own system-level privileges, SQLite works directly with the application that uses it. This makes it extremely lightweight, efficient, and easy to integrate with a wide range of applications.

SQLite provides an interface to a "database" that is a single file on disk. It implements most of SQL standard, including transactions, and it can use indices to speed up queries. However, as a part of its simplicity, it doesn't support some more advanced RDBMS features, such as right outer joins or full outer joins.

Some of the key characteristics of SQLite include:

Serverless: SQLite doesn't run as a server process, which differentiates it from other databases such as MySQL or PostgreSQL. Applications open SQLite files directly, which simplifies setup and administration.

Zero-Configuration: There's no setup or administration needed to start using an SQLite database. This makes it a popular choice for development, testing, and even some production environments.

Transactional: SQLite transactions are fully ACID-compliant, allowing safe access from multiple processes or threads.

Self-Contained: A complete SQLite database is stored in a single cross-platform disk file.

SQLite is used widely both in industry and in academia, for both prototyping and for lightweight applications that don't require the full power of a larger RDBMS or where database needs are modest. As of my knowledge cutoff in September 2021, SQLite is embedded in a wide range of popular software, including most smartphones (both iOS and Android), many web browsers, and countless standalone applications.

1.1.8 Tutorials

Sample Projects

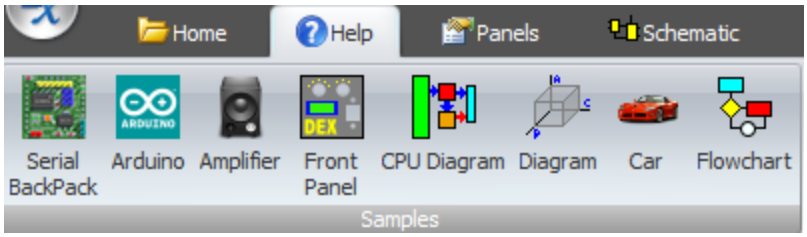
Demonstration Video

A Simple Resistor Ladder

Web Tutorial

1.1.8.1 Sample Projects

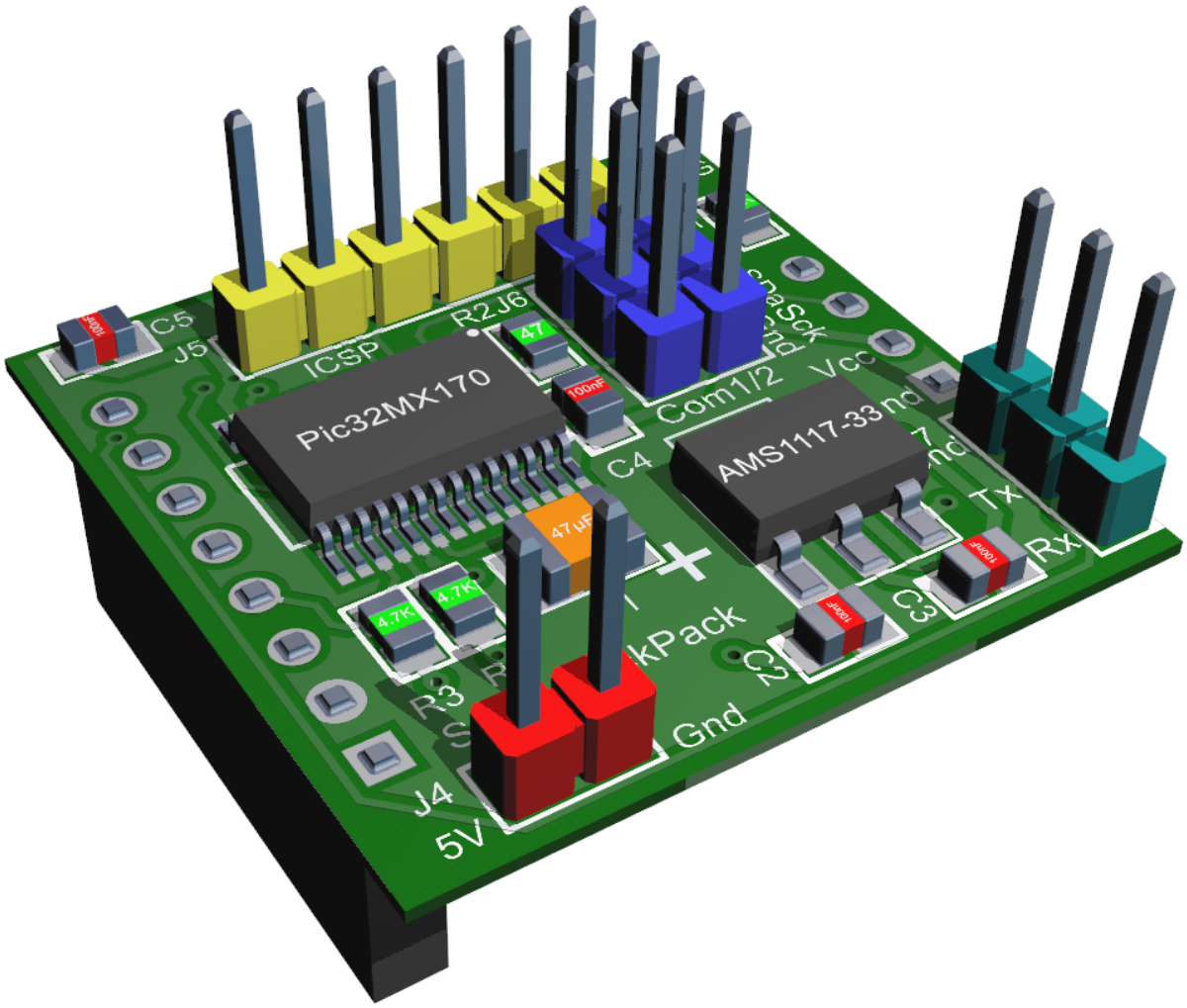
There are several sample projects in the Help→Samples buttons group. These illustrates many of the powerful features and capabilities in AutoTRAX DEX.



Ribbon Menu Samples

The Serial Back-Pack

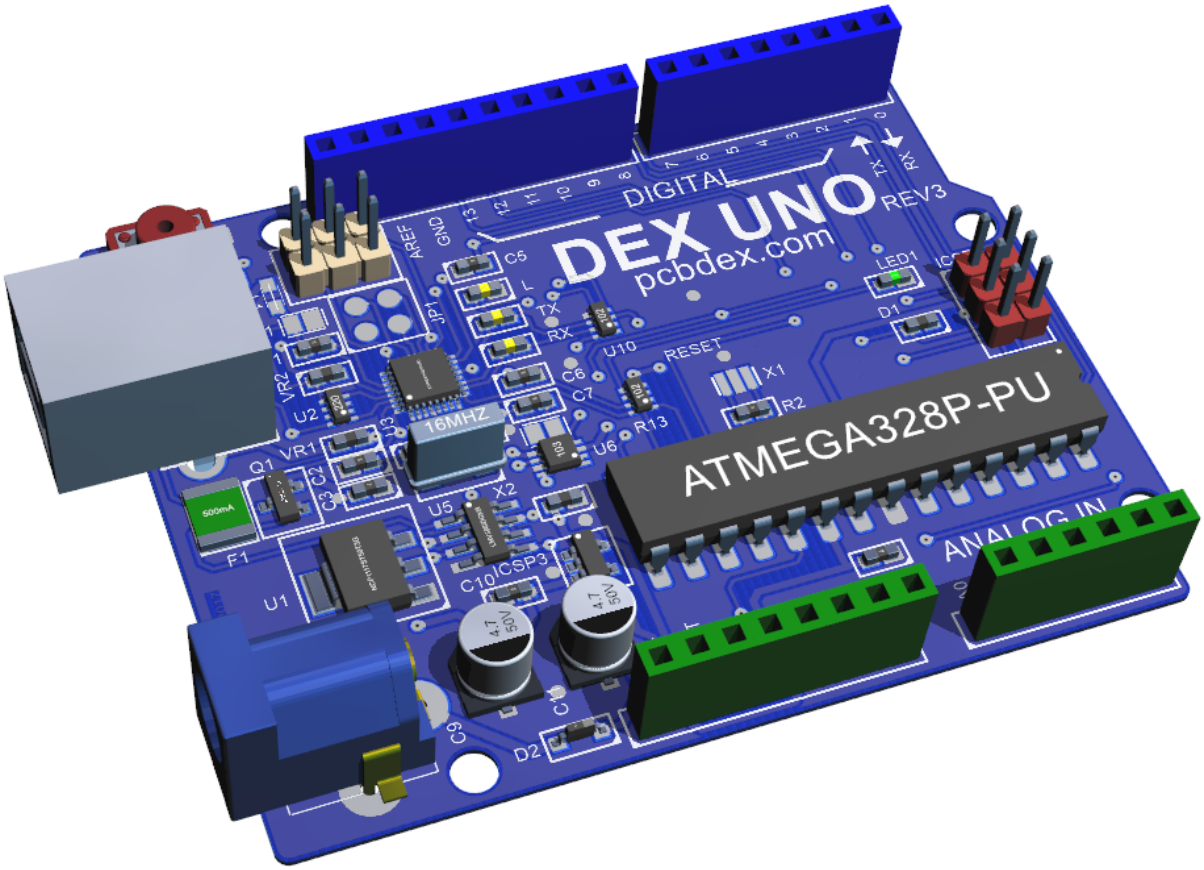
This project was donated by Mick Gulovsen from Melbourne, Australia.



The Serial Back-Pack Sample Project

The AutoTRAX DEX UNO

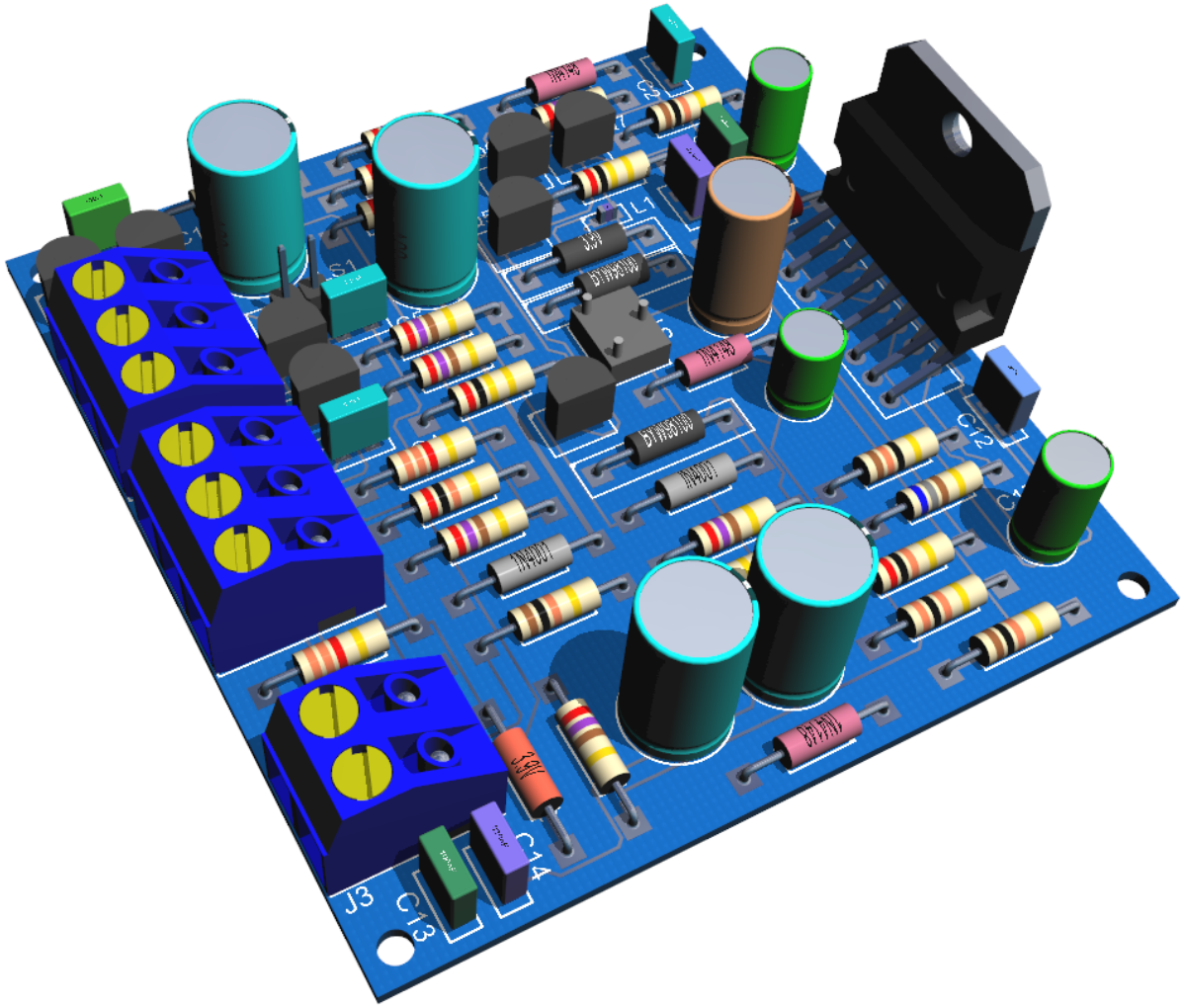
AutoTRAX DEX Uno is a microcontroller board based on the ATmega328P.



The AutoTRAX DEX UNO SAmple Project

The Amplifier

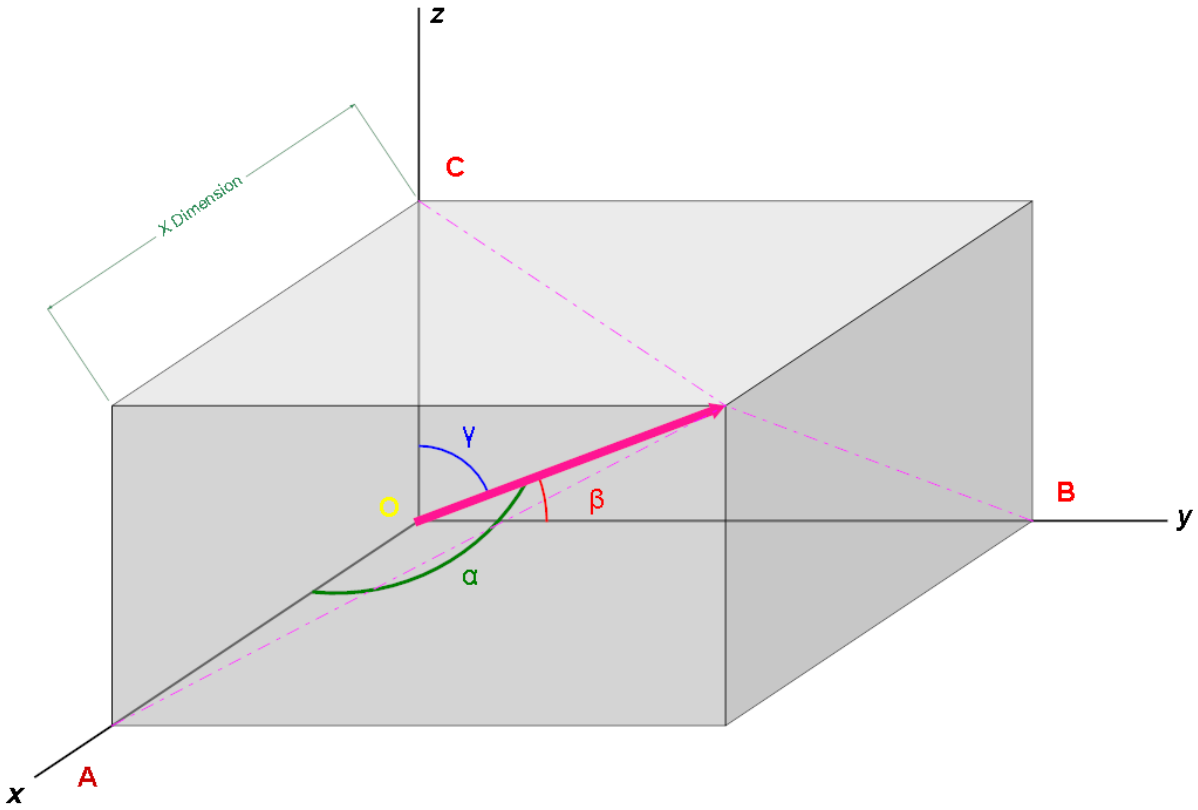
The amplifier project shows a sample design of an amplifier using all TPH components.



The Amplifier SAmple Project

The Diagram

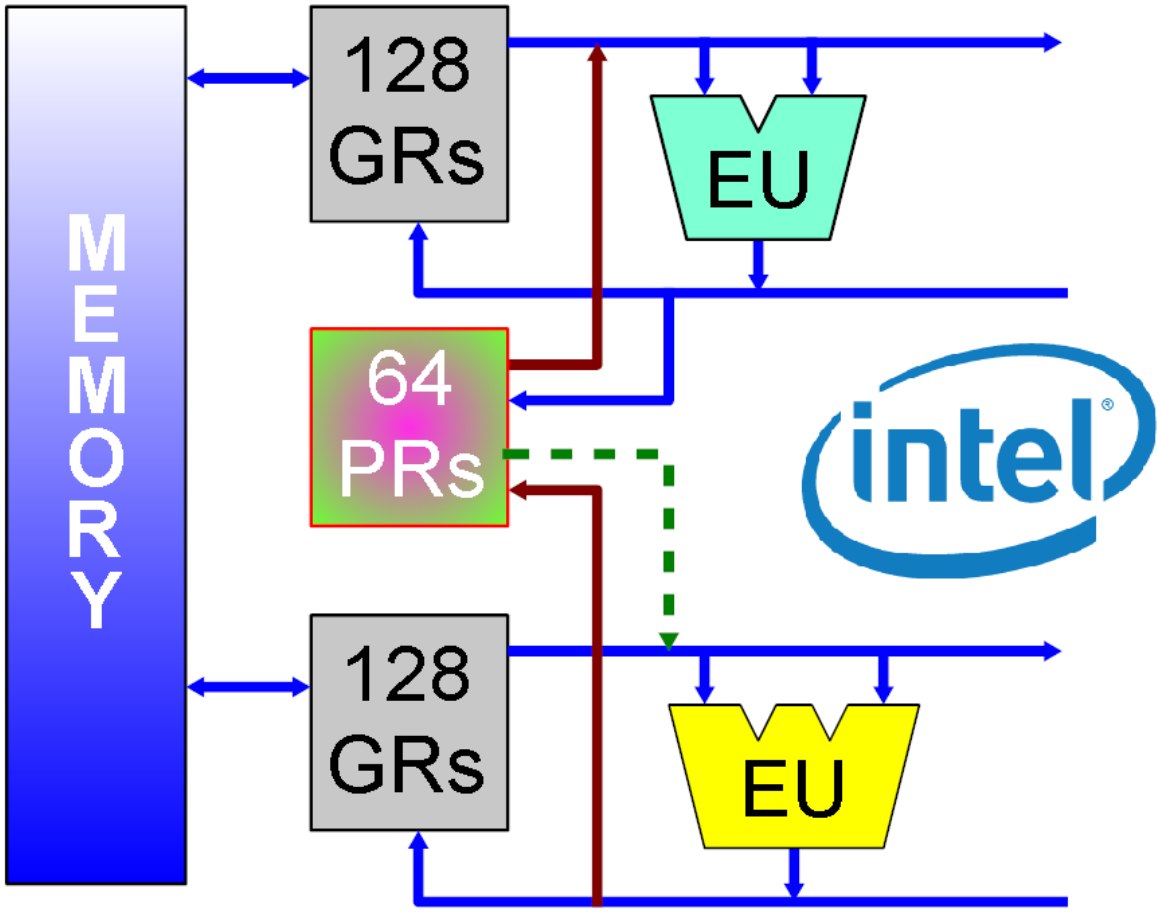
This project is a simple diagram that shows you the graphical powers of AutoTRAX DEX.



The Diagram Sample Project

The CPU Diagram

This project shows you how you can use the powerful graphics capabilities in AutoTRAX DEX to diagram a simple CPU using graphical blocks and connecting wires.

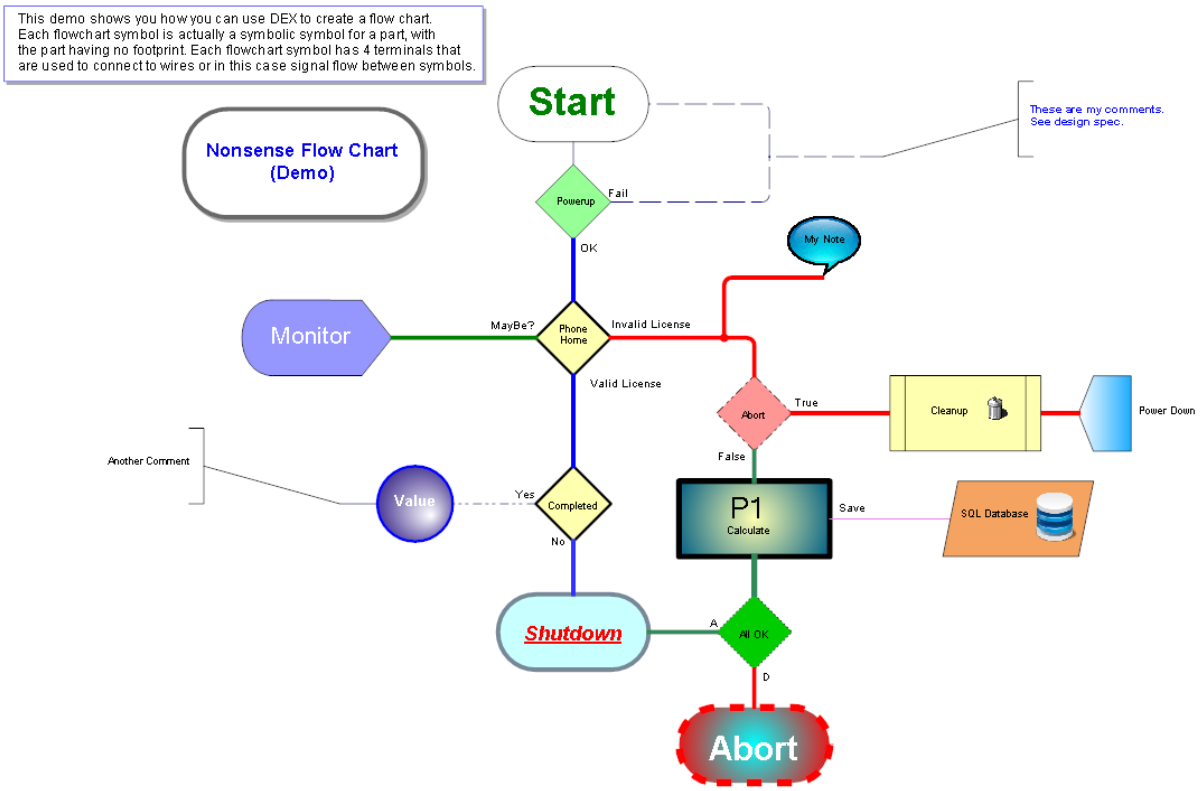


General Organization for IA-64

The CPU Diagram Sample Project

The Flowchart Design

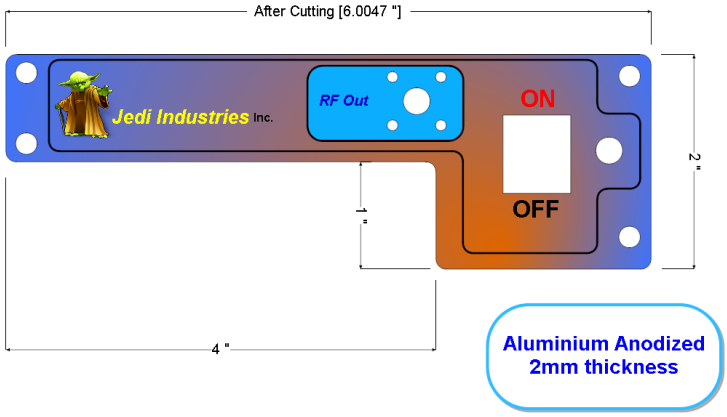
This project shows you part of the electrical circuit in an automobile such as you might see in a typical user manual.



The Flowchart Design Sample Project

The Front Panel

With AutoTRAX DEX you can even design front panel displays. The front panel project contains an example front panel design.



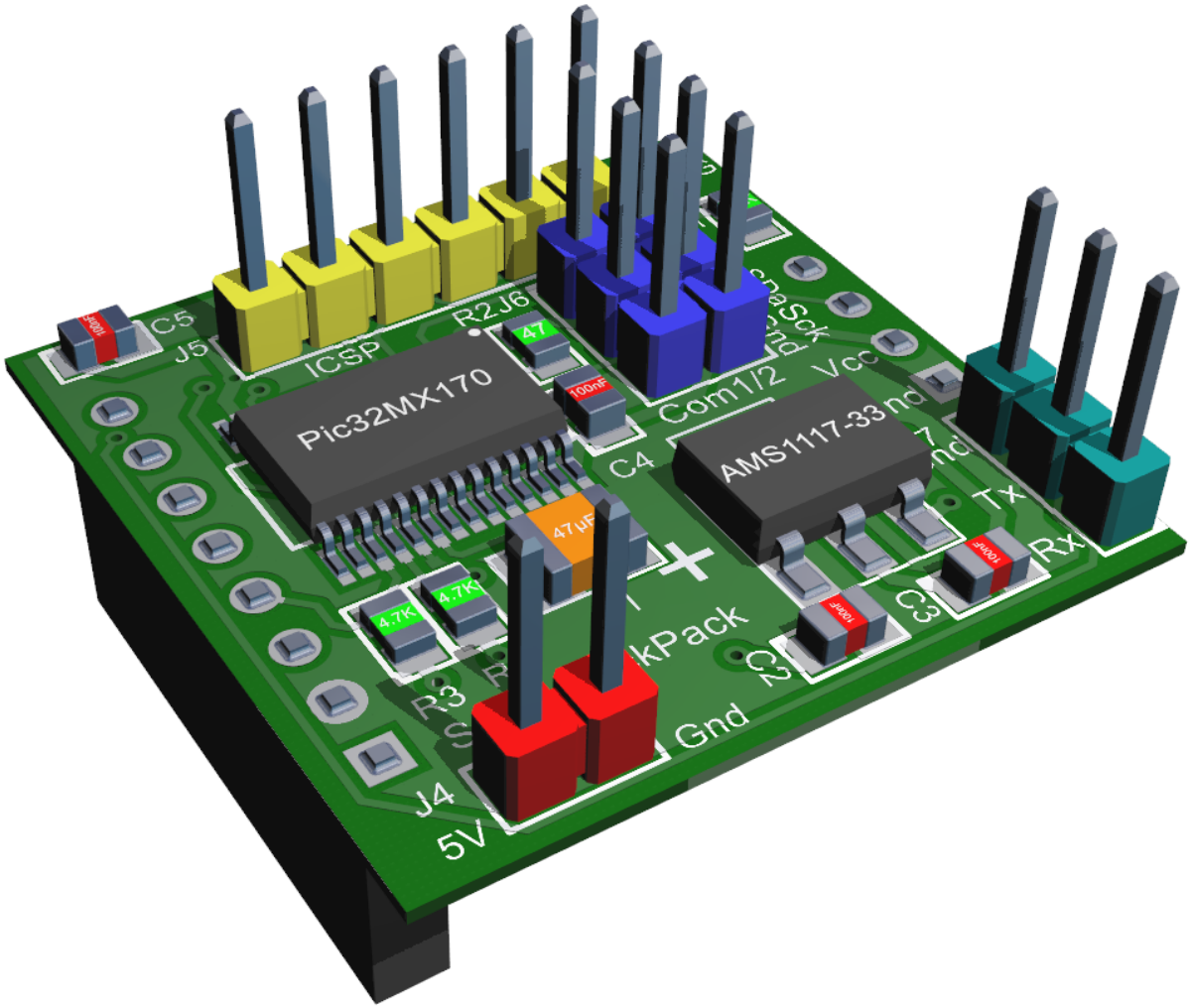
The Front Panel Sample Project

1.1.8.1.1 The Serial Back-Pack

Click Help→Samples→ 

to open the **Serial Backpack** sample project.

This project was donated by Mick Gulovsen from Melbourne, Australia. He has created a tutorial on using AutoTRAX DEX at https://pcbdex.com/HowToMakeAPCB/



The Serial Back-Pack Sample Project

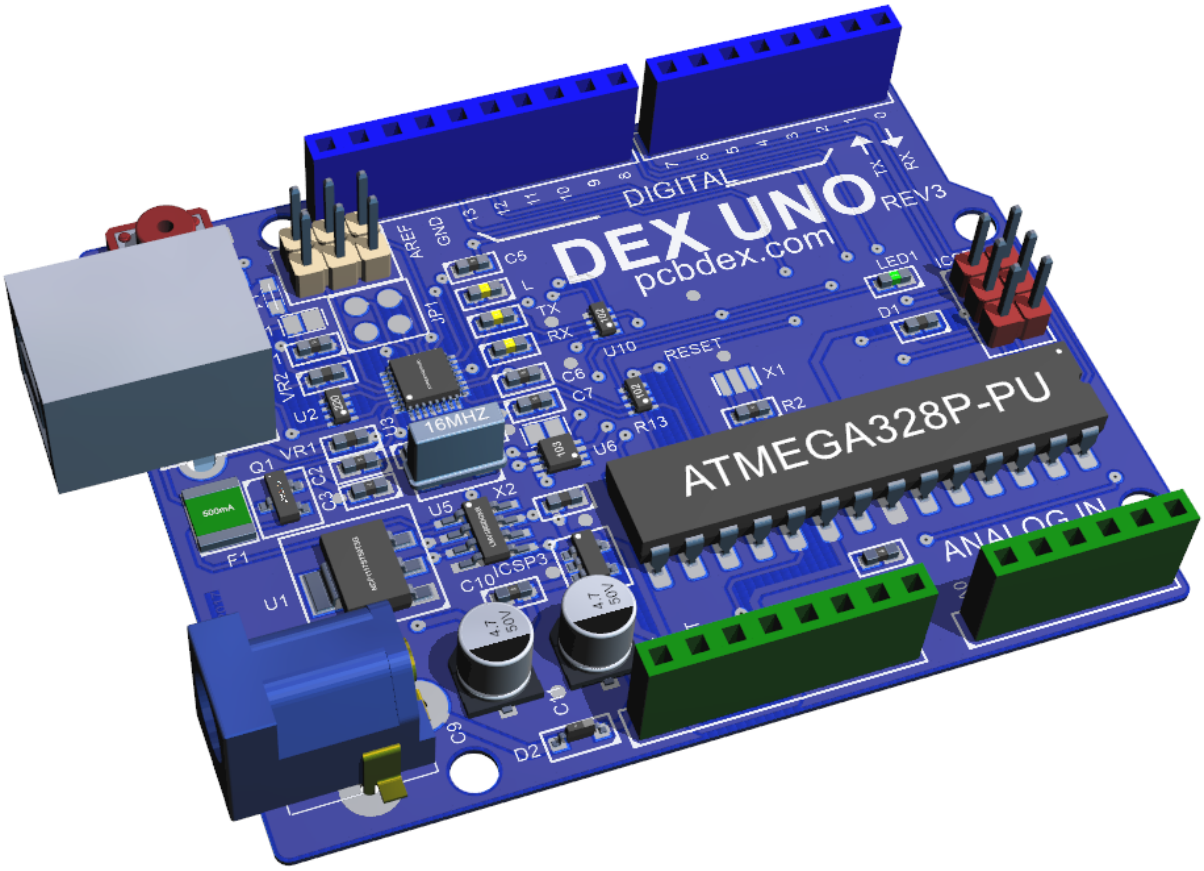
1.1.8.1.2 The DEX UNO

Click Help→Samples→ 

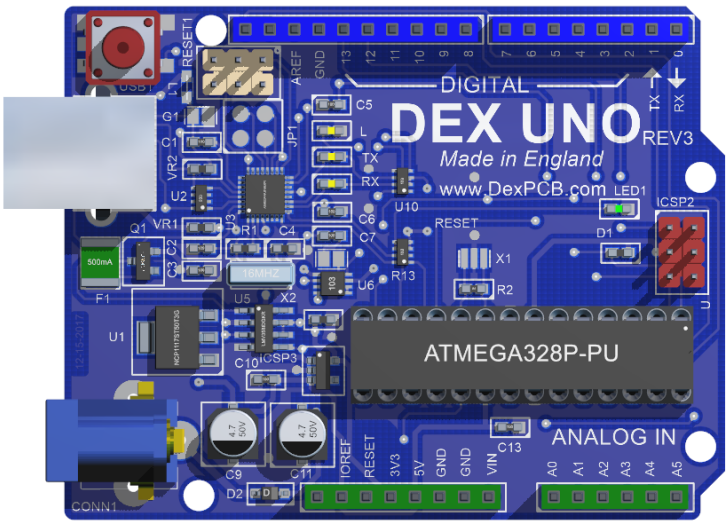
to open the **AutoTRAX DEX UNO** sample project.

AutoTRAX DEX Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

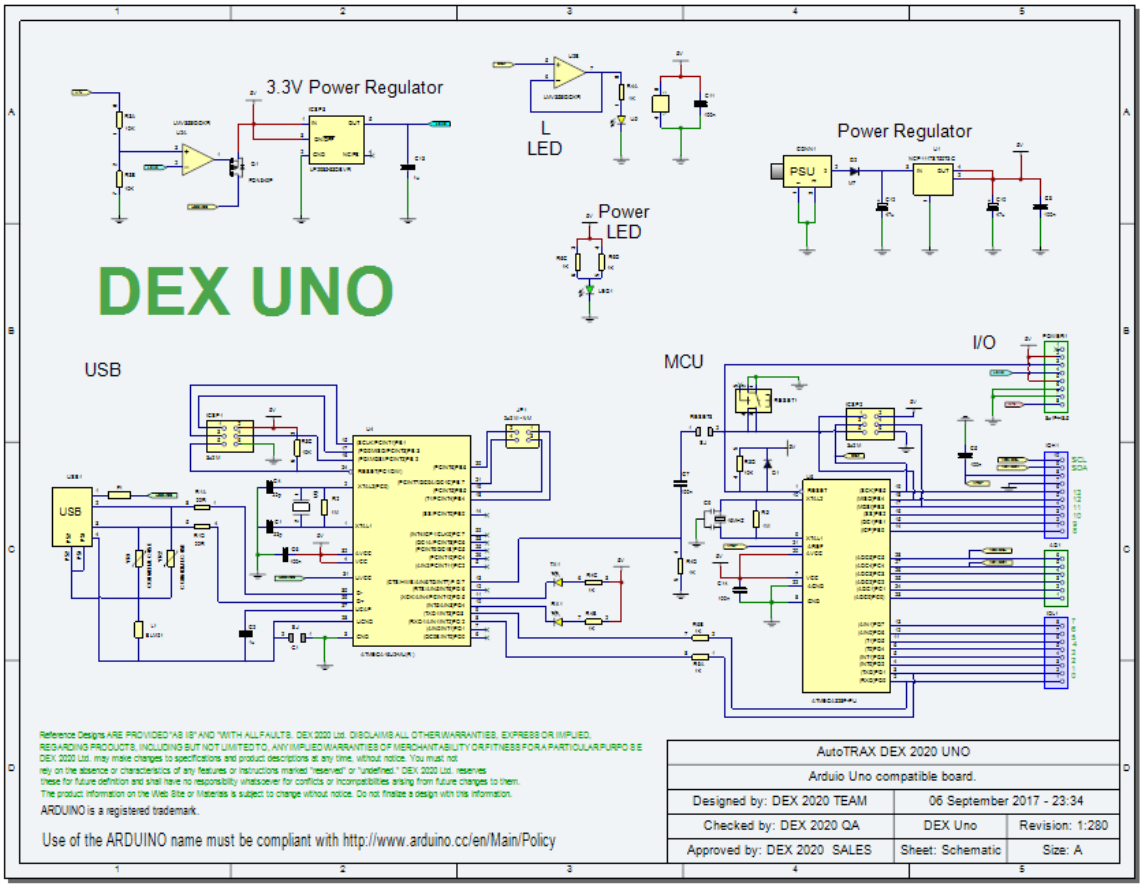
You are free to modify and use; however, you must comply with the Arduino license.



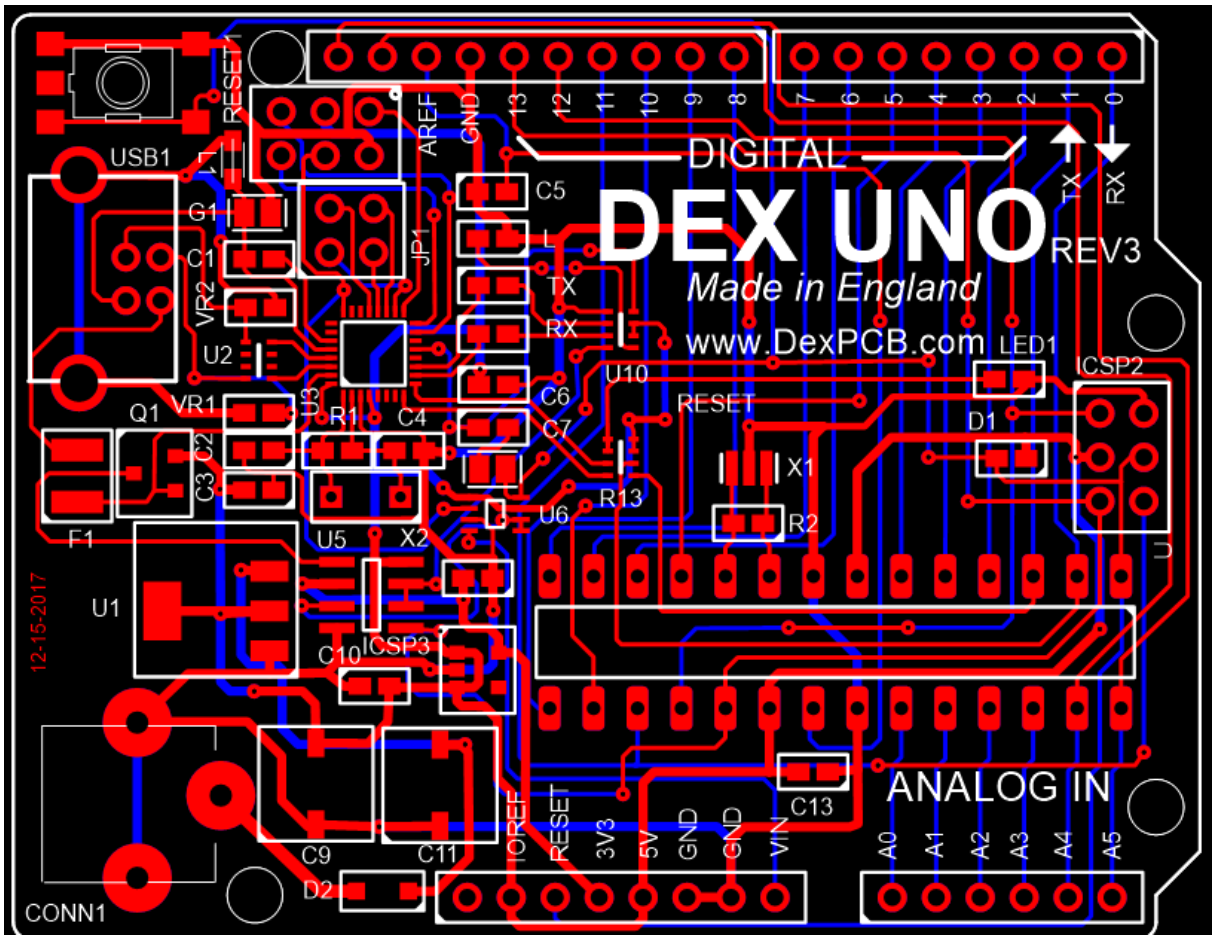
The AutoTRAX DEX UNO Sample Project



Orthographic 3D View from the Top



The Schematic



The PCB

Arduino

Arduino is an open-source electronics platform that uses easy-to-use hardware and software. The hardware consists of a physical programmable circuit board (often referred to as a microcontroller), and the software is a piece of software, or Integrated Development Environment (IDE), that runs on your computer and is used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program.

The Arduino project started in 2005 as a project for students at the Interaction Design Institute Ivrea in Ivrea, Italy. It was aimed at providing a low-cost and easy

way for novices and professionals to create devices that interact with their environment using sensors and actuators.

Since then, Arduino has been widely adopted in a variety of fields, including hobby electronics, education, and industrial applications. It has a large community and a wide array of supplementary add-on "libraries" that enable users to extend the base functionality. It also supports various types of boards beyond the initial Arduino UNO, like the Mega, Nano, Mini, and boards with WiFi and BLE capabilities like the MKR WiFi 1010.

The philosophy of Arduino has always been to make hardware and software accessible to artists, designers, hobbyists, and anyone interested in creating interactive objects or environments. This focus on ease of use makes it an excellent choice for people new to programming and electronics.

Arduino UNO

The Arduino Uno is one of the most popular Arduino boards. It is often the first board many people use when they are learning about Arduino or beginning to experiment with electronics.

Here are some of the key features of the Arduino Uno:

- **Microcontroller:** It is based on the ATmega328P microcontroller.
- **Digital I/O Pins:** It has 14 digital input/output pins. Six of these pins can be used as Pulse Width Modulation (PWM) outputs.
- **Analog Input Pins:** It has 6 analog inputs.
- **Operating Voltage:** It operates at 5 volts.
- **Input Voltage:** Recommended range is 7 to 12 volts, but limits are 6-20 volts.
- **Flash Memory:** 32 KB of which 0.5 KB is used by the bootloader.
- **SRAM:** 2 KB.
- **EEPROM:** 1 KB.
- **Clock Speed:** 16 MHz.

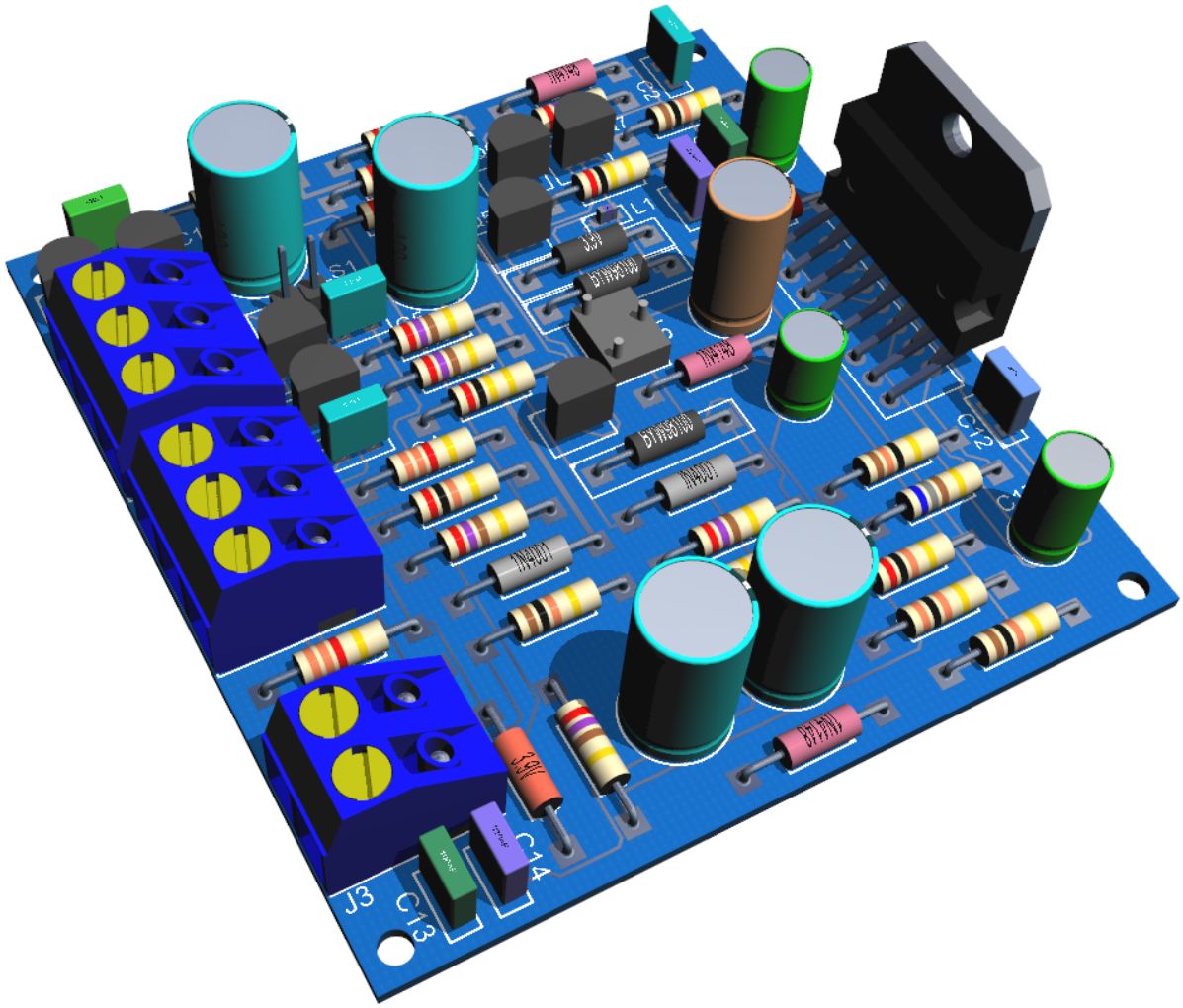
The Arduino Uno is unique in that it includes a built-in USB interface, which allows it to be easily connected to a computer for programming. The board can be powered either from the USB connection or with an external power supply.

Arduino Uno has a wide variety of applications. It can be used to control LEDs, read data from a variety of sensors, control motors, and much more. The ease with which the Arduino can be used to interact with a wide range of hardware makes it a valuable tool for anyone interested in electronics or physical computing.

1.1.8.1.3 The Amplifier

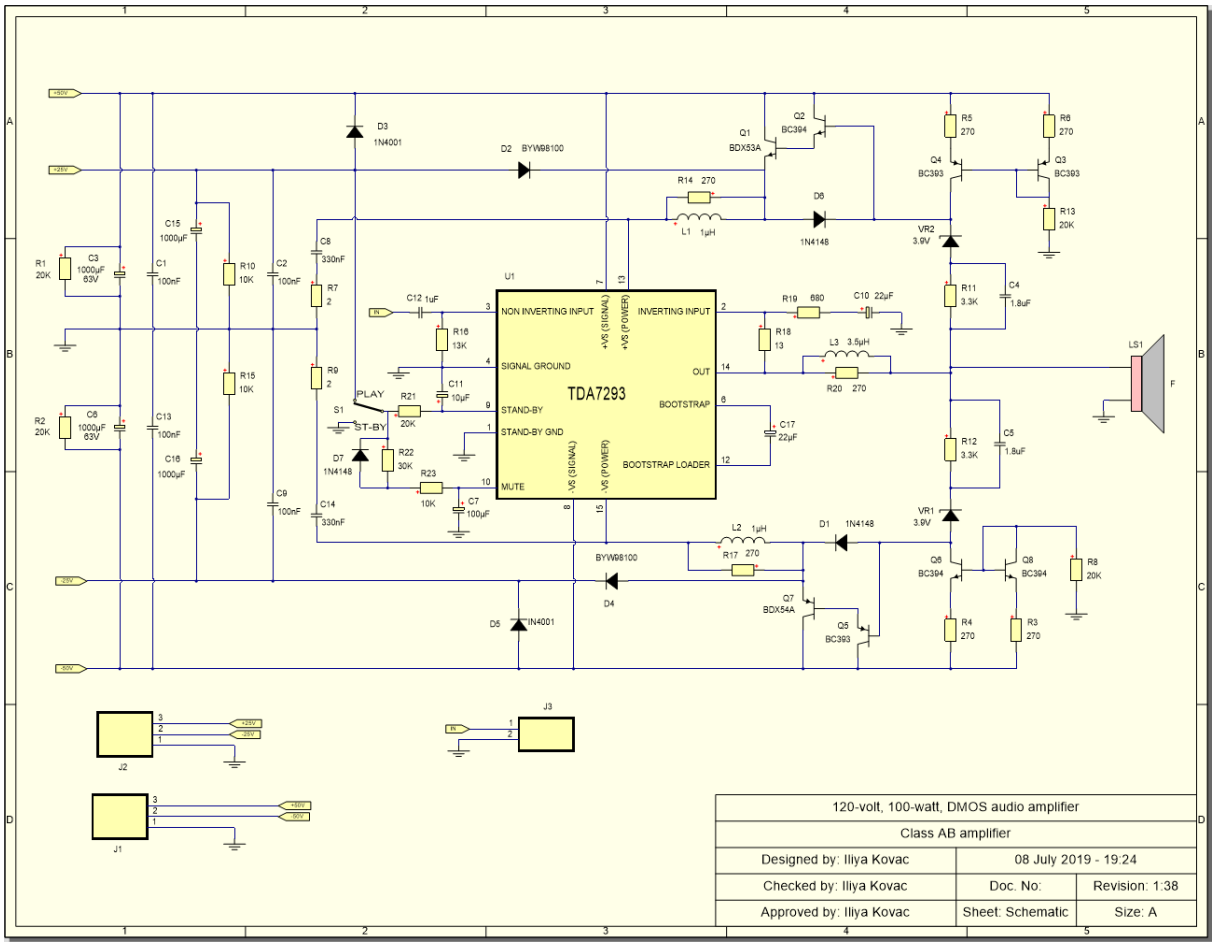
Click Help→Samples→ 

to open the Amplifier sample project.

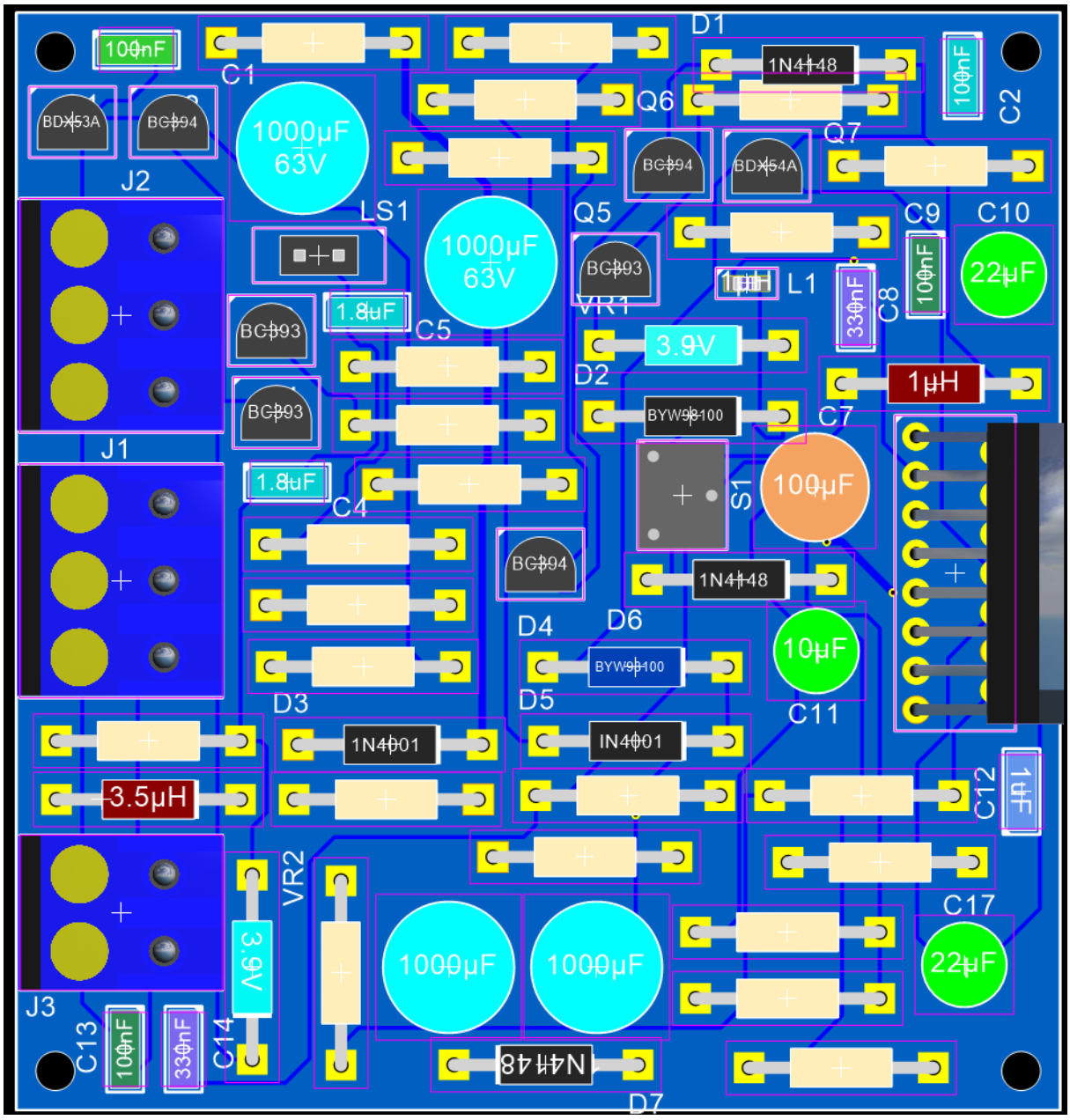


The Amplifier Sample Project

The amplifier project shows a sample design of an amplifier using all TPH components.



The Main Schematic for the Amplifier



The PCB view in 2-D with all layers on and the PCB set to draw as solid.

Electronic Amplifiers

Electronic amplifiers, or simply amplifiers, are devices that increase the power of a signal. They do this by taking energy from a power supply and controlling the output to match the input signal's shape but with a larger amplitude. In other words, an amplifier modulates the output of the power supply to make the output signal stronger than the input signal.

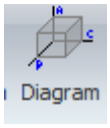
Amplifiers are used in a wide range of electronic devices. Here are some types of electronic amplifiers:

- **Audio Amplifiers:** These are used in home theater systems, radios, and musical instruments (like electric guitars) to increase sound levels.
- **Radio Frequency (RF) Amplifiers:** These amplify signals in the radio frequency range, which are used in wireless communication devices like mobile phones, and in broadcast transmitters.
- **Operational Amplifiers (Op-Amps):** These are used in a wide range of applications, including signal conditioning, filtering, or for performing mathematical operations such as addition, subtraction, integration, and differentiation.
- **Instrumentation Amplifiers:** These are used in the amplification of signals from instruments like strain gauges and thermocouples. They're designed to provide high input impedance and high gain while rejecting unwanted noise.
- **Microwave Amplifiers:** These are used to amplify signals in the microwave frequency range, typically used in radar systems and satellite communications.

The performance of an amplifier is characterized by several parameters, including gain (the ratio of the output signal power to the input signal power), bandwidth (the range of frequencies over which the amplifier provides satisfactory performance), efficiency, linearity, noise figure, and others.

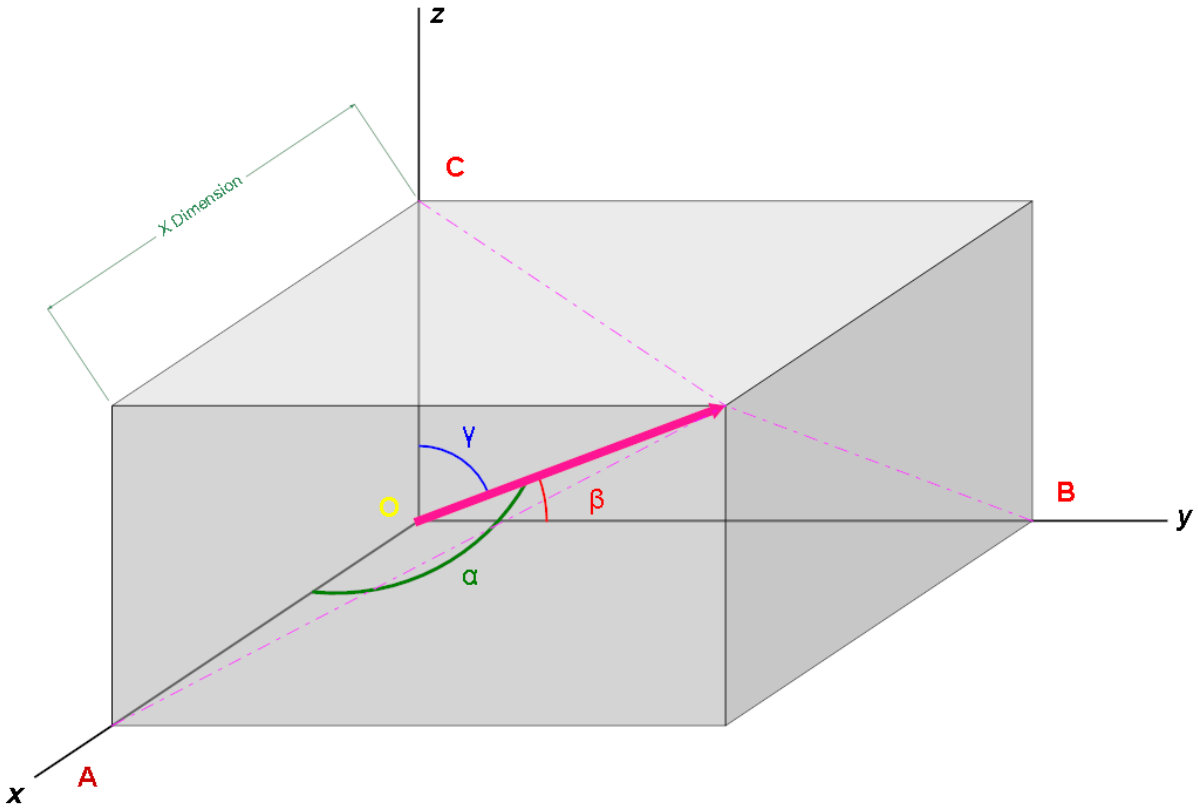
There are also many different types of amplifier circuits, each designed to maximize certain performance parameters depending on the intended application. The most common types are Class A, B, AB, and D, each providing a unique balance of power efficiency, linearity, and distortion characteristics.

1.1.8.1.4 The Diagram

Click Help→Samples→ 

to open the **Diagram** sample project.

This project is a simple diagram that shows you the graphical powers of AutoTRAX DEX.



The Diagram Sample Project

What is a Graphical Diagram

A graphical diagram is a type of visual representation that uses symbols, shapes, lines, and colors to illustrate concepts, ideas, processes, or relationships between different elements. Graphical diagrams can range from simple charts to complex maps and network diagrams, depending on the nature of the information being represented.

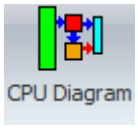
Some common types of graphical diagrams include:

- **Flowcharts:** These are used to illustrate a sequence of operations or a workflow. They often include symbols like rectangles, diamonds, and arrows to represent different stages, decisions, and the flow of information.
- **Bar charts and Histograms:** These are used to represent numerical data or the distribution of data sets.
- **Pie charts:** These are used to represent proportions of a whole.
- **Venn Diagrams:** These are used to illustrate the relationships between different groups or sets, typically showing areas of overlap between them.

- **Mind maps:** These diagrams are used to visually organize information, typically around a single concept or problem, with related ideas branching out from the central topic.
- **Organizational Charts:** These diagrams represent the structure of an organization and the relationships and relative ranks of its parts and positions/jobs.
- **Network diagrams:** These are used in information technology to illustrate the connections between different devices in a network.
- **Process diagrams:** These can include diagrams like PFD (Process Flow Diagram) and P&ID (Piping and Instrumentation Diagram) used in engineering and manufacturing to illustrate the stages of a process.

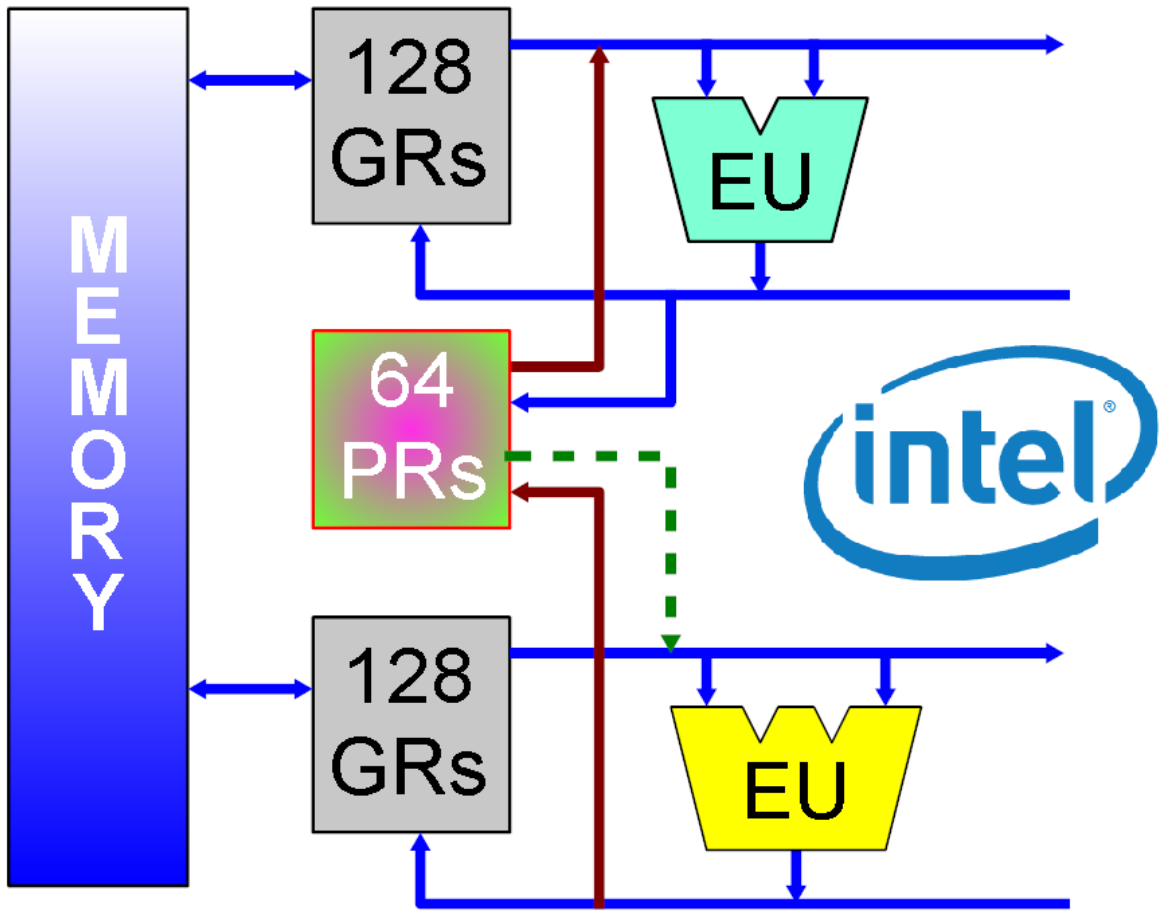
These graphical diagrams are widely used in various fields such as business, education, engineering, and science, because they can communicate complex information in a way that's easy to understand at a glance.

1.1.8.1.5 The CPU Diagram

Click Help→Samples→ 

to open the **CPU Diagram** sample project.

This project shows you how you can use the powerful graphics capabilities in AutoTRAX DEX to diagram a simple CPU using graphical blocks and connecting wires.



General Organization for IA-64

The CPU Diagram Sample Project

What is a CPU Diagram

A CPU (Central Processing Unit) diagram is a graphical representation that illustrates the internal architecture or design of a CPU. It can range from a basic diagram showing only the major components, to a detailed schematic showing the intricate design and interconnections of the CPU at the transistor level.

At a high level, most CPU diagrams will include the following key components:

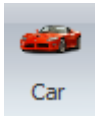
- **Control Unit:** The control unit directs all of the processor's operations, it interprets the instructions from memory and transforms them into a series of signals that the computer executes.
- **Arithmetic Logic Unit (ALU):** The ALU performs basic arithmetic and logic operations.

- **Registers:** Registers are small storage areas that hold data that the CPU is currently processing.
- **Cache Memory:** Cache is a small amount of high-speed memory located within the CPU or in close proximity to it. It is used to temporarily store frequently accessed data to speed up processing.
- **Buses:** These are the communication pathways that transport data between different components of the CPU and the rest of the computer system. Major buses in a CPU include the data bus, address bus, and control bus.
- **Clock:** The clock synchronizes all CPU operations.

In more detailed CPU diagrams, you might see individual execution units, branch predictors, memory management units, and other advanced features, depending on the architecture of the specific CPU. For example, modern CPUs often include multiple "cores," or independent processing units, each with their own control unit, ALU, registers, and sometimes, cache.

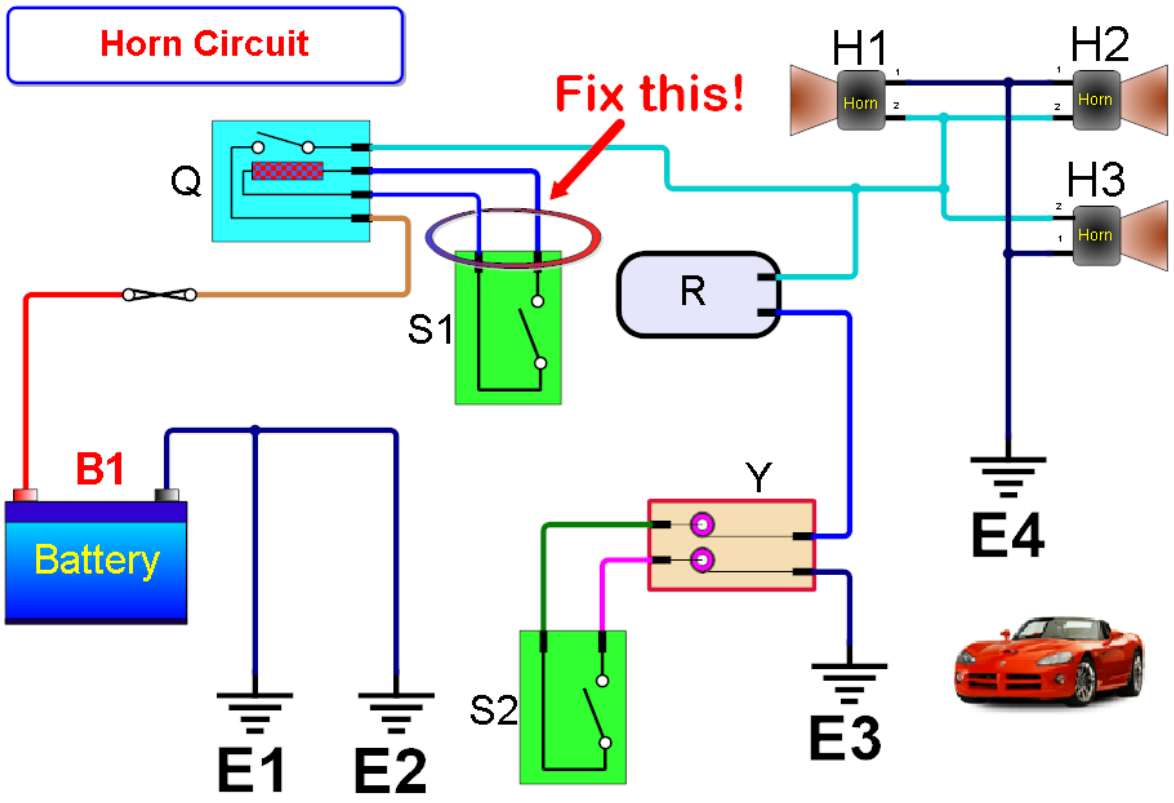
Remember, different CPU architectures (e.g., CISC vs RISC, or Intel vs ARM) have different designs, so their diagrams may highlight different features or have a unique organization. CPU diagrams can be very complex due to the intricacy of modern processors.

1.1.8.1.6 The Car Harness

Click Help→Samples→ 

to open the **Car Harness** sample project.

This project shows you part of the electrical circuit in an automobile such as you might see in a typical user manual.



The Car Electrical Harness Sample Project

What is a Car Electrical Harness

A car electrical harness, often referred to as a wiring harness, is a group of wires, cables, and connectors that relay information and electric power throughout the vehicle. Essentially, it's the vehicle's nervous system, because it transmits signals and electrical power, much like how nerves transmit signals throughout the body.

The wiring harness plays a critical role in connecting various electrical and electronic components of the vehicle, such as the headlights, engine controls, fuel injectors, air conditioning, the radio, and many other components.

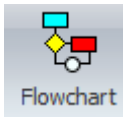
Each wire in the harness is color-coded and often labeled to indicate its specific function, making it easier for auto mechanics to repair or modify the vehicle's electrical system.

The wires are bound together into a single harness to keep the wiring system organized. This makes production easier and the final product safer and more reliable. It also makes installing the wiring systems faster and more efficient, and it helps to protect the wires from damage caused by vibration, abrasion, and moisture.

As cars have become more complex and feature-rich, so too have their wiring harnesses. Today, a modern vehicle might contain several different wiring harnesses, each designed for a specific system (e.g., engine controls, audio systems, lighting systems, etc.). These harnesses are designed to accommodate the

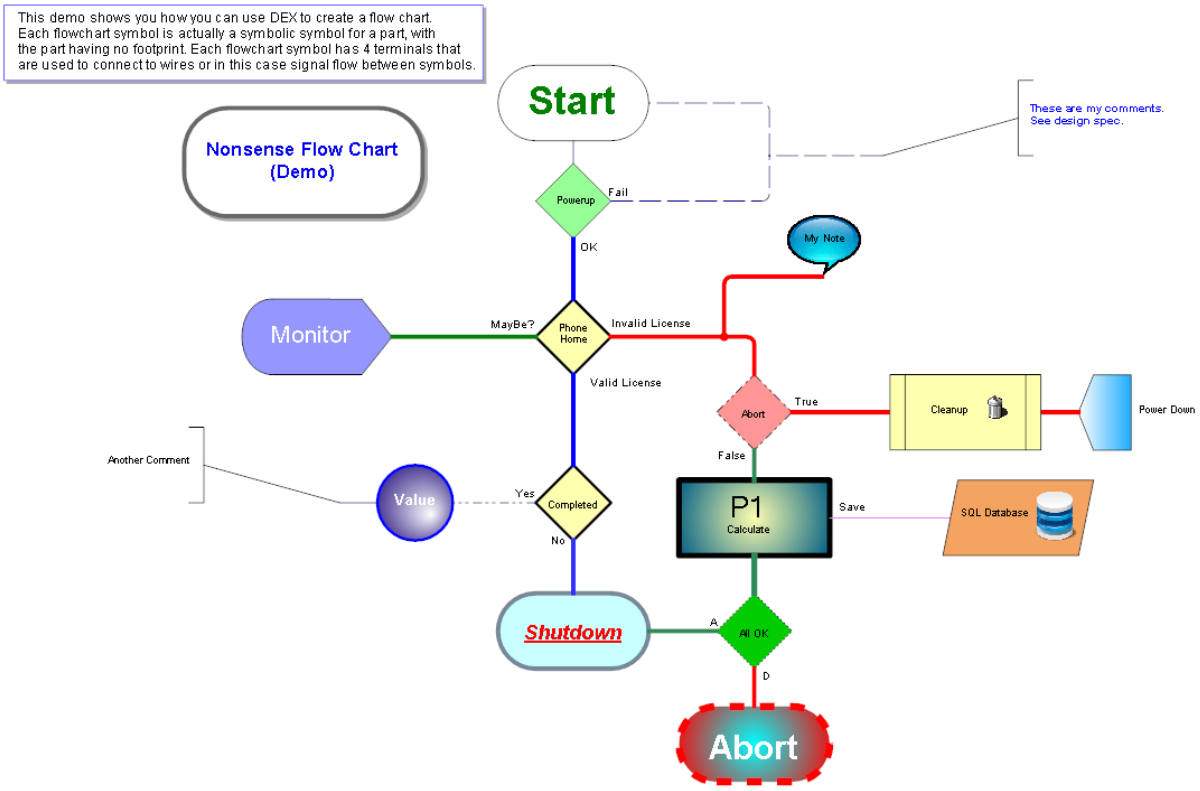
increasing electrical demands of modern vehicles, which include everything from advanced infotainment systems to autonomous driving features.

1.1.8.1.7 The Flowchart Design

Click Help→Samples→ 

to open the **Flow Chart** sample project.

With AutoTRAX DEX you can draw flowchart diagrams such as the one below. The flowchart design project shows you the example flowchart you see below.



The Flowchart Design Sample Project

Flowcharts

A flowchart is a type of diagram that represents a workflow or process. It shows the steps as boxes of various kinds, and their order by connecting them with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analyzing, designing, documenting or managing a process or program in various fields.

Here are some common elements of a flowchart:

- **Start/End Symbols:** These are typically represented as circles or rounded rectangles and signify the beginning or end of a process.

- **Process Step:** Represented as a rectangle, this shows a task or operation that needs to be done.
- **Decision Point:** This is represented as a diamond shape and indicates a point where a decision is required, typically a Yes/No question or True/False condition.
- **Flow Lines/Arrows:** These show the direction of the process flow.
- **Input/Output:** This symbol (often a parallelogram) represents information entering or leaving the process.
- **Predefined Process:** Represented by a rectangle with double-struck vertical edges, this symbol is used to represent a sequence of operations that are defined elsewhere.
- Flowcharts are useful for understanding complex processes, designing and optimizing processes, and ensuring that processes meet certain standards. They're often used in business and engineering disciplines, but can also be useful in other fields. The visual nature of flowcharts makes them easy to understand, and they're a common tool for teaching and explaining processes.

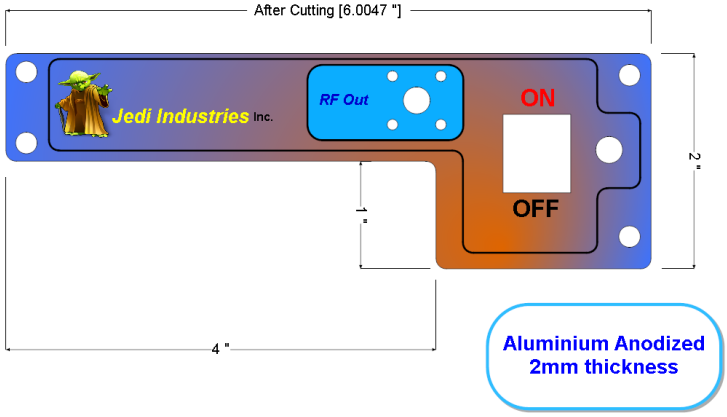
1.1.8.1.8 The Front Panel

Click Help→Samples→ 

to open the **Front Panel** sample project.

With AutoTRAX DEX you can even design front panel displays. An example front panel display is shown below And was designed using the powerful graphics available in AutoTRAX DEX.

The front panel project contains this example front panel design.



The Front Panel Sample Project

Front Panels

Electronic front panels, often referred to as control panels or interfaces, serve as the interaction point between a user and a device. They include a variety of components that are crucial for operating electronic devices and can be found in various types of equipment, including audio systems, scientific instruments, industrial machinery, home appliances, and computer servers.

These front panels can contain elements such as:

- **Display screens:** These may be simple LED/LCD screens for showing basic status indicators, or more complex touch screens which provide detailed control and interaction.
- **Buttons, switches, and knobs:** These allow the user to perform actions like turning the device on/off, adjusting settings, or triggering specific functions.
- **Ports and sockets:** For plugging in external devices or accessories. These could be USB ports, headphone jacks, HDMI ports, or others, depending on the device.
- **Indicator lights:** These provide visual feedback about the status of the device or certain functions, such as power on/off, activity status, warning/error indicators, etc.

The design, layout, and functionality of an electronic front panel can significantly affect the user experience. Thus, considerations like user-friendliness, intuitiveness, aesthetic appeal, and ergonomics are often important in their design.

Further, depending on the application, these panels can be designed to withstand various environmental factors like water, dust, temperature, and mechanical stress. This is especially relevant in industrial and outdoor applications.

Modern trends in electronic front panel design often involve incorporating smart interfaces, such as touch screens, voice control, and even gesture recognition, to enhance user interaction and convenience. Moreover, with the rise of IoT (Internet of Things), these panels can also be designed to connect and communicate with other devices and networks, allowing for remote control and monitoring, among other functionalities.

1.1.8.2 Web Tutorial

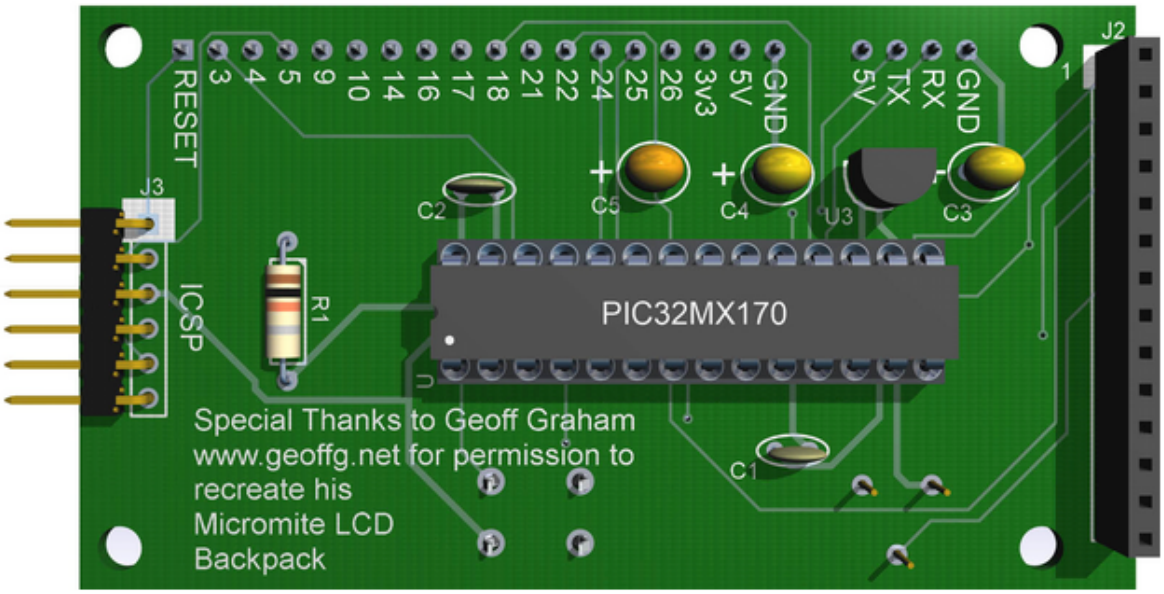
Mick's Tutorial

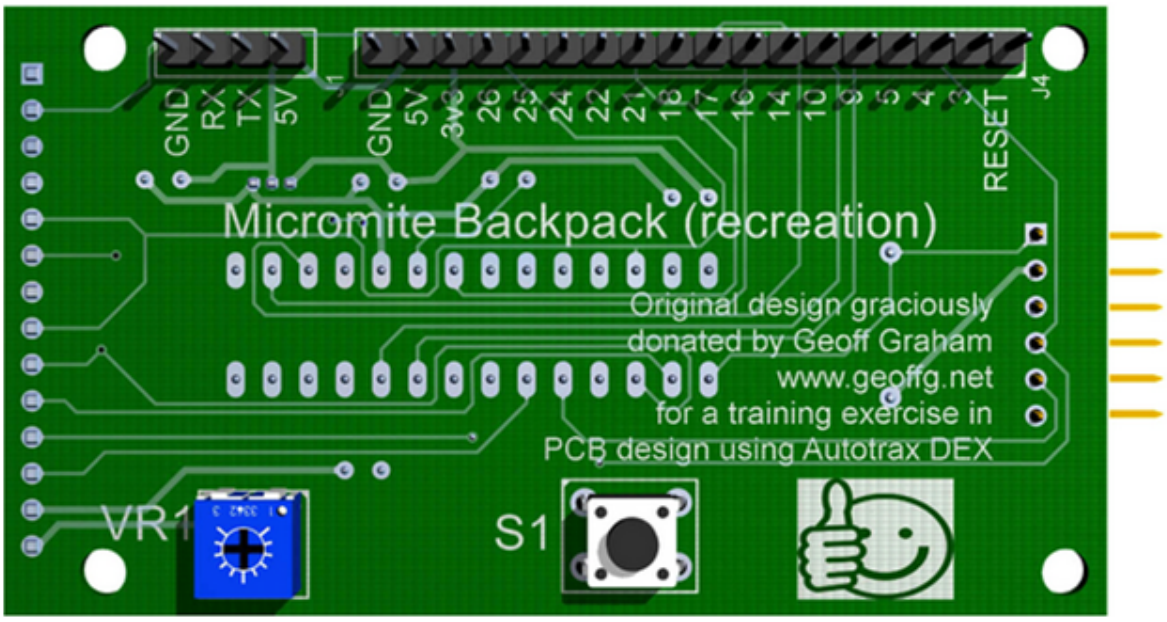
Mick Gulovsen from Melbourne, Australia has created a tutorial on using AutoTRAX DEX at https://pcbdex.com/HowToMakeAPCB.

'I intend to show the `basics' of how to go about entering all of the required data to lay out a PCB, from start to finish, including creating the GERBER files to send to a PCB fabrication house for production. This document will no doubt be full of grammatical and spelling errors and may show methods that are not necessarily the easiest, or best, way to do things. It is the way that I go about using AutoTRAX DEX, you may

do things differently and that is fine. We are all individuals and we all do things in our own ways. I also make no guarantee about the accuracy of this board and in fact it may have serious errors that will make it useless. What is important is the methodology that I use and if you chose to make any PCBs using this design you do so at your own risk. Of course you can always edit the project to fix said errors, if they rear their ugly heads. One last thing is this will NOT be an exact TRACK for TRACK reproduction. You can route manually and exactly replicate the original but that would take a lot more effort and is a moot point as generally you will be designing from scratch'

Mick Gulovsen





1.1.9 Updating Your Version

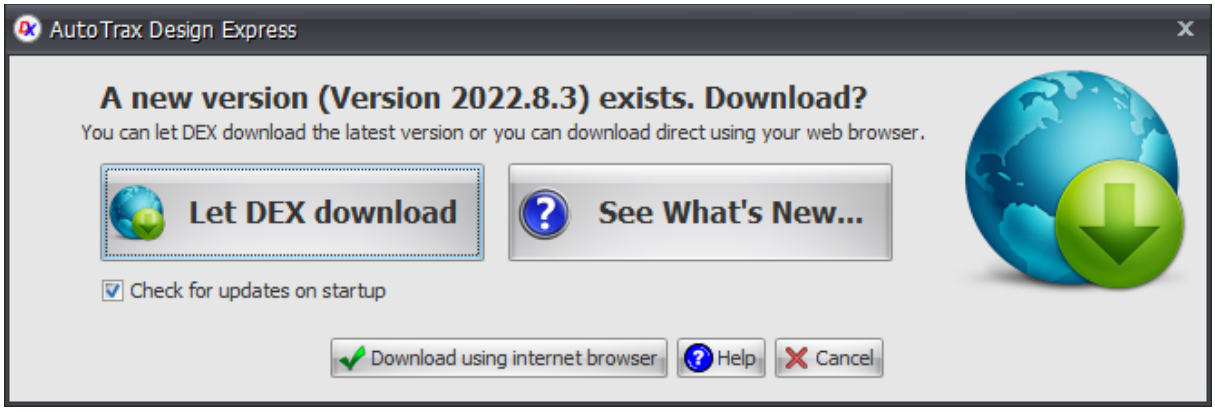
Manual Checking

To manually check for updates click the Home→Account→  button.

Automatically checking for new versions

When AutoTRAX DEX starts it optionally checks to see if there is a new version of AutoTRAX DEX available.

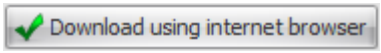
If there is a new version it will display the dialog box below.



Click 

to view the What's new website.

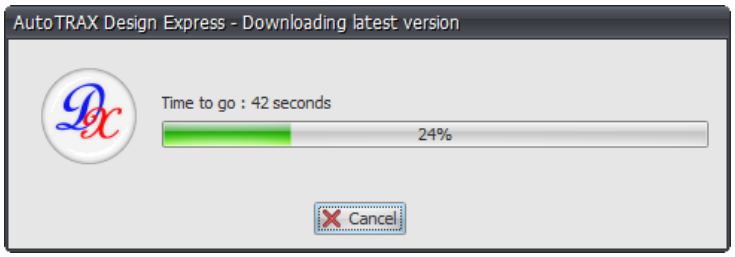
Download

Click 

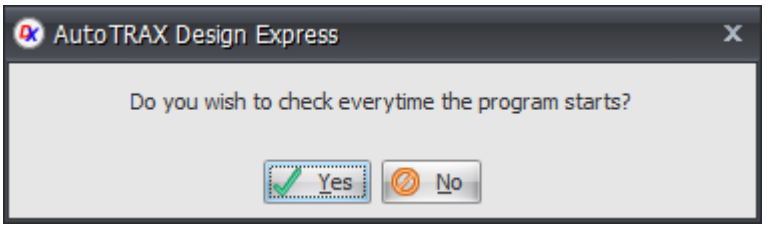
to download direct from the AutoTRAX DEX website or

Click the 

download button to have AutoTRAX DEX download the latest version and install it. This is the easiest way to update and you will not be prompted for install passwords and setup details. When AutoTRAX DEX is downloading the latest version it will display the download progress dialog shown below.



To disable auto-updating click the Cancel button and the following dialog will be displayed. Click on the No button to prevent checks for updates in future.



To re-enable automatically checking for updates, click on the Get Latest Version button 

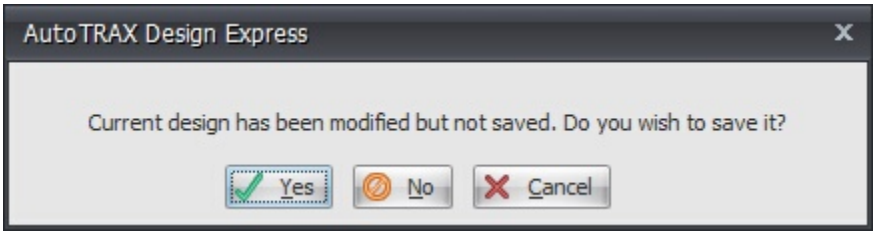
in the Home→Account menu

1.1.10 Closing DEX

To close AutoTRAX DEX click on the Exit button 

in the Home menu.

If you have any unsaved changes, you will be prompted to save your work.



1.1.11 Electronic circuit simulation with SPICE

Electronic circuit simulation with SPICE (Simulation Program with Integrated Circuit Emphasis) is a computer program used to model and analyze the behavior of an electronic circuit. It can be used to predict the performance of an actual component or system under various conditions. The program includes models for both linear circuits, such as resistors, capacitors and inductors, as well as non-linear components such as transistors and diodes. It is widely used in the electronics industry for everything from prototyping circuits to troubleshooting existing systems.

SPICE simulations are made up of two parts: a netlist description of the circuit components and their connections, plus a set of parameters that define how each component behaves within the circuit. The netlist defines which components make up the circuit, where they are connected, and what type of component it is. For example, it might describe a resistor between Node 1 and Node 2, or a capacitor between Node 3 and ground. Each component type has its own set of parameters that describe its behavior in detail: capacitance value for capacitors; resistance value for resistors; voltage drop across a diode; etc.

Once the netlist and parameter sets have been defined, SPICE uses numerical methods to calculate voltages, currents and other electrical properties throughout the circuit over time using Newton's method or other iterative techniques. This allows it to accurately simulate different waveforms by varying either parameters or voltages at specific times during the simulation process. This can be useful in predicting transient behaviors in mixed-signal systems or power supply designs where multiple signals interact over time.

SPICE simulations can also provide information about power consumption over time as well as thermal effects due to current flow through various parts of a system; these properties are important when designing products operating at high frequencies or under high power loads. Additionally, SPICE offers features like Monte Carlo analysis which allow engineers to evaluate how variations in device characteristics affect circuit operation across large numbers of randomly generated

data points—this is especially useful when designing robust systems capable of handling real-world variations in components' values.

Overall, SPICE is an extremely powerful tool for modeling complex electronic circuits before they are built—it allows engineers to identify potential problems before they occur while also helping them optimize their designs for maximum performance within their desired constraints. Its flexibility makes it applicable to virtually any application involving analog/digital components while its ability to accurately predict real world behaviors make it indispensable in modern electronics design processes.

1.2 Designs

AutoTRAX DEX handles 3 different types of files.

- **Projects**
- **Parts**
- **Artwork**

Projects contain schematics, text documents and a PCB and represent the design for your PCB.

Part files contain schematic symbols, text documents and a footprint (land pattern) that contains all the information needed to define the abstract schematic representation for a part together with the footprint required for the PCB and the 3D solid model for the part.

Parts are defined parametrically; this allow a single part to be quickly and easily modified into a different part type. The parametric models define how to create a complete part from only a few simple parameters.

Artwork files contain schematic that do not have any electrical content, such as parts and wires. They are used for creating library artwork to be used in project and part files. They can also be used for any other drawing purposes, as AutoTRAX DEX's powerful graphics allows you to graphically design engineering drawings.

1.2.1 Line Styles, Fill Styles, and Fonts

Line Styles

In computer graphics and design, line styles are used to describe the appearance of lines that compose shapes, paths, or outlines. They add detail, differentiation, or emphasis to different elements of a design or visualization. Here are some of the most common types of line styles:

- **Solid Lines:** The most basic type of line is a solid, unbroken line.
- **Dashed Lines:** These are composed of uniform dashes with spaces in between. The length of the dashes and the gap between them can often be customized.

- **Dotted Lines:** These are composed of uniform dots with spaces in between.
- **Dash-Dot Lines:** These are composed of a repeating pattern of a dash followed by a dot.
- **Double Lines:** These consist of two parallel lines. Double lines are often used in technical or architectural drawings.
- **Wavy Lines:** These lines have a wavy or curved pattern. They are often used to indicate a boundary that is not fixed.

Apart from these basic styles, lines in graphics can be customized in various ways, such as:

- **Width:** The thickness of the line can be adjusted, with thicker lines often used to emphasize certain elements.
- **Color:** The color of the line can be changed to differentiate different elements or match a specific design aesthetic.
- **Transparency:** The line can be made fully or partially transparent, which can create interesting visual effects when lines overlap.
- **Cap Style:** This refers to the appearance of the end of the line, which can be flat, round, or square.
- **Join Style:** This refers to the appearance of the junction between two lines, which can be mitered (pointed), round, or beveled (flat).

Line styles, along with fill styles and other graphical properties, are fundamental aspects of graphical aesthetics, and they can greatly influence the appearance and effectiveness of a design or visualization.

Fill Styles

In computer graphics, fill styles refer to the patterns or colors used to fill shapes, paths, or areas. Fill styles can range from solid colors to gradients and patterned fills. They are a vital part of graphical styling and are used in various applications such as graphic design, web design, and data visualization. Here are some common types of fill styles:

- **Solid Fill:** This is the simplest type of fill, which uses a single, solid color to fill a shape or area.
- **Gradient Fill:** This fill style gradually blends two or more colors together across a shape or area. There are different types of gradients, including linear gradients (colors transition along a line), radial gradients (colors transition outwards in a circular pattern), and angular gradients (colors transition around a central point).
- **Pattern Fill:** This fill style uses a repeating pattern to fill a shape or area. The pattern can be made up of lines, dots, icons, textures, or other small elements.

- **Texture Fill:** Similar to pattern fills, texture fills use a specific image or texture, such as wood grain, fabric, or a photographic image, to fill a shape or area. This can give the filled object a more realistic appearance.
- **Hatch Fill:** This is a type of pattern fill that uses parallel lines to fill a shape. The lines can be horizontal, vertical, diagonal, or a combination. Hatch fills are often used in technical drawings or to denote different regions in graphs.
- **Transparent or Semi-transparent Fill:** This fill style allows for the background or underlying content to be partially visible through the fill.

In many graphics applications, you can customize fill styles with different colors, opacities, and blending modes. By manipulating these aspects, you can create a wide variety of visual effects and styles.

Fonts

Graphical fonts, often referred to simply as fonts, are collections of text characters with a specific style, size, weight, and design. These character sets include letters, numbers, punctuation marks, and sometimes symbols. They are a key element of graphic design, used in everything from website design to print media, and are a crucial part of establishing a visual identity or style.

Fonts fall into several broad categories:

- **Serif Fonts:** These have small decorative lines (serifs) attached at the end of the strokes of the letters. Examples include Times New Roman and Garamond. They're often used in print media and considered to be more traditional and formal.
- **Sans-Serif Fonts:** These don't have decorative lines at the ends of the strokes. Examples include Arial and Helvetica. They have a more modern, clean look and are commonly used in digital media.
- **Monospace Fonts:** Each character in these fonts is the same width. Examples include Courier and Consolas. These fonts are often used in coding or for typewriter-like aesthetics.
- **Display Fonts:** These are decorative fonts that are used for large headings or logos, rather than body text. They come in a wide variety of designs.
- **Script Fonts:** These mimic handwriting and can be divided further into formal types that resemble calligraphy and cursive writing, and casual types that resemble everyday handwriting.
- **Symbol or Dingbat Fonts:** These consist of symbols or decorative elements instead of standard characters. An example would be the Webdings or Wingdings fonts.

The selection of a font can greatly influence the perception of a piece of text. It can evoke specific moods, convey professionalism or creativity, and contribute to the

readability and legibility of the text. A key principle in graphic design is to use fonts appropriately and consistently, considering the context, audience, and purpose of the design.

Default Graphics Settings

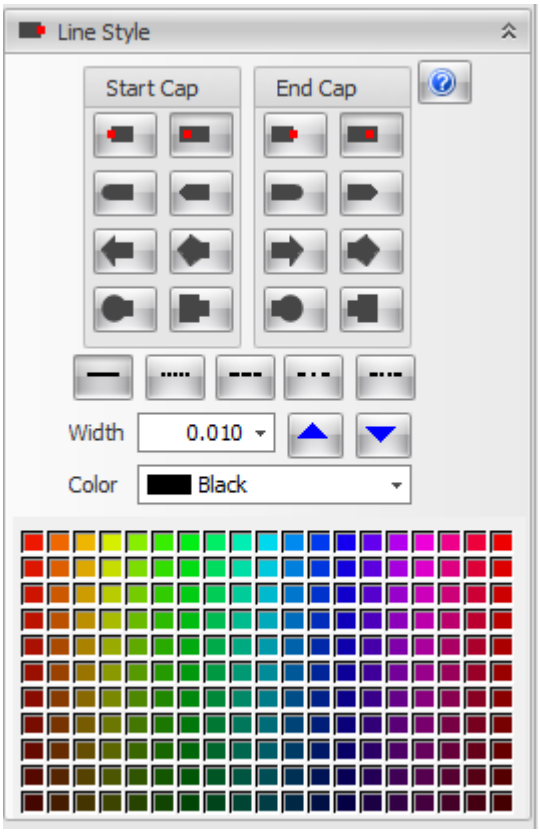
1.2.1.1 Line Styles

The video below demonstrates editing the line style for a line ... *(you will need to be connected to the Internet to view it)*

Many different objects in AutoTRAX DEX have line style, these include:

- Lines and polylines.
- Polygons, rectangles, curves, ellipses and circles.

Pictures



The Line style properties dialog is shown on the left.

Start and End Cap

These control the capping at the start and the end of lines. Only objects with open ends can have end caps, e.g.,lines, polylines and open curves.

Dash Styles

The dash style buttons below the cap styles sets the line style as solid or a variety of dash styles.

Width

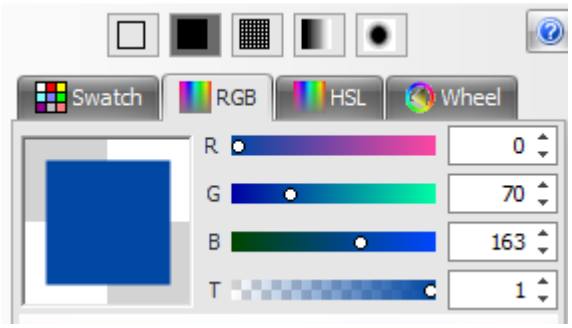
This sets the width of the line.

Color

This sets the color of the line.

1.2.1.2 Fill Styles

If you select an object or several objects that can be filled then the fill style properties of the object will be displayed in the properties panel.
At the top of the fill styles properties panel you will see a collection of buttons as shown below which allow you to select the fill style.
You can apply 4 different fill styles.

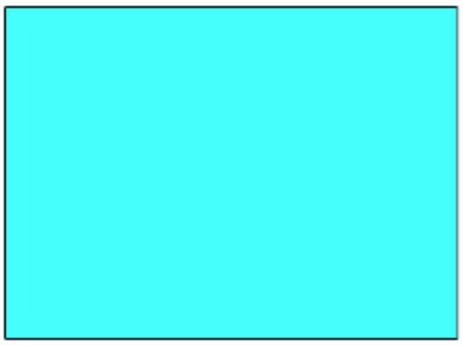


The top part of the full style editor

 **No Fill**

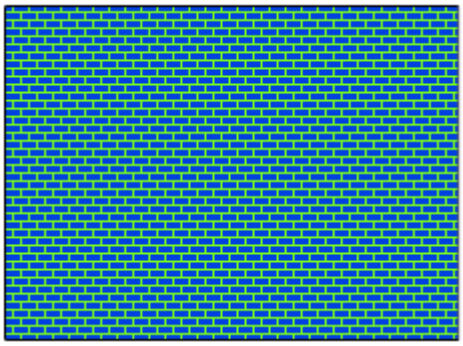
Transparent or no fill

Solid Fill



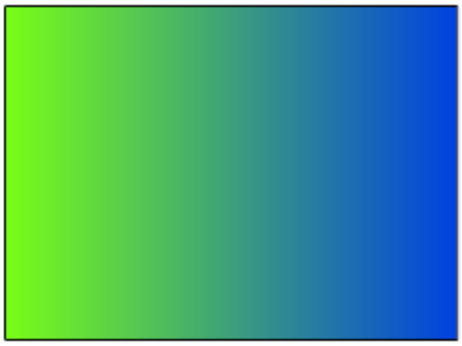
Solid Fill

Hatch Fill



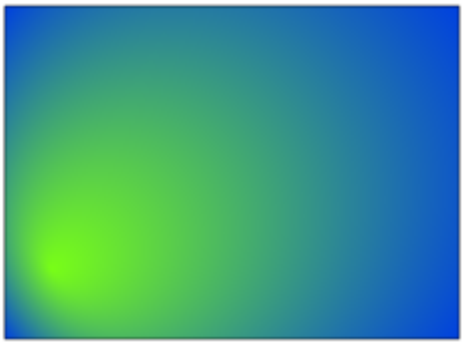
Hatch Fill

Linear Fill



Linear Fill

Radial Fill



Radial Fill

1.2.1.2.1 No Fill

You can set the fill style to be transparent or no fill. When object as no fill you will be able to see through the area of the object that is normally filled.
A transparent fill is a type of fill style where an area or shape is filled with transparency rather than color. This means that whatever is behind the shape or area will be visible through it.

This feature is often used in graphic design and digital artwork to layer objects and create depth, or to allow a background image or color to show through a shape or text. For instance, if you were creating a logo and wanted the background (like a photo or a webpage) to show through certain parts of the logo, you could use a transparent fill for those parts.

Transparency in a fill can also be partial, often referred to as "opacity" or "alpha". A partially transparent fill will blend the color of the fill with whatever is behind it. For instance, a shape with a red fill at 50% opacity placed over a blue background would show as a blend of red and blue. The degree of transparency can usually be

adjusted using an "opacity" or "alpha" slider or input in most graphic design software.

Transparent fills can create complex visual effects and are a crucial tool in creating layered, dynamic compositions in graphic design.

Note: if you set the fill style to be no fill or transparent make sure the object has a line or body style that is not transparent otherwise you will not be able to see the object. However, it will be selectable using the window select tool, you be will not be able to select it by clicking on it.

Click on the transparent button  to set the fill style to transparent.

Transparent or no fill

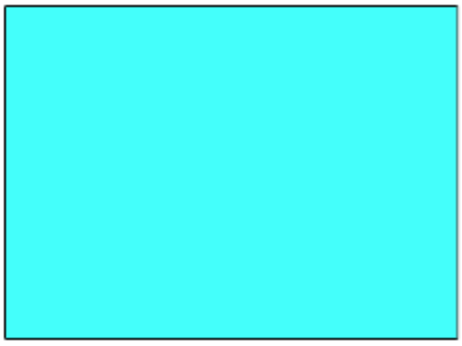
1.2.1.2.2 Solid Fill

Solid fill is the most basic type of fill in graphic design and computer graphics. As its name suggests, it refers to filling a shape, area, or object with a solid, unbroken, and uniform color.Solid fills are used widely in digital design, from creating backgrounds for websites or presentations to coloring in shapes in a digital illustration. They can be used to draw attention to specific elements, provide contrast, or to add color to a design.

In AutoTRAX, you can choose the color of the solid fill from a color wheel, a palette of swatches, or by entering specific color values such as RGB (Red, Green, Blue), CMYK (Cyan, Magenta, Yellow, Black), or HEX codes.

While solid fill is simple, it is also highly versatile. By choosing different colors and applying them to different shapes or elements, you can create a wide variety of designs. It is also often used in combination with other fill types or effects (like gradients, patterns, textures, or strokes) to create more complex graphics.

For example, the rectangle shown below is being filled with a solid fill of cyan.



A rectangle with a solid fill of cyan

To fill an object first select it.

Now you have two options to set the color:

the first option and the quickest is to click on the desired color in the color bar at the base of the viewport.

The second method is to set the color using the fill properties in the selected objects properties panel.

What you see is a standard AutoTRAX DEX color chooser.

The solid fill control is shown below.

Solid Fill Control

1.2.1.2.3 Hatch Fill

A hatch fill, commonly known simply as hatching, is a type of fill pattern made up of parallel lines. It's often used in graphic design, computer graphics, and technical drawings to add texture, depth, or to differentiate between different areas.

The characteristics of the lines in a hatch fill, such as their direction, density, and thickness, can be varied to achieve different effects. These variations result in different types of hatching, including:

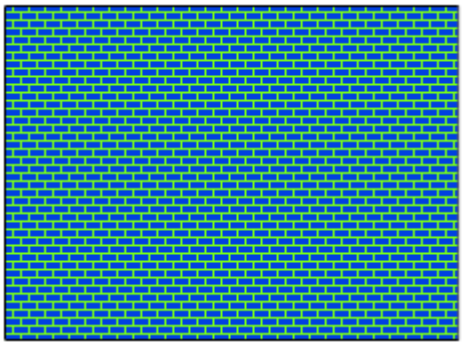
- **Parallel Hatching:** This consists of parallel lines that can be horizontal, vertical, or diagonal. This is the simplest form of hatching.

- **Cross Hatching:** This is created by layering sets of parallel lines at different angles. The lines can be layered two or more times, and the intersections of these layers create a darker area, which can be used to indicate shadows or darker tones.
- **Contour Hatching:** This type of hatching follows the contour of the shape being filled. This can be used to emphasize the shape's three-dimensionality.

Hatch fills can also be combined with colors or other fill types for more complex effects. For instance, a shape might be filled with a color and then overlaid with a hatch fill to create a textured color fill.

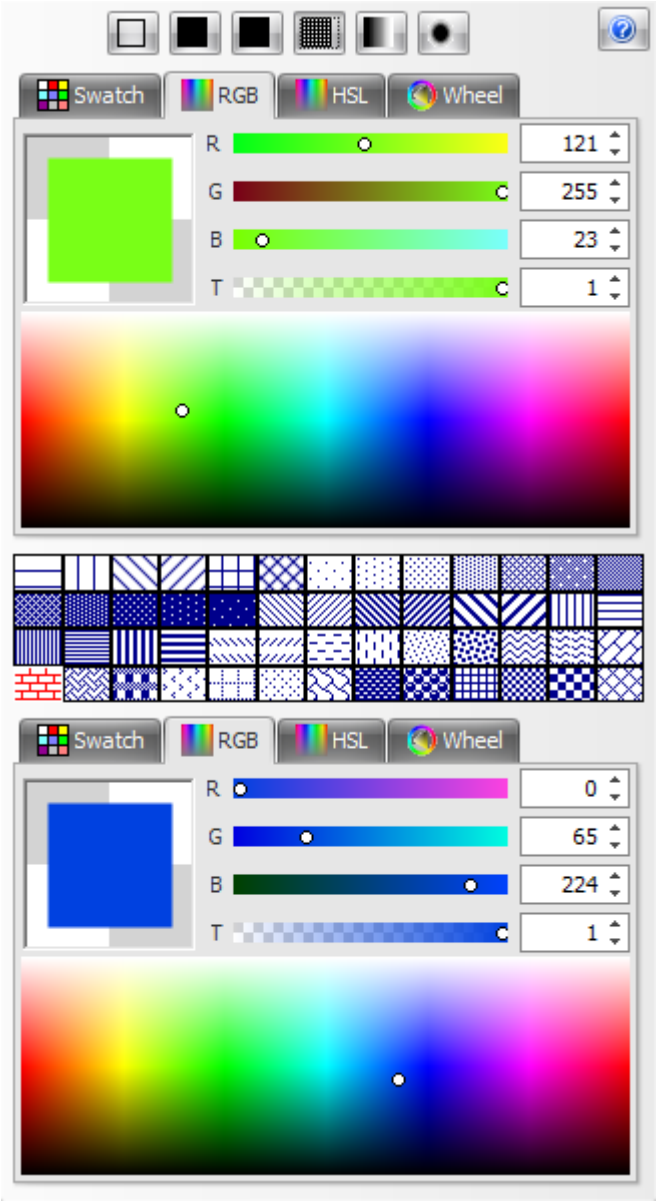
Hatching is a flexible and powerful tool in computer graphics, enabling you to create a variety of textures and shading effects while maintaining a clear distinction between different areas or elements of a design.

In AutoTRAX a hatch fill is where a shape is filled with a pattern. The pattern uses two different colors.



Hatch Fill

The hatch fill control is shown below. It consists of two separate color choosers are shown below. Use either of the color choosers to select the desired color.



Hatch Fill Control

1.2.1.2.4 Radial Fill

A radial fill is a type of gradient fill where colors or shades transition radially, originating from a central point and gradually changing towards the outer edge of the shape. This fill style is often used to create effects of light and shadow, depth, or a three-dimensional look in graphic design.

There are two main types of radial fills:

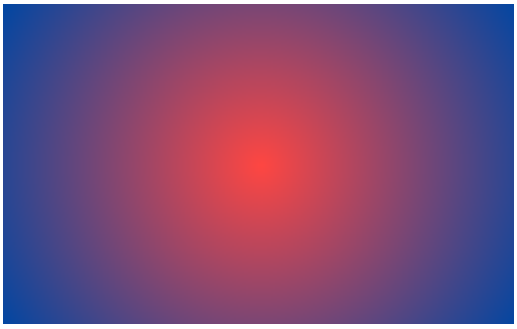
- **Symmetrical Radial Fill:** The color transition occurs symmetrically from a central point, creating a circular or elliptic pattern. It's akin to a drop of ink spreading evenly on a piece of paper.

- *Asymmetrical Radial Fill:* The color transition is skewed or elongated, creating an elliptical pattern that doesn't necessarily originate from the geometric center of the shape. This can be used to create more complex light and shadow effects.

In many graphics software, radial fill options allow you to adjust the position of the center point, the rate and direction of color transition, and the colors used. Multiple colors can be used in a single radial fill, transitioning from one to another in the order and rate specified by the designer.

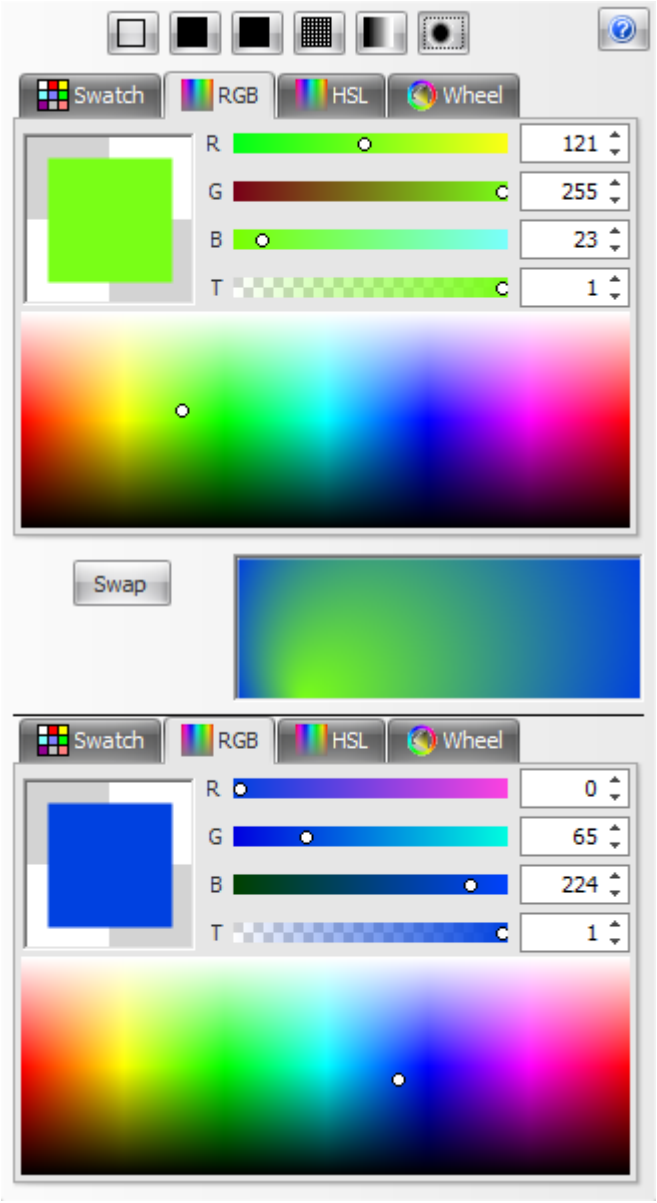
Radial fills are commonly used in various fields like graphic design, digital illustration, and data visualization. They can add depth and visual interest to shapes, backgrounds, and graphical elements.

In AutoTRAX a Radial fill is where you fill an object with a color pattern that is circular with the central color merging into an outer color as shown below.



Radial fill

The radial fill control is shown below.



Radial Fill Control

It consists of two separate color users. The top color chooser sets the central color while the bottom color chooser sets the outer color. You can set the position of the central color by dragging on the central color position control.

The central color position selector

Click the Swap button to exchange the two colors.

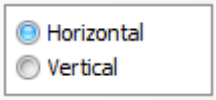
1.2.1.2.5 Linear Fill

Linear fill is where the color changes from one color to the second along a straight line as shown below.

Linear Fill

Click the Swap button to exchange the two colors.

The colors have been swapped

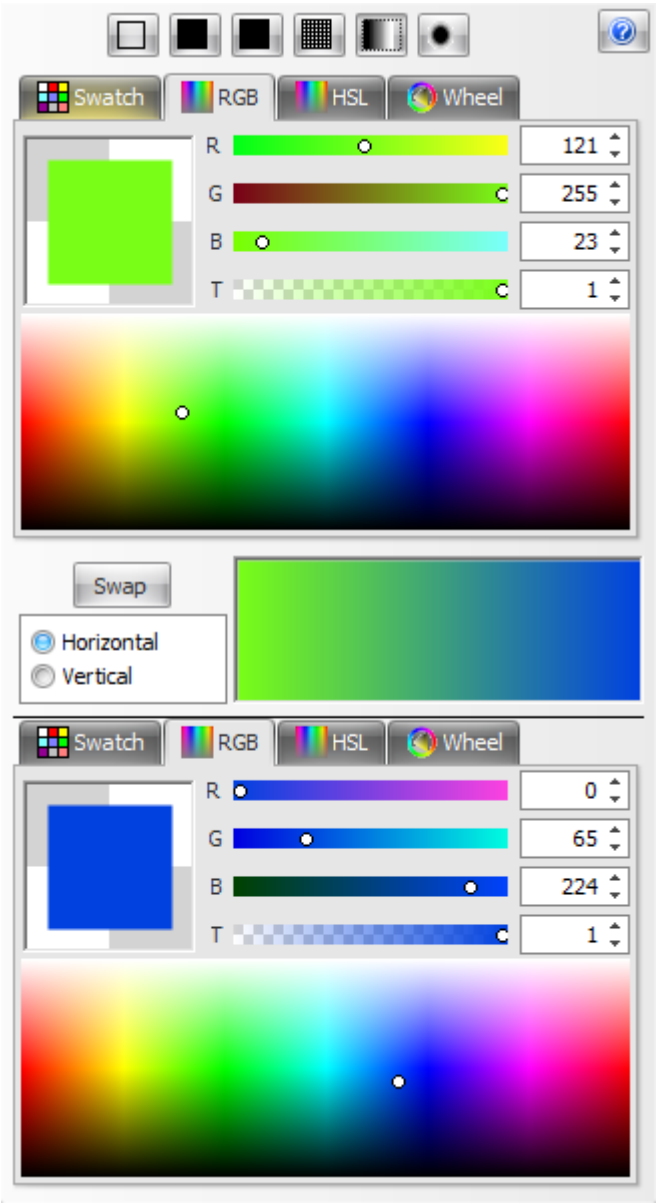
Check either the horizontal or vertical radio button 

to set the direction of color change.

Color direction changed to vertical

The linear fill control is shown below. The top color chooser sets the first color while the bottom, chooser sets the second color.

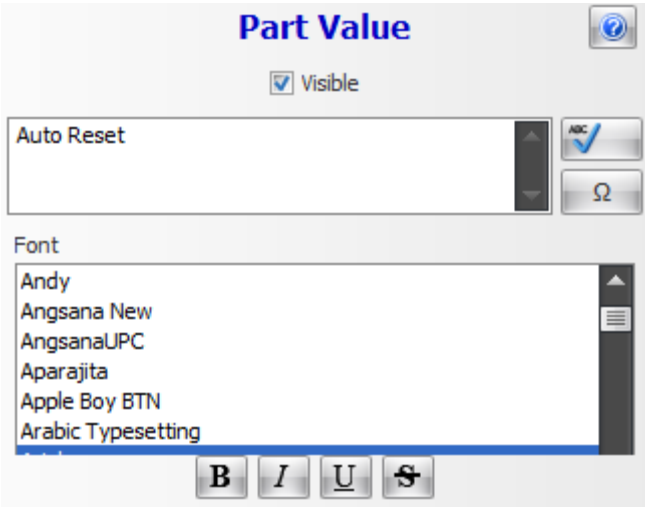
Linear filled preview



Linear Fill Control

1.2.1.3 Fonts

The Font Editor dialog is shown below.



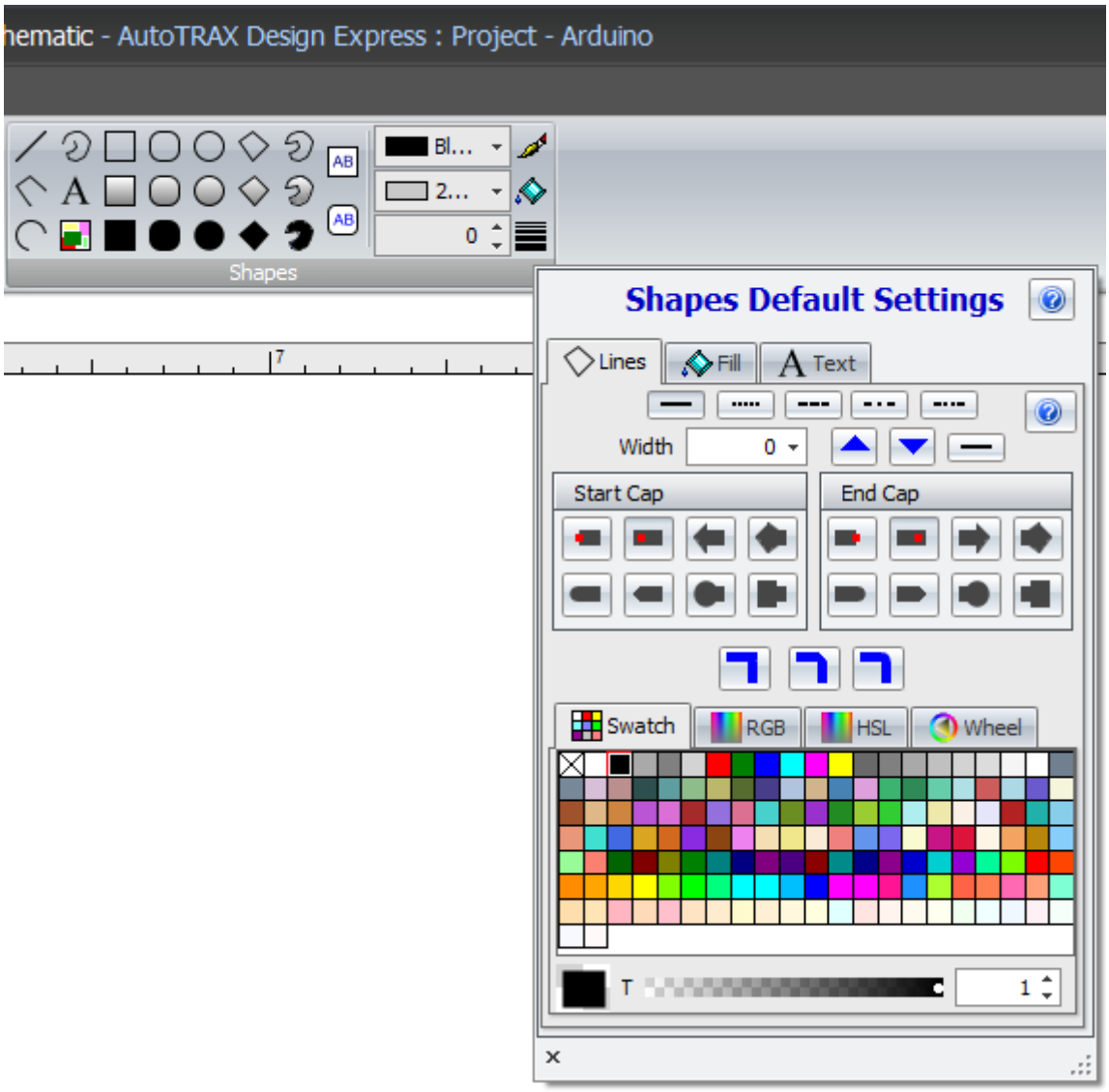
Font Editor dialog

1.2.1.4 Default Graphics Settings

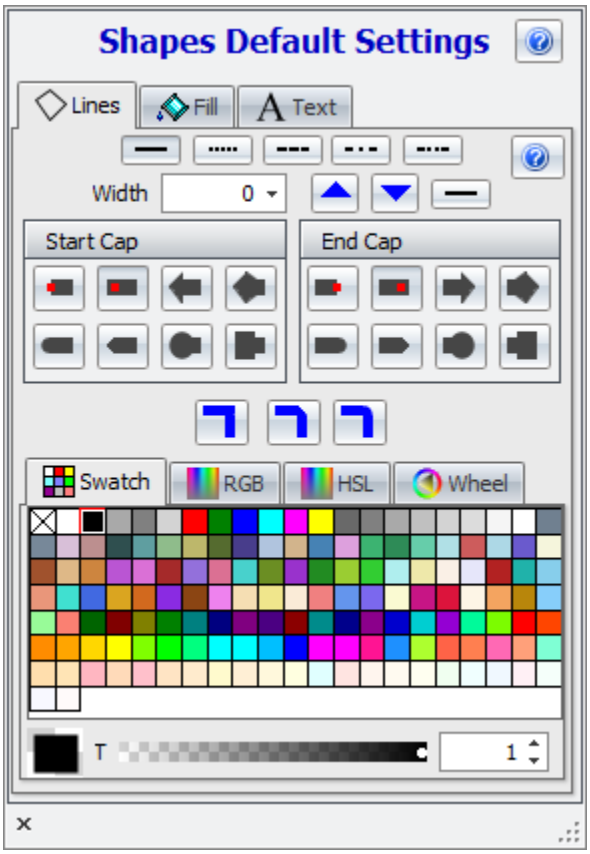
To set the default line/fill and text settings for graphics, click on the small button at the bottom right of the ***Add→Shapes*** ribbon button group.

The default settings pop-up consist of three separate tabs.

- The first tab sets the line of border style.
- The second tab sets the fill style.
- The third tab sets the font properties.

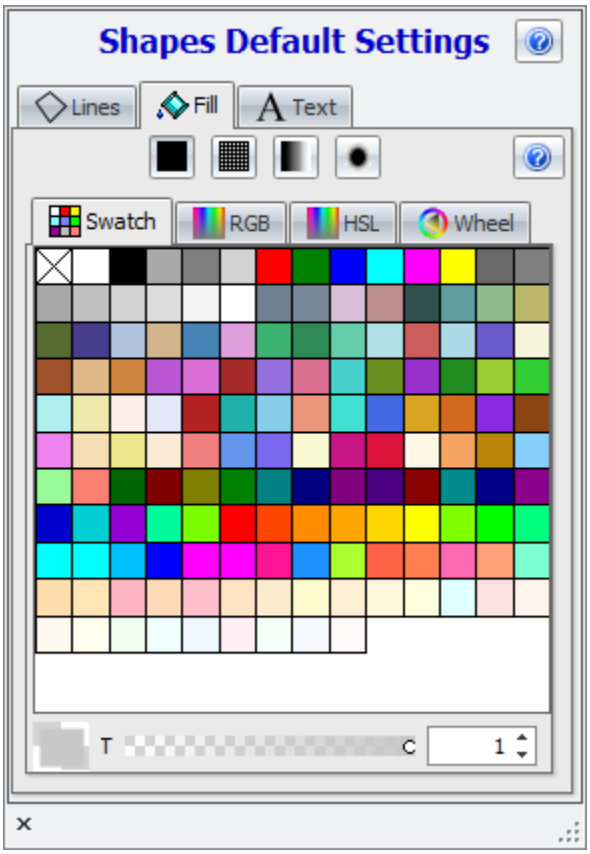


Shapes Default Setting pop up dialog



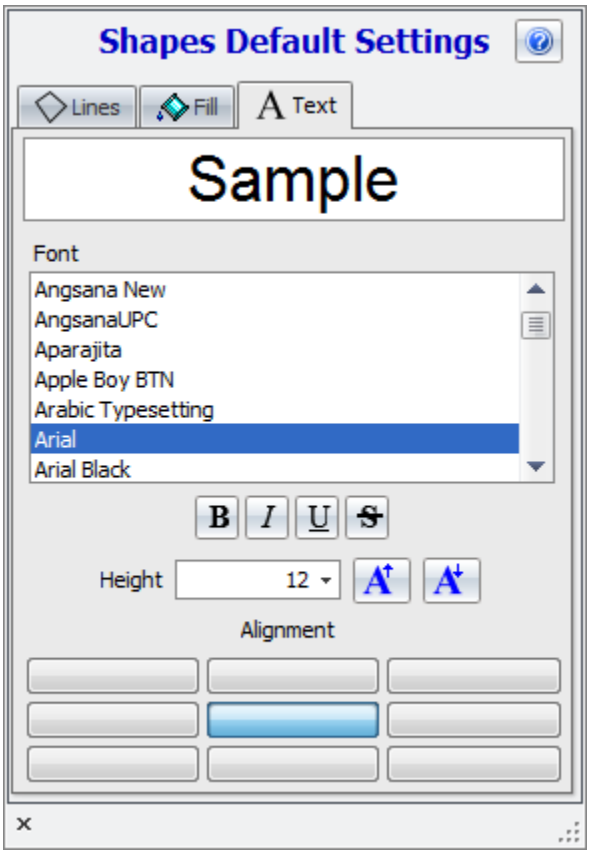
Lines properties

See Line Styles for more info.



Fill properties

See Fill Styles for more details



Font properties

See Fonts for more details

1.2.2 Drawing Aids

AutoTRAX DEX has several drawing aids to help add, edit and align parts, pads, footprints and graphics.

Arranging and Editing Objects

Coordinates

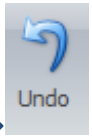
Guides

Snaps and the Grid

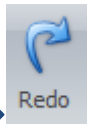
Measuring Distances and Angles

1.2.2.1 Undoing and Redoing Your Changes

Every change (except viewpoint changes) you to make to your design or part is reversible.

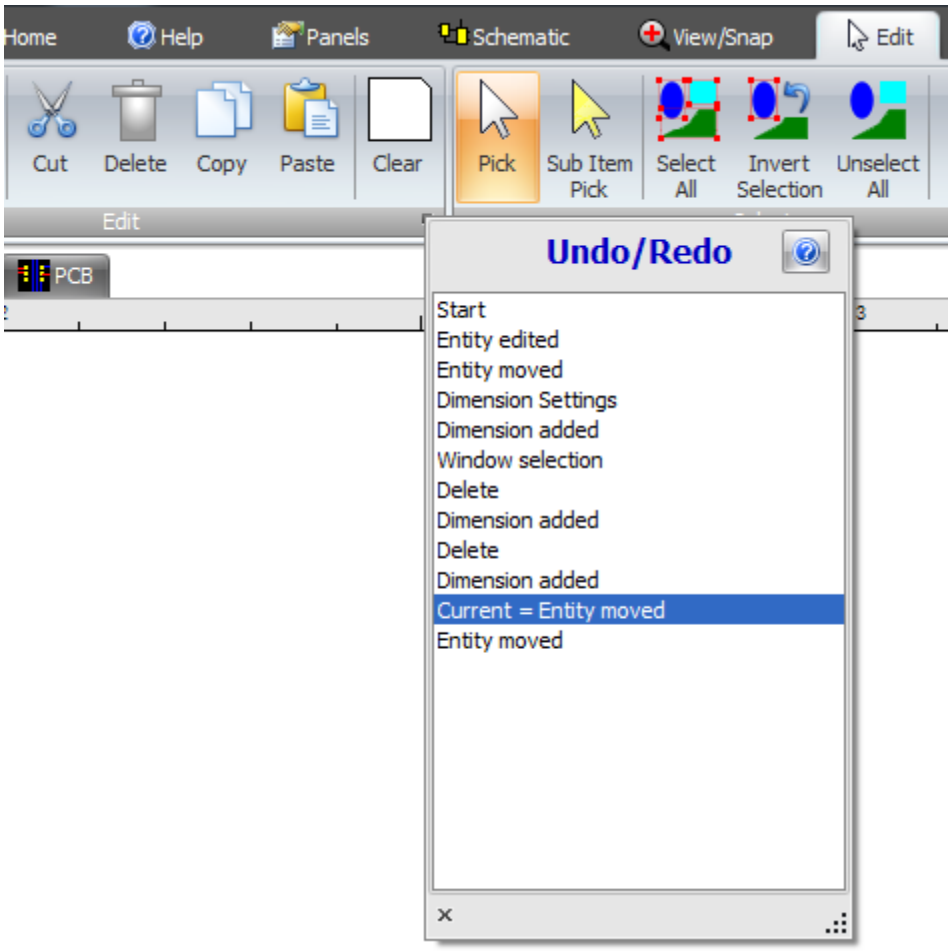
To undo a change click the **Edit→Undo/Redo→** 

button.

To reverse an undo and restore a change, click the **Edit→Undo/Redo→** 

button.

Each change you make to a design or part is named with a name that reflects the action taken. You can view all changes made in the current session by clicking on the  button the lower left of the **Edit→Edit** ribbon button group. The Undo/Redo pop-up will be displayed as shown below. Click on any of the named states to revert the current design to that state. Clicking the  button displays this help topic.



You can also restore previous states that occurred before the current session using the Restore Backup command.

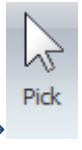
1.2.2.2 Arranging and Editing Objects

In AutoTRAX DEX you can edit and arrange all the objects in a sheet.

1.2.2.2.1 Selecting Items

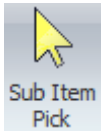
There are 2 picking modes in AutoTRAX DEX

Pick

In this mode only top level objects are picked. Click the Edit→Select→ 

button.

Sub-pick

In this mode you can pick objects that are components of other objects. This is good for selecting objects inside a group or symbol/footprint. Click the Edit→Select→ 

button.

Single Selection

To select single objects, click on the object in the viewport.

If you hold the ***Shift*** key then the object you select will be added to the selected items if it is not already selected else it will be removed from the selected item.

Window Pick

To select multiple objects ***left-click and hold*** in an area not occupied by any objects and drag the mouse. As you drag the mouse you will see all objects inside the rectangle defined by the initial mouse press and the current location of the mouse are selected. The objects must be completely inside the picking rectangle.

Select All

Click the Edit→Select→ 

button to select all objects on a sheet.

Invert selection

Click the Edit→Select→ 

button to invert the selection. All selected objects become unselected and all previously unselected objects become selected.

Unselect All

Click the Edit→Select→ 

button to unselect all objects in a sheet.

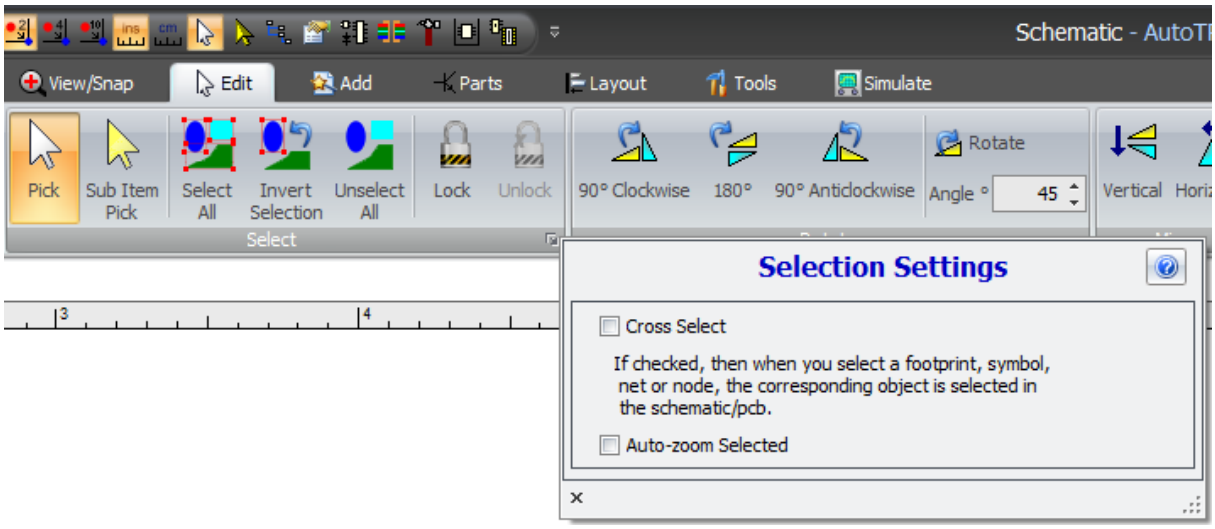
1.2.2.2.2 Selection Settings

When selecting items on a schematic you can optionally automatically select the corresponding PCB item, i.e. if you select a part symbol in a schematic the

corresponding footprint in the PCB will also be selected. Similarly if you select a footprint in the PCB then the corresponding part symbol in a schematic will be selected. To enable/disable this check the **Cross Select** checkbox in the Selection Settings popup dialog.

To view the Selection Settings popup click the button at the bottom right of the Edit→Select ribbon button group.

Automatically zooming to the selected object in its viewport is possible. To enable/disable this feature check the **Auto-zoom Selected** checkbox in the Selection Settings popup dialog.



1.2.2.2.3 Locking Objects

You can lock objects to prevent them from being edited or deleted.

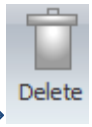
To lock an object:

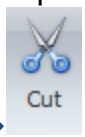
1. Select it.
2. Then tick the lock checkbox in its properties dialog in the Properties Panel.

Similarly, to unlock an object:

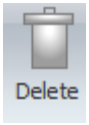
3. First select it.
4. Then clear the lock checkbox in its properties dialog in the Properties Panel.

1.2.2.2.4 Deleting Objects

To delete objects first Select them and then click the Edit→Edit→ 

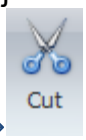
button. You can also delete them and place a copy of the deleted objects on the clipboard by clicking the Edit→Edit→ 

button. Either way, you can always undo you changes.

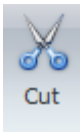
Pressing the ***Delete*** key is the same as clicking on the 

button.

1.2.2.2.5 Moving Objects to the Clipboard

To move objects from the design and place them onto the clipboard, click the Edit→Edit→ 

button. You can always undo the change.

Pressing ***CTRL+x*** is the same as pressing the 

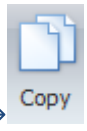
button.

1.2.2.2.6 Clearing Sheets and the PCB

To clear all the contents of a schematic or a PCB click the Edit→Edit→ 

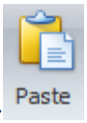
button.

1.2.2.2.7 Copying Objects

To copy selected objects click the Edit→Edit→ 

button. This copies the objects to the clipboard. You can then paste them onto sheets and into other applications.

1.2.2.2.8 Pasting Items from the Clipboard

To paste objects from the clipboard, click the Edit→Edit→ 

button. There are three different types of objects that can be pasted from the clipboard:

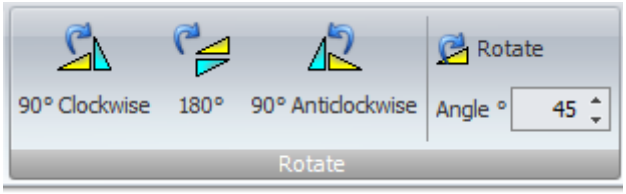
- **Images/pictures**. These are added to the current sheet as image objects.
- **Text**. Text is added as a text object to the current sheet.
- **Objects copied to the clipboard from AutoTRAX DEX**. These are added to the current sheet as selected objects. Once added, the objects are selected and will follow the movement of the mouse in the viewport. Left click the mouse to place the selected objects in their final position.

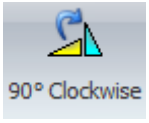
1.2.2.2.9 Rotating Objects

To rotate objects first select them.

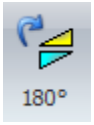
You can quickly rotate them by pressing the Space bar.

You can also rotate them using any of the commands in Edit→Rotate ribbon button group.

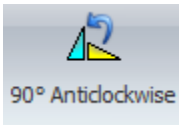


Click the 

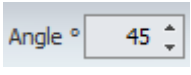
button to rotate the selected objects clockwise by 90°.

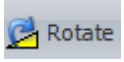
Click the 

button to rotate the selected objects by 180°.

Click the 

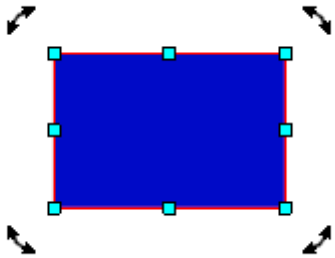
button to rotate the selected objects anticlockwise by 90°.

To rotate selected objects by a set angle, enter the angle in the 

control and click the 

button.

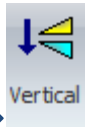
You can also scale objects by dragging the rotate handles.



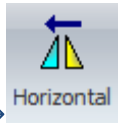
Selected box showing scale and rotate handles.

1.2.2.2.10 Mirroring Objects

To mirror objects about the horizontal or vertical axis first select them.

Click the Edit→Mirror→ 

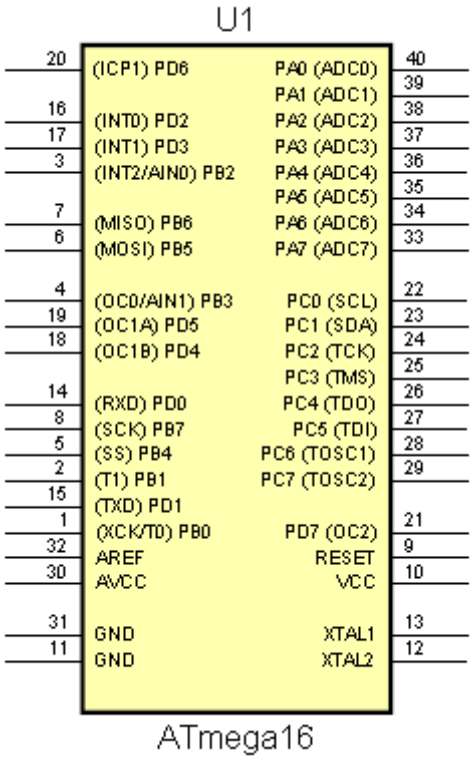
button to mirror the selected items about the vertical Y axis.

Click the Edit→Mirror→ 

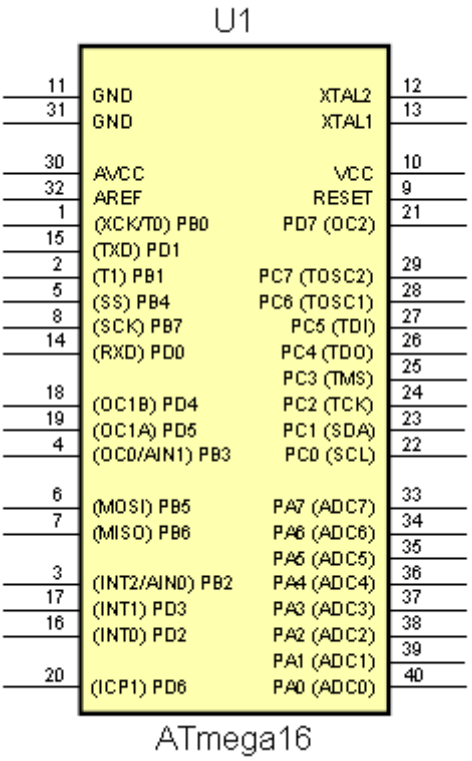
button to mirror the selected items about the horizontal X axis.

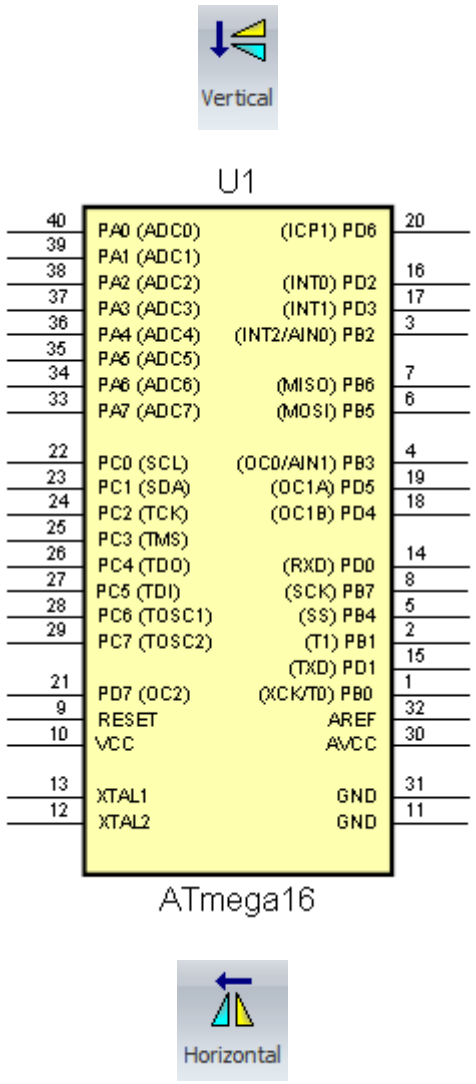
Mirroring Part Symbols

If you mirror a part symbol, the text orientation is maintained but symbol terminals change sides.



Original



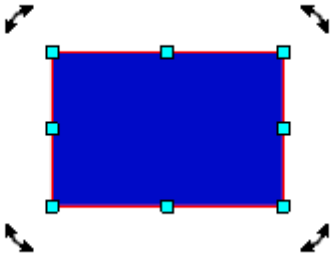


1.2.2.2.11 Scaling Objects

To scale objects first select them.

To scale Up/down by a percentage click the Up button in the Up Down % 200 Scale button group. Click the Down button to scale down. Set the percentage in the % 200 control.

You can also scale object by dragging the scale handles.



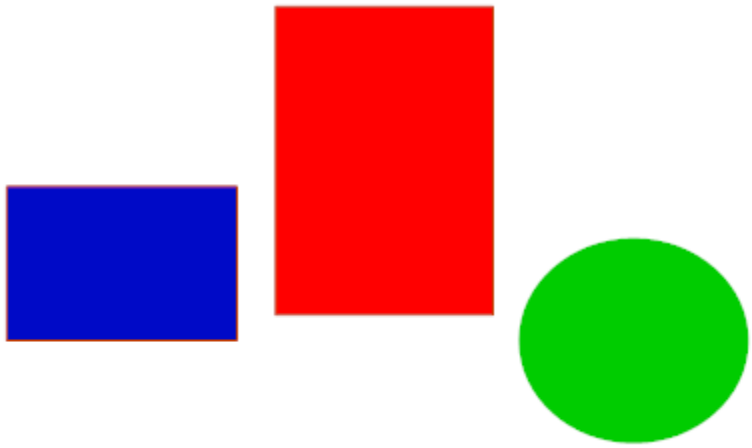
Selected box showing scale and rotate handles.

1.2.2.2.12 Aligning Objects

Aligning graphical objects is a process in graphic design that involves positioning elements in relation to each other or to the overall layout. Good alignment can make a design more organized, balanced, and aesthetically pleasing.

It's important to remember that while these tools are useful, alignment also depends on the designer's judgement. Good alignment is not just about precision, but also about creating a design that is balanced, harmonious, and communicates the intended message effectively.

To align 2 or more objects you must first select them.

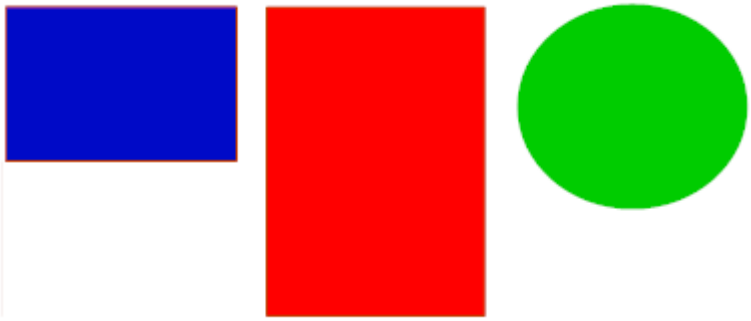


Original

Align Top

Click the Layout→Align→ 

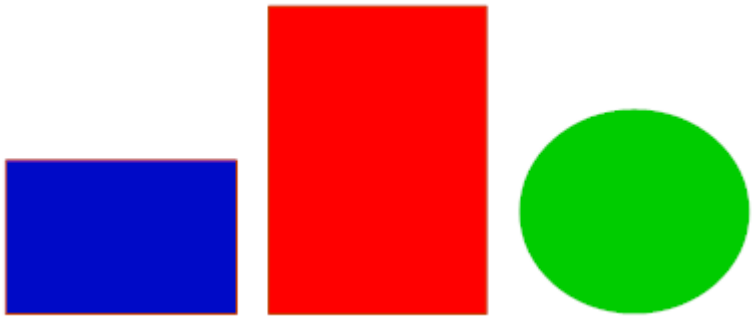
button.



Align Bottom

Click the Layout→Align→ 

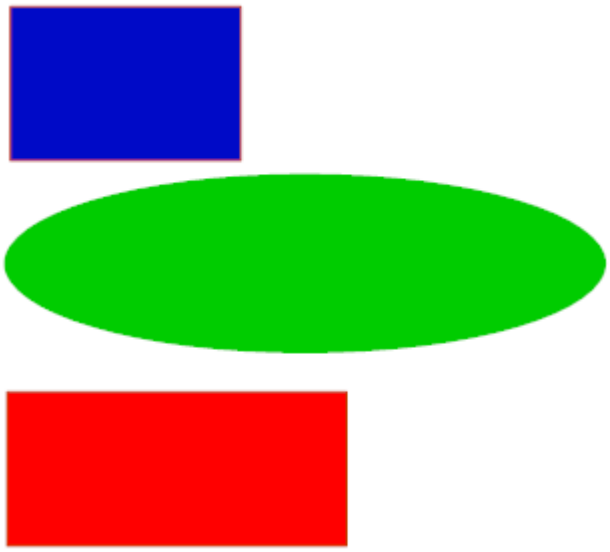
button/



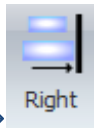
Align Left

Click the Layout→Align→ 

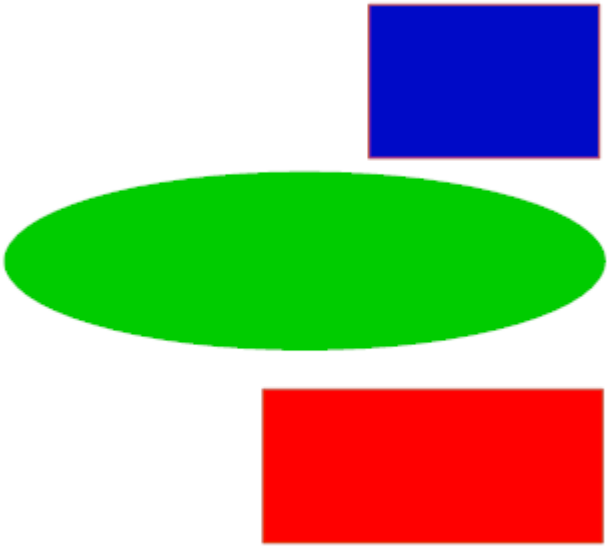
button.



Align Right

Click the Layout→Align→ 

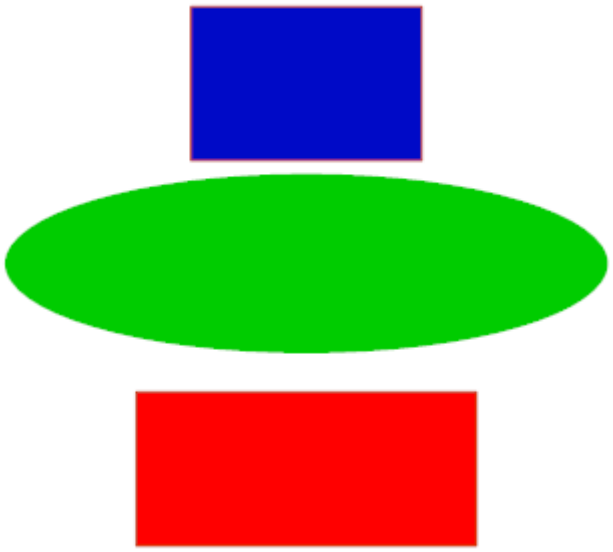
button.



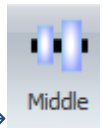
Align Center

Click the Layout→Align→ 

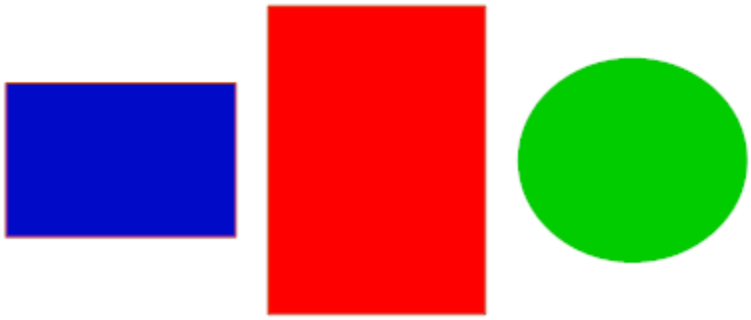
button.



Align Middle

Click the Layout→Align→ 

button.



1.2.2.2.13 Grouping Objects

In graphic design and digital illustration, "grouping" is a feature that allows you to combine multiple individual graphical elements or objects into a single unit. This can make it easier to manage, move, transform, or apply effects to those elements as if they were one object.

Grouped elements can still be edited individually, without ungrouping them. To do this, you usually need to double click on the group or use a specific tool or mode (like the Direct Selection tool in Adobe Illustrator or Edit Group mode in Sketch).

Grouping can be particularly useful when:

Working with Complex Designs: If your design contains many elements, grouping related ones can make the design more manageable. For example, you might group all the elements that make up a logo, an icon, or a section of a layout.

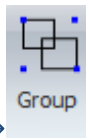
Aligning or Distributing Objects: If you want to align or distribute multiple objects as if they were a single unit, you can group them.

Applying Effects or Styles: If you want to apply the same effect (like a drop shadow, stroke, or fill color) or transformation (like rotation, scale, or translation) to multiple objects at once, you can group them.

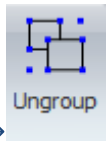
Preventing Accidental Changes: If you have elements that are positioned correctly in relation to each other and you don't want to accidentally move or edit them individually, you can group them.

In AutoTRAX, you can group objects by selecting them and using a command like "Group" (often found in the right-click menu or the main menu, and typically having a keyboard shortcut like Ctrl+G. You can also ungroup objects if you want to manage them individually again.

Remember, while grouping can make complex designs more manageable, overuse can also make a design hard to navigate, especially for other people who might work on it. So it's important to use it judiciously and keep your groups organized.

To group objects so that they can be moved, rotated and scaled as a single object first select them and then click the Layout→Group→ 

button.

To ungroup a group first select it and then click the Layout→Group→ 

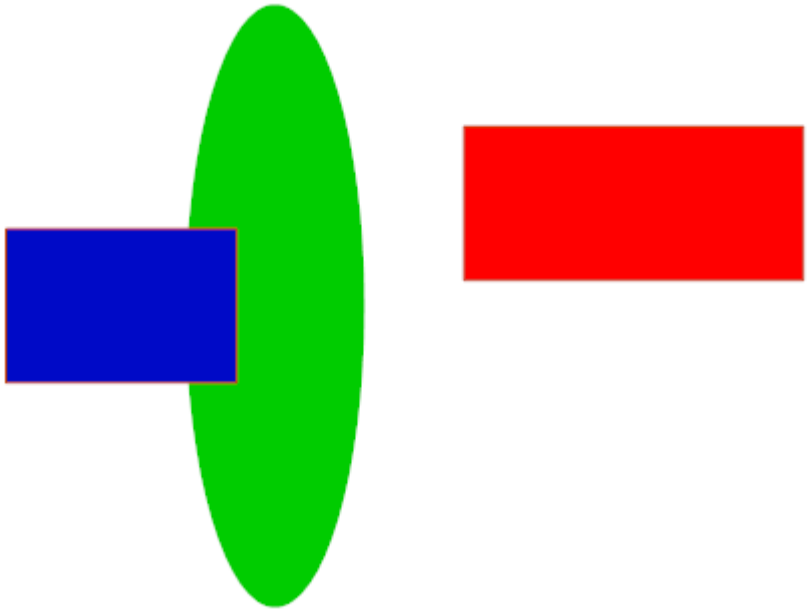
button.

1.2.2.2.14 Distributing Objects

In AutoTRAX, "distribution" is a command that allows for the equal spacing of objects along the vertical or horizontal axis. This can be very useful in creating diagrams, charts, graphs, or any design that requires precision and uniformity in the placement of objects.

Remember that distribution usually respects the outermost objects in the selection. For example, if you're distributing objects horizontally, the objects at the far left and right will stay where they are, and the rest will be spaced evenly between them.

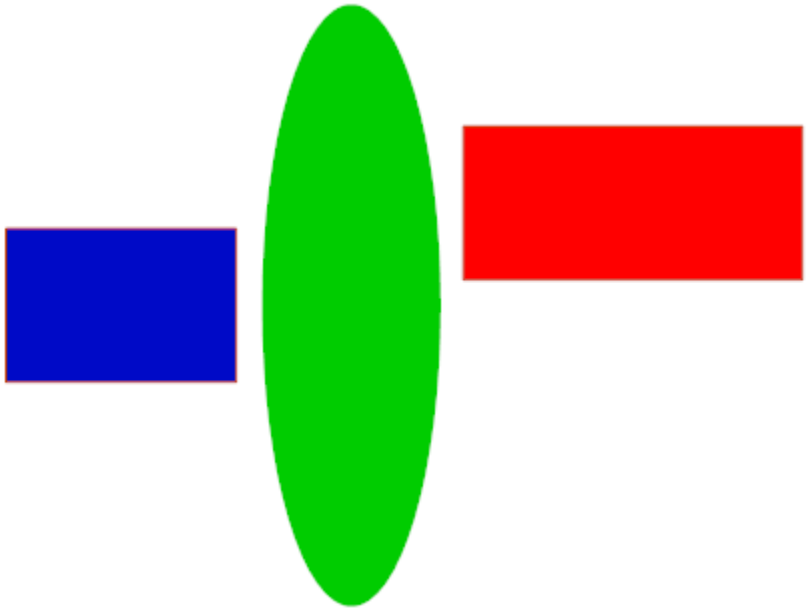
Distributing objects can be a huge time-saver and can help ensure precision and consistency in your diagrams or designs. As with all design tools, though, it's important to use it judiciously and in service of your overall design goals.



Original

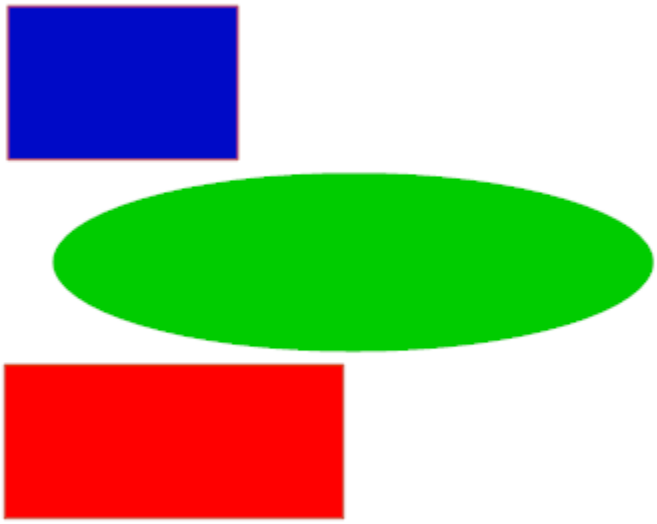
Distributing Horizontally

To distribute objects evenly in a horizontal direction, click the button in the Layout→Distribute button group.



Distributing Vertically

To distribute objects evenly in a vertical direction, click the button in the Layout→Distribute button group.



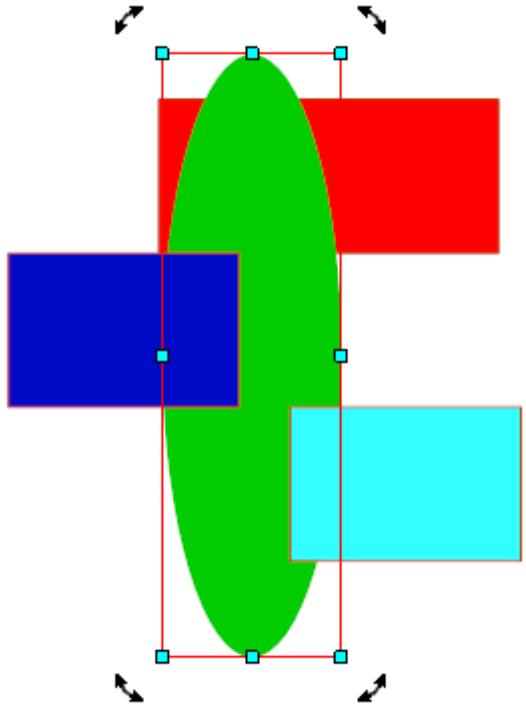
1.2.2.2.15 Reordering Objects

In graphic design and digital art, stacking refers to the order in which objects or layers appear on the Z-axis, that is, from front to back. This concept is also known as "Z-order," "stacking order," or "layer order."

Objects created later typically appear in front of those created earlier, and objects can be moved forward or backward in the stack to control which ones appear in front of others.

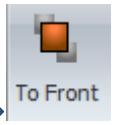
Changing the stacking order of objects can help create the illusion of depth, overlap elements in interesting ways, or simply ensure that the right objects are visible in a complex design. As always, the best practices for stacking and reordering depend on the specific needs and goals of your design.All objects are stacked vertically as they are created.

In AutoTRAX uou can change the stacking order.

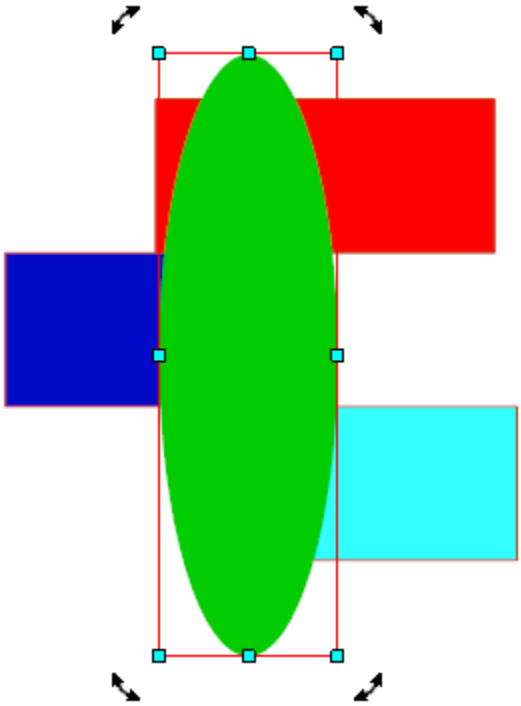


Original

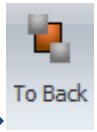
Bringing Objects to the Front



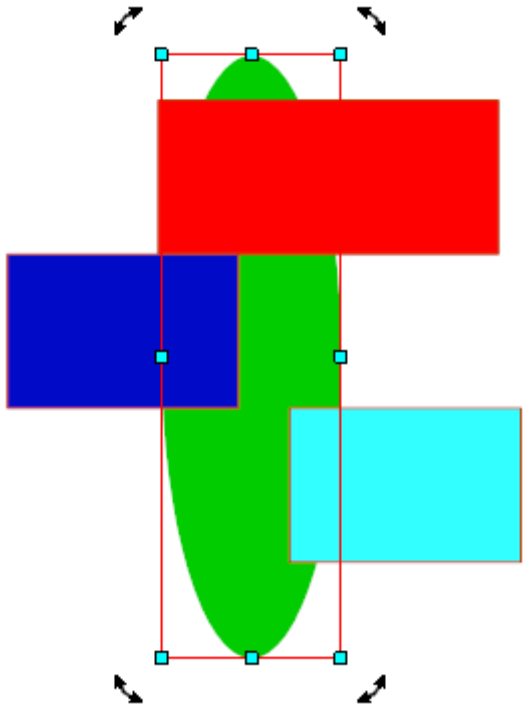
To bring objects to the front, first select them. Then click the Layout→Order→ button.



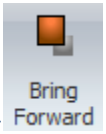
Sending Objects to the Back



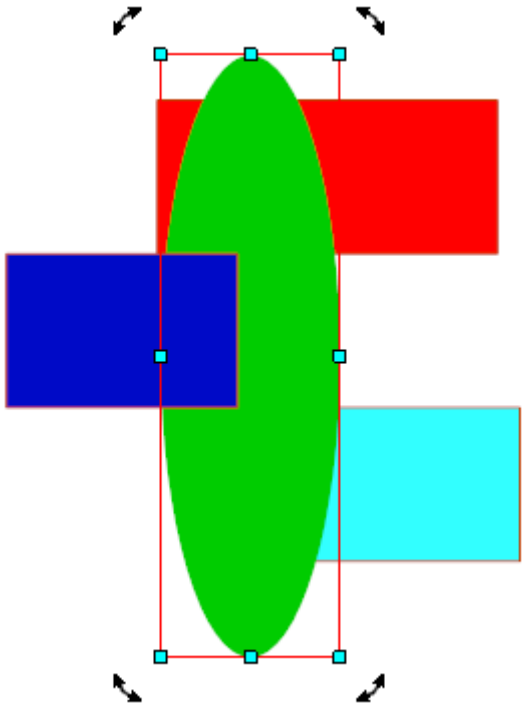
To send objects to the back, first select them. Then click the Layout→Order→ button.



Bringing Objects Up 1 Level

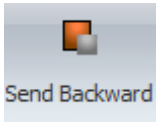
To bring objects up one level, first select them. Then Layout→Order→ 

button.

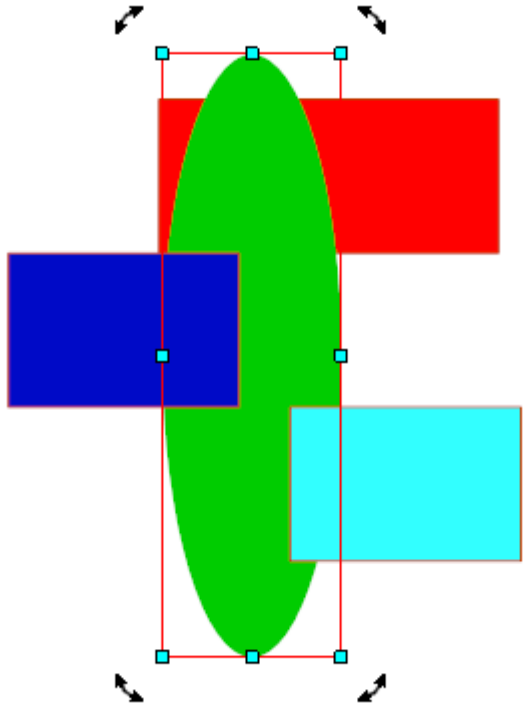


Sending Objects Down 1 Level

To send objects down one level, first select them. Then click the Layout→Order→



button.



1.2.2.2.16 Creating Arrays of Objects

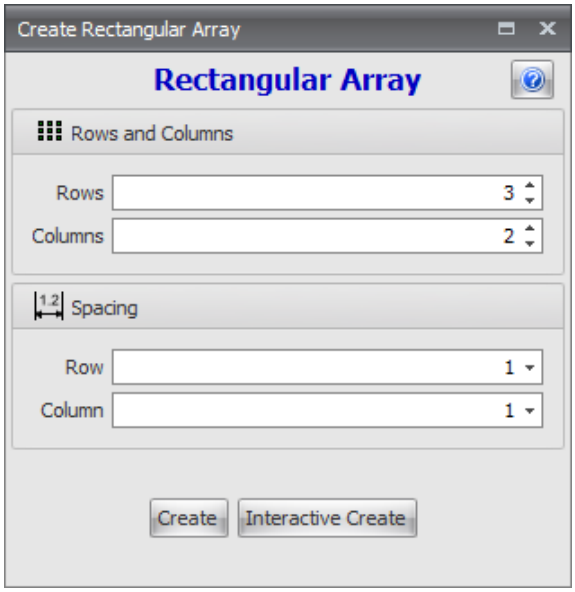
You can create a rectangular or circular array of selected objects.

1.2.2.2.16.1 Creating Rectangular Arrays

To create a rectangular array from one or more objects first select them.

Next click the Edit→Array→ 

button. This dialog box shown below will appear.



Rows and Columns

Rows

Enter the number of rows for the array.

Columns

Enter the number of columns for the array.

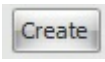
Spacing

Row

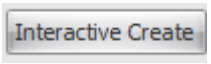
Enter the row (vertical) spacing.

Column

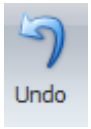
Enter the column (horizontal) spacing.

Click the 

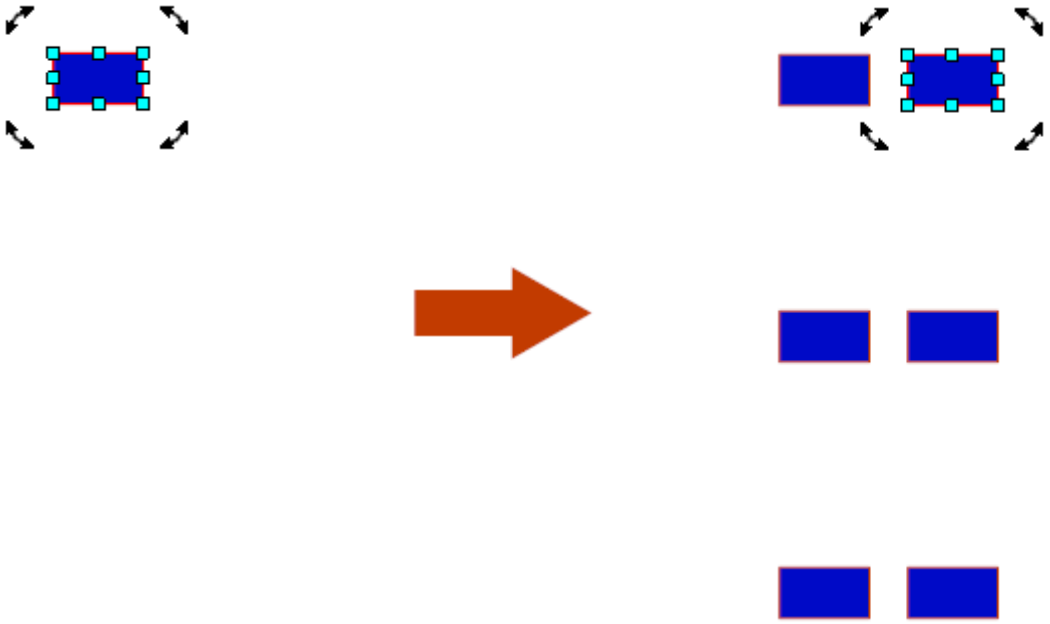
button to create the array.

Click the 

button to interactively create the array. As you drag the mouse in the viewport the array of objects will dynamically re-size according to the position of the mouse. Click the left mouse button to finalize the placement of the objects being arrayed.

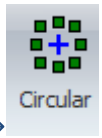
Remember, you can always click the 

button to undo a change. This lets you easily experiment.

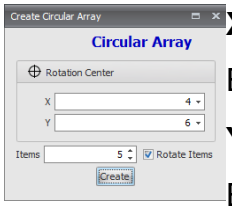


1.2.2.2.16.2 Creating Circular Arrays

To create a circular array from one or more objects first select them.

Next click the Edit→Array→ 

button. This dialog box shown below will appear.



X

Enter the center X coordinate of the rotation center.

Y

Enter the center Y coordinate of the rotation center.

Items

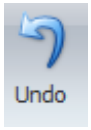
Enter the number of items.

Rotate Items

Check to rotate the items as they are copied.

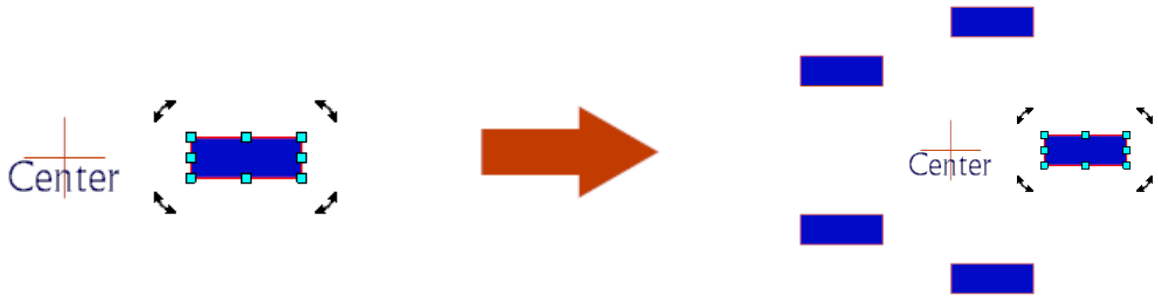
Click the 

button to create the array.

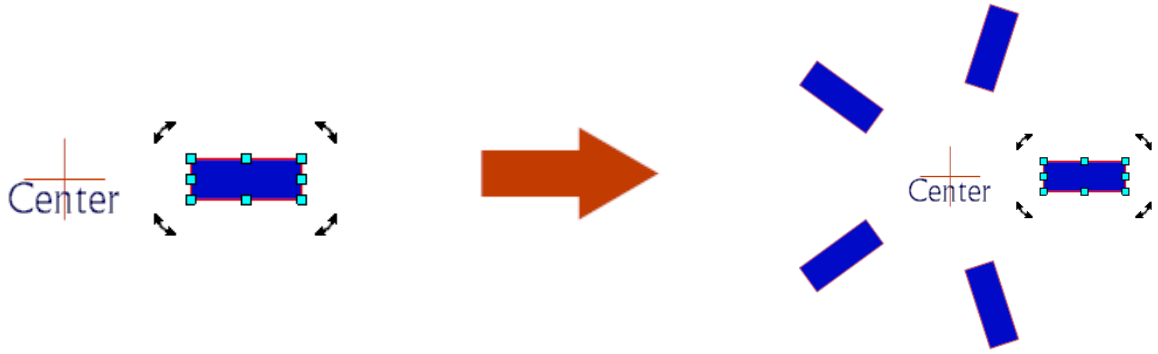
Remember, you can always click the 

button to undo a change. This lets you easily experiment.

Without Rotation



With Rotation

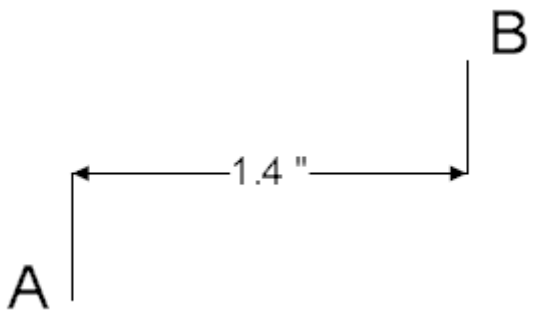


1.2.2.3 Coordinates

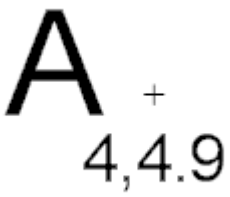
A coordinate is a graphics symbol that displays its X,Y position on a sheet.

It is very much like a Dimension but instead of displaying the distance between 2 points (a relative distance or vector) it displays the position.

Coordinates have no electrical significance.

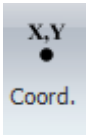


A Dimension showing the horizontal distance between 2 points, A and B



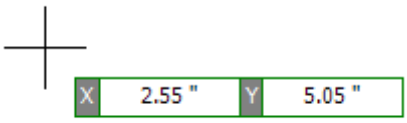
A Coordinate showing the position of point A

1.2.2.3.1 Adding Coordinates

To add a coordinate to a sheet click on the 

button in the Add→Dimension button group.

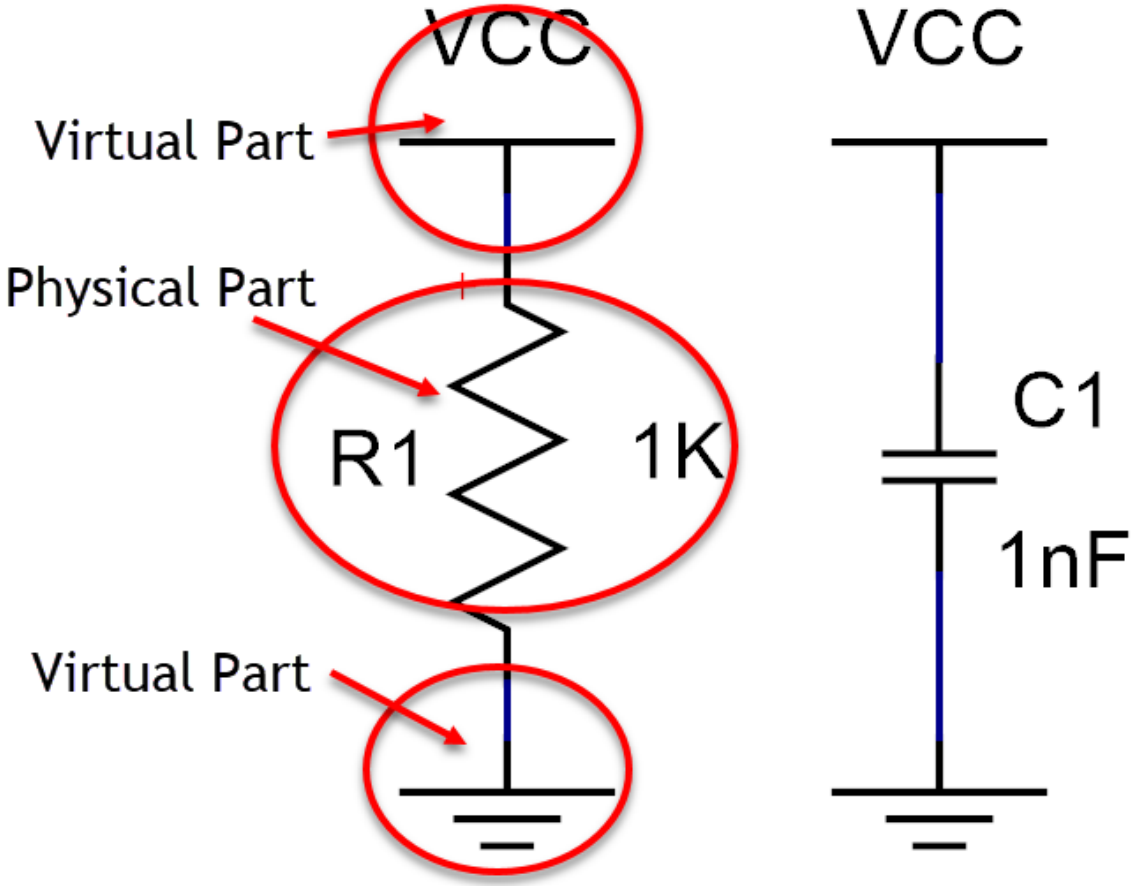
As you move the mouse cursor in the viewport, you will see the position cursor displayed below. You can press the ***'s'*** key to turn snap on/off.



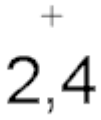
Adding a Coordinate

When the coordinate is at the position you want to place it, left-click on to finish creating it.

You can enter the value of the X and Y coordinates by first pressing the **Enter** key followed by the X coordinate, the **Enter** key, the Y coordinate and finally the **Enter** key to complete creating the coordinate.



Entering the X and Y coordinate using the keyboard



Completed Coordinate

1.2.2.3.2 Editing Coordinates

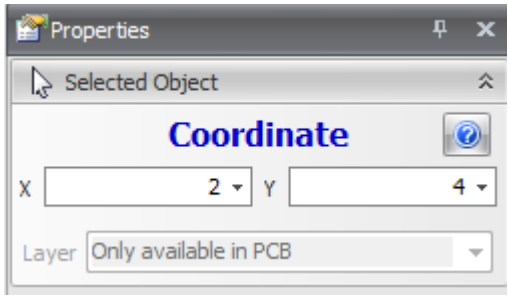
There are 2 ways to edit a coordinate, dragging it or using the Properties panel.

Dragging

You can hold down the left mouse button on a coordinate to select it and then drag the mouse to move it. You will see the X and Y values of the coordinate change as you move it. Press the ***'s'*** key to turn snap on or off.

Properties Panel

If you select a coordinate and the Properties panel is visible, the coordinate's properties dialog will be shown.



X

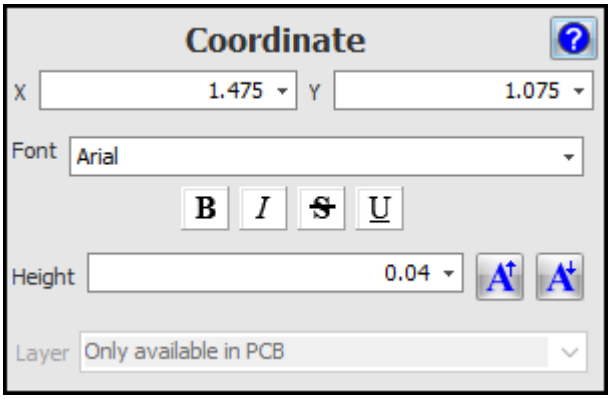
The X or horizontal position of the coordinate.

X

The Y or vertical position of the coordinate

Pressing the button displays this help page.

1.2.2.3.3 Coordinate Editor



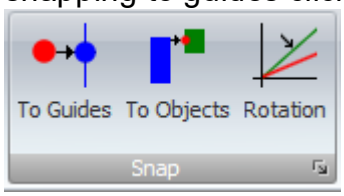
The Coordinate Editor

1.2.2.4 Guides

You can add guides to graphical sheet's to act as visual guides, or to act as snap guides.

Snapping to Guides

You can snap objects to the horizontal, vertical or point guides. To toggle snapping to guides click on the 

button in the View/Snap ribbon tab's 

button group.

If Snapping to Guides is enabled then when objects are created or moved, they will snap to guides when they are close to them.

There are 4 types of guides

Horizontal Guides

Horizontal Guides will snap the vertical Y coordinate of objects.

Vertical Guides

Point Guides

Angled Guides

Vertical Guides will snap the horizontal X coordinate of objects.

Point Guides will snap both the horizontal X and the vertical Y coordinates of objects

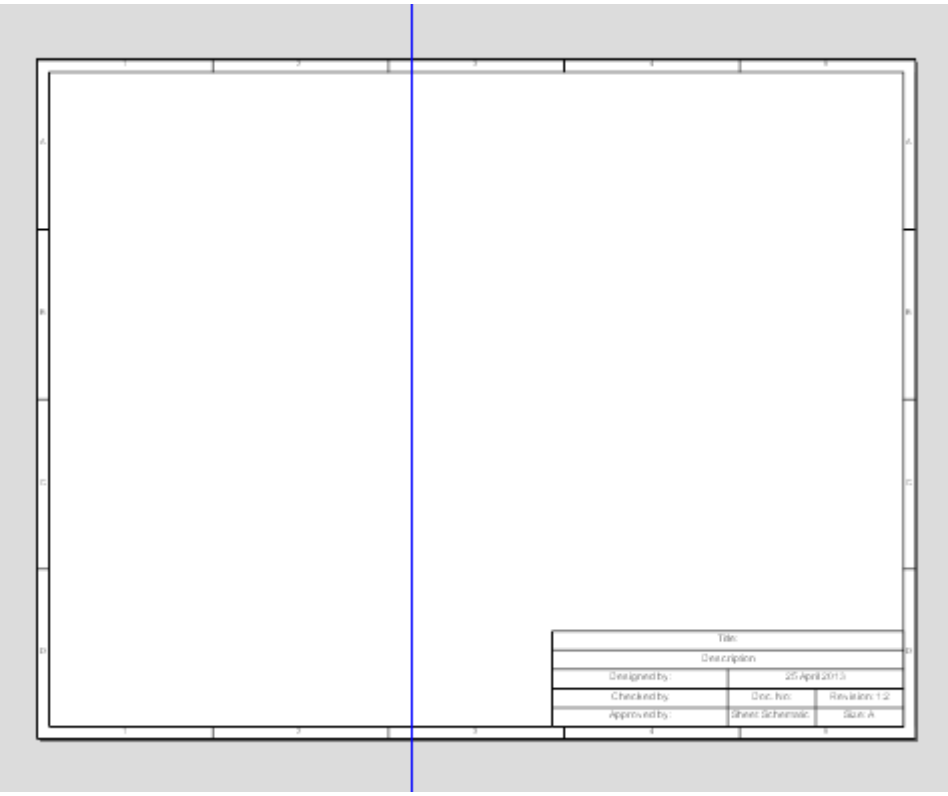
Angled Guides will snap both the horizontal X and the vertical Y coordinates of objects to the Angled Guide

Guides are a powerful tool in AutoTRAX DEX that help you layout your design and also serve to snap objects to a custom grids. This video show you how to add horizontal and vertical guides as well as point guides.

1.2.2.4.1 Vertical Guides

Vertical guides are vertical lines that act as:

- Visual Clue
- X coordinate snap points when adding or moving objects.

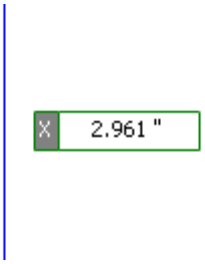


A vertical guide (blue line)

Adding Vertical Guides

There are 2 ways to add a vertical guide to a sheet.

1. The first and easiest is to hold down left mouse button on the vertical ruler to the left of the sheet (If rulers are visible) and drag it into position.
2. The second method is to click on the  button in the Add→Guides button group.



The vertical guide and its horizontal position during initial creation

In both cases, as you drag the guide to position, you can enter a number value by first pressing the **Enter** key and then the X coordinate or click the left mouse button to place the guide.

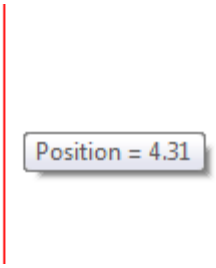
You can press the ***'s'*** key at any time to turn snap on or off.

Editing Vertical Guides

There are 2 ways to edit a vertical guide, dragging it or using the Properties panel.

Dragging

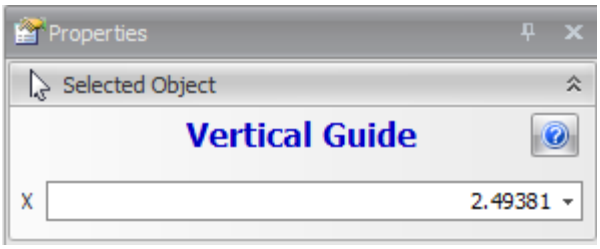
You can hold down the left mouse button on a vertical guide and drag the mouse to move the guide; release the mouse button when the guide is where you want it. You will see the X values of the coordinate change as you move it. Press the ***'s'*** key to turn snap on or off.



Vertical guide's horizontal position displayed during dragging

Properties Panel

If you select a vertical guide and the Properties panel is visible, the vertical guide properties dialog will be shown.



Vertical guide properties

X

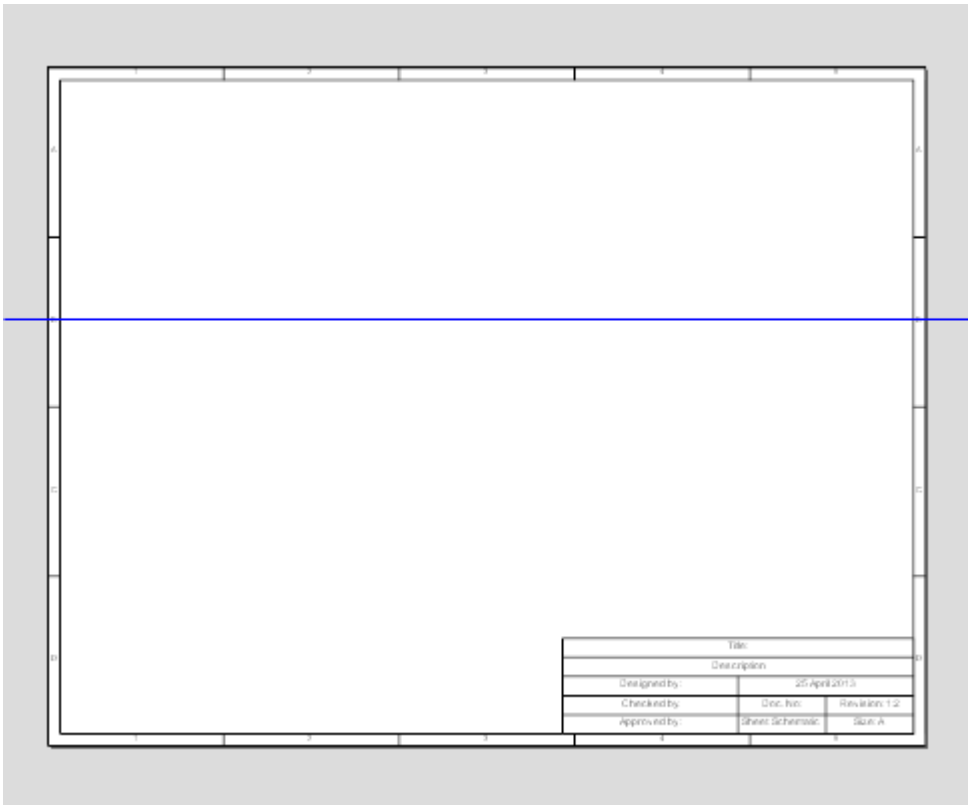
The X or vertical position of the vertical guide

Pressing the button displays this help page.

1.2.2.4.2 Horizontal Guides

Horizontal guides are horizontal lines that act as:

- Visual clues
- Y coordinate snap points when adding or moving objects.

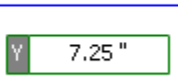


A horizontal guide (blue line)

Adding Horizontal Guides

There are 2 ways to add a horizontal guide to a sheet.

1. The first and easiest is to hold down the left mouse button on the horizontal ruler at the top of the sheet (If rulers are visible) and drag it into position.
2. The second method is to click on the button in the **Add→Guides** button group.



The horizontal guide and its horizontal position during initial creation

In both cases, as you drag the guide to position it you can enter a number value by first pressing the **Enter** key and then the Y coordinate or click the left mouse button to place the guide.

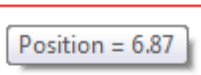
You can press the ***'s'*** key at any time to turn snap on or off.

Editing Horizontal Guides

There are 2 ways to edit a horizontal guide, dragging it or using the Properties panel.

Dragging

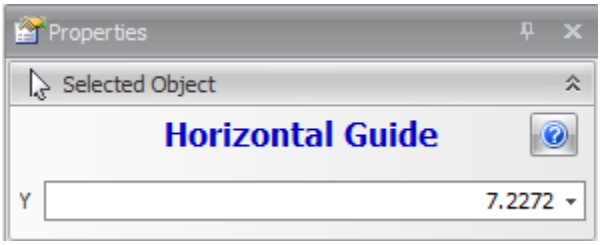
You can hold down the left mouse button on a horizontal guide and drag the mouse to move the guide. You will see the Y value of the coordinate change as you move it. Press the ***'s'*** key to turn snap on or off.



Horizontal guide's vertical position displayed during dragging

Properties Panel

If you select a horizontal guide and the Properties panel is visible, the horizontal guide properties dialog will be shown.



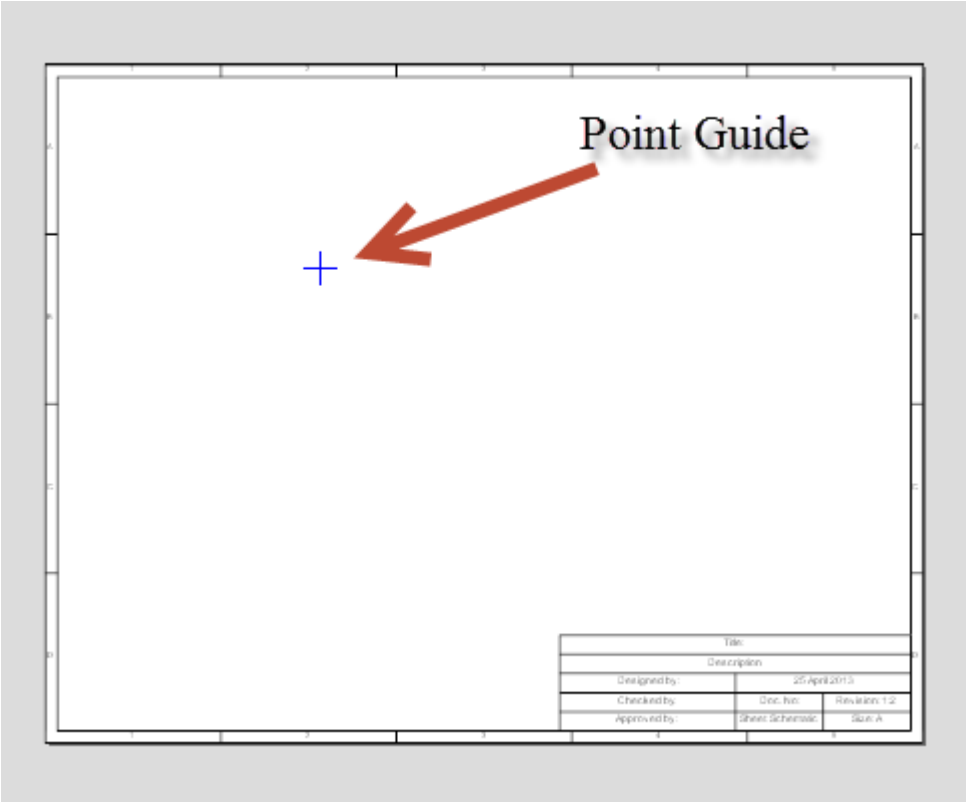
Horizontal guide properties

The Y or vertical position of the horizontal guide

Pressing the  button displays this help page.

1.2.2.4.3 Point Guides

Point guides are points that act as a visual clue and also as an X,Y coordinate snap point when adding or moving objects.



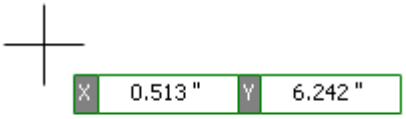
A point guide (blue cross)

1.2.2.4.3.1 Adding Point Guides

There are 2 ways to add a point guide to a sheet.

1. The first and easiest is to hold down the left mouse button on the ruler origin in the upper left hand corner of the rulers (If rulers are visible) and drag it into position.

2. The second method is to click on the  button in the Add→Guides button group.



3.

The point guide and its X,Y position during initial creation

In both cases, as you drag the guide to position it you can enter a number value by first pressing the **Enter** key and then the X and Y coordinate or click the left mouse button to place the guide.

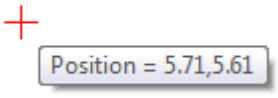
You can press the ***'s'*** key at any time to turn snap on or off.

1.2.2.4.3.2 Editing Point Guides

There are 2 ways to edit a point guide, dragging it or using the Properties panel.

Dragging

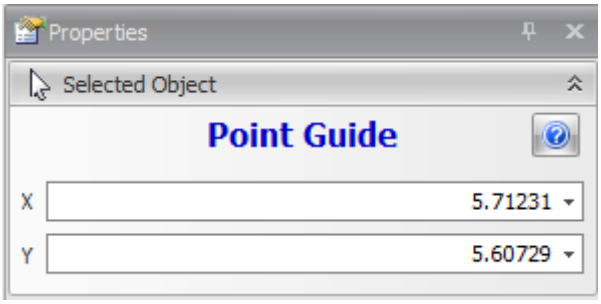
You can hold down the left mouse button on a guide and then drag the mouse to move the guide. You will see the X, Y values of the point change as you move it. Press the ***'s'*** key to turn snap on or off.



Point guide's vertical position displayed during dragging

Properties Panel

If you select a point guide and the Properties panel is visible, the point guide properties dialog will be shown.



Point guide properties

X

The X or horizontal position of the point guide.

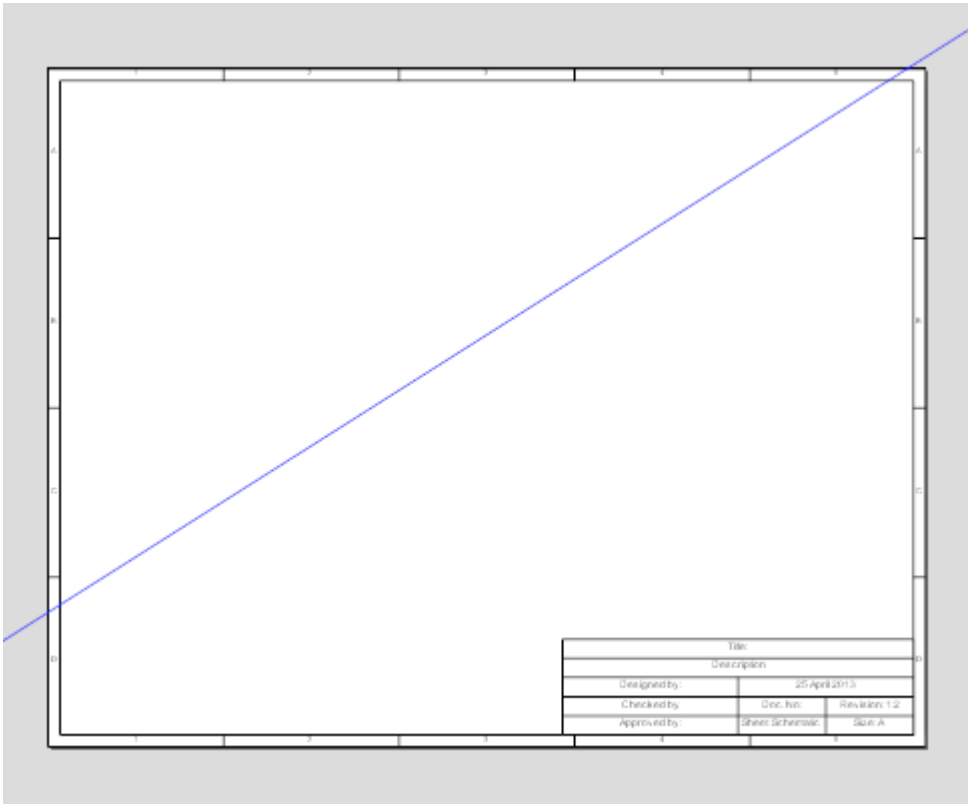
X

The Y or vertical position of the point guide

Pressing the  button displays this help page.

1.2.2.4.4 Angled Guides

Angled guides are angled lines that act as a visual clue and also as an X,Y coordinate snap point when adding or moving objects.



An angled guide (blue diagonal line)

1.2.2.4.4.1 Adding Angled Guides

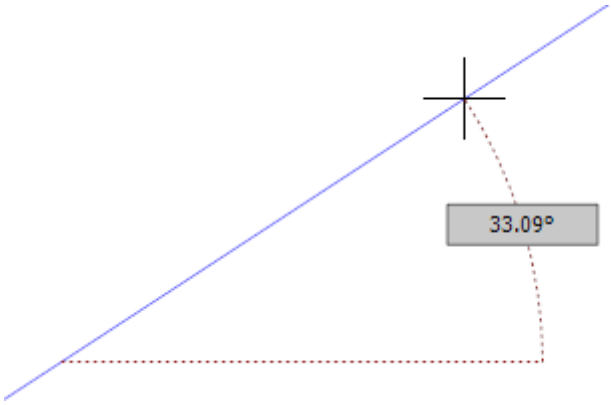
Unlike the other guides there is only one way to add an angled guide to a sheet and that is by clicking on the  button in the **Add→Guides** button group.

The angled guide and its crossing point during initial creation

As you drag the guide to position it you can enter a number value by first pressing the **Enter** key and then the crossing coordinate or click the left mouse button to define the crossing point..

You can press the ***'s'*** key at any time to turn snap on or off.

Now as you drag the mouse, the angle of the angled guide will vary. Left-click to finish creating the angled guide or you can enter a number value for the angle in degrees by first pressing the **Enter** key and then entering the angle.



The angle of the angled guide as you drag the mouse

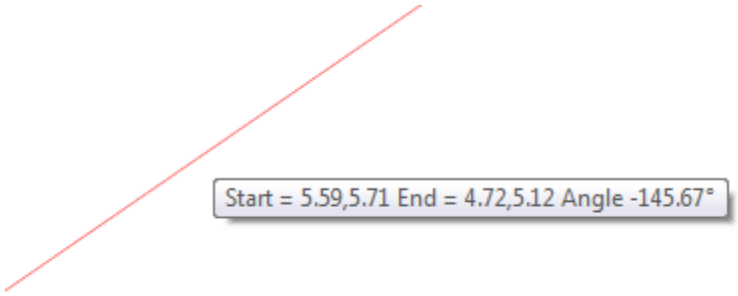
1.2.2.4.4.2 Editing Angled Guides

There are 2 ways to edit an angled guide, dragging it or using the Properties panel.

Dragging

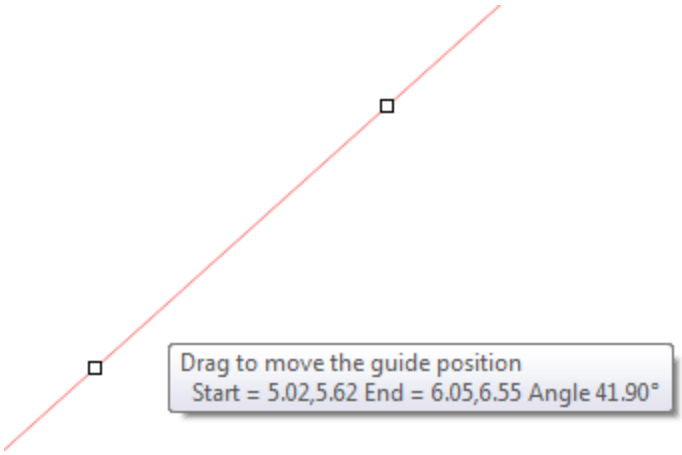
You will see the angled guide in red together with 2 point manipulators. Drag any of them to edit the angled guide.

You can hold down the left mouse button on an angled guide manipulator and drag the mouse to move the guide. Press the ***'s'*** key to turn snap on or off.

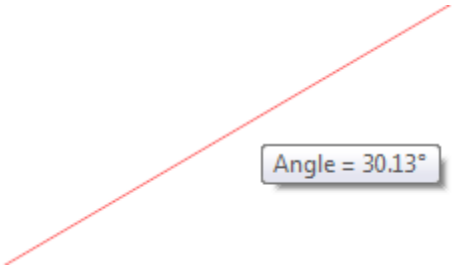


Dragging an angled guide

Or, you can drag on of the 2 guide manipulators. You will see the angle change as you move it.



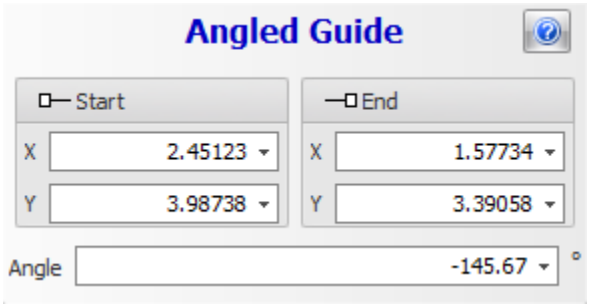
Angled guides manipulators



Guide during dragging a manipulator point

Properties Panel

If you select an angled guide and the Properties panel is visible, the angled guide properties dialog will be shown.



Angled guide properties

X

The X or horizontal position of the guides start or end.

X

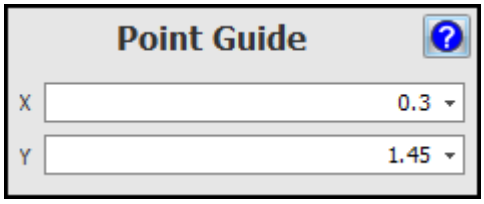
The Y or vertical position of the guides start or end.

Angle

The angle of the guide in degrees.

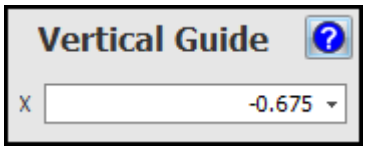
Pressing the  button displays this help page.

1.2.2.4.4.3 Point Guide Editor



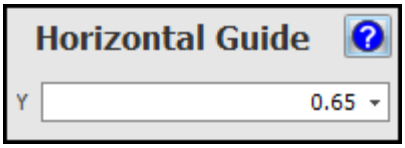
The Point Guide Editor

1.2.2.4.4.4 Vertical Guide Editor



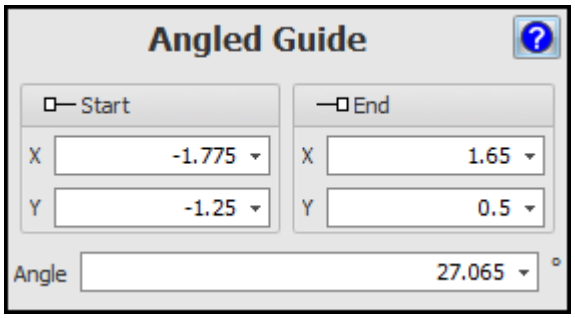
The Vertical Guide Editor

1.2.2.4.4.5 Horizontal Guide Editor



The Horizontal Guide Editor

1.2.2.4.4.6 Angle Guide Editor



The Angled Guide Editor

1.2.2.5 Snaps and the Grid

AutoTRAX DEX can optionally display a grid to help you place items. The grid can display as a graph grid, a line grid or a dot grid see the Grids

You can optionally snap objects to the rectangular grid. The snap can be to whole grid spacing or 2-10 divisions of the grid spacing.

You can also optionally have a Rotation Snap when rotating objects.

Symbol and Wire Grid

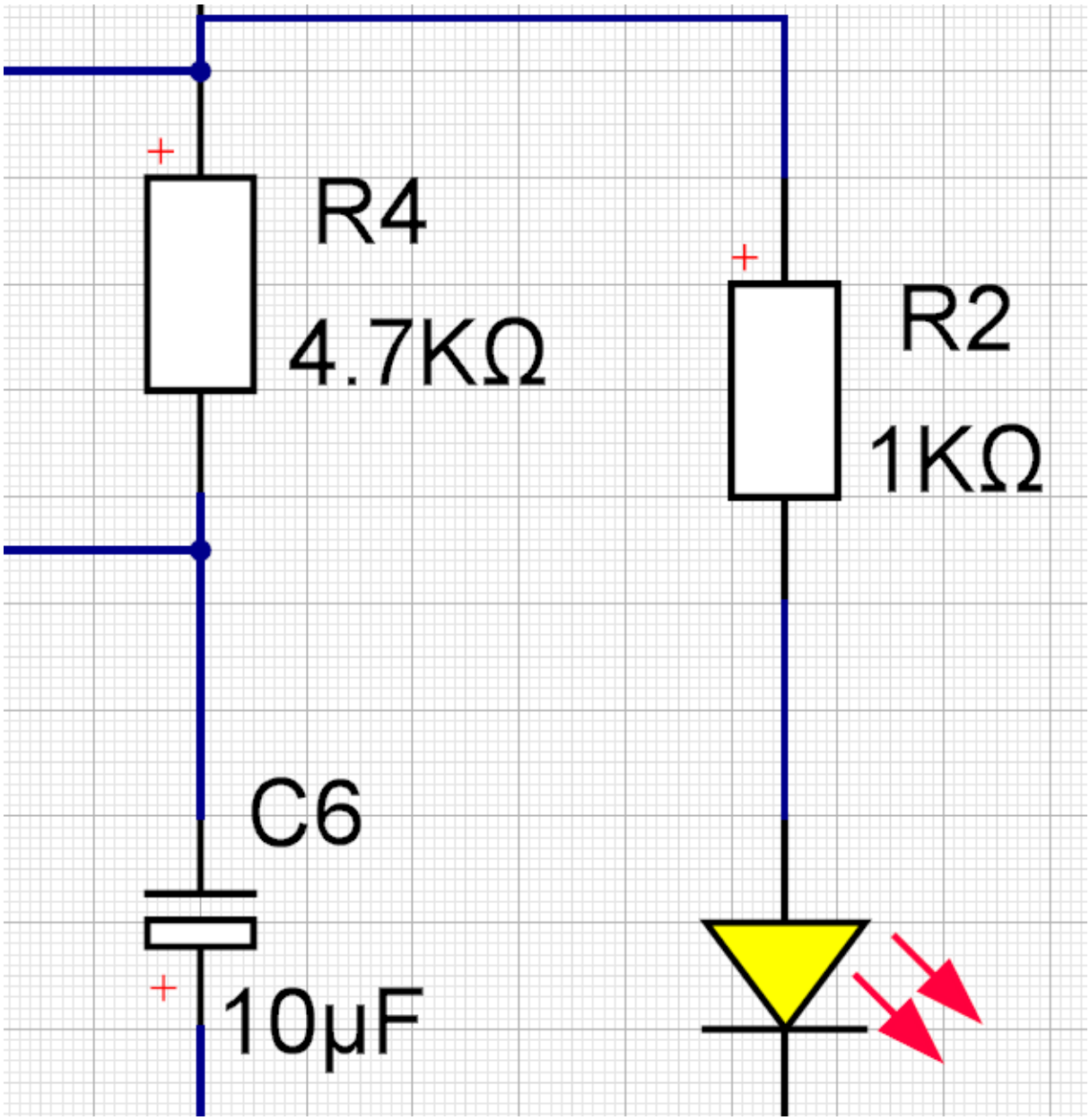
1.2.2.5.1 The Grids

There are 3 different types of grids possible in AutoTRAX DEX, the graph, the line and the dot grids.

Use the Snap Settings pop up to change grid types.

The Graph Grid

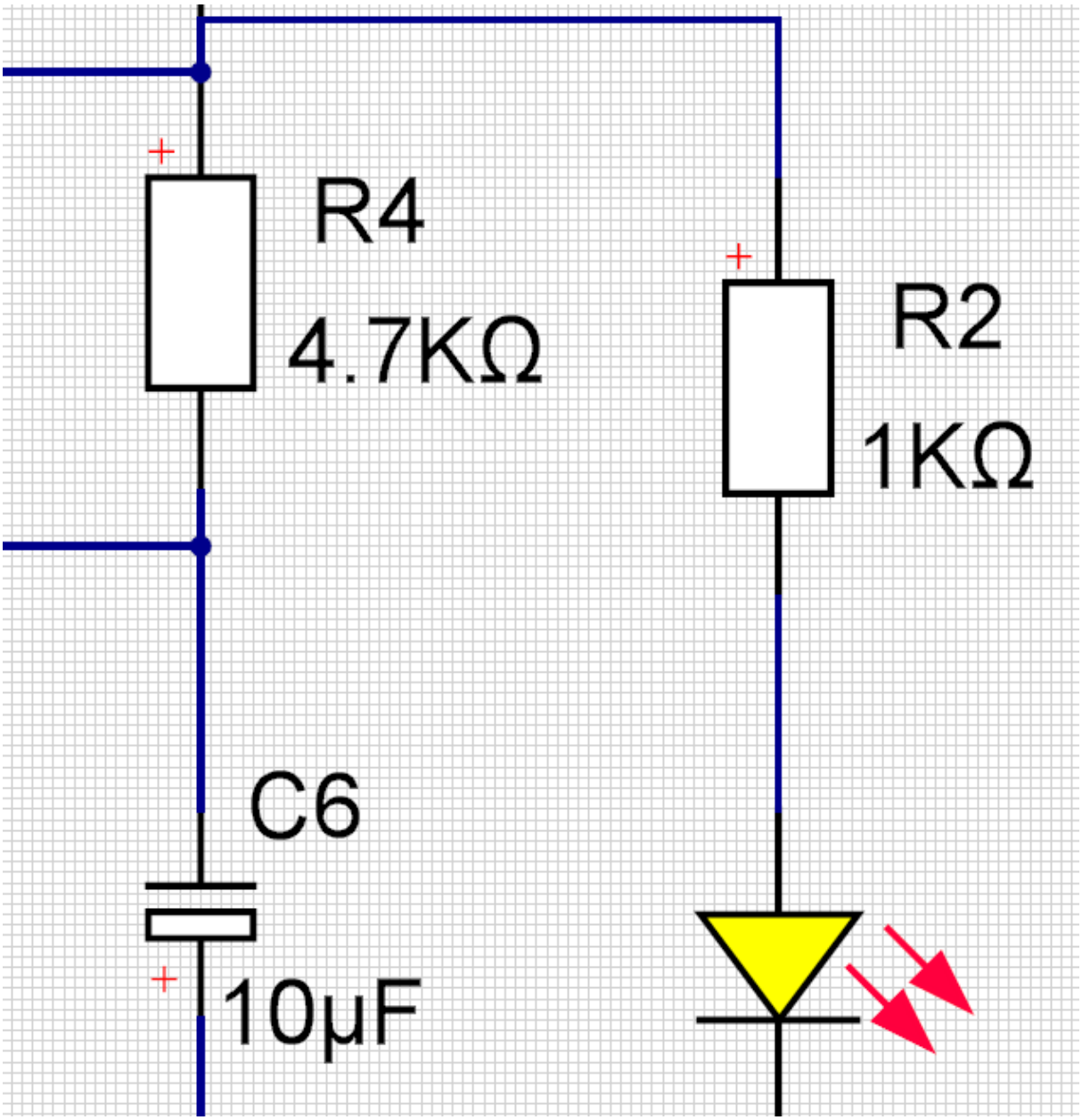
The graph grid is like a sheet of graph paper. As you zoom in and out the graph paper adjusts to show a grid-like structure with strong major divisions and lighter minor division. When you zoom in, the graph paper will subdivide to show more grid lines.



The Graph Grid

The Line Grid

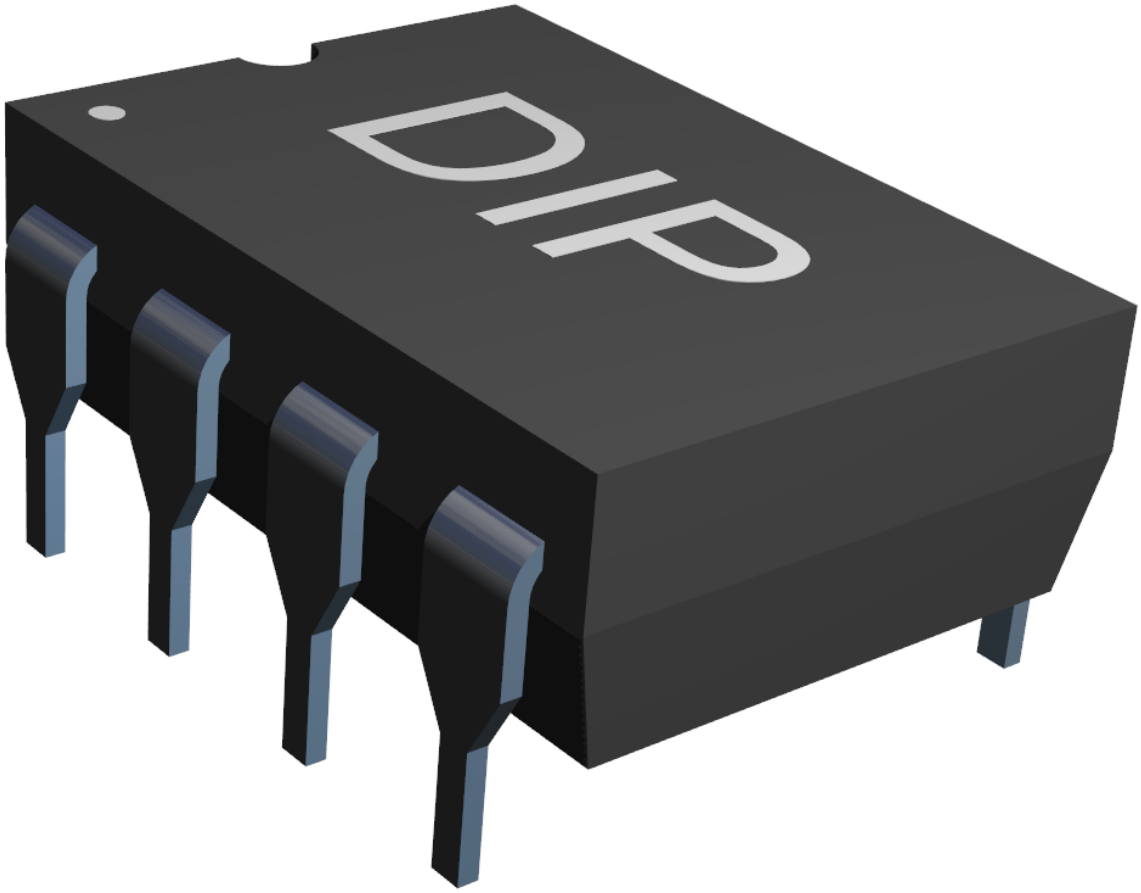
The line grid displays as a set of horizontal and vertical lines as shown below.



The Line Grid

The Dot Grid

The dot grid is drawn as a series of points at the major snap points.



The Dot Grid

1.2.2.5.2 The Origin

The "origin" typically refers to the starting point or reference point from which the positions of all other elements are determined. This point is often defined as the (0,0) coordinate in a 2D or 3D space.

2D Graphics: In 2D graphics, such as in graphic design software or web design, the origin is typically at the top left corner of the screen or canvas. Moving right from the origin increases the X coordinate and moving down increases the Y coordinate. So, a point can be defined anywhere on the canvas using an (X,Y) coordinate, where X is the distance from the origin horizontally and Y is the distance vertically.

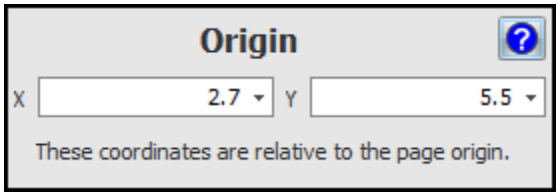
3D Graphics: In 3D graphics, used in 3D modeling, animation, or video games, the origin is the point where the X, Y, and Z axes intersect. The X and Y axes define a horizontal plane (usually corresponding to width and depth), and the Z axis defines the vertical direction (height). So, a point in 3D space can be defined with an (X,Y,Z) coordinate, where X, Y, and Z are the distances along each axis from the origin.

The origin is a fundamental concept in graphics and geometry, as it provides a consistent and predictable way to define the position of elements. Transformations such as translations (moving an object), rotations, and scaling (changing size) are also often performed relative to the origin, or an object's own local origin (often its geometric center or a user-defined point).

The origin can be editing by display it and the dragging it or changing it's position in the properties panel.

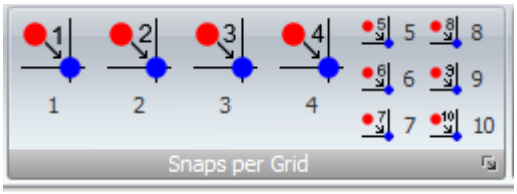
Both the horizontal and the vertical roller in the origin box have context menus that are command items to set the origin.

1.2.2.5.3 Origin Editor



The Origin Editor

1.2.2.5.4 Setting the Snaps per Grid



You can set the number of snaps per grid spacing from 1 to 10 using the Snaps per Grid button group in the View/Snap ribbon menu tab.

Click on the button once to set the snaps per grid and enable snap to grid Click on the same button again to disable snap to grid. Click on another button to set the snaps per grid as shown in the button.

1.2.2.5.5 Rotation Snap

In AutoTRAX, "rotation snap" or "angle snap" is a feature that allows the rotation of objects to snap to specified angle increments. This feature provides precision and consistency when rotating objects.

For example, with rotation snap set to 45 degrees, as you rotate an object, it will snap to 0, 45, 90, 135, 180, 225, 270, 315, and 360 degrees. This allows for precise alignment of objects at exact angles.

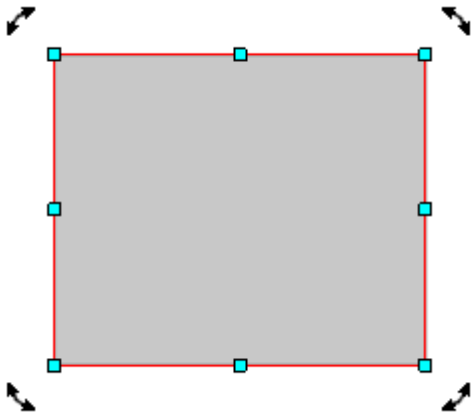
The value of the rotation snap angle can typically be set to any desired value in the software's settings, allowing for custom rotation increments.

This feature can be particularly helpful when creating symmetrical designs or when exact alignment of objects is needed. Most software allows for rotation snap to be easily toggled on or off, giving designers flexibility to rotate objects freely or with precision as needed.

You can snap to a rotation angle when rotating objects. Use the Snap Settings pop up.

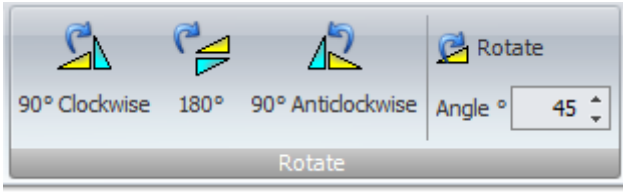
You can rotate objects by selecting the objects and:

- Pressing the **Space key** or
- Dragging any of the rotate handles at the 4 corners



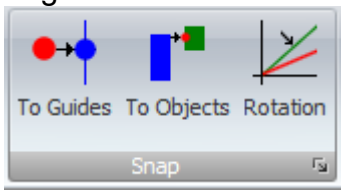
Selected item showing rotate handles

- Clicking on any of the rotate command buttons in the Rotate group of the Edit ribbon tab



1.2.2.5.6 Snapping to Guides

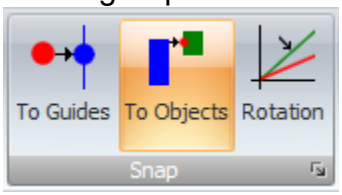
You can snap objects to the horizontal, vertical or point guides. To toggle snapping to guides click on the 

button in the View/Snap ribbon tab's 

button group.

1.2.2.5.7 Snapping to Objects

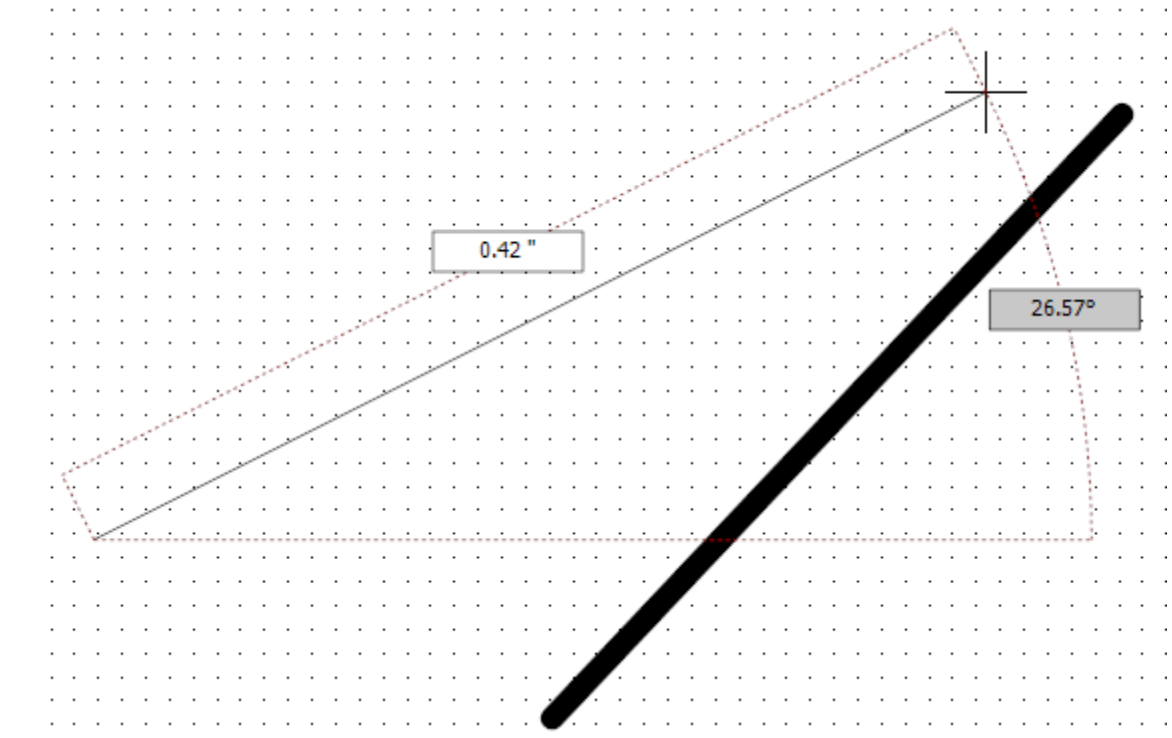
You can snap object to others as you draw them.

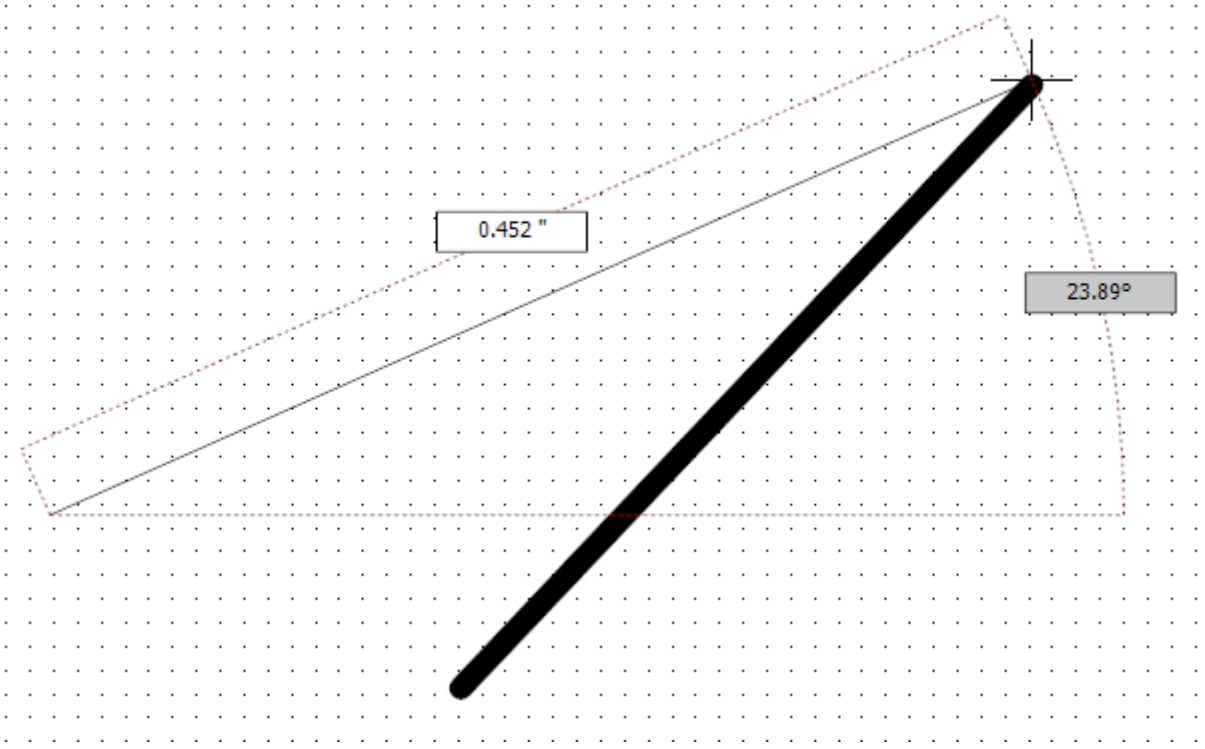
Select the Snap To Objects button in the Snap button group in the **View/Snap** ribbon tab to turn snapping to objects on or off. 

Certain graphic objects have intrinsic snap point, for instance lines have intrinsic snap points at the start and end of the line.

If you have snap to objects on and you add say a line, as you drag the line to define it and you move closer to the start or end of another line, the end of the line you are adding will snap to the closed end point of the other line.

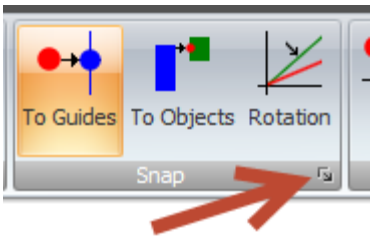
As you drag the end of a line that you are creating, when you get close to the end of another line, the new line will snap to an end point of the second line.





1.2.2.5.8 Snap Settings

To set the grid and snap settings click on the small button at the bottom right of the Snap button group in the View→Snap ribbon tab...



The Grid/Snap Setting dialog will appear as shown below.

 Clicking to display this help topic.

Snap Spacing

This is the major snap grid spacing. You change the actual snaps per grid using the buttons in the **Snaps Per Grid** button group.

Graph

Click to display the grid as a graph grid.

Line

Click to display the grid as a line grid.

Dot

Click to display the grid as a dot grid.

Draw To Units

If checked, grid lines/points will be spaced on the measurements units you are using e.g. inches or cm.

Draw to Snap Spacing

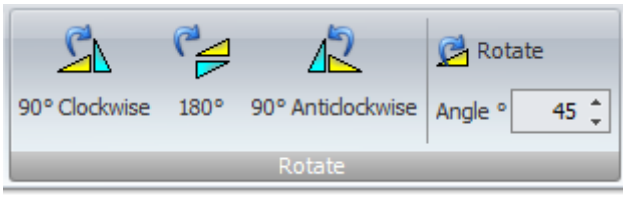
If checked, grid lines/points will be spaced on the **Snap Spacing** you define.

Snap Rotation Angle

This is the rotation snap angle in degrees for rotating selected objects when you press the **space bar**.

Menu Rotation Angle

This is the rotation snap angle in degrees for rotating selected objects when you use the menu buttons.



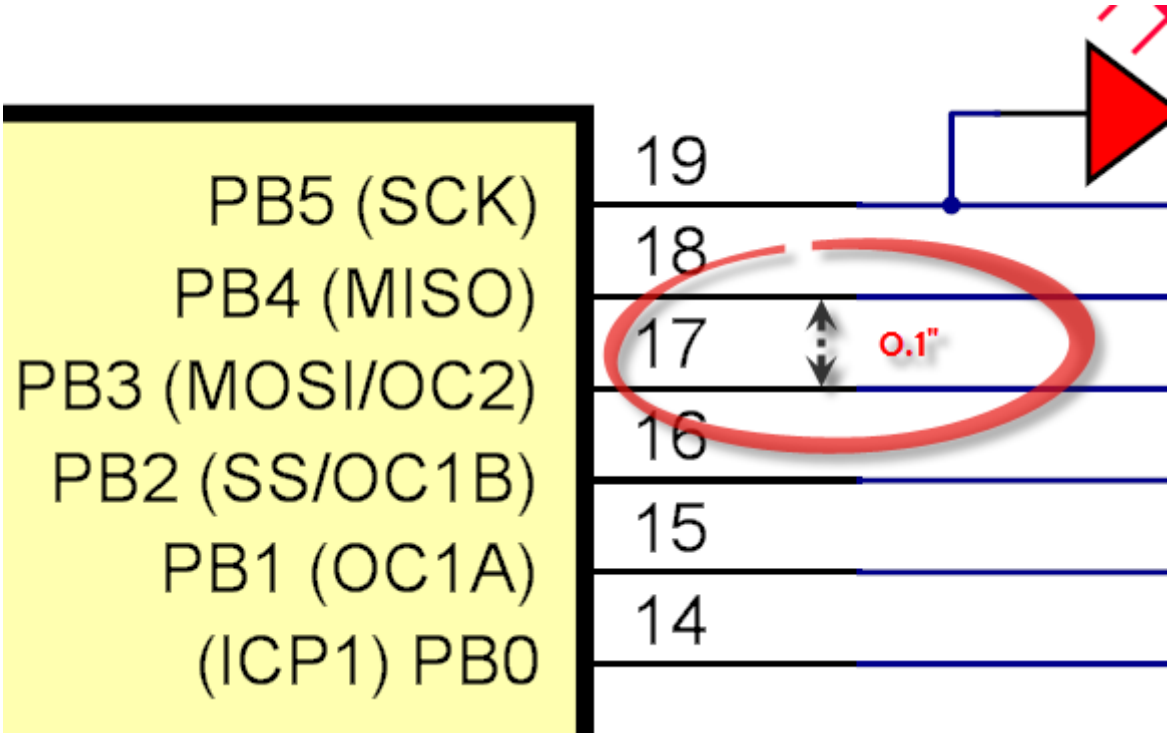
1.2.2.5.9 Symbol and Wire Grid

AutoTRAX DEX has a fixed grid for schematic terminal magnets, schematic terminals and schematic wires and buses. This is 0.05". This is 1.27 mm.

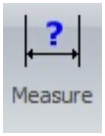
This was chosen for visibility reasons and to ensure it is easy to lay out wires vertically/horizontally with no small snap compensating section. A unit mm setting such as 1mm is too small and 2mm too large.

The AutoTRAX DEX schematic wire auto-router is grid based and uses a grid of 0.05".

You cannot change this. Setting the Snaps per Grid or the Snap Settings has no effect on adding/editing wire, moving terminals or changing the border size/position.



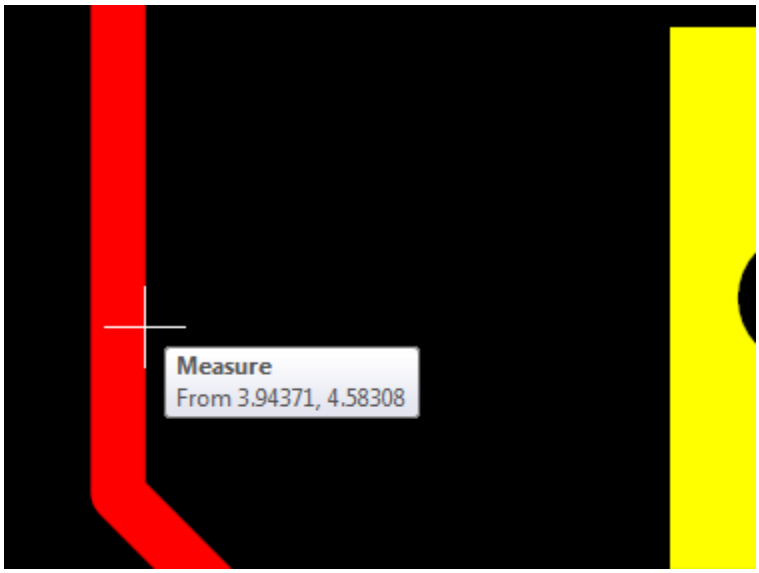
1.2.2.6 Measuring Distances and Angles

To measure distances click the Measure distance button 

in the **Tools→Misc**. ribbon menu.

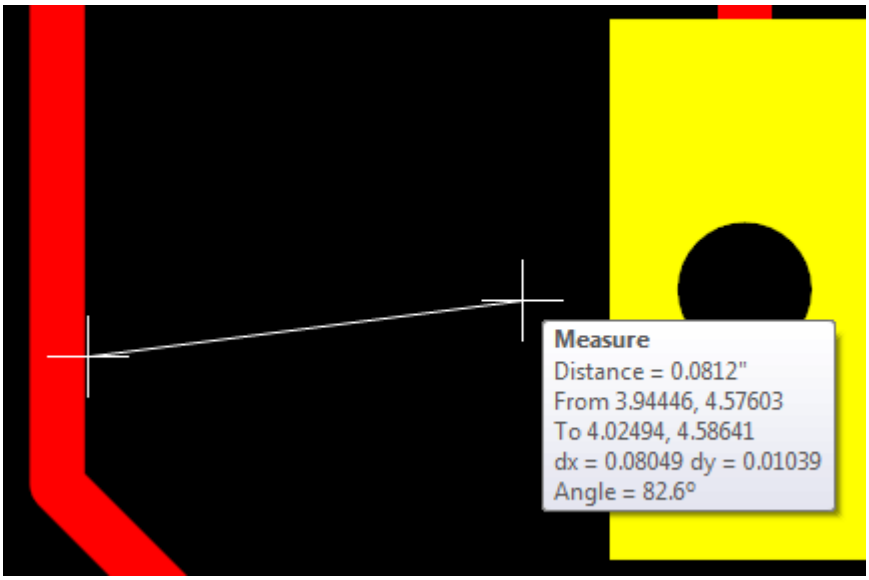
Often in your design you will want to measure the size of an object or the distance between objects. This video show you how to quickly use the measure tool.

1. Move the cursor to the start point from which you wish to begin measuring. You can turn snap on or off by pressing the ***'s'*** key.



Move the cursor to the start point

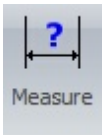
2. **Left-click** to begin measuring. Now, as you drag the cursor, the distance will be displayed. Again you can turn snap on or off by pressing the ***'s'*** key.



Drag the mouse to measure the distance

3. **Left-click** again to end the measurement.

If the Auto-repeat commands feature enabled, additional measurements by repeating the process above without the necessity of clicking the Measure distance



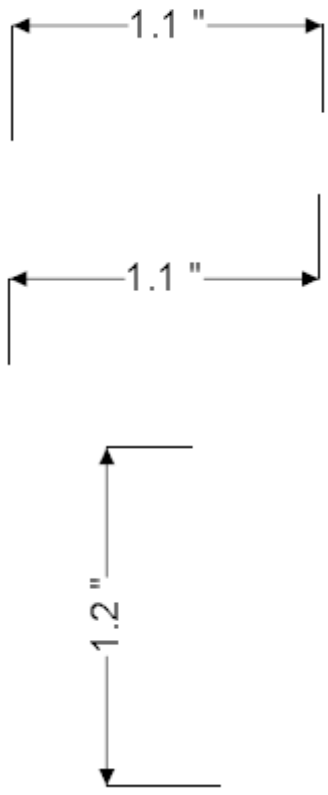
button in the **Tools→Misc**. ribbon menu. If Auto-repeat commands is enabled hit **ESC** to quit measuring.

1.2.2.7 Dimensions

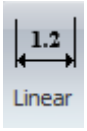
In AutoTRAX DEX a dimension is used to show the distance between 2 points. There are 2 different types of dimensions, linear and aligned dimensions.

Linear Dimensions

Linear dimensions are dimensions that display horizontal or vertical distances.

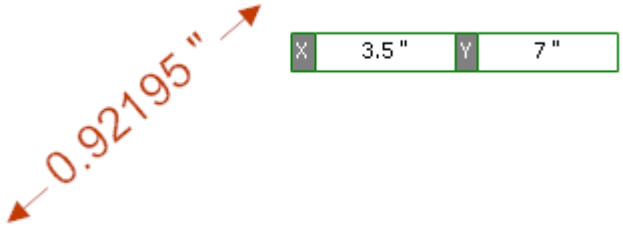


Adding Linear Dimensions

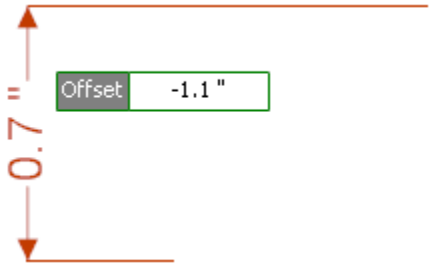
1. To add a linear dimension click the 

in the **Add→Dimension** button group.
2. Now as you move the mouse in the viewport, you will see the start coordinate. ***Left-click*** to define the start point of the dimension or press the **Enter** key and enter the numeric value of the start point.

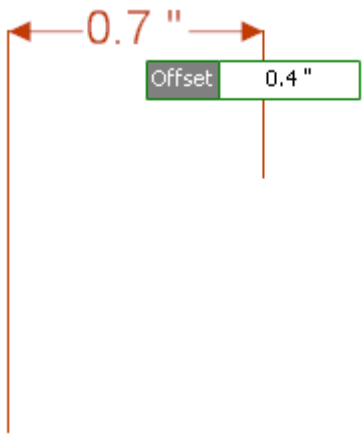
3. Now as you move the mouse the dimension will be drawn and the end point shown.



4. ***Left-click*** to set the end point or press the ***Enter*** key and enter the numeric value of the end point.
5. Now as you move the mouse you will see the dimension offset from the start and end points as shown below. Depending on the position of the mouse, the dimension will show the horizontal or vertical distance. ***Left-click*** again to finally define the dimension or press the ***Enter*** key and enter the numeric value of the offset.



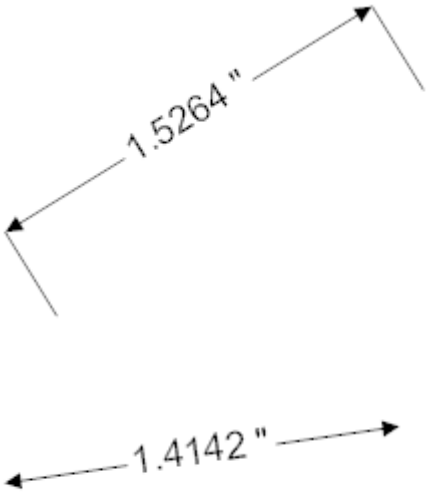
Dimension showing the vertical distance



Dimension showing the horizontal distance

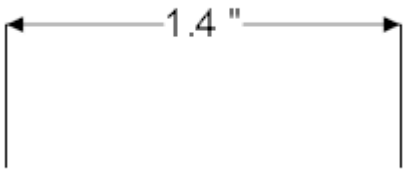
Aligned Dimensions

Aligned dimensions are dimensions that are drawn at an angle and aligned to the dimension being displayed.

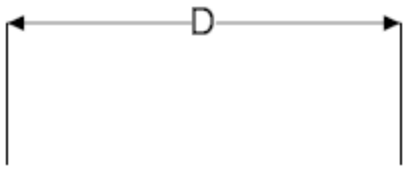


Dimension Labeling

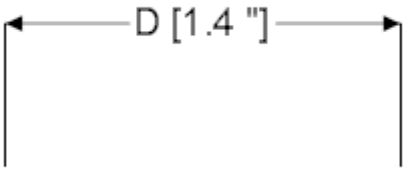
Dimensions can display the value, a label or both.



Just Value



Label only



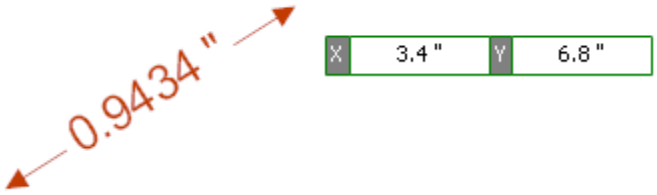
Value and Label

1.2.2.7.1 Adding Aligned Dimensions

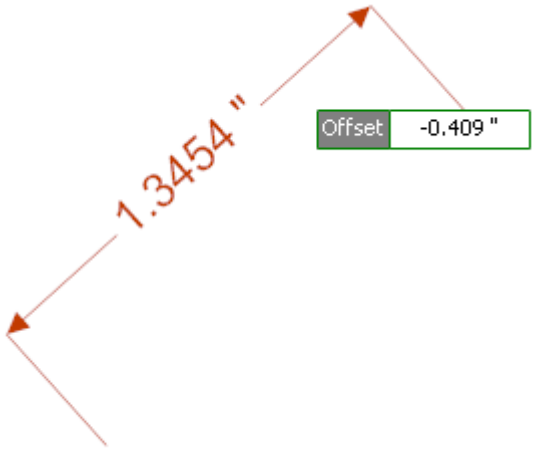
1. To add an aligned dimension click the 

in the **Add→Dimension** button group.
2. Now as you move the mouse in the viewport, you will see the start coordinate. ***Left-click*** to define the start point of the dimension or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to exactly place the starting point.

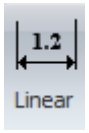
3. Now as you move the mouse, the dimension will be drawn and the end point shown.



4. ***Left-click*** to set the end point or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to exactly place the end point.
5. Now as you move the mouse you will see the dimension offset from the start and end points as shown below. ***Left-click*** again to finally define the dimension or press the ***Enter*** key and enter the numeric value of the offset.

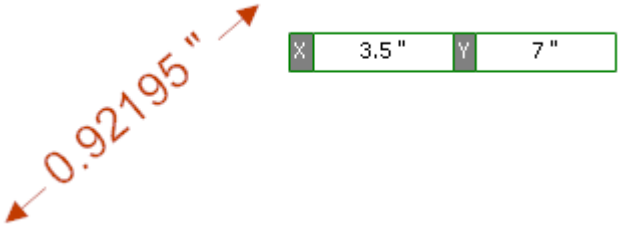


1.2.2.7.2 Adding Linear Dimensions

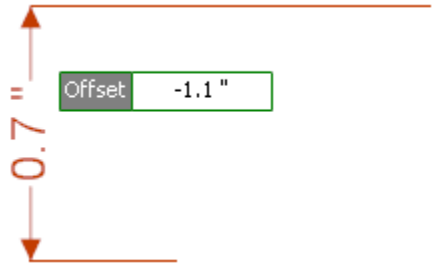
1. To add a linear dimension click the 

in the **Add→Dimension** button group.
2. Now as you move the mouse in the viewport, you will see the start coordinate. ***Left-click*** to define the start point of the dimension or press the **Enter** key and enter the numeric value of the start point.

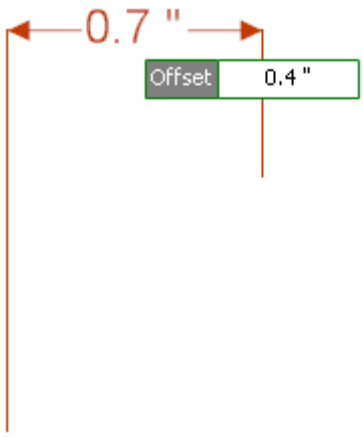
3. Now as you move the mouse the dimension will be drawn and the end point shown.



4. ***Left-click*** to set the end point or press the ***Enter*** key and enter the numeric value of the end point.
5. Now as you move the mouse you will see the dimension offset from the start and end points as shown below. Depending on the position of the mouse, the dimension will show the horizontal or vertical distance. ***Left-click*** again to finally define the dimension or press the ***Enter*** key and enter the numeric value of the offset.



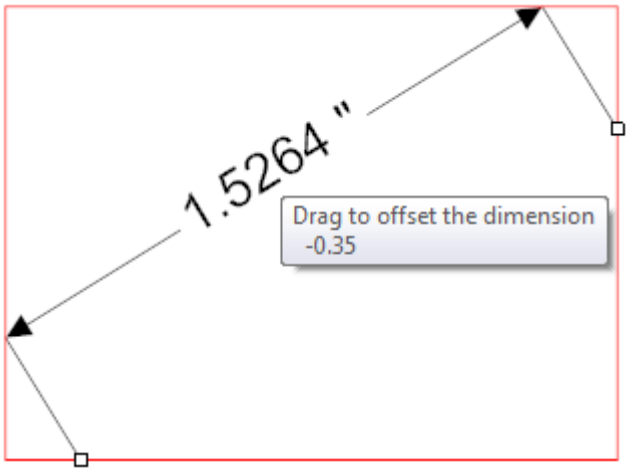
Dimension showing the vertical distance



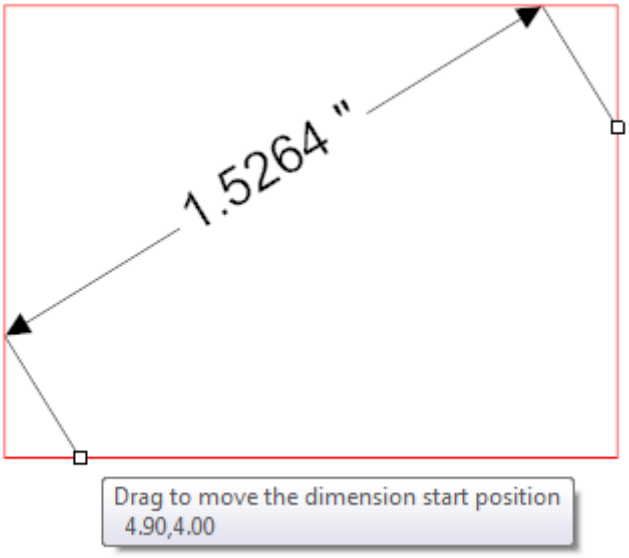
Dimension showing the horizontal distance

1.2.2.7.3 Editing Dimensions

You can edit dimensions by selecting the dimension and dragging the text or text line to change the offset or drag the start/end handles.



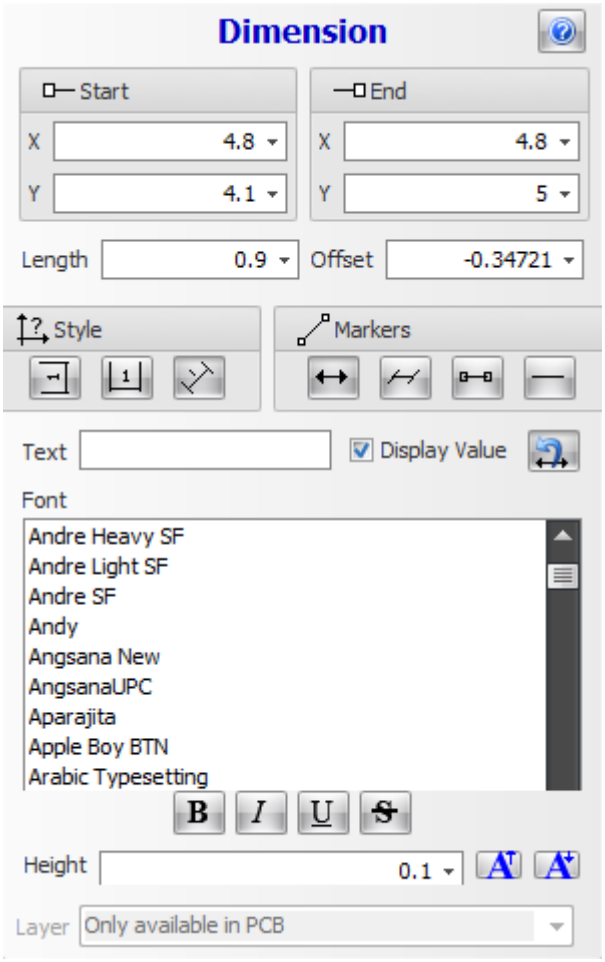
Drag text or test line to change the offset



Drag start or end handle to change dimension points

Using the Properties Panel

In addition you can use the dimensions property dialog in the properties panel.



X

The start or end X coordinate.

Y

The start or end Y coordinate.

Length

The length of the dimension

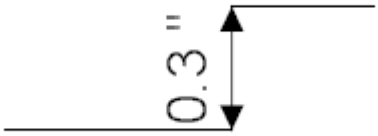
Offset

The distance the dimension is from the start/end points

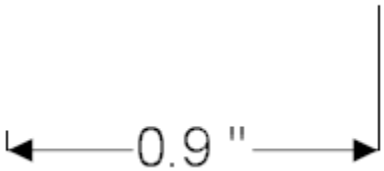
Style

The style of the dimension.

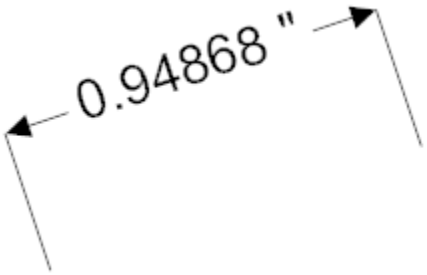
 Vertical



Horizontal



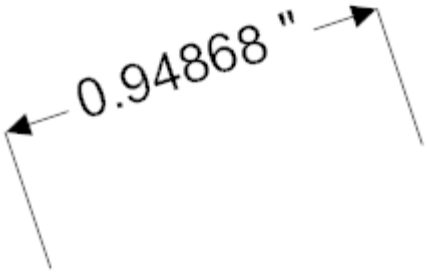
Angled



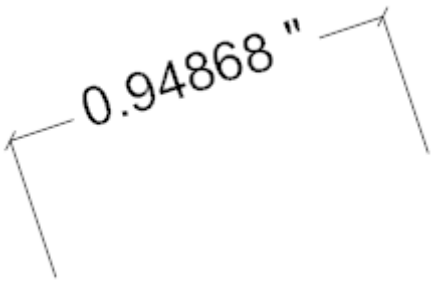
Markers

You can select the graphics markers for the ends of the dimension line.

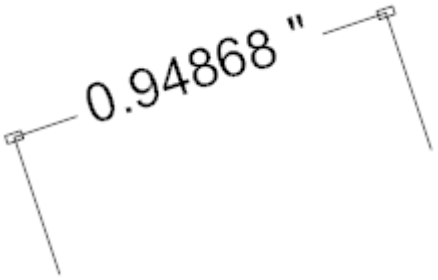
Arrow



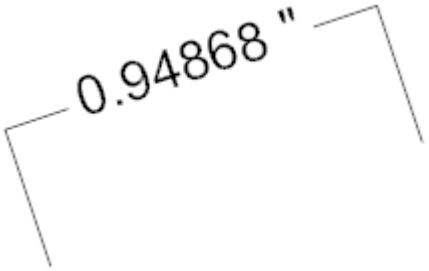
Slash



 Square



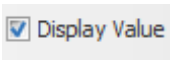
 None



TEXT

Enter an optional text label.

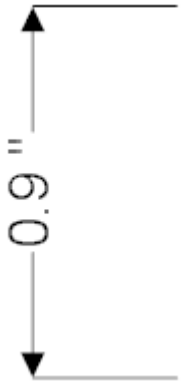
Display Value



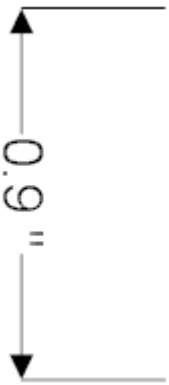
Check to display numeric text for the distance.

Flip

 Check to flip text orientation

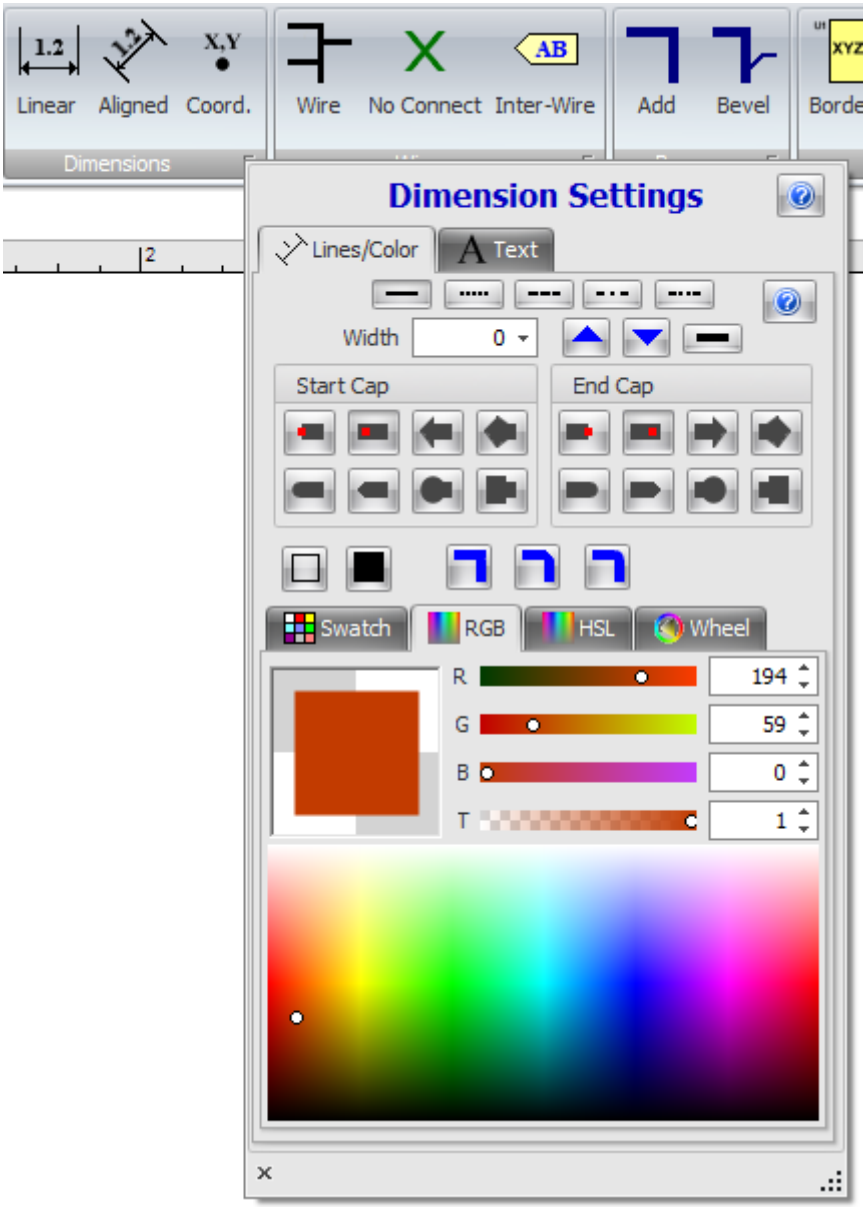


Not Flipped

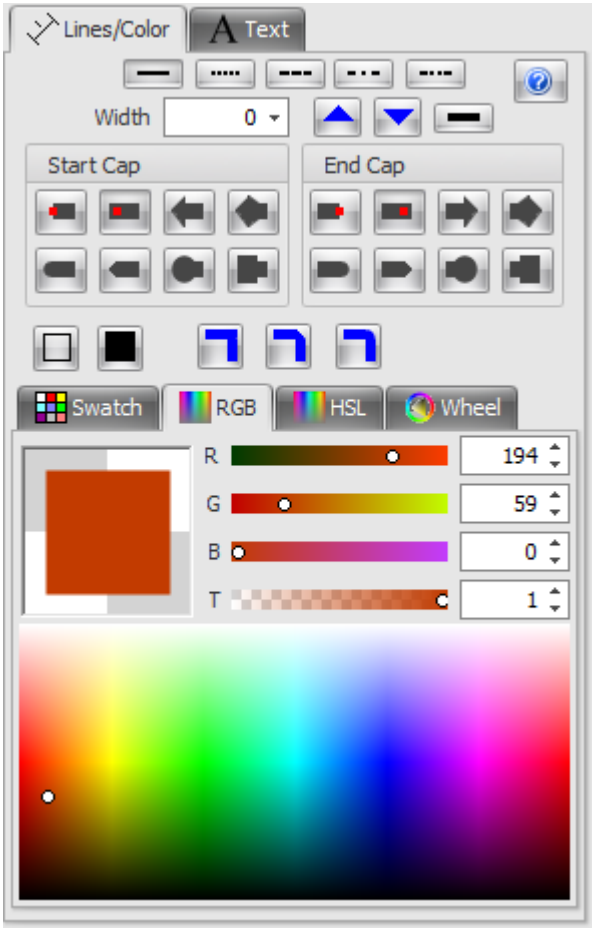


1.2.2.7.4 Dimension Settings

You can set the default properties of dimensions that you will add by clicking on the small button at the bottom left of the **Add→Dimension** button group. You can set the line/text style and text font.

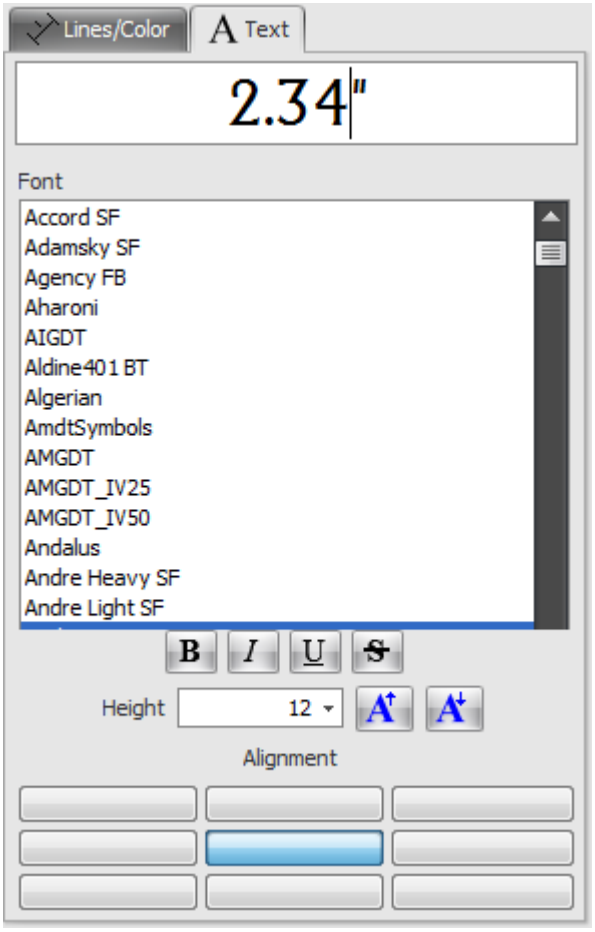


Dimension Settings Pop-up



Dimension line and text style and color

See Line Styles



Dimension text font

See Fonts

1.2.3 Graphics

You can add graphical elements to AutoTRAX DEX.

In schematics graphical elements have no electrical significance, they are not wire connections and have no impact on the PCB.

On PCBs, if you add graphical items to electrical layers, e.g. the top or bottom copper layer, then copper will be placed there.

You can add the following graphics:

- Lines
- Arcs
- Polylines
- Polygons

- Rectangles
- Curves
- Ellipses and Circles
- Text
- Notes
- Pictures

1.2.3.1 Arcs

In AutoTRAX DEX you can add graphical arcs. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Arcs have a width that can be zero for an infinitely thin arc. However, if you add arcs to a PCB you need to make sure their width is at least the size of the manufacturing minimum width.

Arcs also have color and can be semi-transparent. It is even possible to define an invisible arc. If you have snap to objects enabled, objects will snap to invisible arcs.

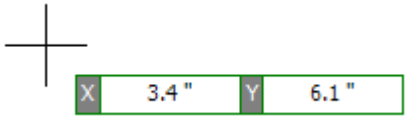
Arcs can also have a line style as shown below.

1.2.3.1.1 Adding Arcs

To add an arc to a graphical sheet, click on the  button in the **Add→Shapes** ribbon menu button group.

To set the Line Styles before adding arcs, use the Default Graphics Settings

Move the mouse inside a graphical sheet's viewport. You will see the arc center cross follow the mouse.

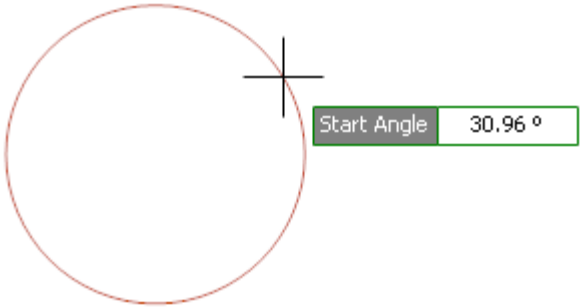


Mouse Center Point

Left-click when the center cross is where you want to start the arc or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab***, and then the Y value and ***Enter/Space/Tab***.

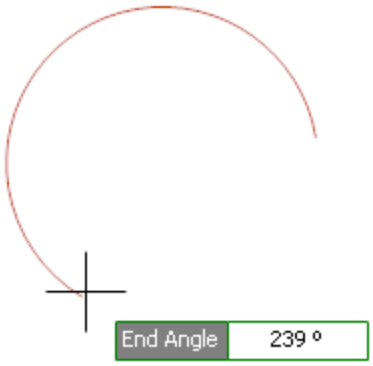
You will next be prompted for the start angle of the arc. Moving the mouse defines the arc's diameter and displays the start angle. Left-click to define the start angle of

the arc or press the ***Enter/Space/Tab*** key followed by the start angle numerical value in degrees.



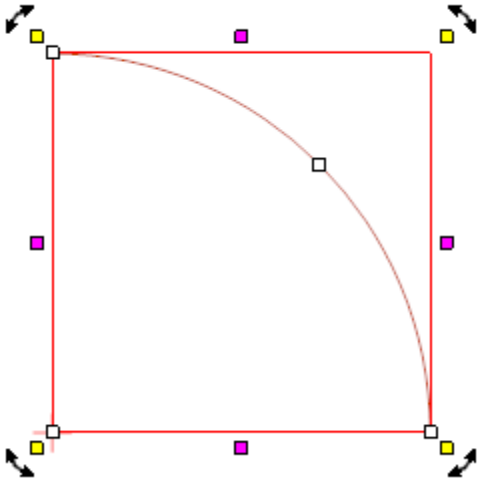
Arc diameter and start angle

Now move the mouse to define the arc's end point . Finally ***left-click*** to define the arc or press the ***Enter/Space/Tab*** key followed by the end angle numerical value in degrees.



1.2.3.1.2 Editing Arcs

To edit an arc first select it.



Drag either of the arc's end manipulators(□) to change the start/end angle of the arc.

Drag the arc's diameter manipulators(□) to change the diameter of the arc

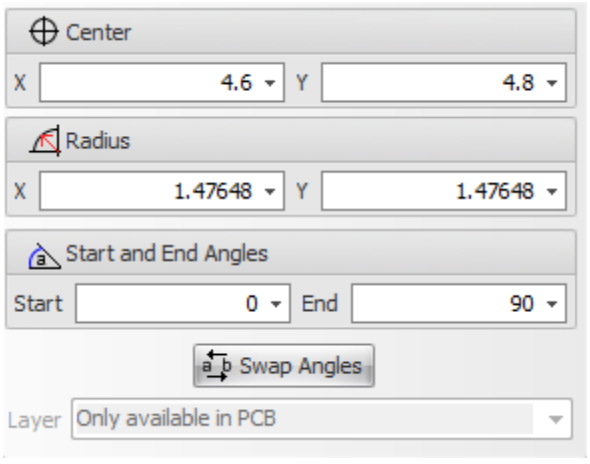
Drag any rotate manipulator to rotate the arc.

Drag any scale manipulator to re-size the arc.

Drag the arc to move it.

Arc Properties Dialog

You can also edit the parameters of the arc using its properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Click the button to show this help topic.

Center

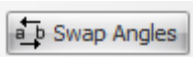
Enter the X and Y coordinates for the center of the arc.

Radius

Enter the X and Y radius for the arc.

Start and End Angle

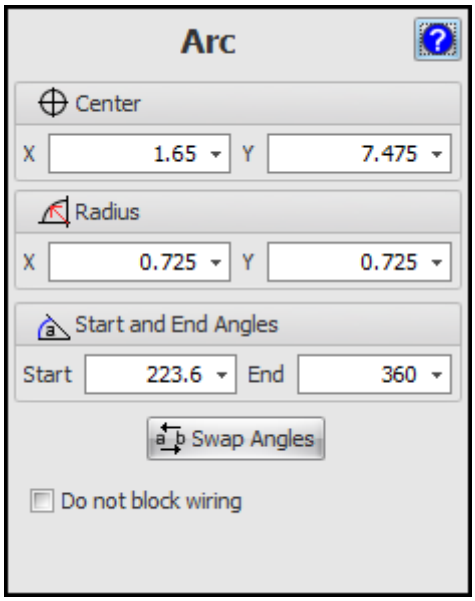
Enter the start and end angle for the arc in degrees.

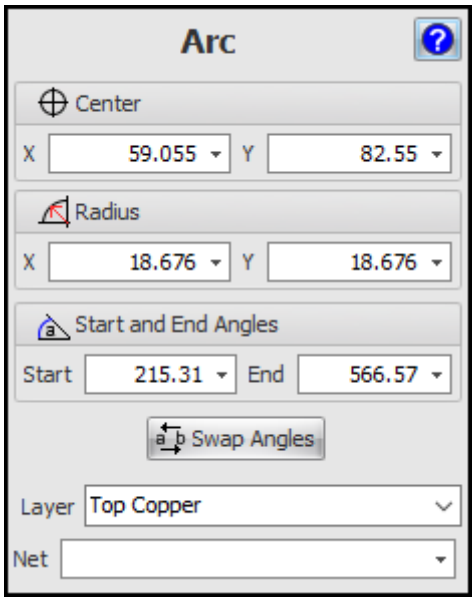
Click 

to swap the start and end angles.

Use the Layers drop down to change the layer of the arc.

1.2.3.1.3 Arc Editor





1.2.3.2 Blocks

Blocks are objects imported from DXF files.

1.2.3.3 Curves

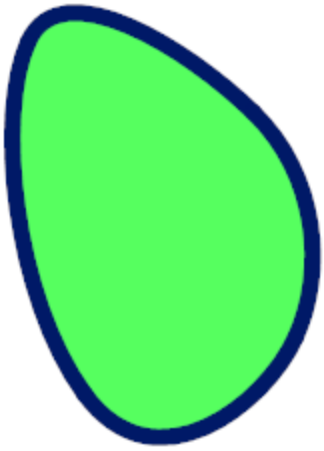
In AutoTRAX DEX you can add graphical curves. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Curves have a width that can be zero for an infinitely thin curve. However if you add curves to a PCB, you need to make sure their width is at least the size of the manufacturing minimum width.

Curves also have color and can be semi-transparent. It is even possible to define an invisible curve. If you have snap to objects enabled, objects will snap to invisible curves.

Open

Closed



Closed with Border and Filled

Closed with No Border and Filled

1.2.3.3.1 Adding Curves

You can add open and closed curves.

To set the Fill Styles and Line Styles before adding curves use the Default Graphics Settings.

An Open Curve

A Closed Curve

To add an open curve to a graphical sheet, click on the  button in the **Add→Shapes** ribbon menu button group.

Curves in AutoTRAX DEX are piecewise bezier curves.

A Bezier curve is a mathematical curve used in computer graphics and related fields to model smooth curves. It is defined by a set of control points, which influence the shape of the curve.

A piecewise Bezier curve is a Bezier curve that is made up of several Bezier curve segments, each defined by its own set of control points. These segments are joined together end-to-end to form a smooth curve.

The advantage of using a piecewise Bezier curve is that it allows for greater flexibility in modeling curves with varying degrees of curvature. By using multiple segments, each with their own control points, it is possible to create a curve that has both sharp corners and smooth curves.

To create a piecewise Bezier curve, one simply needs to define the control points for each segment and then join them together. The resulting curve will be a smooth, continuous curve that can be used for a wide range of applications in computer graphics and related fields.

To add a closed curve to a graphical sheet, click on one of the  buttons in the Add→Shapes ribbon menu button group.

 Adds a hollow curve.

 Adds a filled curve with border.

 Adds a filled curve without a border.

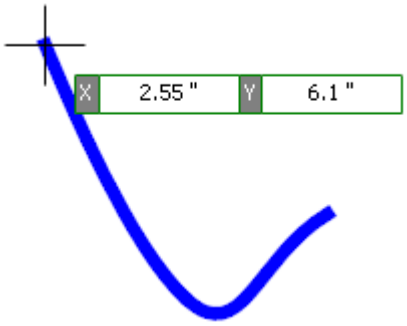
Move the mouse inside the graphical sheet's viewport. You will see the point cross follow the mouse. ***Left-click*** when the point cross is where you want to start a curve or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.

Move to the start of the curve

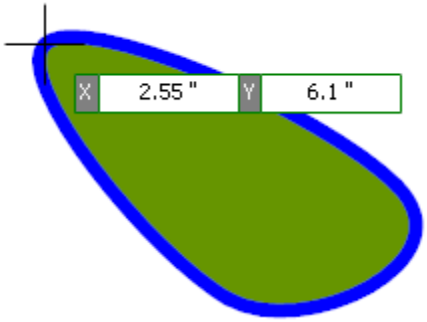
Now as you mouse the mouse the first segment of the curve will be defined.

Move to end of first segment

Left-click to define the second point for the curve or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to exactly place the second point. Now as you move the mouse the curves second segment is being defined.



Open Curve



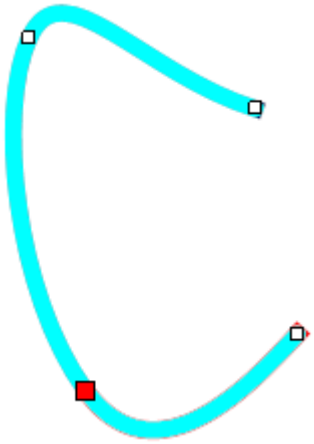
Closed Curve

Continue this to complete the curve. If you are adding an open curve and you place the cursor over the first point of the curve and ***left-click***, the curve will be automatically closed and filled.

Double-click to end the curve with the next segment to be placed or press the ***ESC*** key to end the curve with the last completed curve segment.

1.2.3.3.2 Editing Curves

To edit a curve first select it.



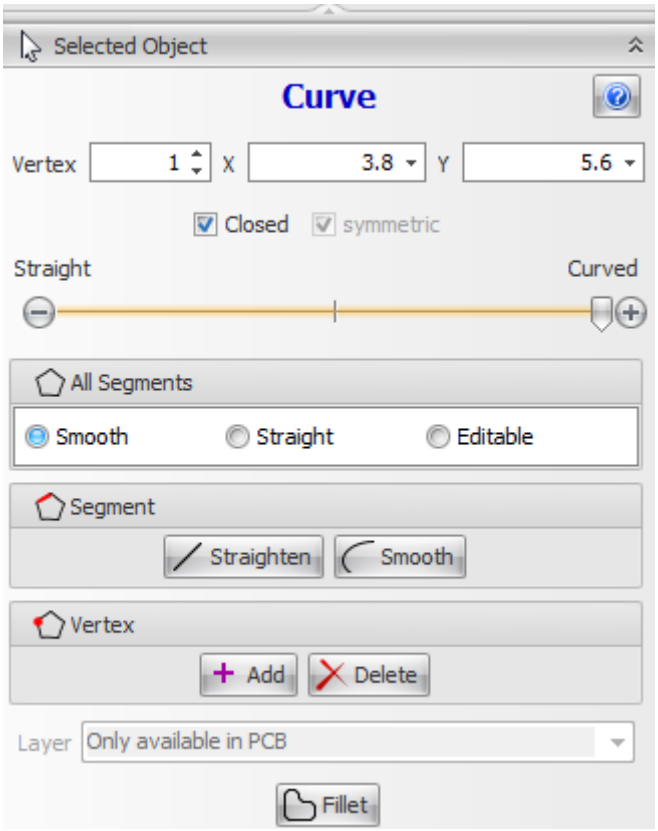
A selected curve

Drag any of the manipulators(□) to change the corners or ends.

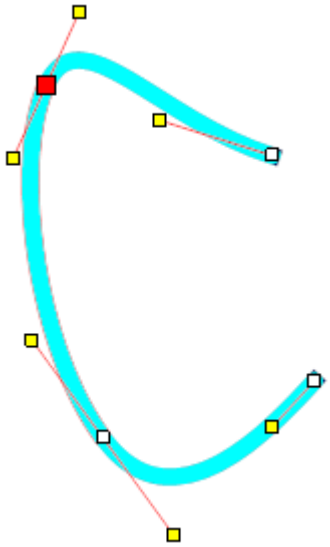
Drag the curve to move it.

Curve Properties Dialog

You can also edit the parameters of the curve using its Properties Panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Curve Properties Dialog



Editable Curve

Click the  button to show this help topic.

Vertex

This is the selected vertex/segment. The selected vertex is drawn as a solid red square. The selected segment id drawn in red.

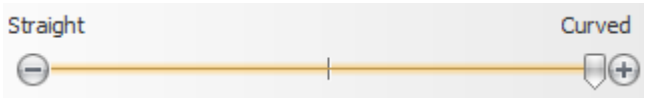
X

The X coordinate of the selected vertex.

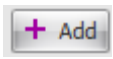
Y

The Y coordinate of the selected vertex.

Tension

Slide the 

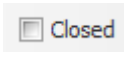
slider to change the 'curvature' of the curve.

Click 

to add a vertex.

Click 

to delete the selected vertex.

Check/uncheck 

close/open the polygon.

All Segments

Check the Smooth button to smooth the entire curve.

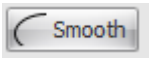
Check the Straight button to straighten the entire curve.

Check the Editable button to make the entire curve editable.

Segment

Click the 

button to straighten the selected segment.

Click the 

button to smooth the selected segment.

Layer

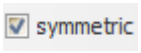
Use this drop-down to set the layer for the rectangle.

Click 

to fillet the polygon

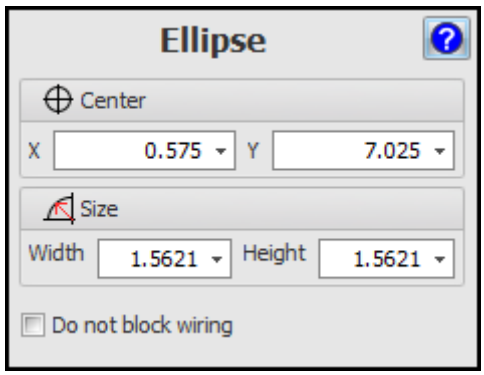
Editable Curves

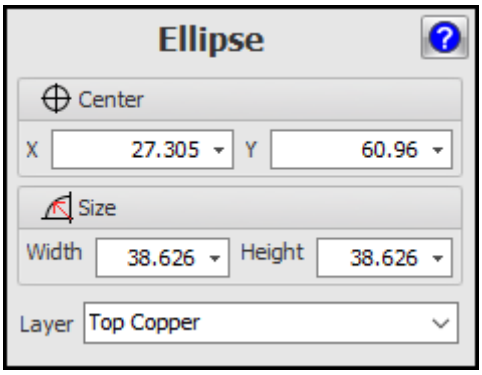
Drag any of the control points  to change the shape.

Uncheck the 

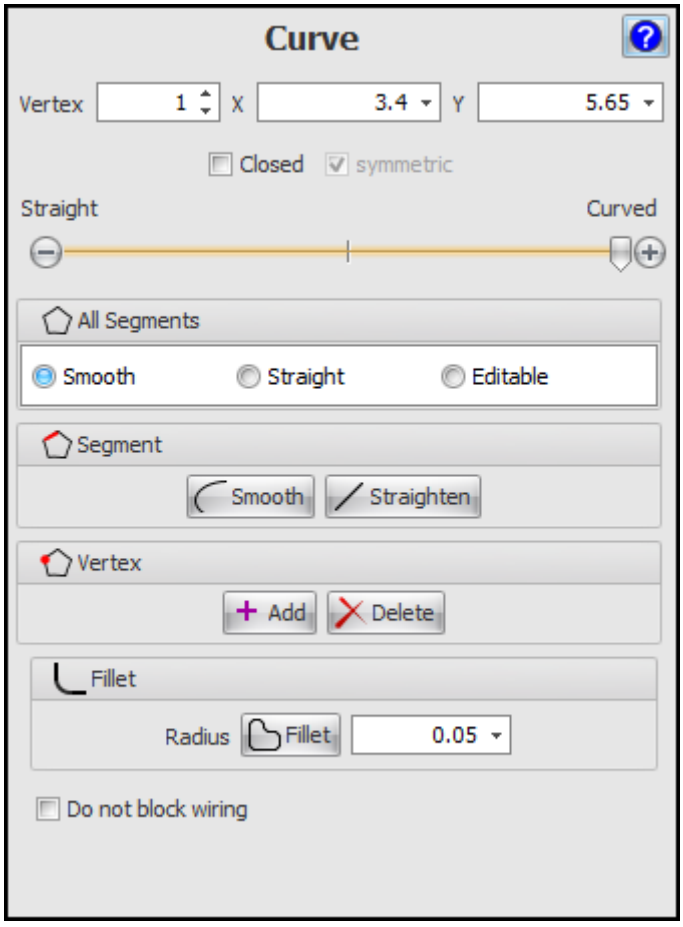
button to make the control points for the vertex independent of each other.

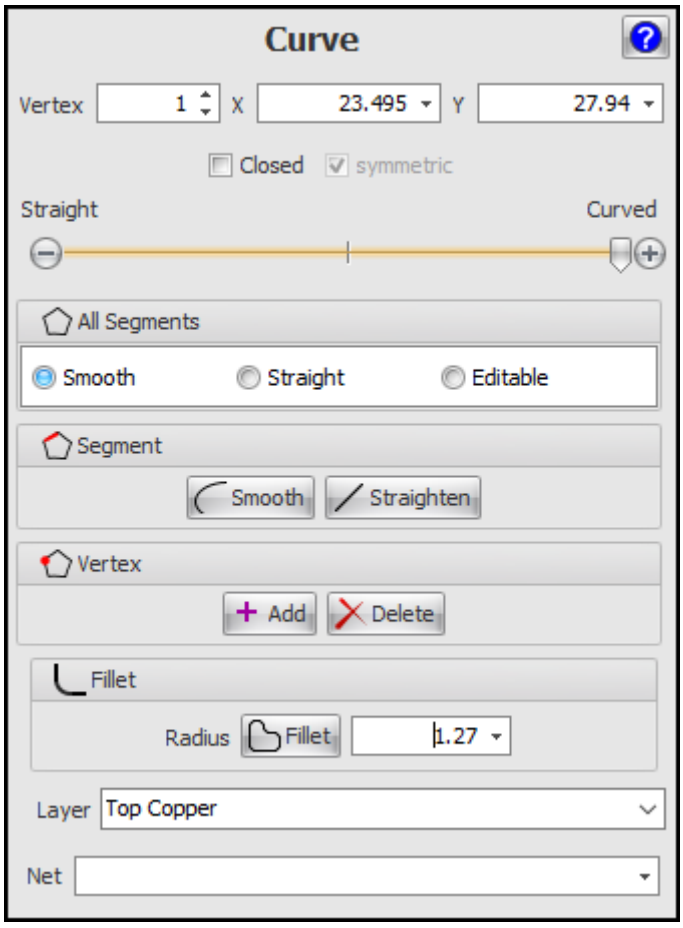
1.2.3.3.3 Circle/Ellipse Editor





1.2.3.3.4 Curve Editors

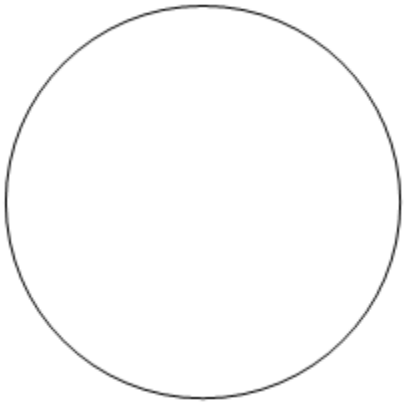




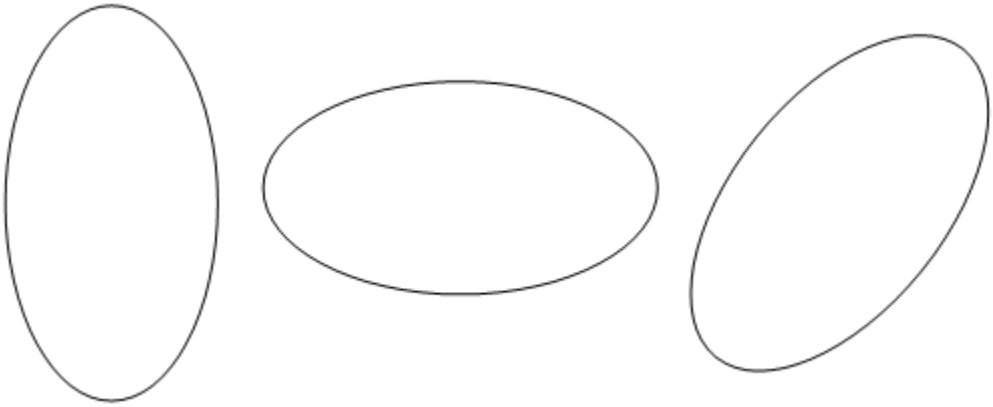
1.2.3.4 Ellipses and Circles

In AutoTRAX DEX you can add graphical ellipses.

A circle is an ellipse where the horizontal and vertical dimensions are the same.



Circle



Ellipses

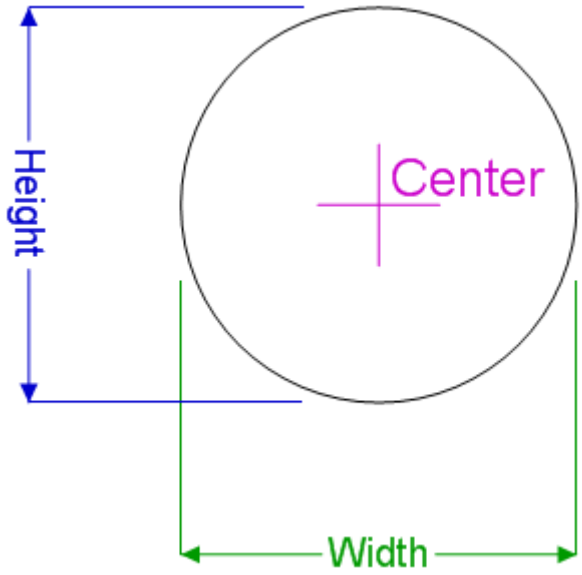
In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Ellipses have a width that can be zero for an infinitely thin ellipse. However if you add ellipses to a PCB you need to make sure their width is at least the size of the manufacturing minimum width.

Ellipses also have color and can be semi-transparent. It is even possible to define an invisible ellipse. If you have snap to objects enabled, objects will snap to invisible ellipses.

Ellipses can also have an ellipse style as shown below.

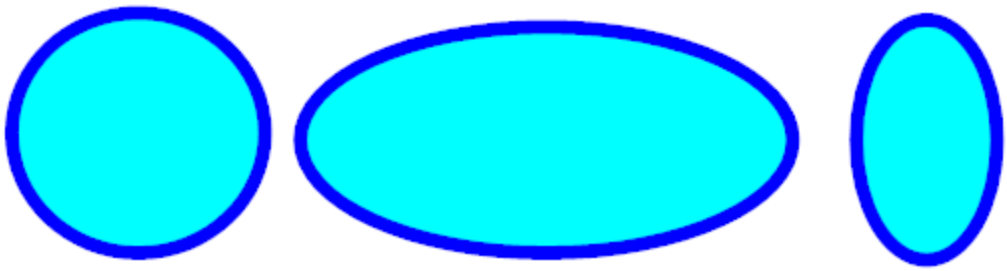
Ellipses are defined by their center and the size. The size is defined by a horizontal width and a vertical height.



1.2.3.4.1 Adding Ellipses and Circles

You can add ellipses or circles. A circle is an ellipse with the x and y radii being the same.

To set the Fill Styles and Line Styles before adding ellipses use the Default Graphics Settings



Ellipses/Circles

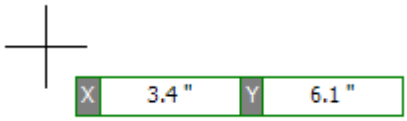
To add an ellipse to a graphical sheet click on one of the  buttons in the **Add→Shapes** ribbon menu button group.

 Adds a hollow ellipse.

 Adds a filled ellipse with border.

 Adds a filled ellipse without a border.

Move the mouse inside the graphical sheet's viewport. You will see the point cross follow the mouse. ***Left-click*** when the point cross is where you want to start an ellipse or circle or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.



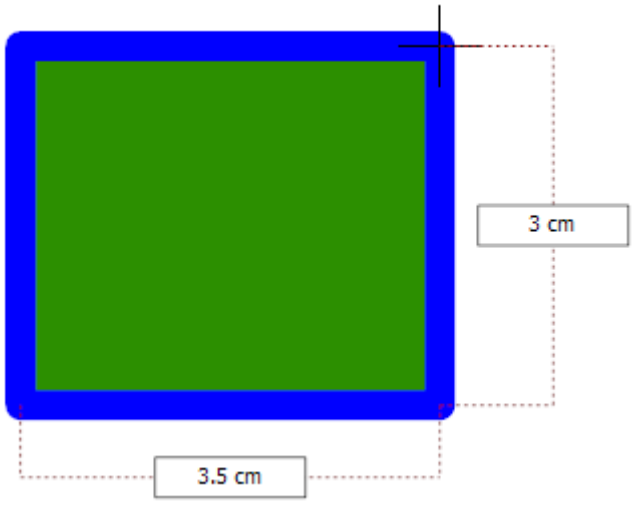
Now as you move the mouse the ellipse will change in size and aspect ratio.

If you hold down the ***Shift*** key as you move the mouse, the ellipse will be centered at the first point.

If you hold down the ***CTRL*** key as you move the mouse, the ellipse will be circle with width and height the same.

If you hold down both the ***Shift*** and ***CTRL*** keys simultaneously as you move the mouse, the circle will be centered at the first point as you move the mouse, with the width and the height being the same.

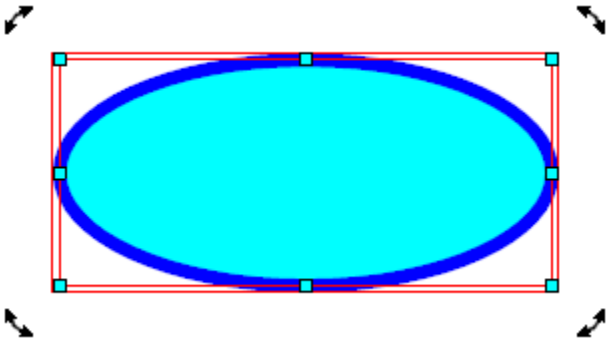
Left-click when the ellipse/circle is the shape you want or press the ***Enter*** key followed by the numeric values for the width and height of the ellipse.



Dragging the mouse defines the ellipses size

1.2.3.4.2 Editing Ellipses and Circles

To edit an ellipse or a circle first select it.



A selected ellipse

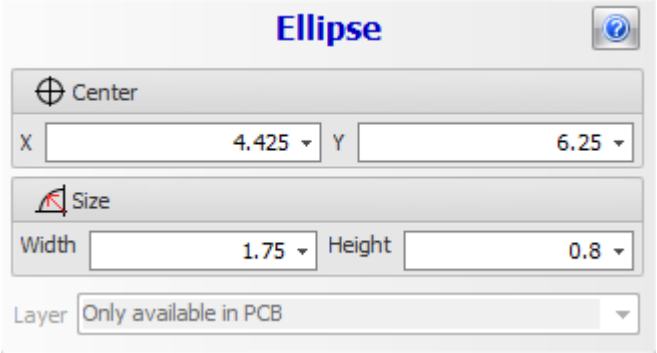
Drag any rotate manipulator to rotate the ellipse.

Drag any scale manipulator to re-size the ellipse.

Drag the ellipse/circle to move it.

Ellipse Properties Dialog

You can also edit the parameters of the ellipse using its properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Ellipse Properties Dialog

X

The X coordinate of the center of the ellipse.

Y

The Y coordinate of the center of the ellipse

Width

The width of the ellipse.

Height

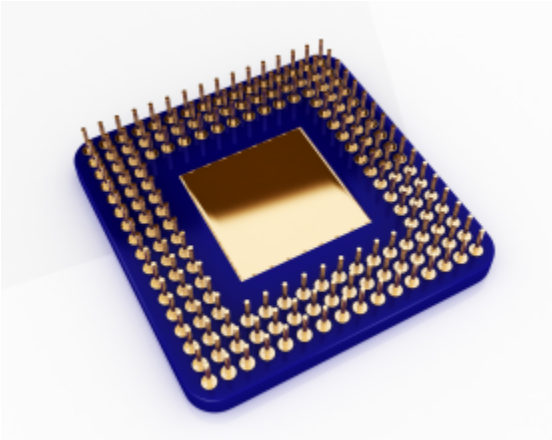
The height of the ellipse

Layer

Use this drop-down to set the layer for the ellipse.

1.2.3.5 Images

In AutoTRAX DEX you can add graphical images. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.



Image

1.2.3.5.1 Adding Images

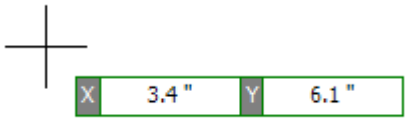
You can pictures/images to your design.

An image with transparency

To add an image to a graphical sheet click on one of the  button in the **Add→Shapes** ribbon menu button group.

An Open Image file dialog will be display. Use this to select the image file to be inserted and click the **Open** button.

Move the mouse inside a graphical sheet's viewport. You will see the point cross follow the mouse. ***Left-click*** mouse button when the point cross is where you want to start the image or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.



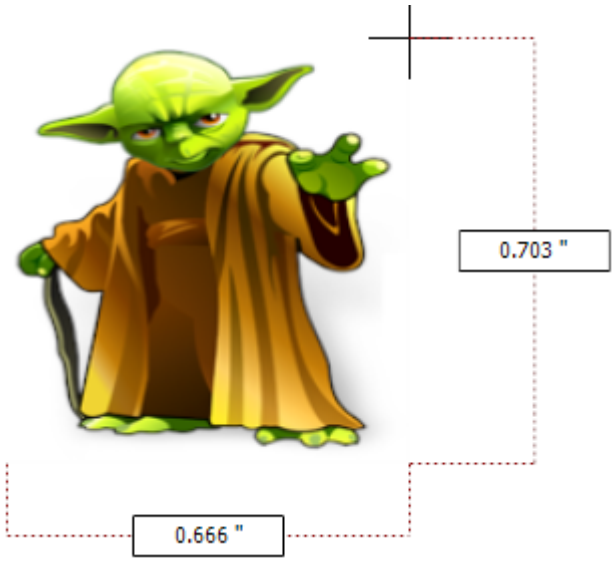
Now as you move the mouse the image will change in size and aspect ratio.

If you hold down the ***Shift*** key as you move the mouse, the image will be centered at the first point.

If you hold down the ***CTRL*** key as you move the mouse, the image will be square with width and height the same.

If you hold down both the ***Shift*** and ***CTRL*** keys simultaneously as you move the mouse, the image will be centered at the first point and will be square with the width and the height the same.

Alternately, you can add an image with a simple click, hold, and drag. When the point cross is where you want to start the image, ***left-click*** and hold the left button. **Left-click and hold** to re-size the image and release the button; the image will be placed. In this mode, if you hit ***Enter*** before releasing the left button, you will be able to enter exact X and Y numerical values for the width and heights respectively. Note, using the ***Shift*** and ***CTRL*** keys as described above also works with this alternate method.



Dragging the mouse defines the image size

1.2.3.5.2 Editing Images

To edit an image first select it.

A selected image

Drag either of the images corner manipulators( ) to change the corners or ends.

Drag any rotate manipulator to rotate the image.

Drag any scale manipulator to re-size the image.

Drag the image to move it.

Image Properties Dialog

You can also edit the parameters of the image using its Properties Panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.

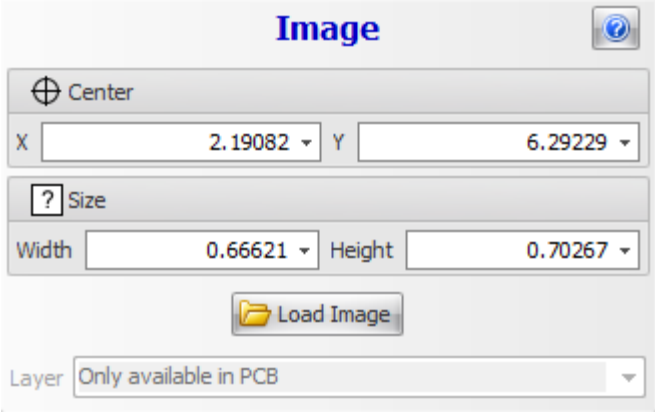


Image Properties Dialog

X

The X coordinate of the center of the image.

Y

The Y coordinate of the center of the image

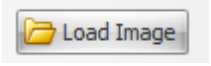
Width

The width of the image.

Height

The height of the image

Load Image

Click the 

button to load a different image.

Layer

Use this drop-down to set the layer for the image.

1.2.3.5.3 Image Editor

1.2.3.6 Lines

In AutoTRAX DEX you can add graphical lines. In schematics they have no electrical significance. However in PCBs, they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

A mathematical line as infinite length and zero thickness correcting.

In AutoTRAX a line is more like a mathematical line segment in that it as a start and endpoint. A line also has:

- thickness
- color
- line style
- optional start and end caps.

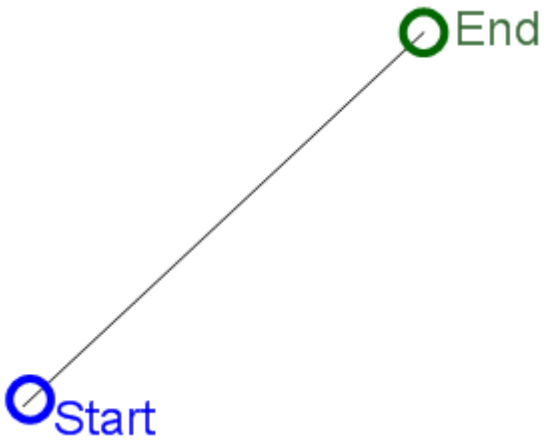
To add a line:

1. First select the Add ribbon tab if it is not already selected.
2. Next click on the add line button.
3. Now move the mouse so the cursor is positioned in the viewport where you want the line to start. As you move the mouse you will see the top coordinate displayed in the top status bar.
4. Hold down the left mouse button and drag the mouse. As you move the mouse you will see the line take shape. You will see the end position of the line displayed in the top status bar along with the length and angle of the line.
5. Release the left mouse button to end adding the line to the schematic.

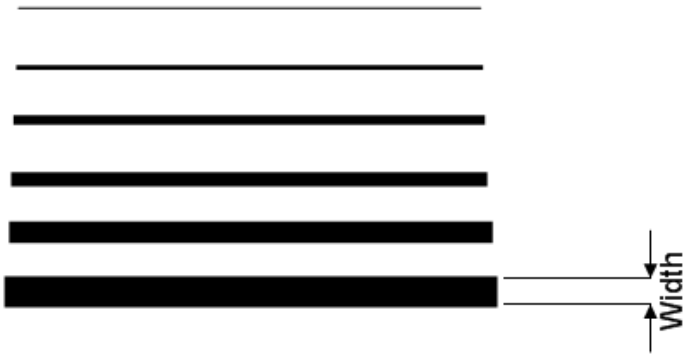
Lines have a width that can be zero for an infinitely thin line. However, if you add lines to a PCB, you need to make sure their width is at least the size of the manufacturing minimum width.

Lines also have color and can be semi-transparent. It is even possible to define an invisible line. If you have snap to objects enabled, objects will snap to invisible lines.

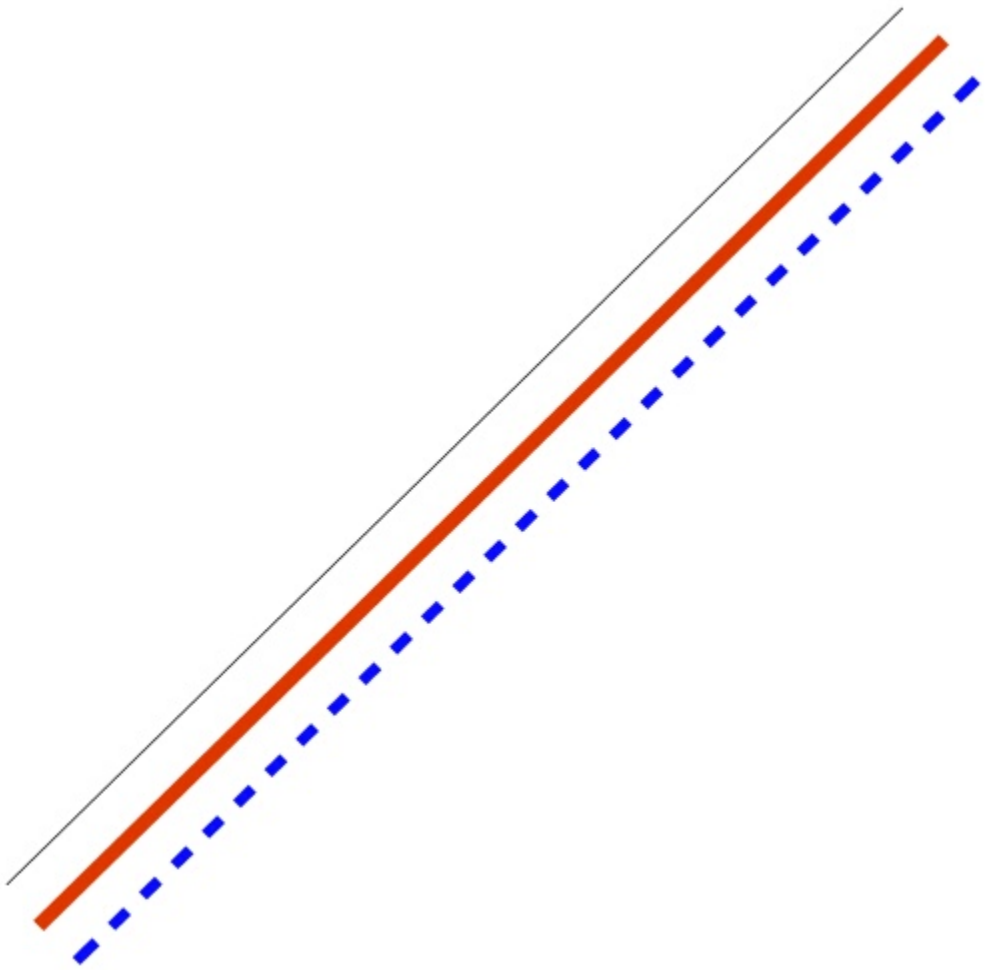
Lines are defined by 2 points, the start point of the line and the end point of the line.



Lines can have different widths.

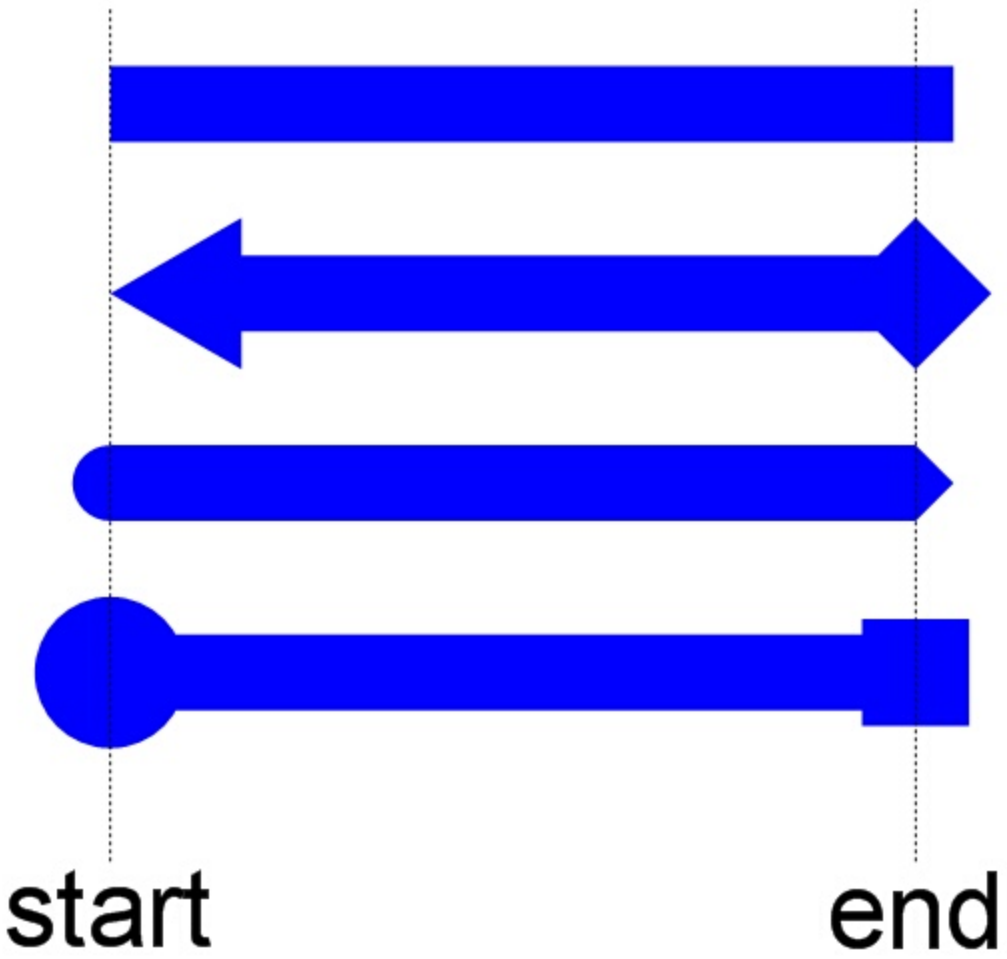


Lines can also have a line style as shown below.



Different line thicknesses and styles

In addition, you can also set the line cap style for lines to any of the cap styles shown below. Each end of the line can have a different style.



Line end caps

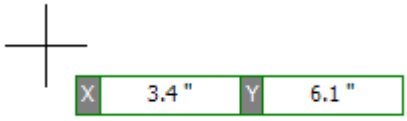
If you have snap to objects enabled, then as you add other objects they can snap to either end of a line or the center of the line.

1.2.3.6.1 Adding Lines

In AutoTRAX DEX you can add graphical lines to both schematics and PCBs.

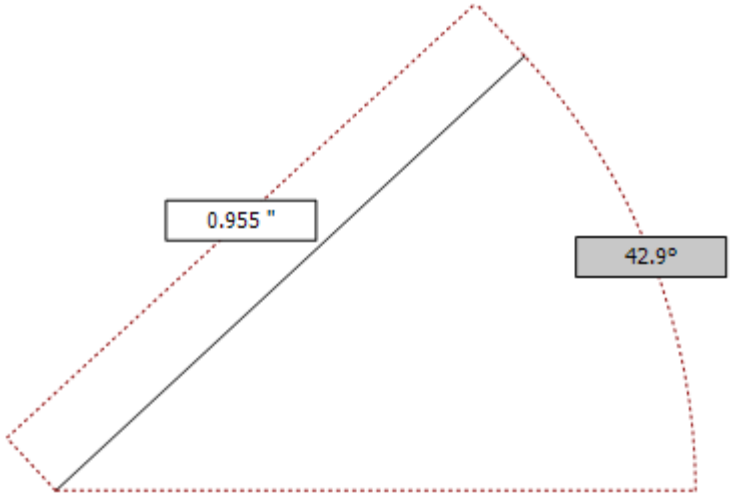
To set the Line Styles before adding lines, use the Default Graphics Settings

1. To add a line, click on the line button in the **Add→Shapes** menu.
2. Now move the mouse to where you want to start the line then:
   a. ***Left-click*** when the start cross is where you want to start of the line to be or
   b. press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.



Start point cursor and input boxes

3. Now as you move the mouse you will see the line being drawn and the end point will follow the cursor. If snap is enabled, both the start and the end points will snap to the grid. At any time you can turn snap on and off by pressing the ***'s'*** key. As you move the mouse, the length and angle of the line displayed. Left click to define the line's end point or press the ***Enter*** key followed by the length, ***Enter***, the angle, and ***Enter*** to exactly define the end point of line. Note that as you move the end of the line, if smart pan is enabled, the viewport will automatically pan so the end point is always visible. If possible, the original view will be restored.



End point cursor and input boxes

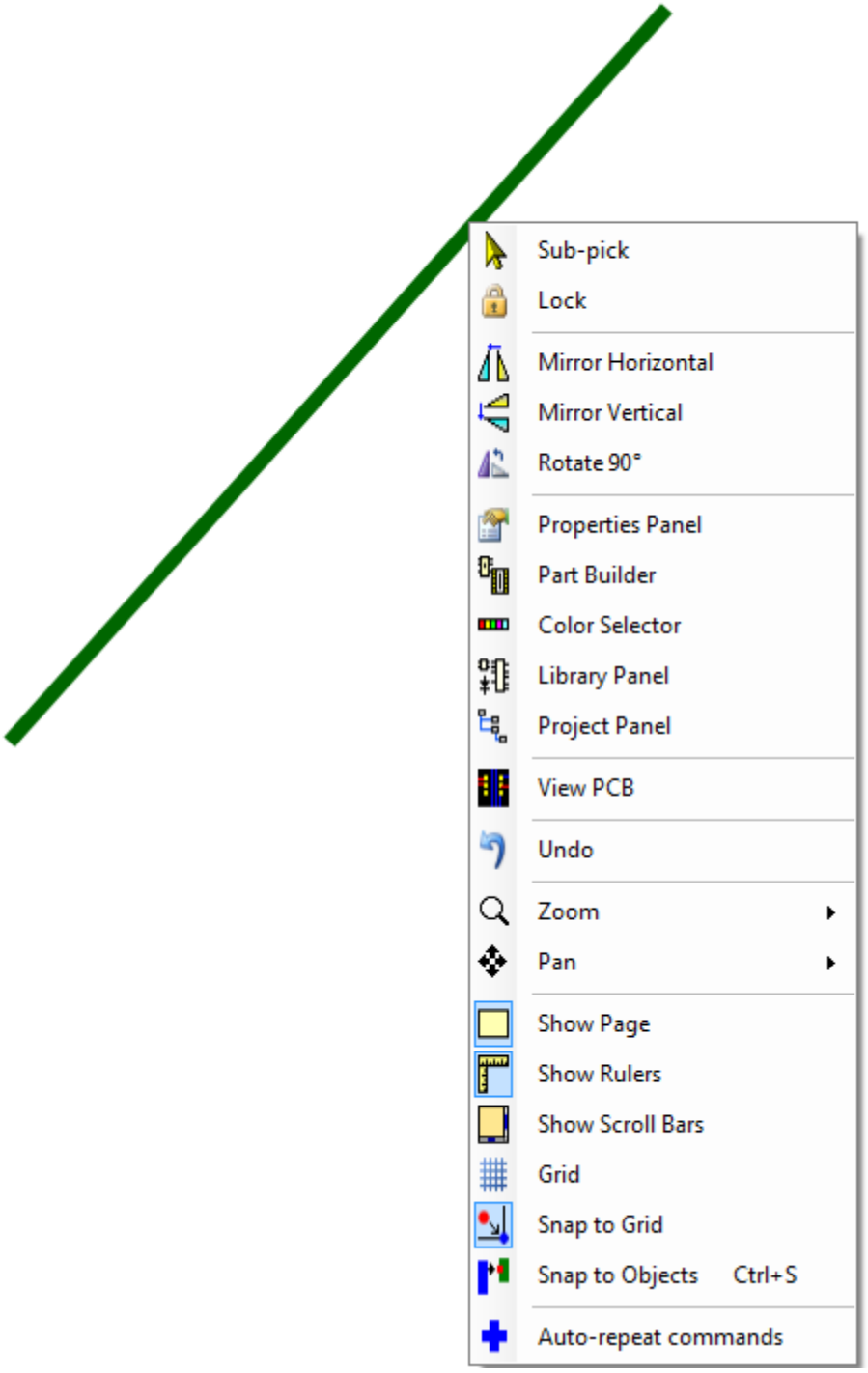
To turn smart-pan on or off, click on the smart-pan button on the status bar.

If you have Auto-repeat commands enabled, then you will be prompted to enter the start point for another line. To cancel adding more lines press either the ***ESC*** key or ***right-click*** and select cancel from the context menu.

Alternately, you can create a line by ***left-clicking and holding*** at the location for the start of the line, moving to the end location for the line and releasing the left button. The line is then added to your design.

1.2.3.6.2 Context Menu

If you ***right-click*** on a line that you have selected, a context sensitive menu will be displayed.

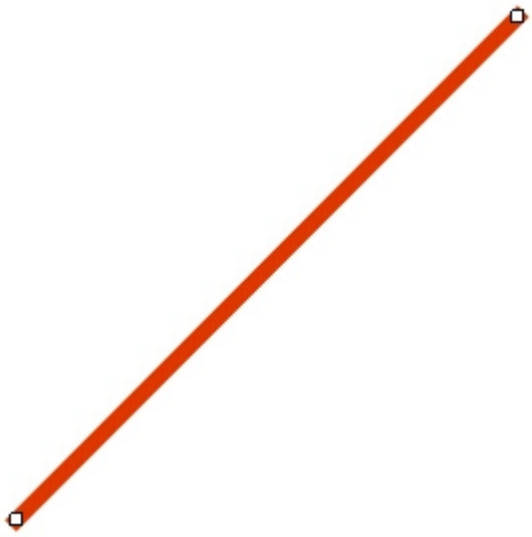


right-click context menu

1.2.3.6.3 Editing Lines

To edit a line first select it.

When a line is selected you will see 2 manipulator points, one at the start of the end and another at the end of the line.



A selected line showing the manipulator points at both ends

Click and hold either point and drag it to set the start or end of the line. If snap is enabled then the end points of the line will snap to the grid as the ends are moved.

To move a line keeping its angle to the horizontal constant, ***left-click*** anywhere on the line other than on either of the end manipulators and drag the line to its new position. If snap is enabled then the line will move in steps as defined by the grid; in this case the end points may not snap to the grid. To snap the end points to the grid, drag the end points individually.

As you drag the end points they may snap to other objects if snap to objects is enabled.

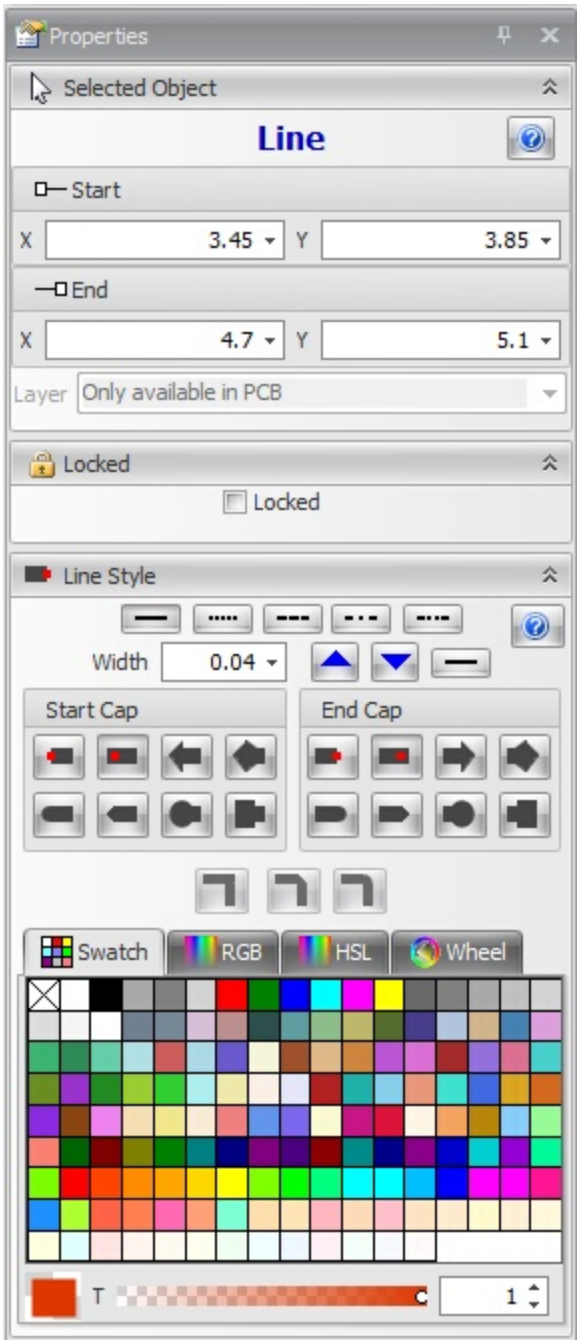
If you have any point, horizontal or vertical guides on your schematic or PCB and snap to guides is enabled, then the end points will snap to them if they are close enough.

You can set the color of the line by either left or ***right-clicking*** on a color button in the color bar at the base of the application viewports.

You can also set the lines parameters using its properties panel.

Line Properties Dialog

You can also edit the parameters of the line using its properties panel as shown below. To view the properties panel first select it. Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Click the  button to show this help topic.

Start

This is the start point of the line.

End

This is the end point of the line.

Layer

Use this to set the layer on which the line lies. *This only applies to PCBs.*

Locked

Check this box to prevent its properties from being altered or the line being moved.

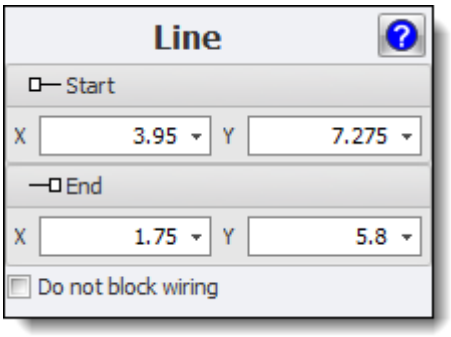
Line Style

Here you can set the style of the line.

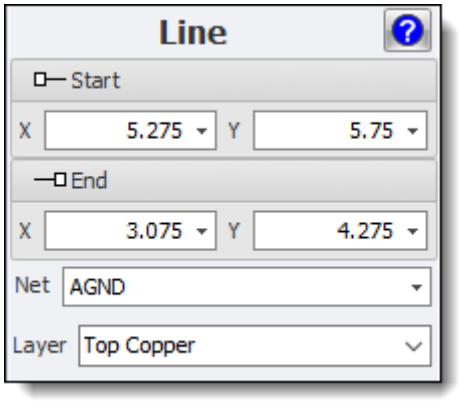
This includes the:

- dash/solid style
- The width
- The start and end caps
- The lines color.

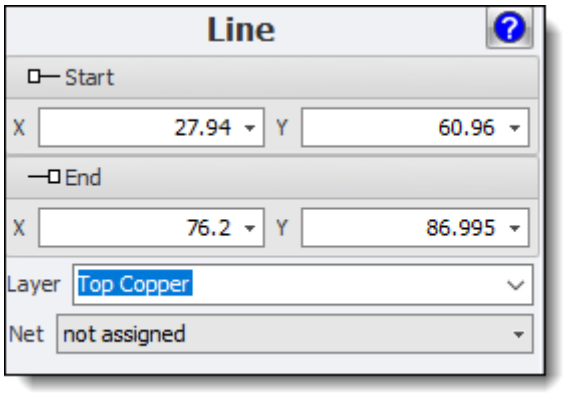
1.2.3.6.4 Line Editor



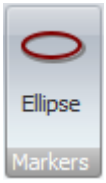
Schematics



PCB. Line on Electrical Layer



1.2.3.7 Markers

To add a marker to your sheet clic Ellipse button 

in the graphics menu.

A Marker

1.2.3.7.1 Marker Editor

The Marker Editor

1.2.3.8 Notes

In AutoTRAX DEX you can add graphical notes. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Notes have a width that can be zero for an infinitely thin line. However if you add lines to a PCB you need to make sure their width is at least the size of the manufacturing minimum width.

Notes also have color and can be semi-transparent. It is even possible to define an invisible note. If you have snap to objects enabled, objects will snap to invisible notes.

1.2.3.8.1 Adding Notes

You can add text notes.

To set the Fill Styles, Line Styles and Font Style before adding notes use the Default Graphics Settings

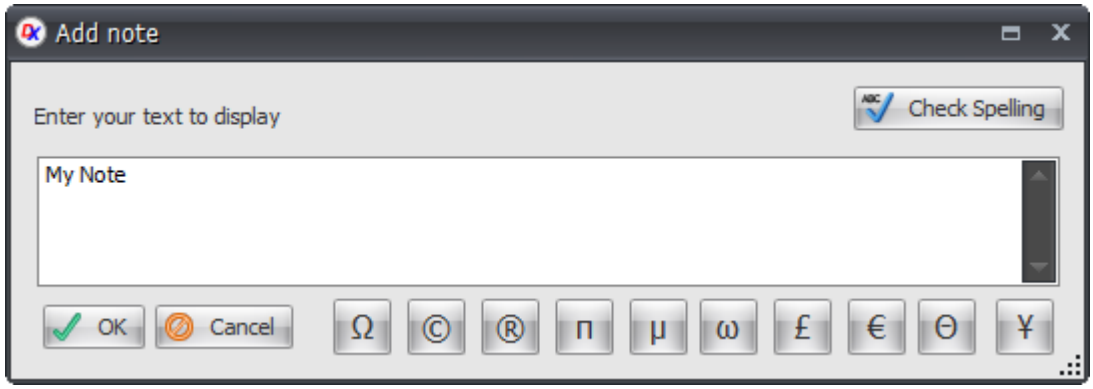
Notes and rounded notes

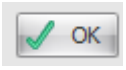
To add a note to a graphical sheet click on one of the  buttons in the **Add→Shapes** ribbon menu button group.

 Adds a rectangular note.

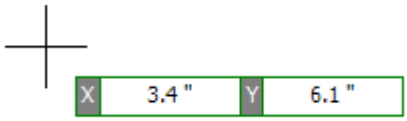
 Adds a Rounded rectangular note.

You will be prompted for the text.



Enter the text and click the  button

Move the mouse inside a graphical sheet's viewport. You will see the point cross follow the mouse. Left-click when the point cross is where you want to place the note or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and ***Enter/Space/Tab*** again to exactly set placement point.



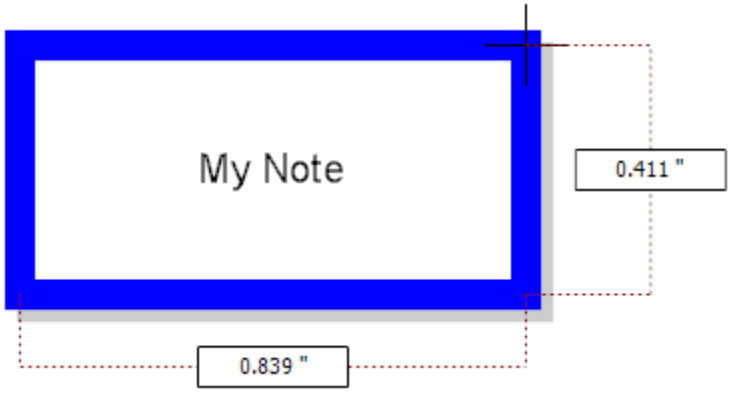
Now as you move the mouse the note will change in size and aspect ratio.

If you hold down the ***Shift*** key as you move the mouse, the note will be centered at the first point.

If you hold down the ***CTRL*** key as you move the mouse, the note will be square with the width and the height the same.

If you hold down both the ***Shift*** and ***CTRL*** keys simultaneously as you move the mouse, the note will be centered at the first point and will be square with the width and the height being the same.

Alternately, you can add a note image with a simple click, hold, and drag. When the point cross is where you want to start the note, ***left-click*** and hold the left button. While holding the left button drag to re-size the note and release the button; the note will be placed. In this mode, if you hit ***Enter*** before releasing the left button, you will be able to enter exact X and Y numerical values for the width and heights respectively. Note, using the ***Shift*** and ***CTRL*** keys as described above also works with this alternate method.



Dragging the mouse defines the rectangles size

1.2.3.8.2 Editing Notes

To edit a note first select it.

A selected note

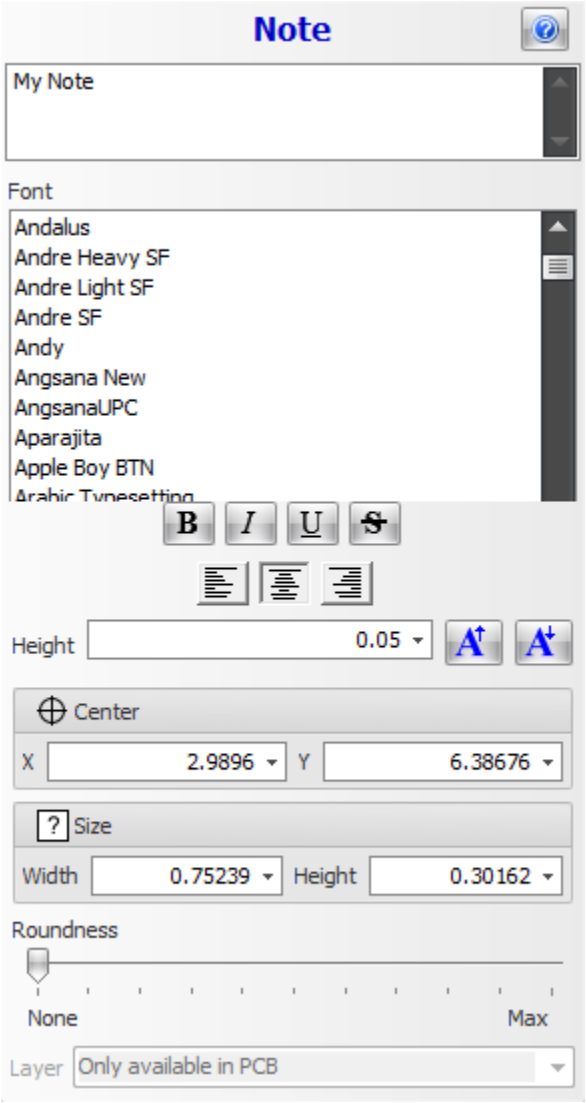
Drag any rotate manipulator to rotate the note.

Drag any scale manipulator to re-size the note.

Drag the note to move it.

Note Properties Dialog

You can also edit the parameters of the note using its properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it and then select Properties Panel from the context menu that opens.



Note Properties Dialog

X

The X coordinate of the center of the note.

Y

The Y coordinate of the center of the note

Width

The width of the note.

Height

The height of the note

Roundness

Drag the slider to change the roundness of the notes corners.

Layer

Use this drop-down to set the layer for the note.

1.2.3.8.3 Note Box Editor

1.2.3.9 Polylines

In AutoTRAX DEX you can add graphical polylines. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Polylines have a width that can be zero for an infinitely thin arc. However if you add polylines to a PCB you need to make sure their width is at least the size of the manufacturing minimum width.

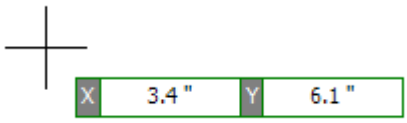
Polylines also have color and can be semi-transparent. It is even possible to define an invisible polyline. If you have snap to objects enabled, objects will snap to invisible polylines.

Polylines can also have a line style as shown below.

1.2.3.9.1 Adding Polylines

To add a polyline to a graphical sheet click on the  button in the Add→Shapes ribbon menu button group.

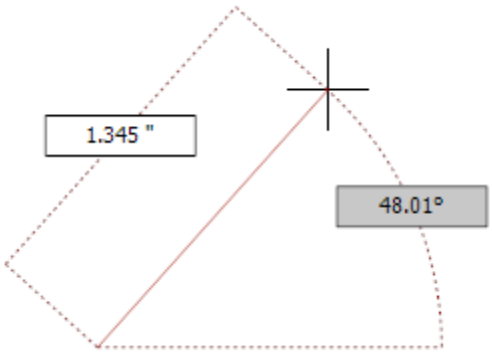
To set the Fill Styles and Line Styles before adding polylines use the Default Graphics Settings



Polyline start point

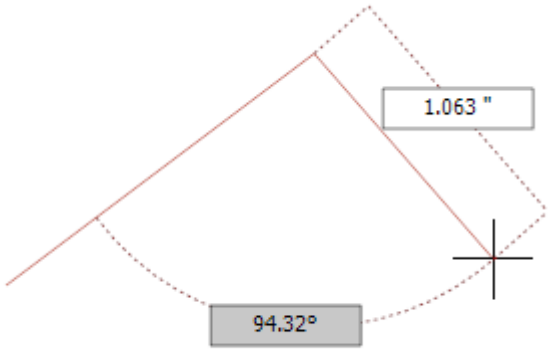
Left-click when the start cross is where you want to start the polyline or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.

Now as you move the mouse, the first segment of the polyline is defined. Left click to define its end point or press the ***Enter/Space/Tab*** key followed by the length, ***Enter/Space/Tab***, the angle, and ***Enter/Space/Tab*** to exactly define the first segment of the polyline.



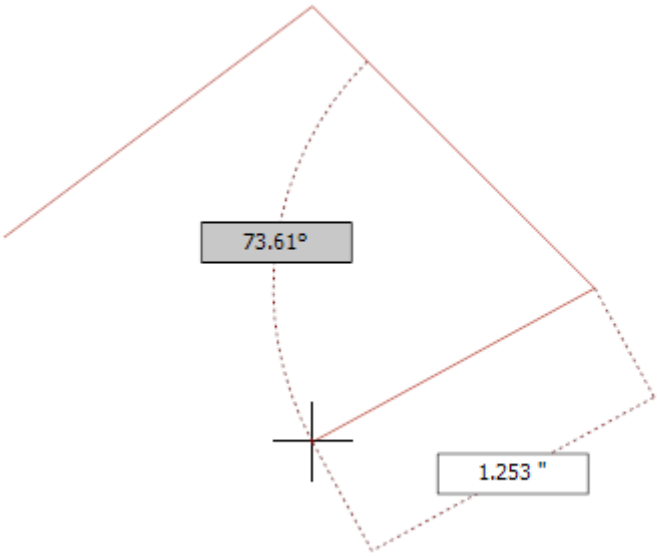
Defining the first segment

Now as you move the mouse, the second segment of the polyline is defined. ***Left-click*** to define its end point or press the ***Enter/Space/Tab*** key followed by the length, ***Enter/Space/Tab***, the angle, and ***Enter/Space/Tab*** to exactly define the second segment of the polyline.



Defining the second segment

Moving the mouse again, the third segment of the polyline is defined. ***Left-click*** to define its end point or press the***Enter/Space/Tab*** key followed by the length, ***Enter/Space/Tab***, the angle, and ***Enter/Space/Tab*** to exactly define the third segment of the polyline.



Defining the third segment

Continue this process to add as many line segments as you wish.

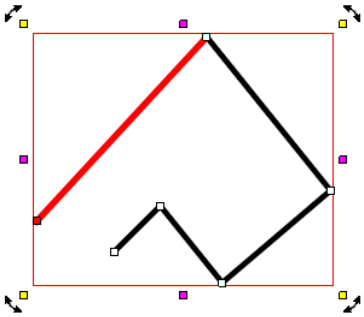
Press the ***ESC*** key to remove the last vertex added.

Double-click to complete adding the polyline.

To close the polyline move the mouse over or very near the start of the polyline and ***left-click.***

1.2.3.9.2 Editing Polylines

To edit a polyline first select it.



A selected polyline

Drag either of the polyline corner manipulators(□) to change the corners or ends.

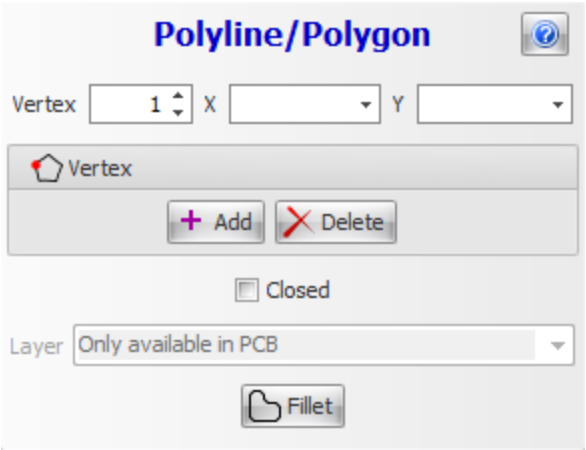
Drag any rotate manipulator to rotate the polyline.

Drag any scale manipulator to re-size polyline.

Drag the polyline to move it.

Polyline Properties Dialog

You can also edit the parameters of the polyline using it's properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Polyline/Polygon Properties Dialog

Click the button to show this help topic.

Vertex

This is the selected vertex/segment. The selected vertex is drawn as a solid red square. The selected segment id drawn in red.

X

The X coordinate of the selected vertex.

Y

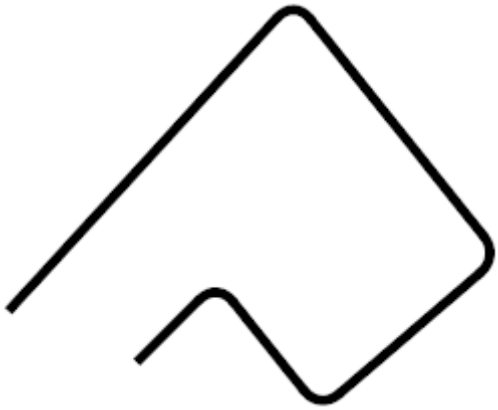
The Y coordinate of the selected vertex.

Click Add to add a vertex.

Click Delete to delete the selected vertex.

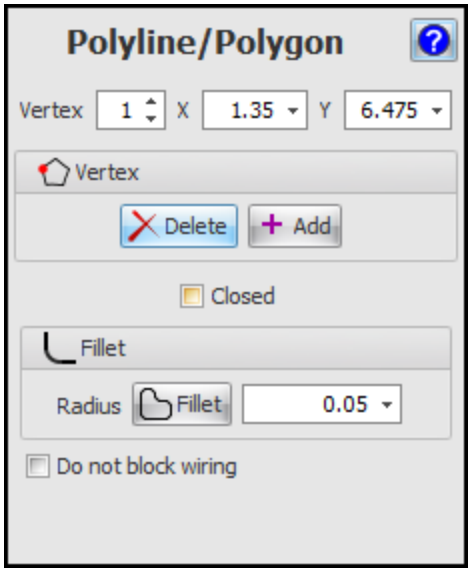
Check/uncheck Closed close/open the polyline.

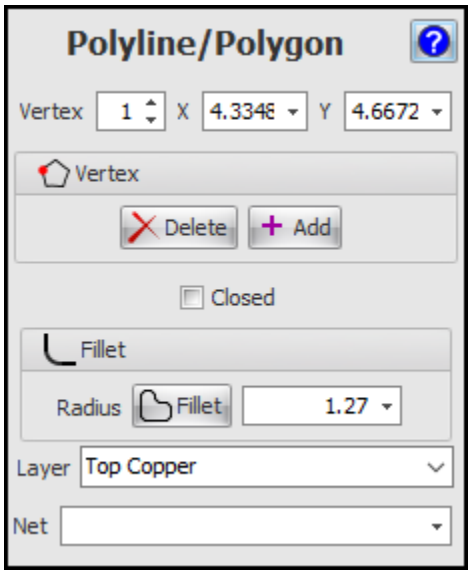
Click Fillet to fillet the polyline



A filleted polyline

1.2.3.9.3 Polyline/Polygon Editor





1.2.3.10 Polygons

In AutoTRAX DEX you can add graphical polygons. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Polygons have a width that can be zero for an infinitely thin polygon. However if you add polygons to a PCB you need to make sure their width is at least the size of the manufacturing minimum width.

Polygons also have color and can be semi-transparent. It is even possible to define an invisible polygon. If you have snap to objects enabled, objects will snap to invisible polygons.

Polygons can also have a line style as shown below.

1.2.3.10.1 Adding Polygons

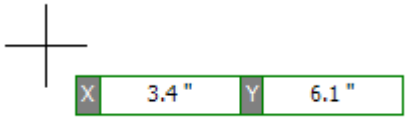
To add a polygon to a graphical sheet click on one of the  buttons in the **Add→Shapes** ribbon menu button group.

To set the Fill Styles and Line Styles before adding rectangles use the Default Graphics Settings

 Draws a hollow polygon.

 Draws a filled polygon with fill different from the outline.

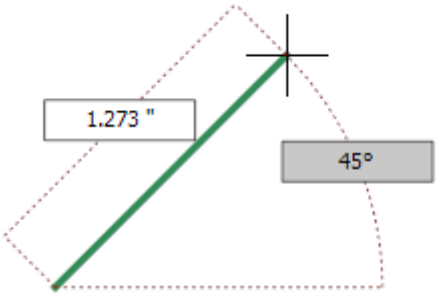
 Draws a filled polygon with no outline.



Polygon start point

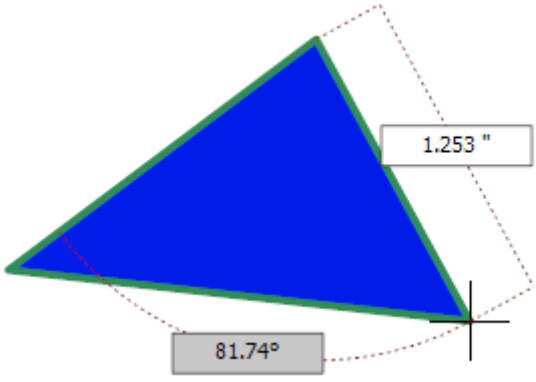
Left-click when the start cross is where you want to start the polygon or press the ***Enter*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.

As you move the mouse, the first segment of the polygon is defined. ***Left-click*** to define its end point or press the ***Enter/Space/Tab*** key followed by the length, ***Enter/Space/Tab***, the angle, and ***Enter/Space/Tab*** to exactly define the first segment of the polygon.



Defining the first segment

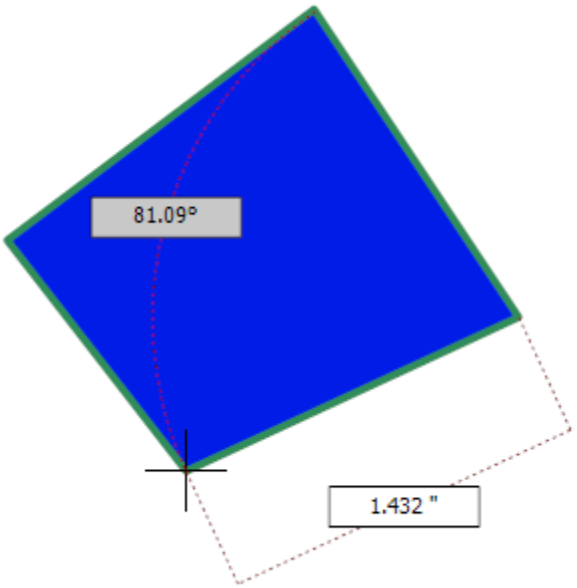
Now as you move the mouse the second segment of the polygon is defined. ***Left-click*** to define its end point or press the ***Enter*** key followed by the length, ***Enter/Space/Tab***, the angle, and ***Enter/Space/Tab*** to exactly define the second segment of the polygon.



Defining the second segment

Moving the mouse again, the third segment of the polygon is defined. ***Left-click*** to define its end point or press the ***Enter/Space/Tab*** key followed by the length,

Enter/Space/Tab, the angle, and ***Enter/Space/Tab*** to exactly define the first segment of the polygon.



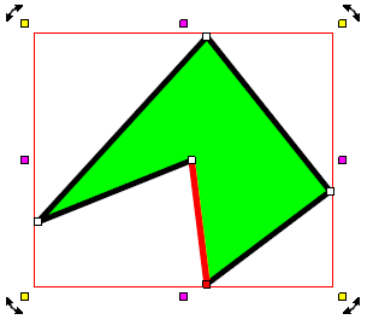
Defining the third segment

Continue this process to add as many line segments as you wish.

Double-click to end the polygon or hit the ***ESC*** key to end the polygon with the last placed line segment.

1.2.3.10.2 Editing Polygons

To edit a polygon first select it.



A selected polygon

Drag either of the polygon corner manipulators(□) to change the corners or ends.

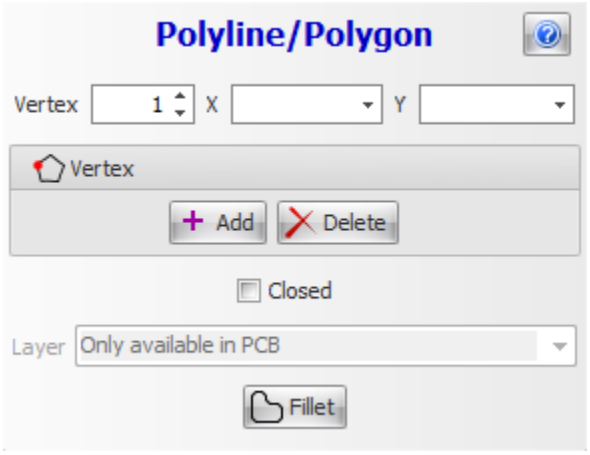
Drag any rotate manipulator to rotate the polygon.

Drag any scale manipulator ■ to re-size the polygon.

Drag the polygon to move it.

Polygon Properties Dialog

You can also edit the parameters of the polygon using its properties panel as shown below. To view the properties panel first select it.Then right-click on it, then select Properties Panel from the context menu that opens.



Polyline/Polygon Properties Dialog

Click the button to show this help topic.

Vertex

This is the selected vertex/segment. The selected vertex is drawn as a solid red square. The selected segment id drawn in red.

X

The X coordinate of the selected vertex.

Y

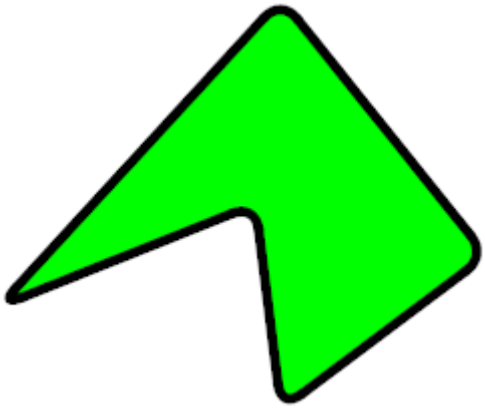
The Y coordinate of the selected vertex.

Click Add to add a vertex.

Click Delete to delete the selected vertex.

Check/uncheck Closed close/open the polygon.

Click Fillet to fillet the polygon



A filleted polygon

1.2.3.11 Rectangles

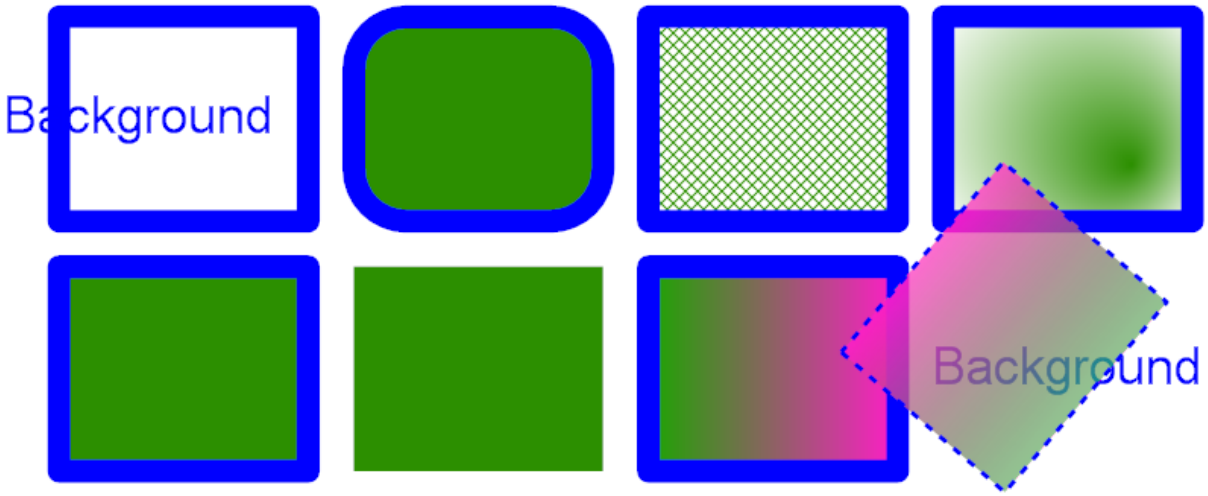
In AutoTRAX DEX you can add graphical rectangles. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

Rectangles have a width and height.

Rectangles also have color and can be semi-transparent. It is even possible to define an invisible rectangle. If you have snap to objects enabled objects will snap to invisible lines.

Rectangles can also have a line style as shown below.

See Fill Styles and Line Styles for more styling information.

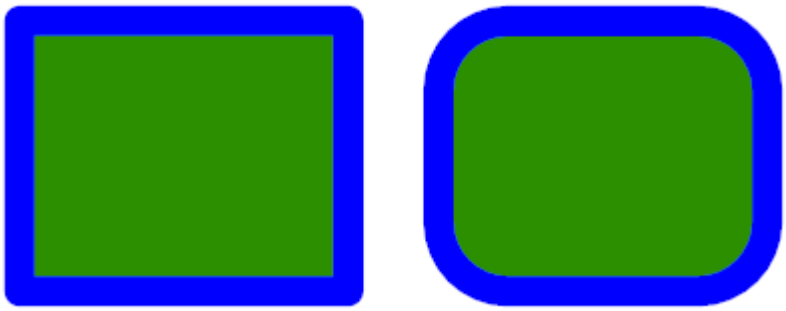


Sample rectangles

1.2.3.11.1 Adding Rectangles

You can add rectangles and rectangles with corners.

To set the Fill Styles and Line Styles before adding rectangles use the Default Graphics Settings.



Rectangles and rounded rectangles

To add a rectangle to a graphical sheet click on one of the  buttons in the **Add→Shapes** ribbon menu button group.

 Adds a hollow rectangle.

 Adds a filled rectangle with border.

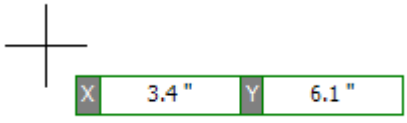
 Adds a filled rectangle without a border.

 Adds a hollow rectangle with rounded corners.

 Adds a filled rectangle with border with rounded corners.

 Adds a filled rectangle without a border with rounded corners.

Move the mouse inside the graphical sheet's viewport. You will see the point cross follow the mouse. ***Left-click*** when the point cross is where you want to start a rectangle or press the ***Enter/Space/Tab*** key followed by the X value, ***Enter/Space/Tab*** key, the Y value, and then ***Enter/Space/Tab*** to exactly place the starting point.



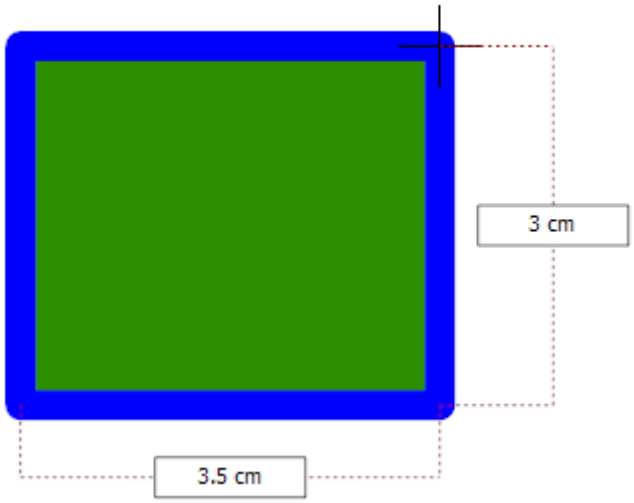
Now as you move the mouse the rectangle will change in size and aspect ratio.

If you hold down the ***Shift*** key as you move the mouse, the rectangle will be centered at the first point.

If you hold down the ***CTRL*** key as you move the mouse, the rectangle will be square with width and height being the same.

If you hold down both the ***Shift*** and ***CTRL*** keys simultaneously as you move the mouse, the rectangle will be centered at the first point and will be square with the width and the height being the same.

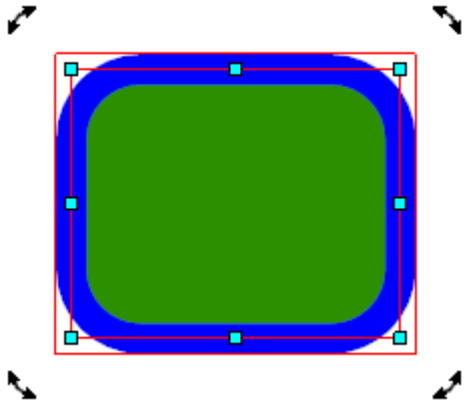
Alternately, you can add a rectangle with a simple ***left click and hold*** then drag. When the point cross is where you want to start a rectangle, ***left-click*** and hold the left button. ***Left-click and hold*** and drag to re-size the rectangle and release the button; the rectangle will be placed. In this mode, if you hit ***Enter*** before releasing the left button, you will be able to enter X and Y numerical values for the width and heights respectively. Note, using the ***Shift*** and ***CTRL*** keys as described above also works with this alternate method.



Dragging the mouse defines the rectangles size

1.2.3.11.2 Editing Rectangles

To edit a rectangle first select it.



A selected rectangle

Drag either of the rectangle's corner manipulators(▫) to change the corners or ends.

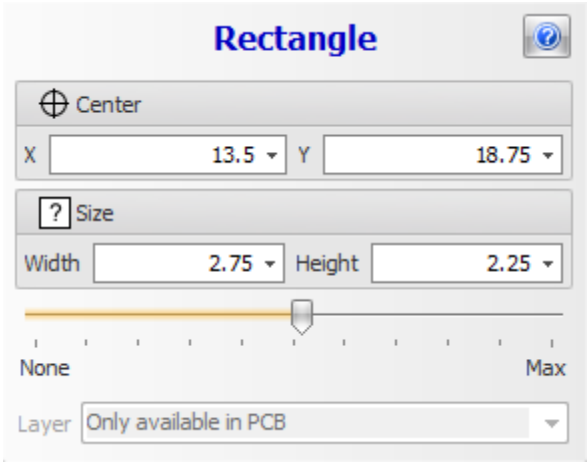
Drag any rotate manipulator to rotate the rectangle.

Drag any scale manipulator to re-size the rectangle.

Drag the rectangle to move it.

Rectangle Properties Dialog

You can also edit the parameters of the rectangle using its properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Rectangle Properties Dialog

X

The X coordinate of the center of the rectangle.

Y

The Y coordinate of the center of the rectangle

Width

The width of the rectangle.

Height

The height of the rectangle

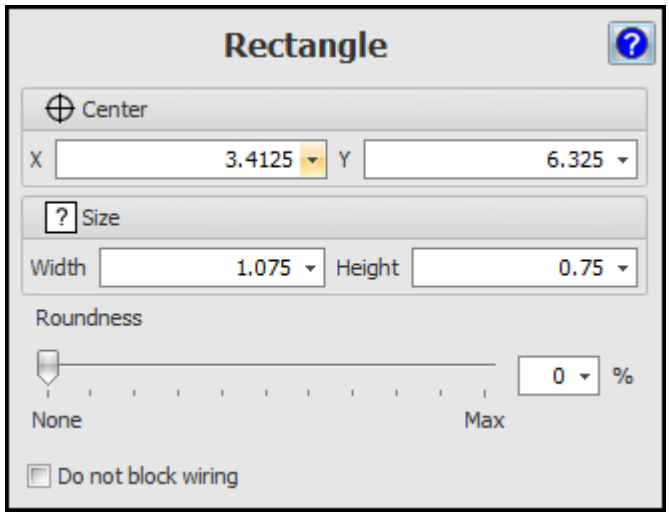
Roundness

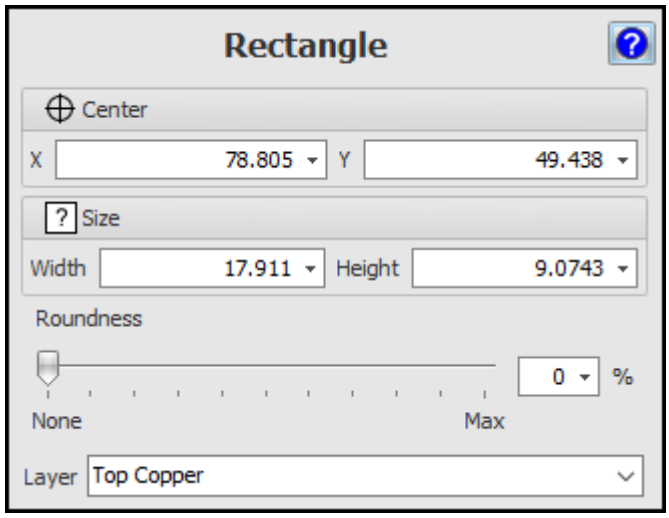
Drag the slider to change the roundness of the rectangles corners.

Layer

Use this drop-down to set the layer for the rectangle.

1.2.3.11.3 Rectangle Editor



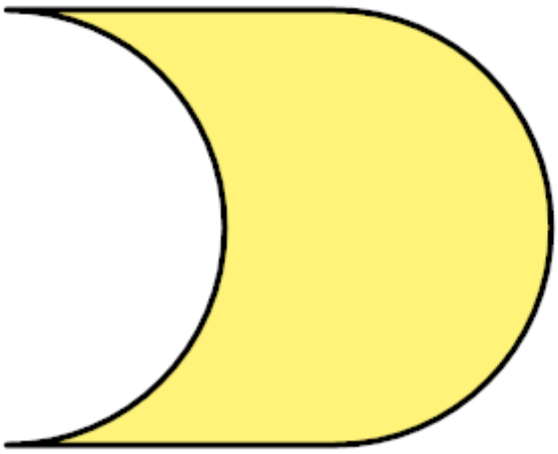


1.2.3.12 Shapes

You can combine two or more graphics objects into a single shape object.

The shape object consists of a shape outline and zero or more holes.

Shapes are useful to creating a complex shaped objects about of simple objects.



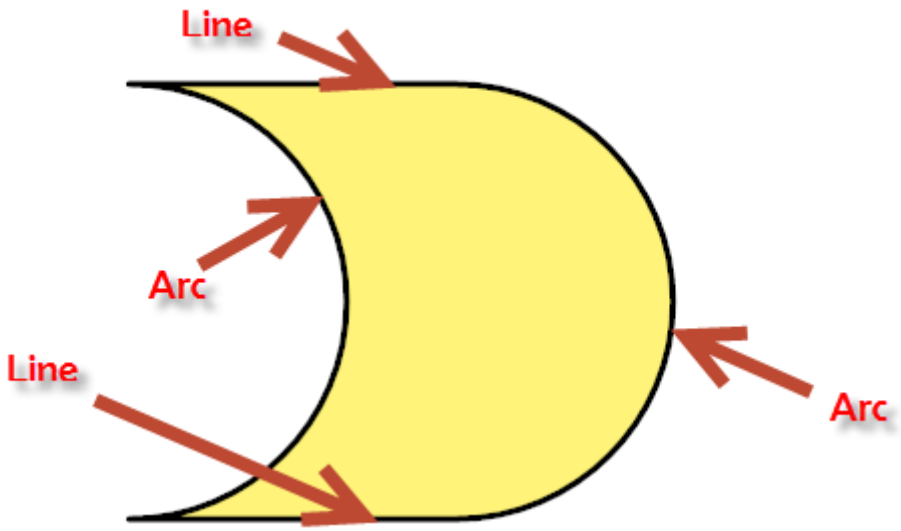
Example shape

The above shape was made by:

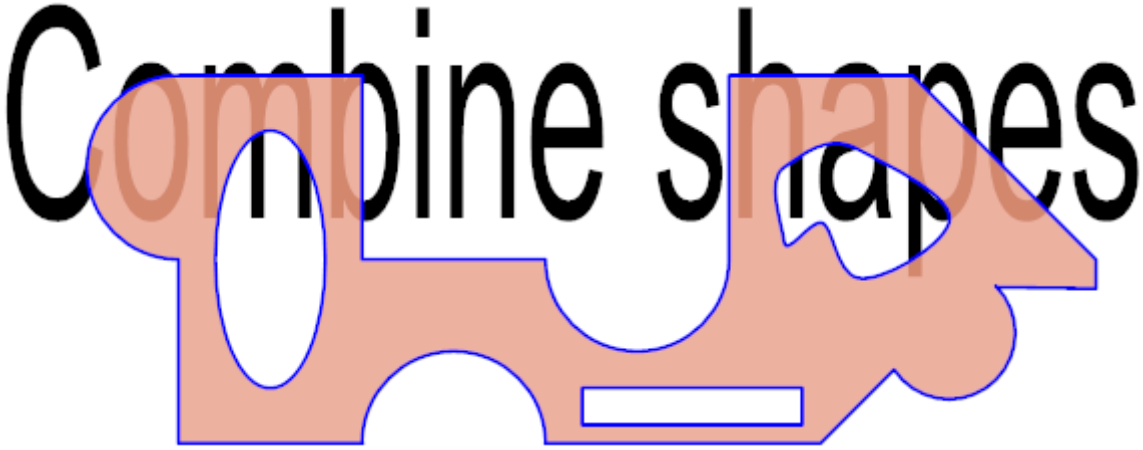
1. Adding a arc to a schematic.
2. Copying the arc and moving it to the left
3. Adding a line connecting the bottom ends of the two arcs.
4. Copying the line and moving it to connect the tops of the two arcs.

5. Selecting all 4 objects and then clicking the Layout→Group→ 

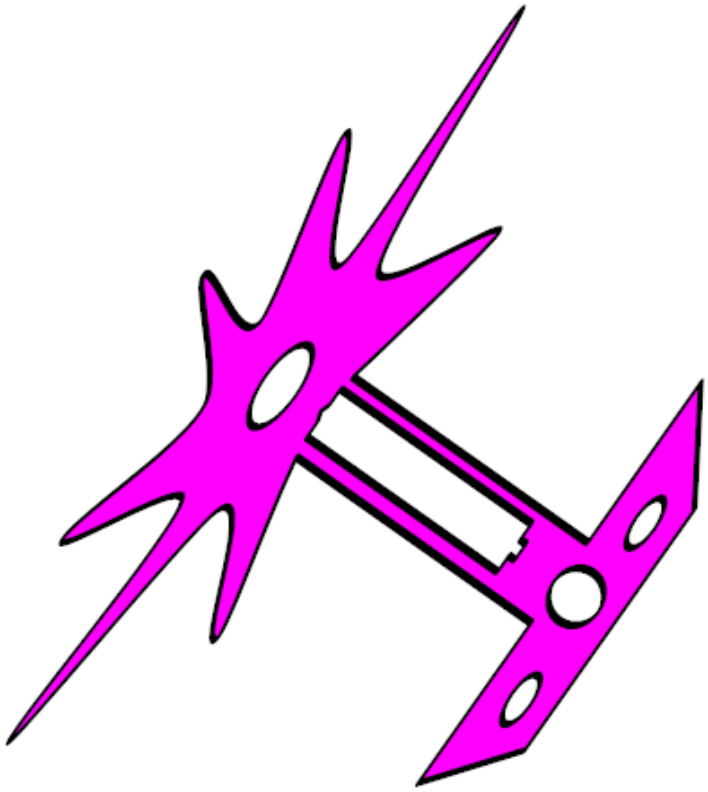
button.
6. Setting the fill and line style.



Below is a cartoon shape created by combining 1 curve, 1 polyline, 2 circles, 1 ellipse and a polygon. Just draw the outline and internal cutouts and combine.



Sample Shape - Made with polylines, arcs, an ellipse, a rectangle and a curve



Sample Shape

Creating Shapes

Editing Shapes

1.2.3.12.1 Creating Shapes

You can combine 2 or more graphics objects into a single shape object.

The shape object consists of a shape outline and zero or more holes.

Shapes are useful to creating a complex shaped objects about of simple objects.

The following objects are converted into the shape outline.

- Lines
- Arcs
- Polylines
- Open Curves

The following objects are converted into the shape holes.

- Rectangles
- Circles
- Ellipses
- Closed Curves
- Polygons

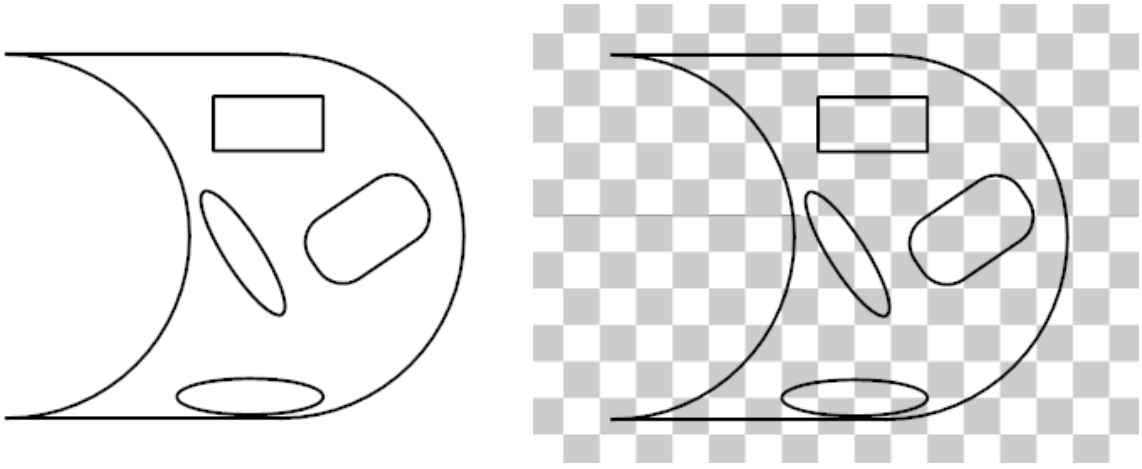
You cannot create a shape object containing:

- Text
- Images
- Dimensions
- Coordinates
- Electrical items such as Symbols, wires, buses, inter-wire connectors, footprints, pads and tracks.

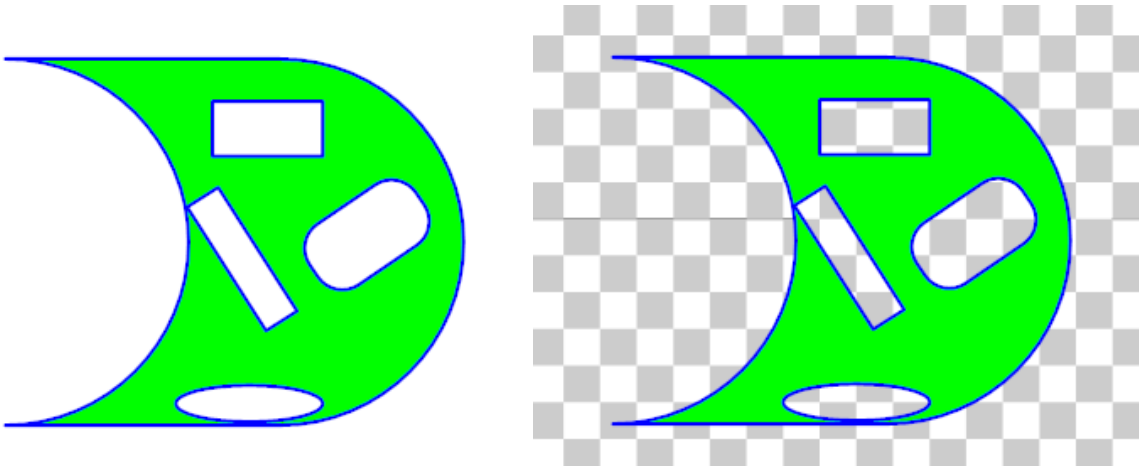
To combine graphics objects so that they can be moved, rotated and scaled as a single object first select them and then click the Layout→Group→ 

button. This button will only be enables if you have selected graphical objects.

All the fill styles of the hole objects will be set to transparent. The fill style of the resultant shape will be the shapes default fill style. And the line style will be the default line style.



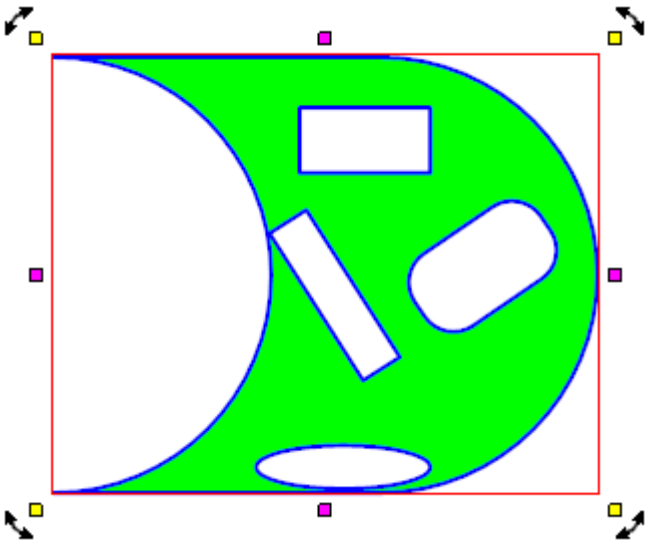
Graphics to convert to a single shape



New Shape object

1.2.3.12.2 Editing Shapes

To edit a shape first select it.



Selected Shape

Drag either of the shape's corner manipulators(▫) to change the corners or ends.

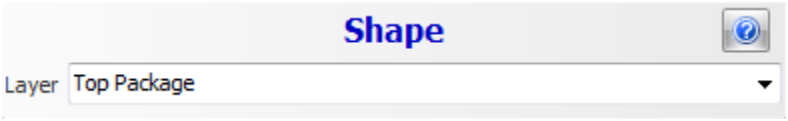
Drag any rotate manipulator to rotate the shape.

Drag any scale manipulator to re-size the shape.

Drag the shape to move it.

Shape Properties Dialog

You can also edit the parameters of the shape using its properties panel as shown below. To view the properties panel first select it. Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Click the  button to show this help topic.

Layer - Select the layer for the shape. This only applies to shapes in PCBs or Footprints.

1.2.3.12.3 Shape Editor

1.2.3.13 Text

In AutoTRAX DEX you can add graphical text. In schematics they have no electrical significance. However, in PCBs they have significance if they are placed on electrical layers such as the top or bottom layer; here they will leave copper traces and will provide an electrical connection to everything that is connected to them.

1.2.3.13.1 Adding Text

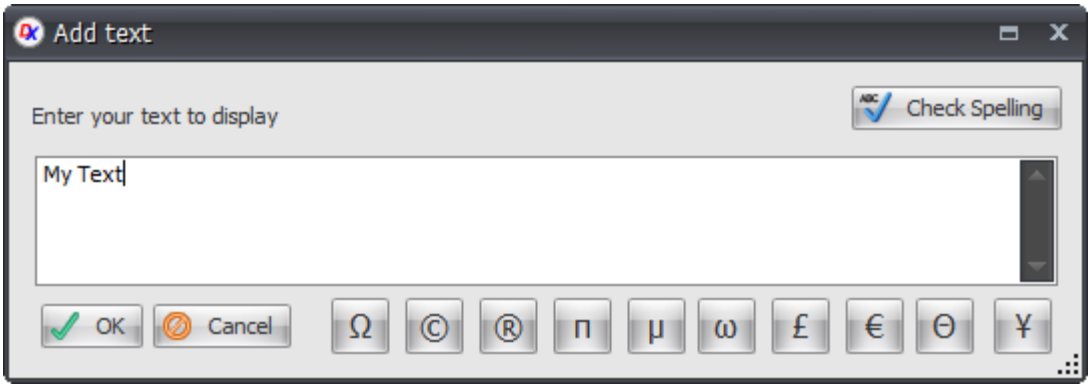
You can add Text.

To set the Fill Styles, Line Styles and Font Style before adding text use the Default Graphics Settings.

Text

To add text to a graphical sheet. click on the A button in the **Add→Shapes** ribbon menu button group.

You will be prompted for the text.



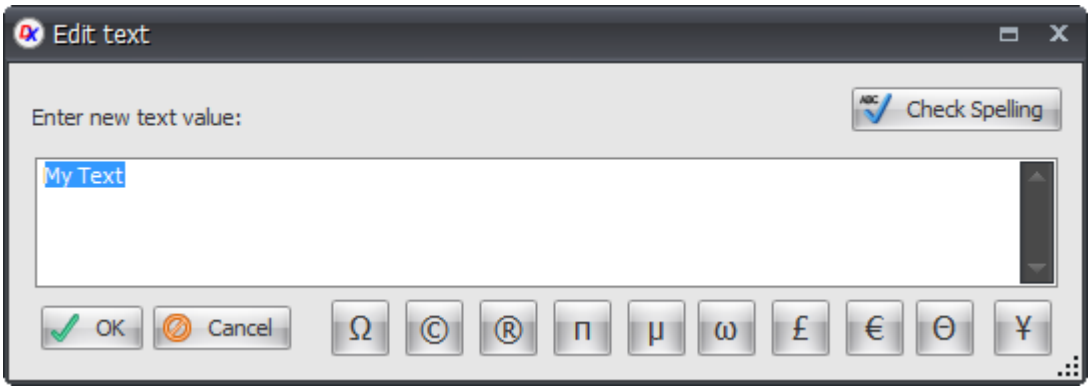
Enter the text and click the OK button.

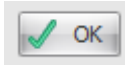
Move the mouse inside a graphical sheet's viewport. You will see the text cross follow the mouse. **Left-click** to place the text.

1.2.3.13.2 Editing Text

Double-click

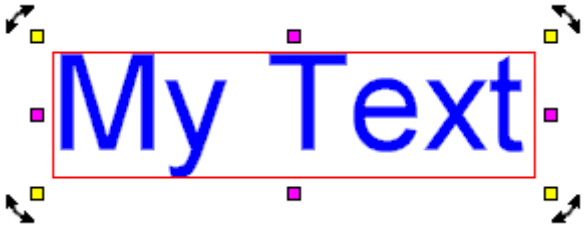
Double-click on the text to edit the text. The edit text dialog will appear.



Enter the new text and click the  button.

Interactive Edit

To edit text interactively first select it.



Selected Text

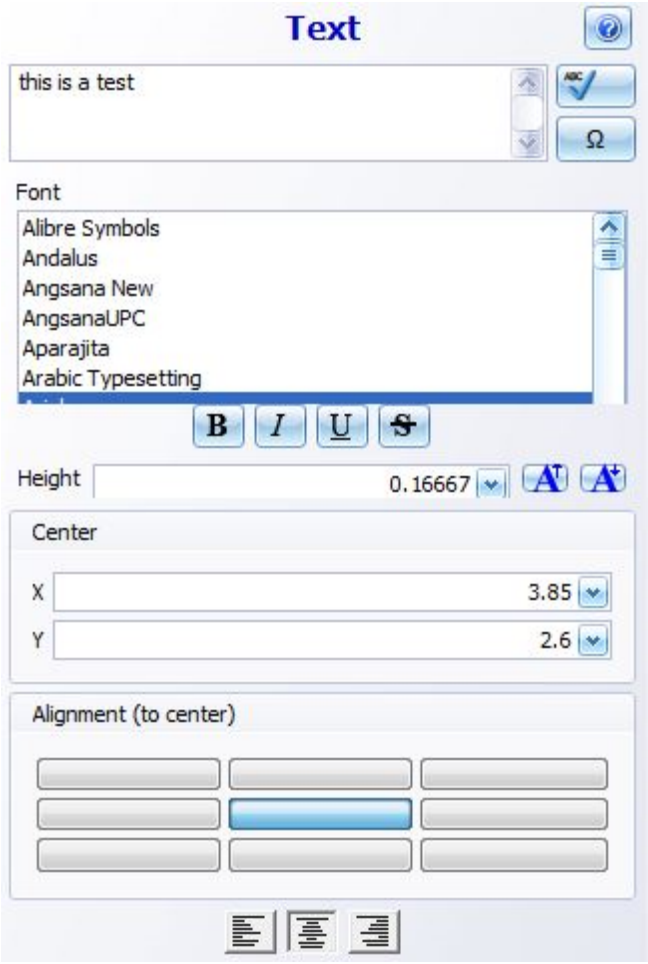
Drag any rotate manipulator  to rotate the text.

Drag any scale manipulator to re-size the text.

Drag the rectangle to move the text.

Text Properties Dialog

You can also edit the parameters of the text using its properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Text Properties Dialog

Text

Text to be placed

Font

Font style of the placed text

Height

The height of the text

X

The X coordinate of the center of the text

Y

The Y coordinate of the center of the text

Center

Center location of the text

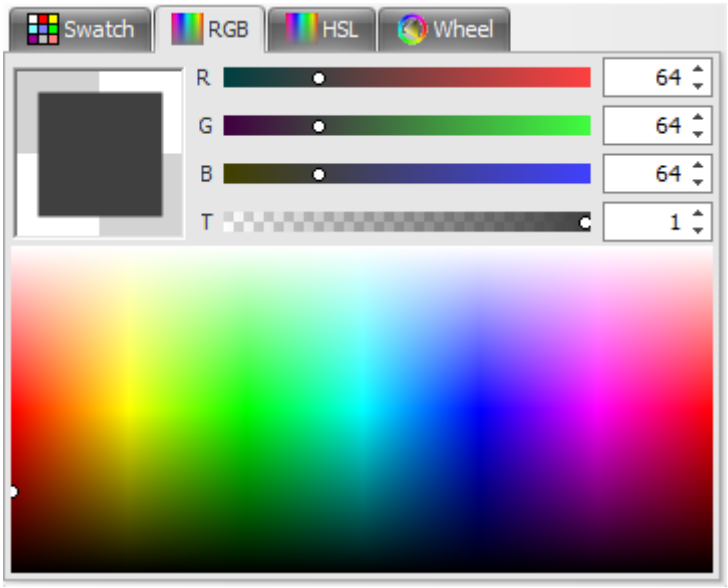
Alignment

Alignment of the text

1.2.3.13.3 Text Editor

1.2.3.14 Colors

The color editor consists of 4 separate editors. Click on the editor's name at the top of the control to display the editor.



The Color Swatch Editor

The RGB Color Editor

The HSL Color Editor

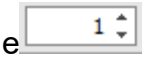
The Color Wheel Editor

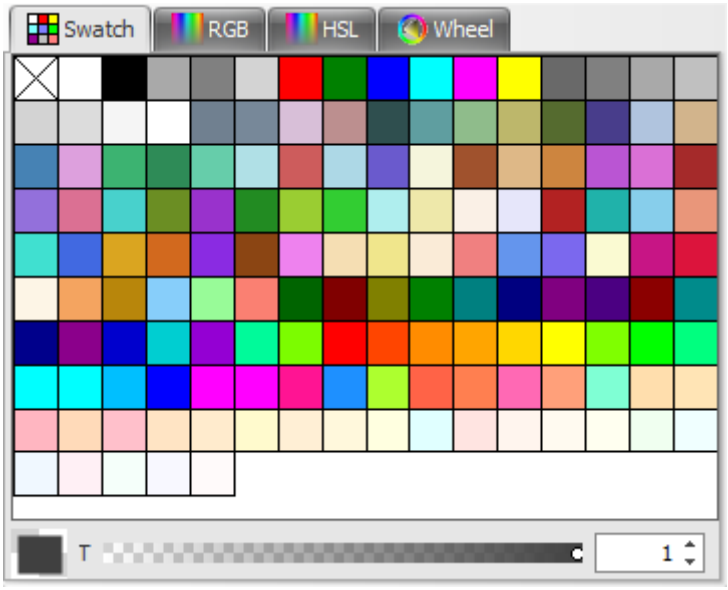
1.2.3.14.1 The Color Swatch

The color swatch editor is shown below.

Click on any of the colored square to select the color.

Transparency

Drag the transparency slider  to the left to increase the transparency of the color and drag the slider to the right to reduce the transparency. You can also enter the transparency in the  box. The transparency value is between 0 and 1 where 0 is totally transparent.



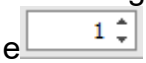
1.2.3.14.2 RGB

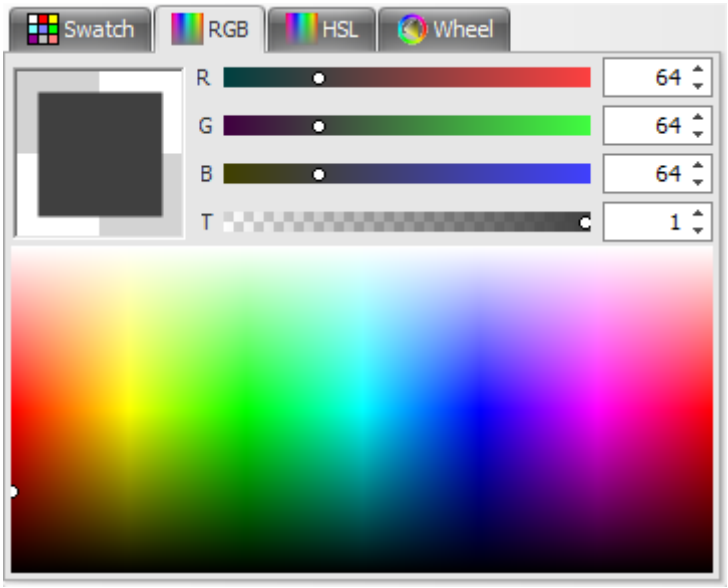
The RGB editor is shown below. RGB is an abbreviation for red–green–blue.

R(Red) Drag the  slider to change the red color component. You can enter a value between 0 and 255.

G(Green) Drag the  slider to change the green color component. You can enter a value between 0 and 255.

B(Blue) Drag the  slider to change the blue color component. You can enter a value between 0 and 255.

T(Transparency) Drag the transparency slider  to the left to increase the transparency of the color and drag the slider to the right to reduce the transparency. You can also enter the transparency in the  box. The transparency value is between 0 and 1 where 0 is totally transparent.

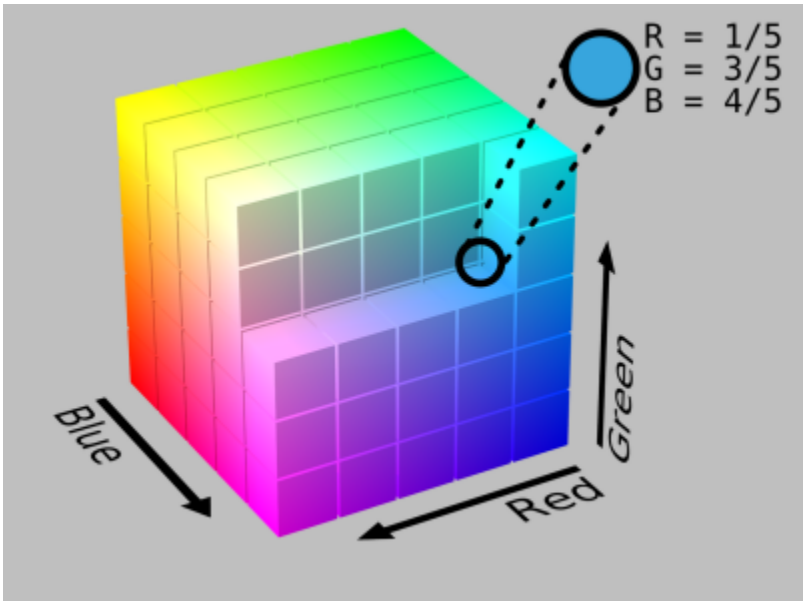


Background

An RGB color space is any additive color space based on the RGB color model. A particular RGB color space is defined by the three chromaticity's of the red, green, and blue additive primaries, and can produce any chromaticity that is the triangle defined by those primary colors. The complete specification of an RGB color space also requires a white point chromaticity and a gamma correction curve.

An RGB color space can be easily understood by thinking of it as "all possible colors" that can be made from three colorants for red, green and blue. Imagine, for example, shining three lights together onto a white wall: one red light, one green light, and one blue light, each with dimmer switches. If only the red light is on, the wall will look red. If only the green light is on, the wall will look green. If the red and green lights are on together, the wall will look yellow. Dim the red light and the wall will become more of a yellow-green. Dim the green light instead, and the wall will become more orange. Bringing up the blue light a bit will cause the orange to become less saturated and more whitish. In all, each setting of the three dimmer switches will produce a different result, either in color or in brightness or both. The set of all possible results is the gamut defined by those particular color lamps. Swap the red lamp for one of a different brand that is slightly more orange, and there will be a slightly different gamut, since the set of all colors that can be produced with the three lights will be changed.

An LCD display can be thought of as a grid of thousands of little red, green, and blue lamps, each with their own dimmer switch. The gamut of the display will depend on the three colors used for the red, green and blue lights. A wide-gamut display will have very saturated, "pure" light colors, and thus be able to display very saturated, deep colors.



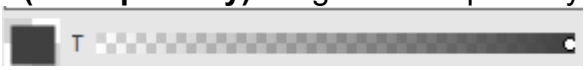
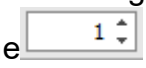
1.2.3.14.3 HSL

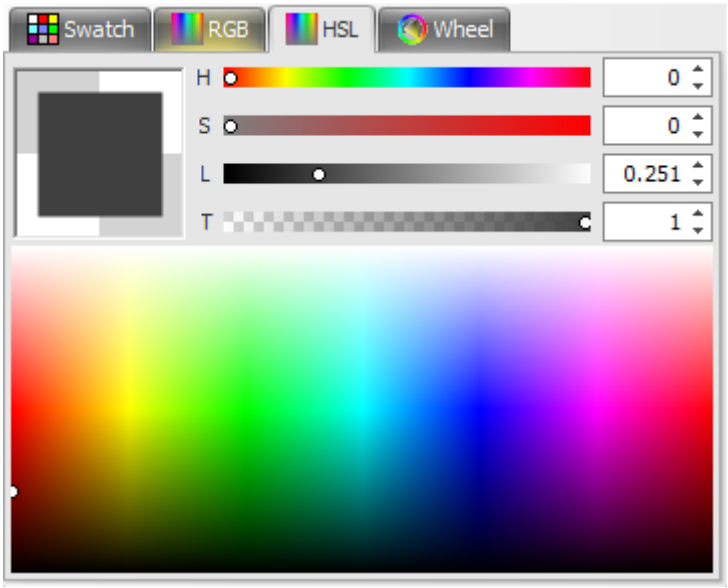
The HSL (Hue-Saturation-Luminosity) editor is shown below.

H(Hue) Drag the  slider to change the Hue. You can enter a value between 0 and 360.

S(Saturation) Drag the  slider to change the Saturation. You can enter a value between 0 and 1.

L(Luminosity) Drag the  slider to change the Luminosity. You can enter a value between 0 and 1.

T(Transparency) Drag the transparency slider  to the left to increase the transparency of the color and drag the slider to the right to reduce the transparency. You can also enter the transparency in the  box. The transparency value is between 0 and 1 where 0 is totally transparent.



Background

HSL is a common cylindrical-coordinate representation of points in an RGB color model. The representation rearrange the geometry of RGB in an attempt to be more intuitive and perceptually relevant than the Cartesian (cube) representation. Developed in the 1970s for computer graphics applications, HSL is used today in color pickers, in image editing software, and less commonly in image analysis and computer vision.

HSL stands for hue, saturation, and lightness, and is often also called HLS.

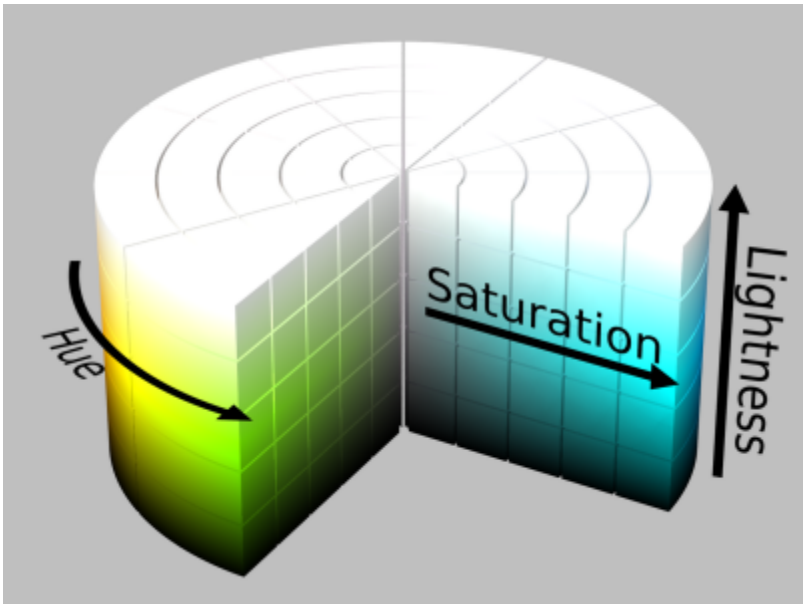
In the cylinder, the angle around the central vertical axis corresponds to "hue", the distance from the axis corresponds to "saturation", and the distance along the axis corresponds to "lightness", "value" or "brightness". Because HSL is a simple transformation of device-dependent RGB models, the physical colors they define depend on the colors of the red, green, and blue primaries of the device or of the particular RGB space, and on the gamma correction used to represent the amounts of those primaries. Each unique RGB device therefore has a unique HSL space to accompany it, and numerical HSL or HSV values describe a different color for each basis RGB space.

This representations is used widely in computer graphics, and one or the other of them is often more convenient than RGB, but both are also criticized for not adequately separating color-making attributes, or for their lack of perceptual uniformity. Other more computationally intensive models, such as CIELAB or CIECAM02 better achieve these goals.

Basic principle

HSL has cylindrical geometries (see below), with hue, their angular dimension, starting at the red primary at 0°, passing through the green primary at 120° and the blue primary at 240°, and then wrapping back to red at 360°. In each geometry, the

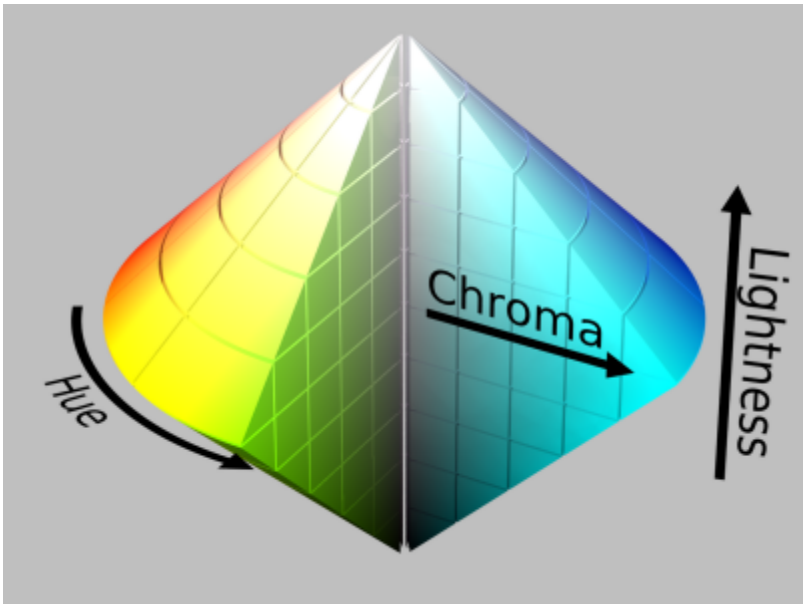
central vertical axis comprises the neutral, achromatic, or gray colors, ranging from black at lightness 0 or value 0, the bottom, to white at lightness 1 or value 1, the top. In both geometries, the additive primary and secondary colors – red, yellow, green, cyan, blue, and magenta – and linear mixtures between adjacent pairs of them, sometimes called pure colors, are arranged around the outside edge of the cylinder with saturation 1; in HSL they have lightness ½. In HSL, both tints and shades have full saturation, and only mixtures with both black and white – called tones – have saturation less than 1.



If we plot hue and HSL lightness against chroma rather than saturation, the resulting solid is a bicone or cone, respectively, not a cylinder. Such diagrams often claim to represent HSL directly, with the chroma dimension deceptively labeled "saturation".

Because these definitions of saturation – in which very dark (in both models) or very light (in HSL) near-neutral colors, for instance ■ or □, are considered fully saturated – conflict with the intuitive notion of color purity, often a conic or bi-conic solid is drawn instead, with what this article calls chroma as its radial dimension, instead of saturation. Confusingly, such diagrams usually label this radial dimension "saturation", blurring or erasing the distinction between saturation and chroma. As described below, computing chroma is a helpful step in the derivation of each model. Because such an intermediate model – with dimensions hue, chroma, and HSV value or HSL lightness – takes the shape of a cone or bicone, HSV is often called the "hexcone model" while HSL is often called the "bi-hexcone model"

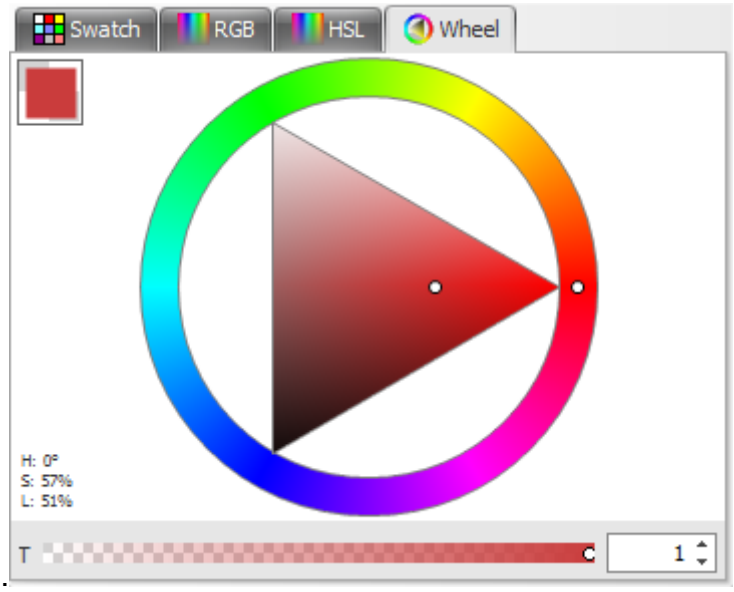
HSL cylinder

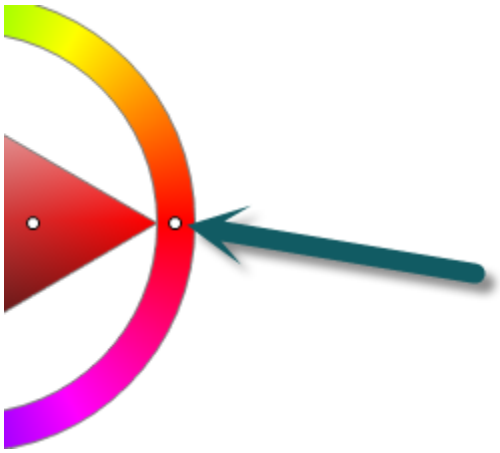


Hue and HSL lightness against chroma

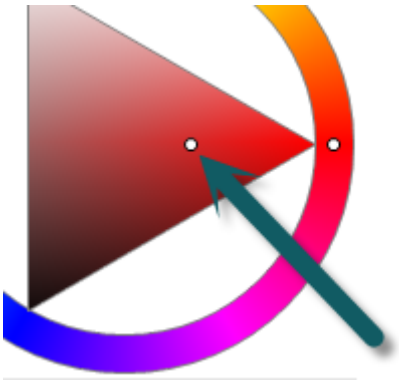
1.2.3.14.4 The Color Wheel

The color wheel editor is shown below



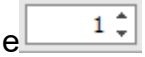


Drag the small dot in the outer ring to change the Hue.



Drag the small dot in the central triangle to change the saturation/luminosity.

Transparency

Drag the transparency slider  to the left to increase the transparency of the color and drag the slider to the right to reduce the transparency. You can also enter the transparency in the  box. The transparency value is between 0 and 1 where 0 is totally transparent.

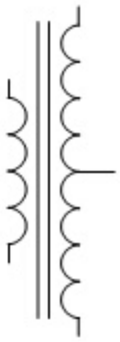
Background

A Color Wheel or color circle is an abstract illustrative organization of color hues around a circle that shows relationships between primary colors, secondary colors, complementary colors, etc.

1.2.4 Artworks

Artwork files contain schematic symbols that do not have any electrical content, such as parts and wires. They are used for creating library artwork to be used in project and part files. They can also be used for any other drawing needs, as AutoTRAX DEX's powerful graphics allows you to graphical design engineering drawings.

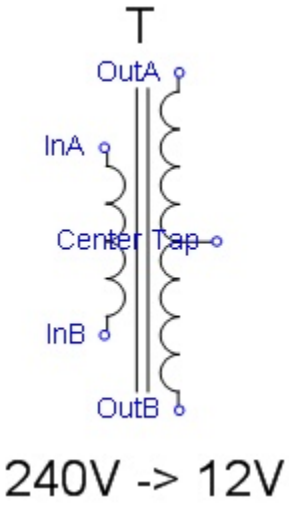
Below is an example of artwork for a transformer with a center tap. It consists of a collection of arcs and lines. There are no symbol terminals.



You would use this when creating a custom transformer part. To do this:

1. Create a new part.
2. Drag the artwork from the library.
3. Add the symbol terminals.
4. Save the part.

Below is a part created using the artwork and with terminals added



You can also add artwork from the library onto another artwork file that you are editing. This allows you to build complex art using a bottom up design methodology.

You can also add artwork to the PCB. However if the artwork is drawn with a black pen, it will appear drawn black on black on the PCB, so you will need to change its color and layer to make it visible. Artwork will initially be placed on the top document layer.

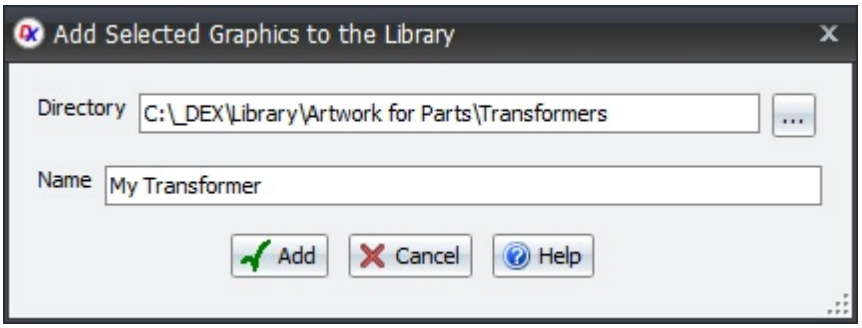
1.2.4.1 Adding Artwork to the Library

To add artwork to the library:

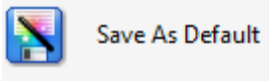
1. Select the graphics for the artwork in the schematic. Any electrical items such as symbol terminals, although selected, will not be added to the artwork file.
2. In the Library Panel, ***right-click*** and select **Add Selected Graphics**. The dialog box shown below will be displayed.
3. Enter the name for the artwork and click the ***OK*** button. You can change the target directory by typing it in or clicking on the browse ... button.

Once added, you can drag the artwork from the library panel onto any schematic.

You can edit the saved artwork file in the library panel by ***double-clicking*** on its name or ***right-clicking*** on its name and select **Edit Selected Part**.



1.2.4.2 Setting The Default Artwork

You can set the default artwork template by first creating an artwork, setting it up exactly as you want all new artworks to be and then click the 

in the File Menu. Then, every time you create a new artwork, it will start off as a mirror image of the default artwork you saved.

The file is named Default.artwork and is in the library '___SystemDoNotRemove' directory .

You can always reset the default artwork to the factory default using the File Settings.

1.2.5 Parts

A part consists of a footprint, and optional 3-D model, 1 or more symbols and ideally a datasheet coupled with hyperlinks to both the supplier and the manufacturer.

Normally a part consists of just a single symbol, but the part may have 2 or more symbols such as the old TTL devices which have multiple gates and you want a separate symbol for each gate.

When you start designing a part, the first thing you need to do is find the datasheet for the part. The datasheet should define the physical package, the preferred footprint or land pattern and the pin assignment details and what each pin does. Without the parts datasheet you are flying blind and just guessing as to what you are designing.

So how to find the datasheet?

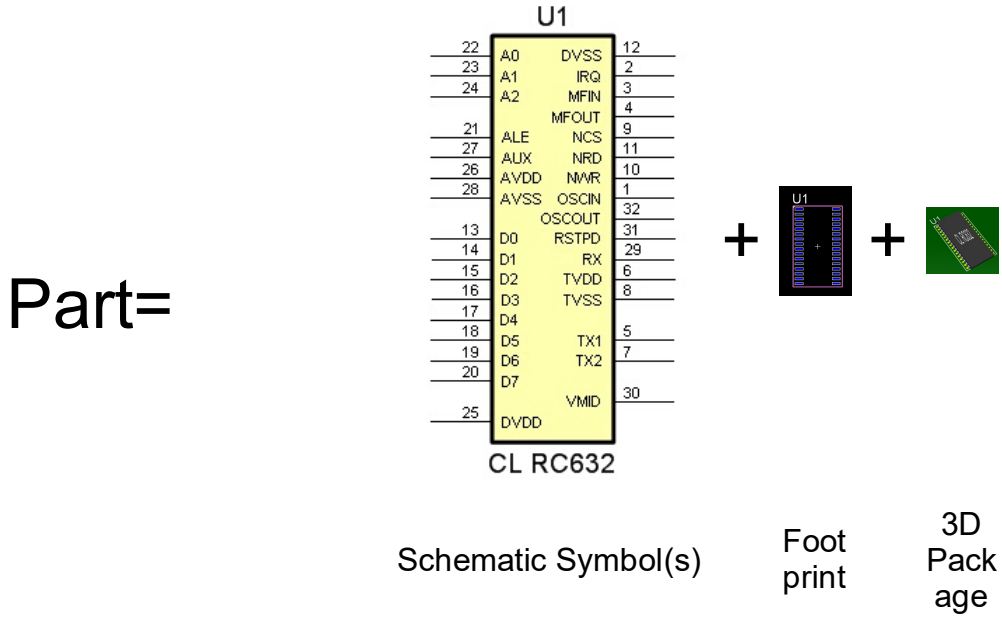
In AutoTRAX DEX there are two different types of symbols:

Symbols with terminal magnets.

Fully custom symbols without terminal magnets such as a traditional NPN transistor or an operational amplifier.

Part files contain schematic symbols, text documents and a footprint (land pattern) that contains all the information needed to define the abstract schematic representation for a part together with the footprint required for the PCB and the 3D solid model for the part.

Parts are defined parametrically and this allows a single part to be quickly and easily modified into a different part type. The parametric models define how to create a complete part from only a few simple parameters.



This video gives you a quick overview of what parts are in AutoTRAX DEX

Parts in AutoTRAX

1.2.5.1 What is a part

What is a part? That is a good question. For AutoTRAX DEX a part can be one of several different things. These are physical parts, which can either be electrical or purely mechanical and non-physical parts which can also be electrical but can also be meta data such as sheet title blocks or just graphics.

Physical Parts

The first, and most obvious type is a physical part.

In AutoTRAX DEX, a physical part as many different representations. These are symbolic shapes, physical land patterns and 3-D physical models

The first representation is a logical or symbolic form. This is where the part is represented by a graphical symbol.

The second representation is as a 2-D footprint or land pattern. This 2-D representation is a collection of objects which include pads, silkscreen artwork,

reference IDs, placement points, PCB holes, PCB cutouts and even the PCB shape or outline.

The third and final representation is a 3-D model that shows you are way part actually exist in reality. This is the closest of the three representations that matches what a physical part really looks like.

Most parts consist of:

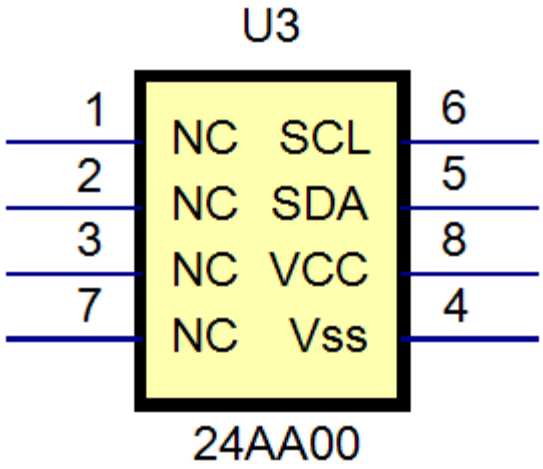
- 1 or more symbols
- 1 footprint
- 1 3D model
- An optional Spice simulation model.

Symbols

A part can consist of one or more symbols. Each symbol represents the electrical characteristics of the part and also is the main visualisation of the part.

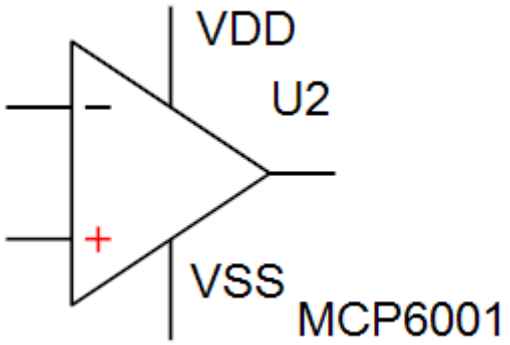
There are 2 types of symbols. Terminal Magnet based symbols and custom symbols.

Terminal Magnet based symbols



A symbol using a terminal magnet

Custom Symbols



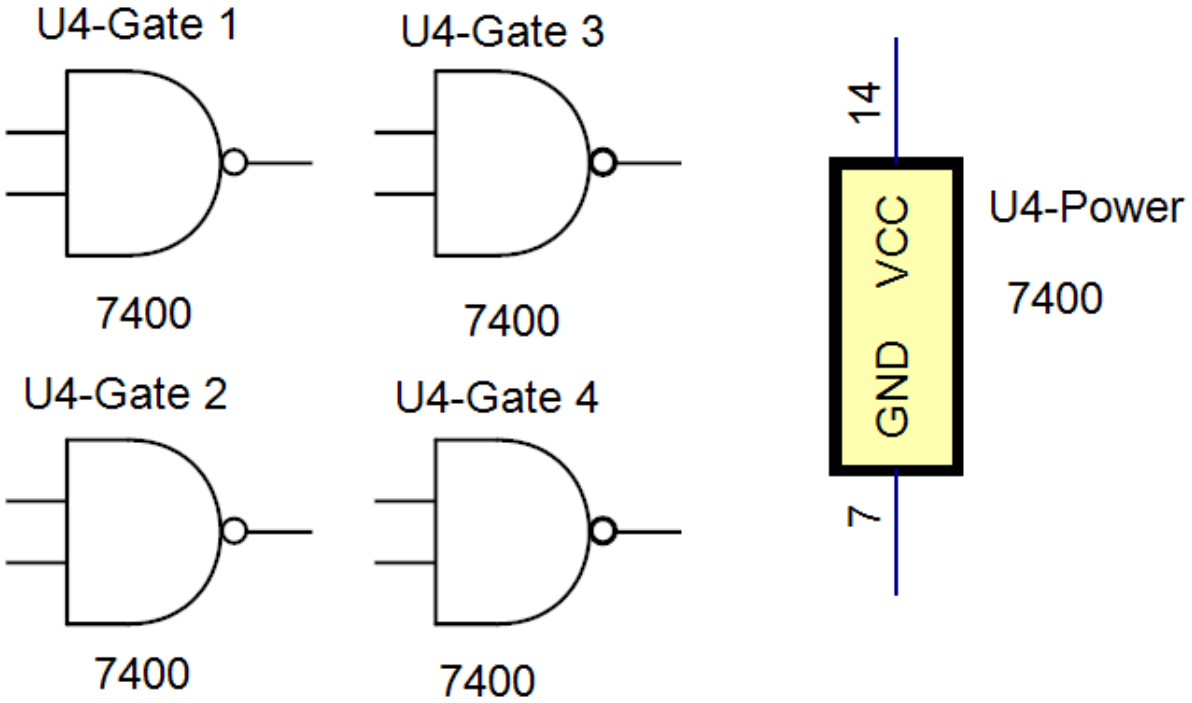
A symbol not using terminal magnet

Single Symbol

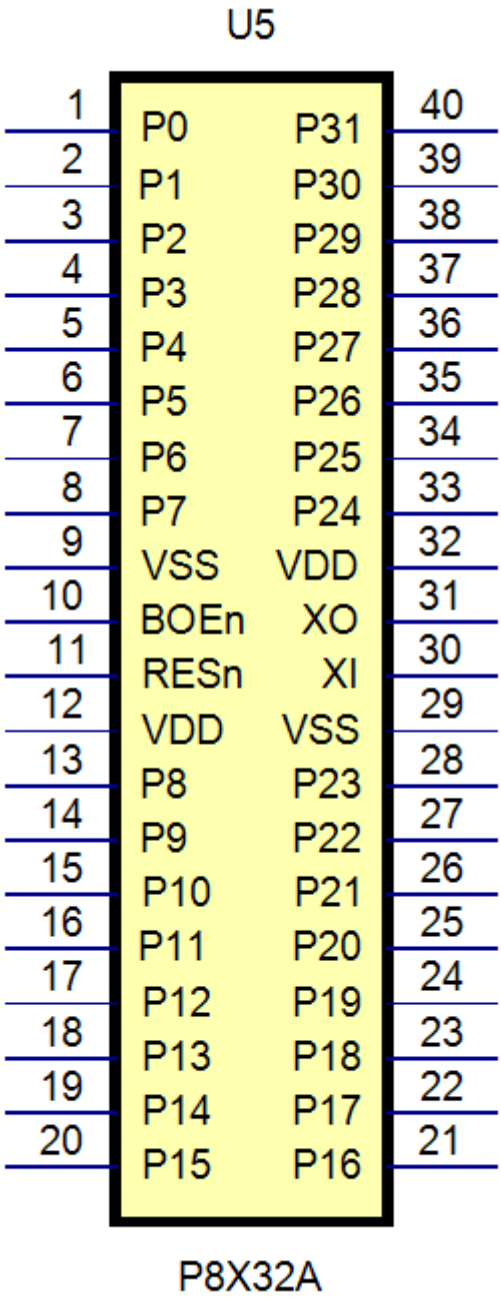
Most parts consist of a single symbol.

Multiple Symbols

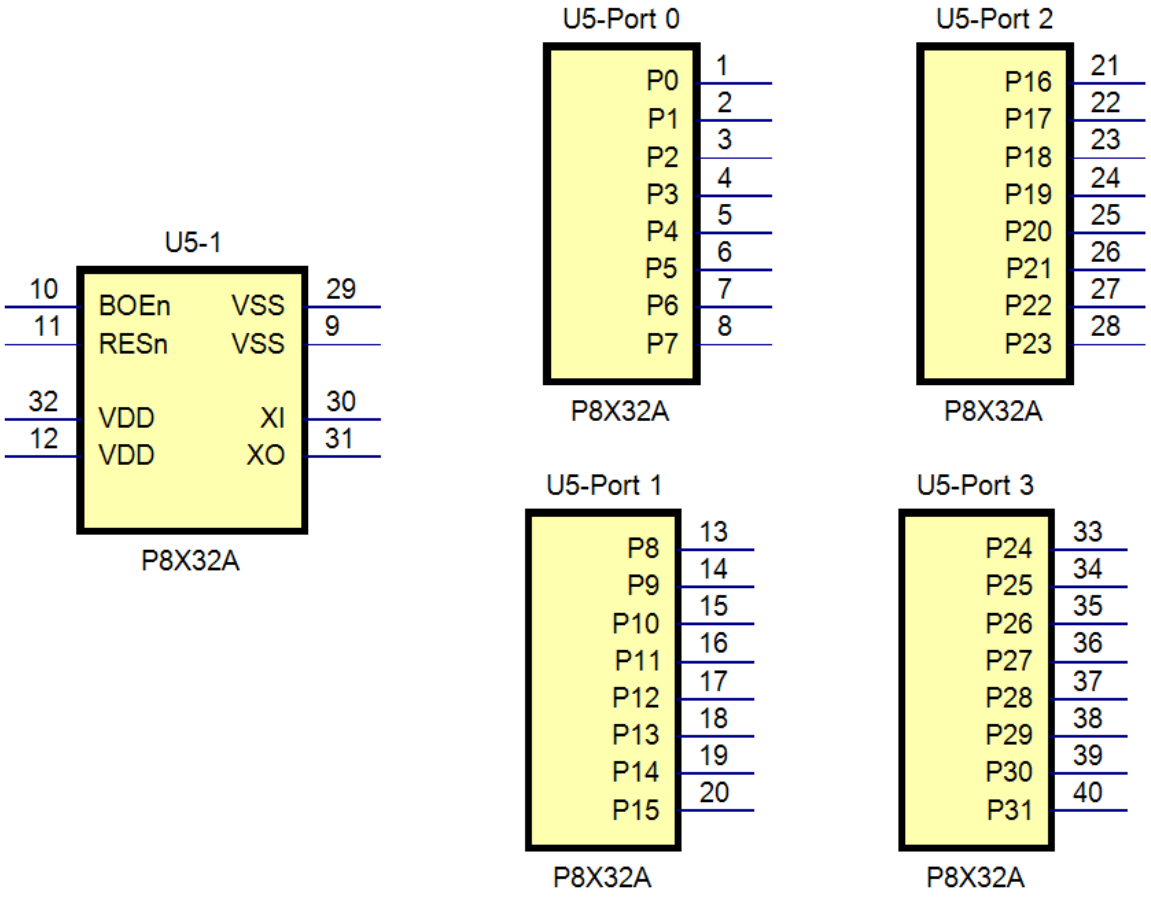
A part can have more than one symbol. Here you have a symbol that has been represented by 4 different logical get symbols and one power connection symbol using a terminal magnet.



Or you can split a symbol as many different symbols as you wish. For instance below the single symbol on the left has been split into five separate symbols whether functionality has been grouped into each symbol. This makes the understanding of the design a lot easier. You can split a symbol into multiple symbols at you have placed a symbol onto a schematic sheet.



The single symbol

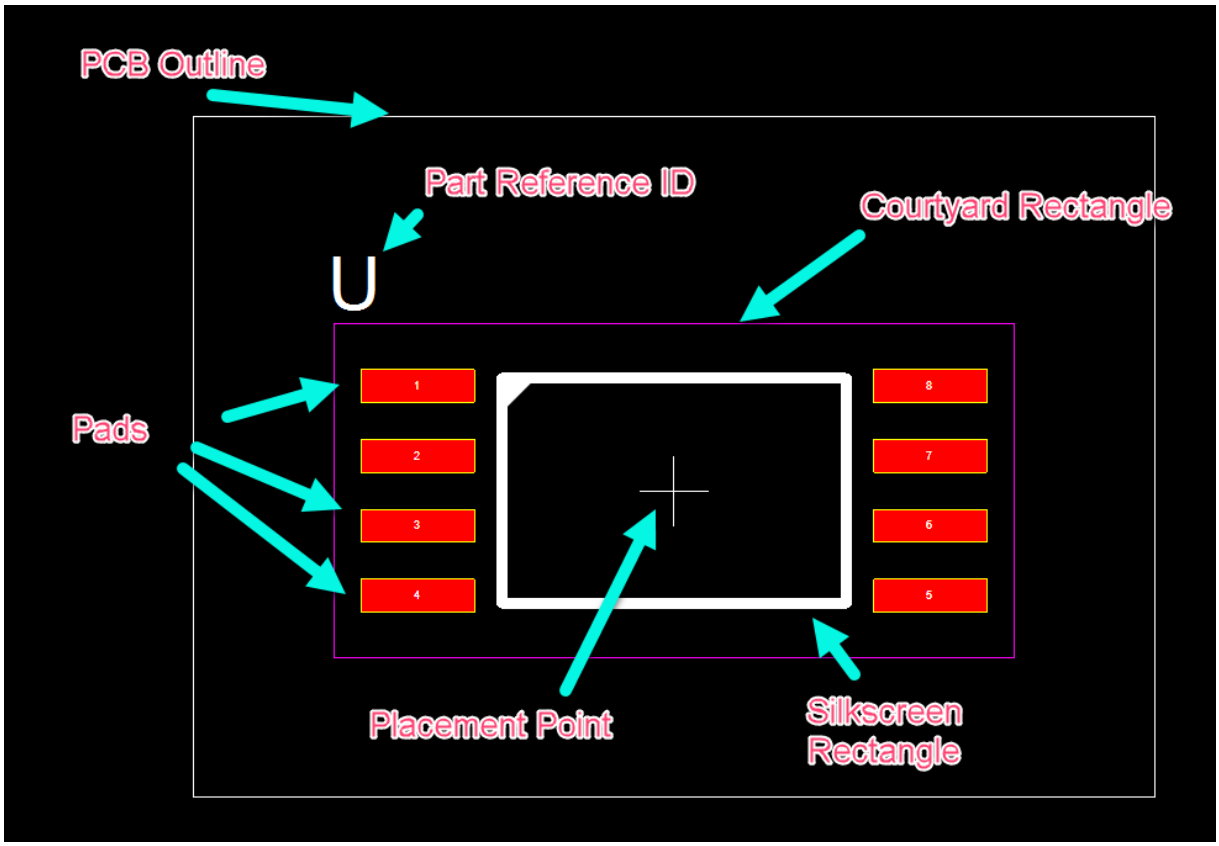


The above symbol split into five separate symbols to clearly reflect the functionality of the device

Footprints or Land-Patterns

A footprint is literally the footprint of a part on the PCB. The most obvious part of the footprint is the collection of electrical contact pads the electrically connect the electrical terminals of a part to copper tracks which also connect to contact pads belonging to the footprints of other parts.

In addition the contact pads act as 'glue' point to stick the parts to the PCB. The 'glue' is not an organic based adhesive but instead is an alloy (solder) that melts at just above 200°C and bonds to the copper pads and the leads of the part.



In addition to the contact pads (pads) the footprint also have:

- A Footprint Reference ID
- A Placement Point
- A Silkscreen Pattern
- A Courtyard rectangle

The 3D Model

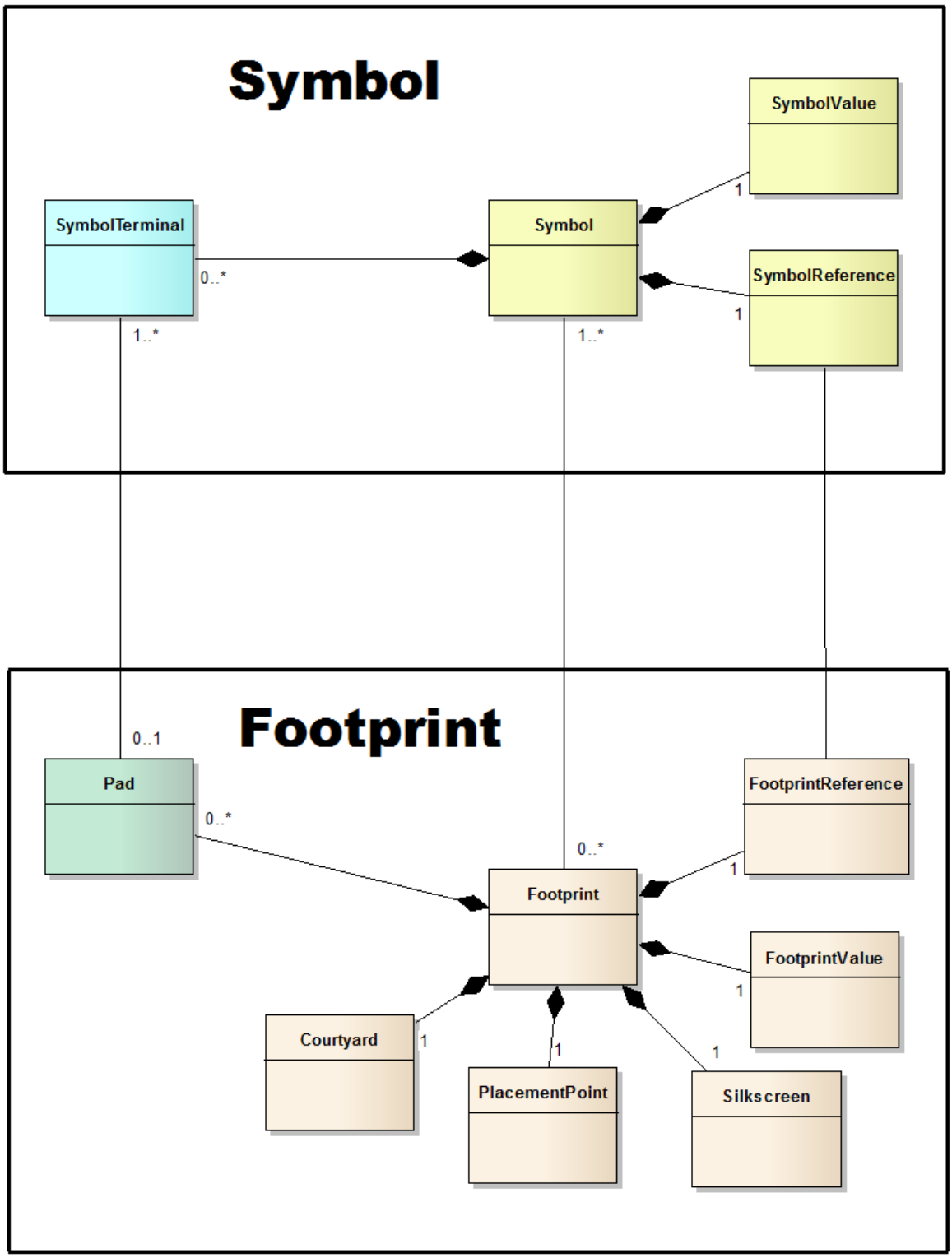
Each part usually has an associated 3D model to represent the part in the 3D view of the PCB. Parametric parts generate the 3D model automatically saving you a great deal of time - 3D models are difficult and time consuming to make. On the right is a 3D model for a capacitor that has been automatically created from the parameters defining the parts.

3D models can be very large in memory size and can greatly increase the size of your part and project files by several megabytes for each part. The beauty of parametric parts is that the model is generated 'on the fly' when needed from the parameters of the parts. This greatly reduces the storage required from several megabytes to just 10's of bytes.

It is also possible to import a 3D model either created by yourself or somebody in your team or imported from the Internet.

1.2.5.2 Part

The class diagram for the main classes in a AutoTRAX DEX part are shown below.



1.2.5.3 Tutorials

Overview of Parts

Overview of Parts in AutoTRAX DEX

Parametric Parts

AutoTRAX DEX comes with many parametric parts. A parametric part is a part than can automatically generate a 2D footprint/land pattern and a matching 3D model. All from a few simple parameters.

Creating a Capacitor

This video show you how to quickly create a custom block capacitor.

Creating a part for an ATMEGA16U2

This video show you how to quickly create a part for a Microchip ATMEGA16U2 high-performance, low-power Microchip 8-bit AVR RISC-based microcontroller.

Importing 3D Models To Use in a PCB Part

This video shows you how to import a 3-D model from an external source into AutoTRAX DEX. The part is also modified and the materials changed to give a more realistic appearance.

Automatic Generation of 3D model PCB Header

This video show the header produced by AutoTRAX DEX.

Renumbering Pads for PCB parts

This video show you how to easily renumber pads in either the part editor or on actual PCBs by simply clicking on each pad in turn.

Create a PCB Footprint by Tracing From a PDF

This video show you how to create a PCB footprint from an image captured from a PDF displayed in a browser. The image is copied and pasted into the footprint designer and pads are traced over the image to produce the final footprint. This uses AutoTRAX DEX.

Creating a Part for a Button Battery

This video shows you how to create the part for a button battery in AutoTRAX DEX. It shows you how to create the footprint/land pattern, the symbol and how to import a 3D model.

Creating a Custom SOT223 Transistor

This video show you how to create a custom SOT223 for a BSP16T1 High Voltage Transistor in AutoTRAX DEX .

Add a Thermal Pad to a Footprint

This video shows you how to add a thermal pad to a footprint to help conduct heat away from the device.

Creating a Power Mosfet Part in AutoTRAX DEX

This video show you how to create a power mosfet device which uses both a custom pad configuration and a custom mosfet symbol.

Creating round, rectangular and polygonal pads and offsetting pad holes

This video shows you how to add round/elliptical, rectangular and polygonal pads to a custom part footprint/land pattern. It also shows you how to offset pad holes.

Creating a custom potentiometer part

This video show you how to create a part for a Bourns 3362 potentiometer. It show you how to create the custom footprint/land pattern and the custom schematic symbol. It also show you how to import a 3D model and position in. Finally it shows the device being used on a PCB.

Creating a part with 4 schematic symbols (Split part)

This video show you how to create a 4000 CMOS logic device (two triple input NOR gates and an inverter) using AutoTRAX DEX. This demonstrates how to create probably one of the most complex parts for AutoTRAX DEX. Once you master how to create one of these, all other parts will seem easy.

Customizing a parametric footprint

This video show you how to take a 4x5 quad AutoTRAX DEX device and add pads and modify pads. It also show you how to add 3D objects.

Capturing Pin Details from a PDF

This demonstration will show you how to quickly capture pin details for a schematic symbol from a PDF document. This will also work with text documents.

How to modify a placed part using the library

This video shows you how to modify a part by saving it to the library, editing the part in the library and then dragging the modified part over your original part.

1.2.5.3.1 Overview of Parts

Overview of Parts in AutoTRAX DEX

1.2.5.3.2 Parametric Parts

AutoTRAX DEX comes with many parametric parts. A parametric part is a part than can automatically generate a 2D footprint/land pattern and a matching 3D model. All from a few simple parameters.

1.2.5.3.3 Creating a Capacitor

This video show you how to quickly create a custom block capacitor.

1.2.5.3.4 Creating a part for an ATMEGA16U2

This video show you how to quickly create a part for a Microchip ATMEGA16U2 high-performance, low-power Microchip 8-bit AVR RISC-based microcontroller.

1.2.5.3.5 Importing 3D Models To Use in a PCB Part

This video shows you how to import a 3-D model from an external source into AutoTRAX DEX. The part is also modified and the materials changed to give a more realistic appearance.

1.2.5.3.6 Automatic Generation of 3D model PCB Header

This video show the header produced by AutoTRAX DEX/Active3D.

1.2.5.3.7 Renumbering Pads for PCB parts

This video show you how to easily renumber pads in either the part editor or on actual PCBs by simply clicking on each pad in turn.

1.2.5.3.8 Create a PCB Footprint by Tracing From a PDF

This video show you how to create a PCB footprint from an image captured from a PDF displayed in a browser. The image is copied and pasted into the footprint designer and pads are traced over the image to produce the final footprint. This uses AutoTRAX DEX

1.2.5.3.9 Creating a Part for a Button Battery

This video shows you how to create the part for a button battery in AutoTRAX DEX It shows you how to create the footprint/land pattern, the symbol and how to import a 3D model.

1.2.5.3.10 Creating a Custom SOT223 Transistor

This video show you how to create a custom SOT223 for a BSP16T1 High Voltage Transistor in AutoTRAX DEX

1.2.5.3.11 Add a Thermal Pad to a Footprint

This video shows you how to add a thermal pad to a footprint to help conduct heat away from the device.

1.2.5.3.12 Creating a Power Mosfet Part

This video show you how to create a power mosfet device which uses both a custom pad configuration and a custom mosfet symbol.

1.2.5.3.13 Creating round, rectangular and polygonal pads and offsetting pad holes

This video shows you how to add round/elliptical, rectangular and polygonal pads to a custom part footprint/land pattern. It also shows you how to offset pad holes.

1.2.5.3.14 Creating a custom potentiometer part

This video show you how to create a part for a Bourns 3362 potentiometer. It show you how to create the custom footprint/land pattern and the custom schematic symbol. It also show you how to import a 3D model and position in. Finally it shows the device being used on a PCB.

1.2.5.3.15 Creating a part with 4 schematic symbols (Split part)

This video show you how to create a 4000 CMOS logic device (two triple input NOR gates and an inverter) using AutoTRAX DEX This demonstrates how to create probably one of the most complex parts for AutoTRAX DEX Once you master how to create one of these, all other parts will seem easy.

1.2.5.3.16 Customizing a parametric footprint

This video show you how to take a 4x5 quad AutoTRAX DEX device and add pads and modify pads. It also show you how to add 3D objects.

1.2.5.3.17 Capturing Pin Details from a PDF

This demonstration will show you how to quickly capture pin details for a schematic symbol from a PDF document. This will also work with text documents.

1.2.5.3.18 How to modify a placed part using the library

This video shows you how to modify a part by saving it to the library, editing the part in the library and then dragging the modified part over your original part.

1.2.5.4 Don't Trust Part Libraries

You should never trust a parts library; even your own. You must always check each and every part that you use in your design.

Parts contained quite a few very specific details and when you have a library of tens of thousands of parts it is very easy for one of the parts to have an error. This error could easily cause your PCB not to work and will cost you both time and money to fix. So, check, check and check again.

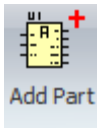
1.2.5.5 The Parts Catalog

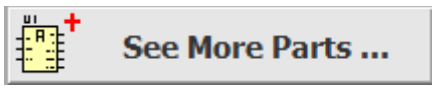
The Parts Catalog is a collection of over 47,000 parts that you can use use in your designs by simply dragging the part from the Add Parts Dialog onto any of your schematics.

➔*All parts come complete with symbols, footprint/land pattern and 3D model.*

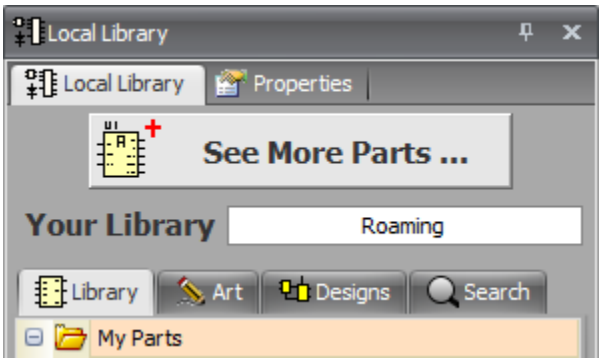
Adding Parts from the Parts Catalog

➔To display the Add Parts Dialog:

- Click the 

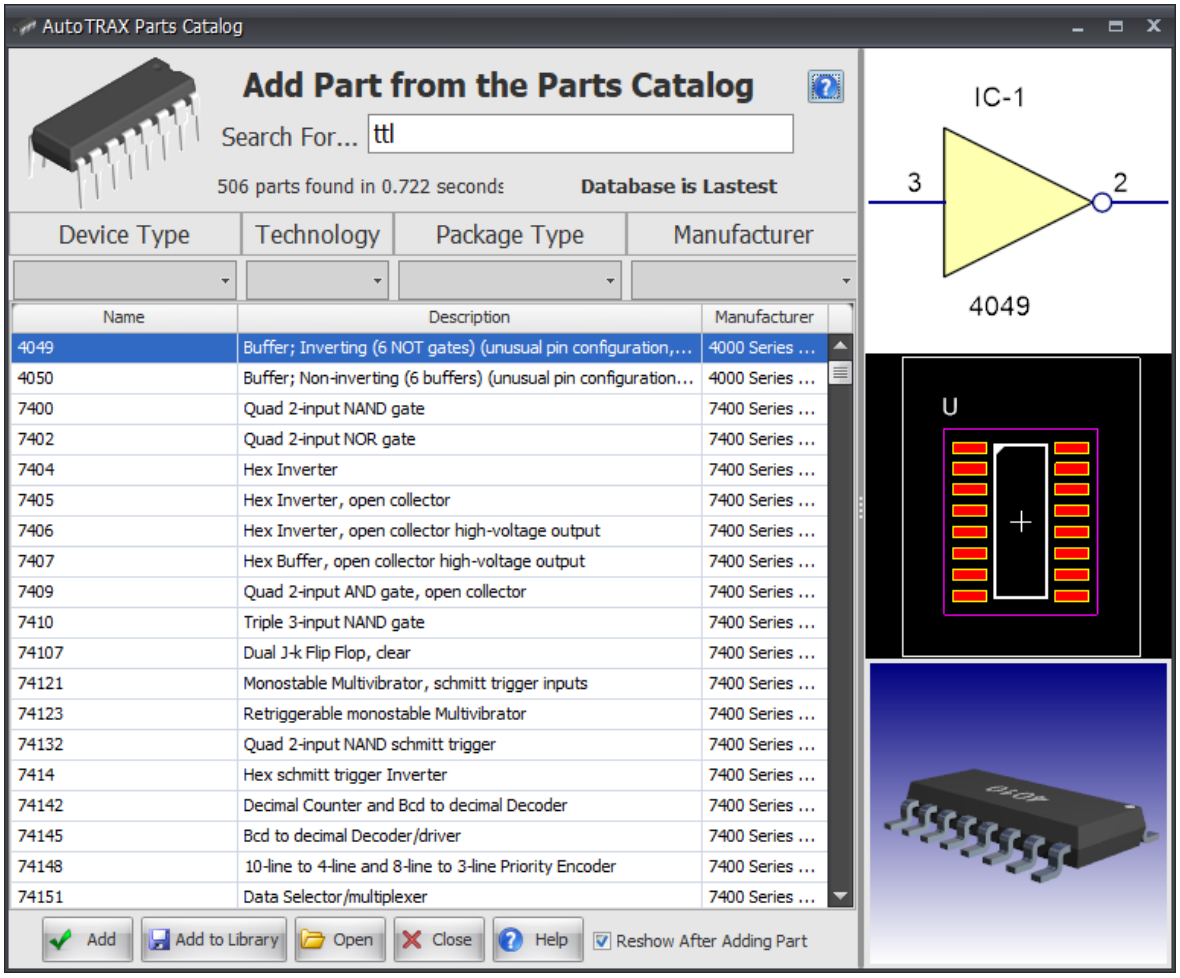
button in the Parts->Library ribbon menu or
- Click the 

button in the Library Panel

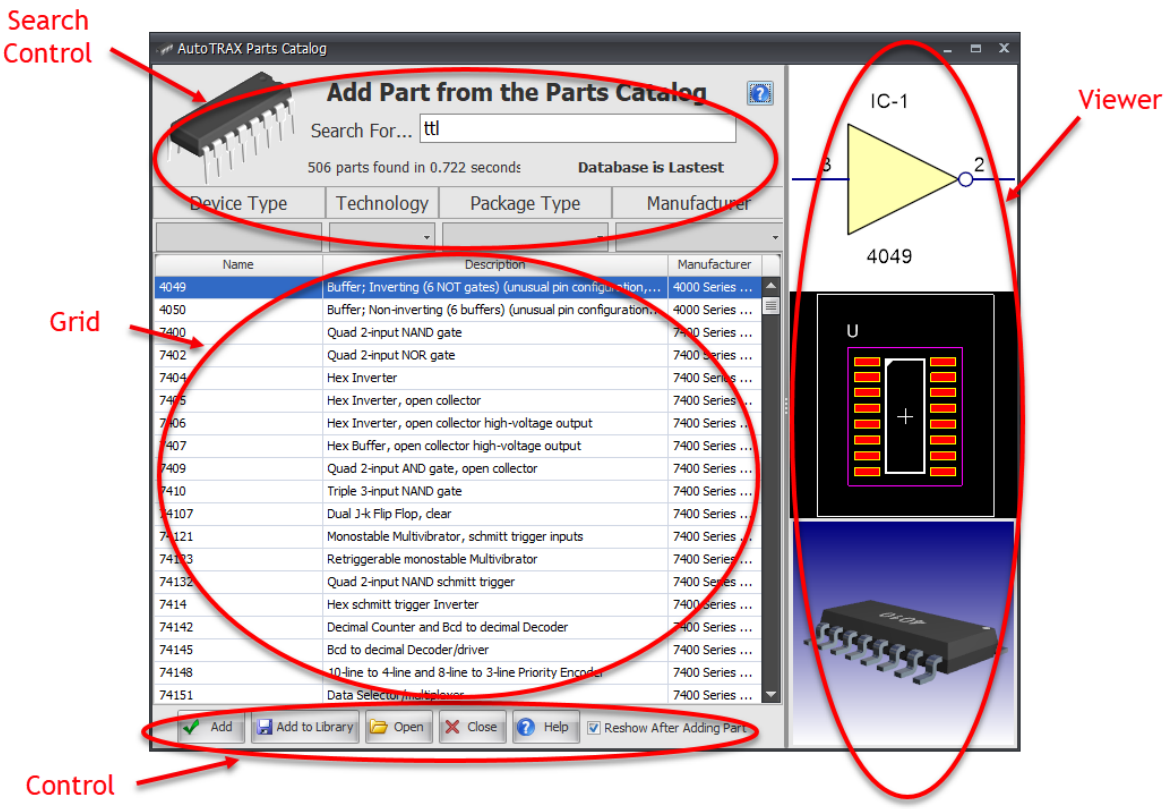


The Add Parts Dialog box is shown below.

The Library Panel



The Add Parts Dialog



The Add Parts Dialog Details

Updating the Parts Catalog

➔AutoTRAX DEX automatically checks for a update to the Parts Catalog and will automatically download it. The update is done in the background. You need to be connected to the Internet for the check and download to work.

Manually Downloading the Parts Catalog

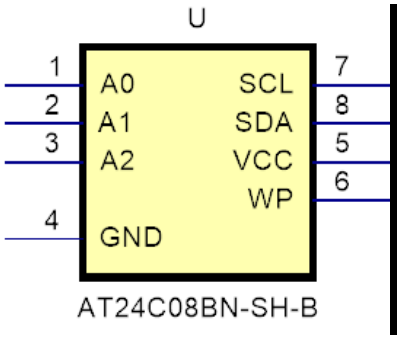
➔You can manually download the Parts Catalog setup program here.
https://pcbdex.com/Download/Parts

Previewing a Part

You can preview a part by clicking on a line in the Grid View for the part. 3 views will display:

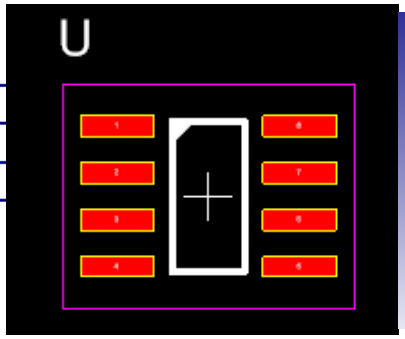
The Symbol for the Part

If the part has more than one symbol, only the first one will be displayed.



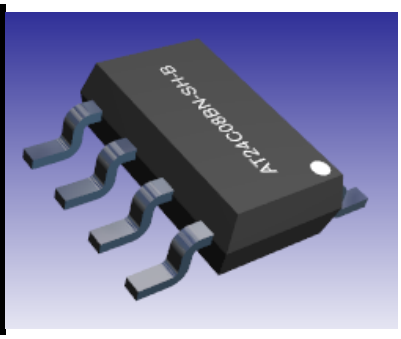
The Part's Symbol

The Footprint / Land Pattern



The Parts Footprint

The Part in 3D



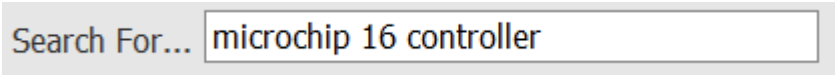
3D View of the Part

You can pan/zoom all 3 viewports using the mouse

Keyword Search

The Add Parts Dialog is a modeless dialog that lets you search for parts using one or more keywords and up to 4 filters.

➔You enter the search keywords in the Search For text input box.



Enter the keywords and separate each one with 1 or more spaces.

Each keyword does not have to be a full match for the word you are looking for. The keyword can be a partial match. For instance alleg will match Allego and all other words containing alleg,

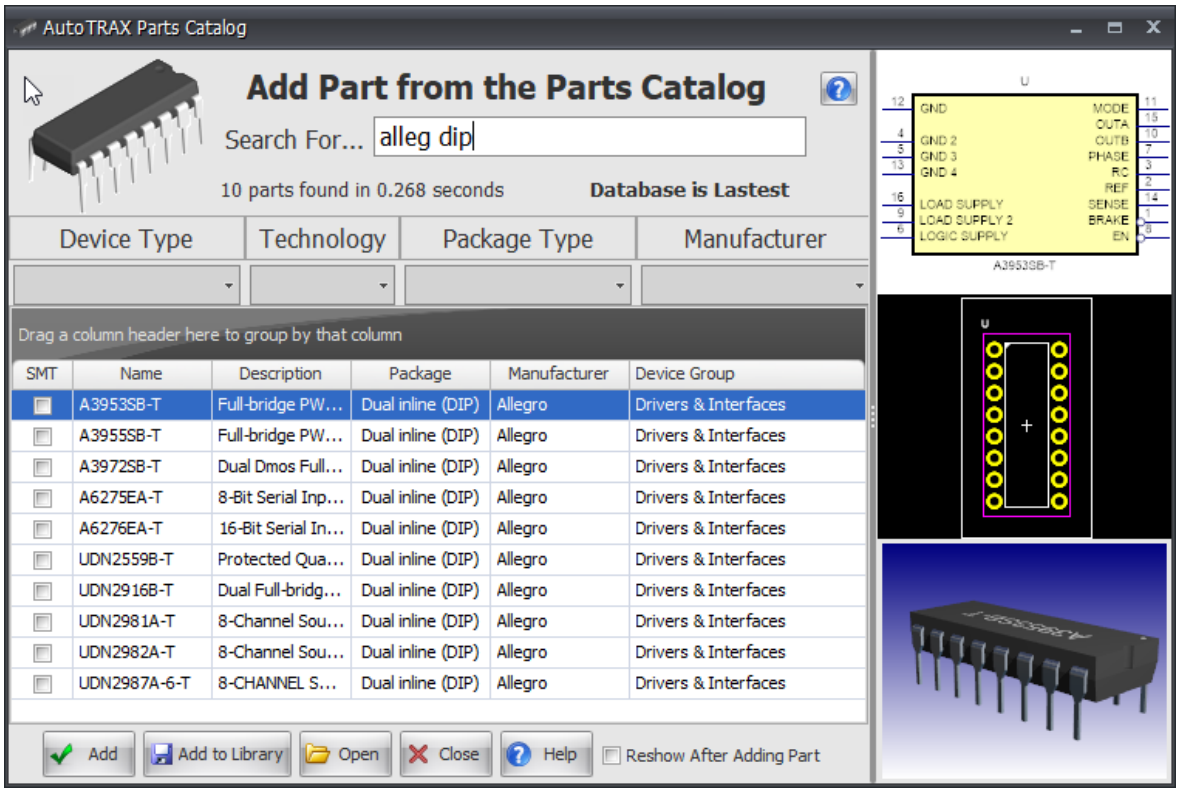
The case of the keyword does not matter.

The search is made in the following parameters for all parts.

- Name

- Description
- Package Type
- Device Group
- Manufacturer
- Technology

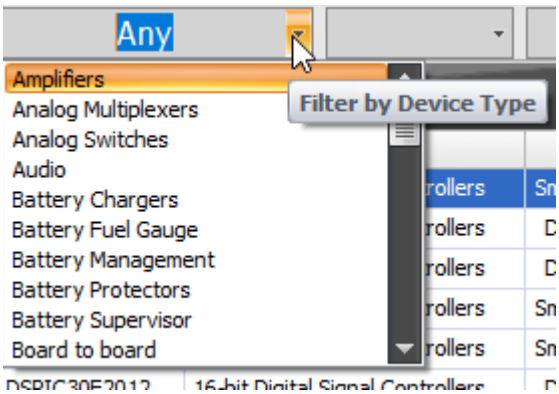
If you enter 2 or more keywords then each keyword must find a match. For instance searching for ***alleg dip*** match all parts with ***alleg*** and ***dip*** in their details as shown below.



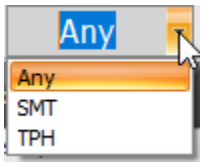
Filters

➔You can filter your search for parts by:

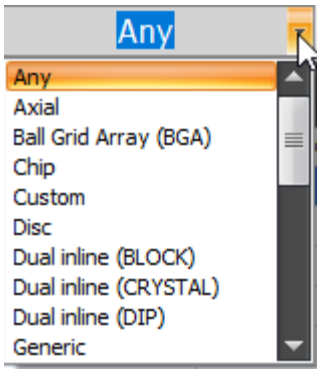
- **Device Types**



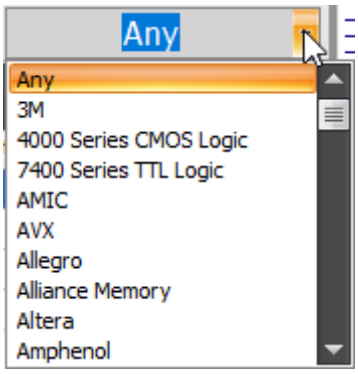
- **Technology**



- **Package Type**



- **Manufacturer**



To add any of the above 4 filter types, click on the small down arrow at the right hand side of the box below the title of the filter type.

A pop-up list of filters will be displayed: click on one of them and the list of parts found will be updated automatically.

You probably will not use the filters as you can use keywords instead.

The Grid View

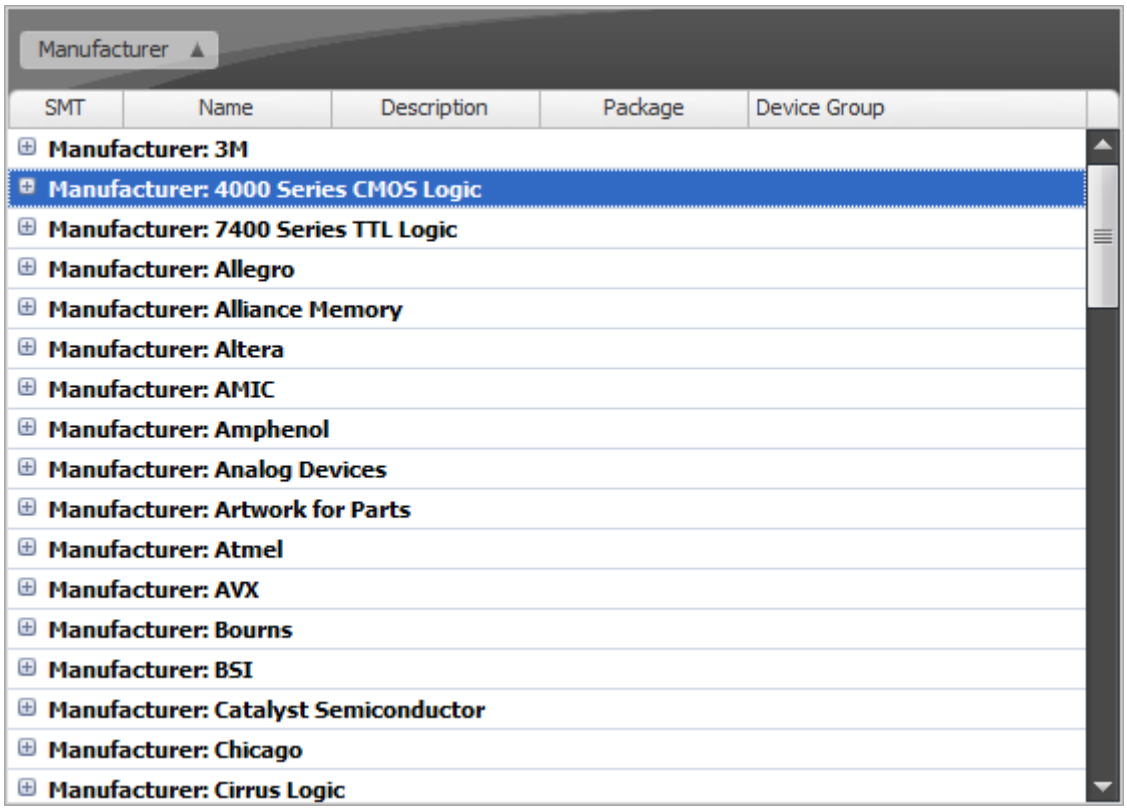
➔All parts found are displayed in the Grid View.

Name	Description	Package	Manufacturer	SMT
DSPIC30F2010-20I-SO	16-bit Digital Signal Controllers	Small Outline (SOIC)	Microchip	☑
DSPIC30F2010-20I-SP	16-bit Digital Signal Controllers	Dual inline (DIP)	Microchip	☐
DSPIC30F2011-20I-P	16-bit Digital Signal Controllers	Dual inline (DIP)	Microchip	☐
DSPIC30F2011-20I-SO	16-bit Digital Signal Controllers	Small Outline (SOIC)	Microchip	☑
DSPIC30F2012-20I-SO	16-bit Digital Signal Controllers	Small Outline (SOIC)	Microchip	☑
DSPIC30F2012-20I-SP	16-bit Digital Signal Controllers	Dual inline (DIP)	Microchip	☐
DSPIC30F2012-30I-SO	16-bit Digital Signal Controller	Small Outline (SOIC)	Microchip	☑
DSPIC30F2012-30I-SP	16-bit Digital Signal Controllers	Dual inline (DIP)	Microchip	☐
DSPIC30F3010-20I-ML	16-Bit Digital Signal Controllers	Quad Flatpack (QFP)	Microchip	☑
DSPIC30F3010-30I-SO	16-bit Digital Signal Controller	Small Outline (SOIC)	Microchip	☑
DSPIC30F3010-30I-SP	16-Bit Digital Signal Controllers	Dual inline (DIP)	Microchip	☐
DSPIC30F3011-20I-P	16-Bit Digital Signal Controllers	Dual inline (DIP)	Microchip	☐
DSPIC30F3011-20I-PT	16-Bit Digital Signal Controllers	Quad Flatpack (QFP)	Microchip	☑

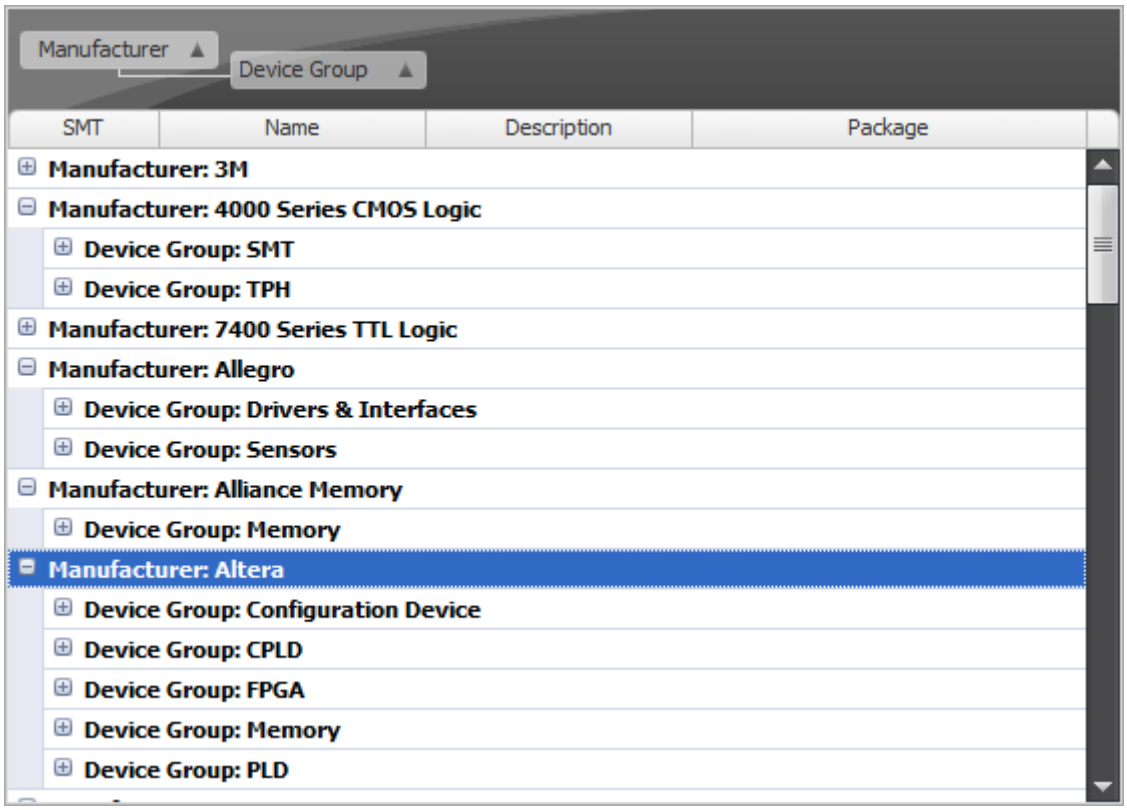
Typical Grid View

➔Click on a row to select a part

Drag a column header to the top to group by the column. You can drag more than 1 column

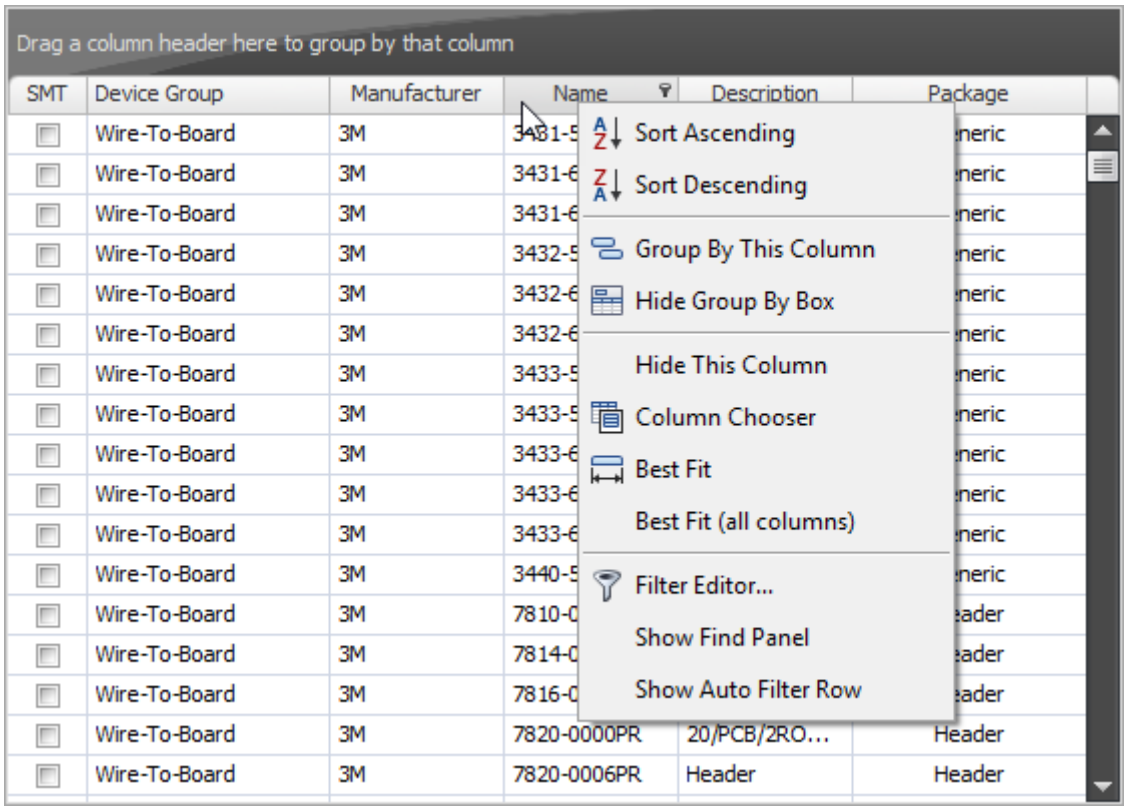


Grouping by Manufacturer



Grouping by Manufacturer Then by Device Group

Right click on the column header for Context menu for column options

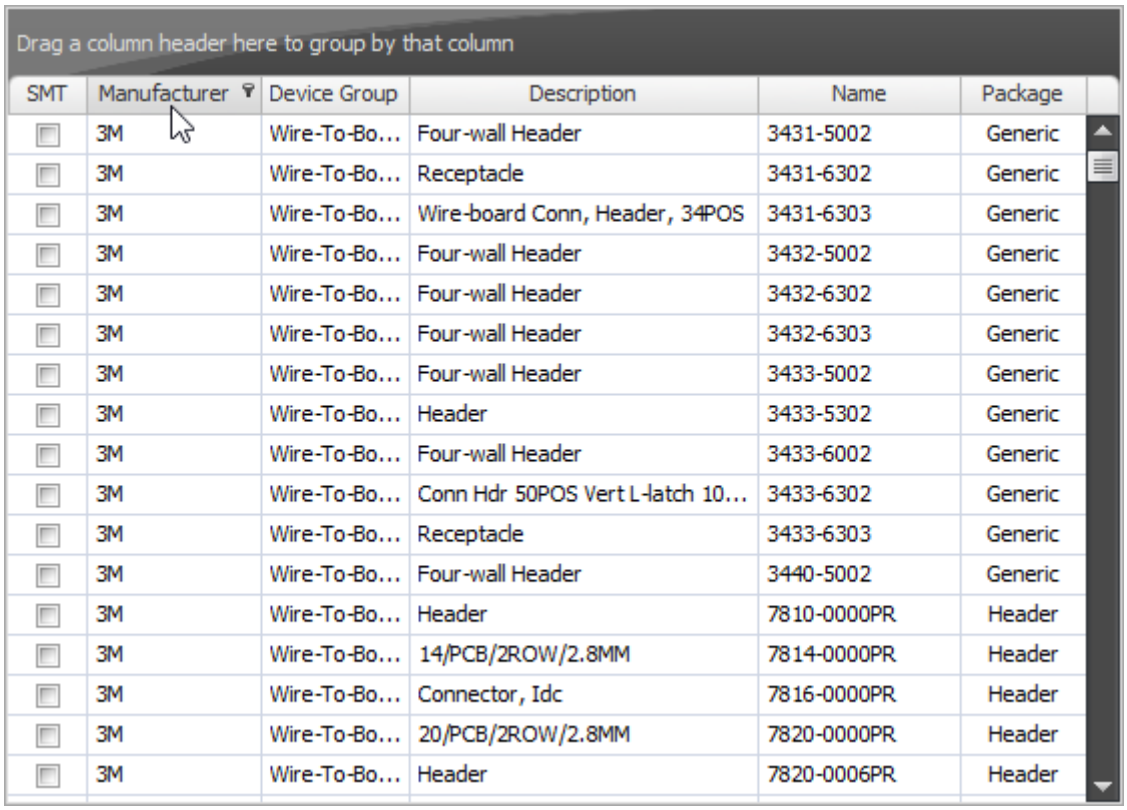


Right Click Context Menu

Name	Description	Manufacturer
7826-0006PR	Header	3M
7834-0000PR	34/PCB/2ROW/2.8MM	3M
7834-0006PR	Header	3M
7840-0000PR	40/PCB/2ROW/2.8MM	3M
7850-0000PR	50/PCB/2ROW/2.8MM	3M
7850-0006PR	Header	3M
7860-0000PR	60/PCB/2ROW/2.8MM	3M
N2510-6V0C-RB-WD	Header	3M
N2520-6V0C-RB-WE	Header	3M
N2534-6V0C-RB-WF	Header	3M
N3314-5202RB	Conn Header 14PS R/A Short Latch	3M
N3314-5302RB	Wall Header	3M
N3314-6002RB	Wire-board Conn, Header, 14POS	3M
N3314-6202RB	Header, Straight, L/latch, 14WAY	3M
N3314-6302RB	Conn Header 14POS STR Long Latch	3M
N3372-5002RB	Wire-board Conn, Header, 60POS	3M
N3372-5302RB	Header, Right Angle, S/latch, 60WAY	3M
N3372-6002RB	Wire-board Conn, Header, 60POS	3M
N3372-6302RB	Conn Header 60POS STR Long	3M

Reordered Columns and Some Columns Hidden

Drag a column to reorder columns and/or re-size columns



Drag a column header here to group by that column

SMT	Manufacturer	Device Group	Description	Name	Package
	3M	Wire-To-Bo...	Four-wall Header	3431-5002	Generic
	3M	Wire-To-Bo...	Receptacle	3431-6302	Generic
	3M	Wire-To-Bo...	Wire-board Conn, Header, 34POS	3431-6303	Generic
	3M	Wire-To-Bo...	Four-wall Header	3432-5002	Generic
	3M	Wire-To-Bo...	Four-wall Header	3432-6302	Generic
	3M	Wire-To-Bo...	Four-wall Header	3432-6303	Generic
	3M	Wire-To-Bo...	Four-wall Header	3433-5002	Generic
	3M	Wire-To-Bo...	Header	3433-5302	Generic
	3M	Wire-To-Bo...	Four-wall Header	3433-6002	Generic
	3M	Wire-To-Bo...	Conn Hdr 50POS Vert L-latch 10...	3433-6302	Generic
	3M	Wire-To-Bo...	Receptacle	3433-6303	Generic
	3M	Wire-To-Bo...	Four-wall Header	3440-5002	Generic
	3M	Wire-To-Bo...	Header	7810-0000PR	Header
	3M	Wire-To-Bo...	14/PCB/2ROW/2.8MM	7814-0000PR	Header
	3M	Wire-To-Bo...	Connector, Idc	7816-0000PR	Header
	3M	Wire-To-Bo...	20/PCB/2ROW/2.8MM	7820-0000PR	Header
	3M	Wire-To-Bo...	Header	7820-0006PR	Header

Columns Reordered and Resized

Adding Parts to Your Design

You must add the parts to a schematic, not to a PCB or 3D PCB view. In AutoTRAX DEX, schematics are central to designs. PCB are derived from information in schematics together with further input directly to the PCB, such as part placement, layer details, PCB outline and routing.

Double Click - recommended

To add a part to a design using double click :

1. Double click the left mouse button any part of a line describing the part in the Grid View
2. The Add Parts Dialog will be hidden if it obscures the target schematic and you will see the symbol for the part being dragged inside the schematic
3. Click the left mouse button to place the part's symbol. The footprint is automatically added to the PCB
4. If you have **Reshow After Adding Part** checked, the Add Parts Dialog will appear and you can then add more parts or close the dialog

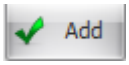
Drag and Drop

To add a part to a design using drag and drop:

1. Hold down the left mouse button over any part of a line describing the part in the Grid View and drag the mouse
2. Once drag as started, the Add Parts Dialog will be hidden if it obscures the target schematic and you will see the symbol for the part being dragged inside the schematic
3. Release to left mouse button to place the part's symbol. The footprint is automatically added to the PCB
4. If you have **Reshow After Adding Part** checked, the Add Parts Dialog will appear and you can then add more parts or close the dialog

Click the Add Button

To add a part to a design using the add button :

1. Click the 

button any part of a line describing the part in the Grid View
2. The Add Parts Dialog will be hidden if it obscures the target schematic and you will see the symbol for the part being dragged inside the schematic
3. Click the left mouse button to place the part's symbol. The footprint is automatically added to the PCB
4. If you have **Reshow After Adding Part** checked, the Add Parts Dialog will appear and you can then add more parts or close the dialog

After you have added a part:

- **Get the Datasheet**

Then check and check again:

- Symbol Terminals Names
- Pad Names
- Footprint dimensions
- Symbol terminals are connected to the correct Pads

There is nothing worse than getting a collection of board back from the manufacturer and having to rework them to fix errors.

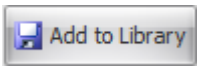
Reshowing the Add Parts Dialog After Adding Parted

Check the 

checkbox to automatically reshow the Add Parts Dialog after a part is added.

Adding Parts to Your Local Library

To add a part to your local library:

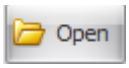
1. Select the part by clicking on the Grid View line describing the part
2. Click the 

button
3. You will be then prompted for the file directory to save it to
4. Select the directory
5. Set the part name

1. Click 

Editing a Part

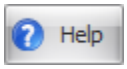
You can edit and part ans save it to any location on your computer or network.

1. Select the part by clicking on the Grid View line describing the part
2. Click the 

button to edit the part
3. Save the part

You cannot save the part back to the Parts Catalog, The Parts Catalog is read only.

Getting Help

➔Clicking the  button will display this help

The SQLite Database

➔AutoTRAX DEX uses SQLite:

- SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine.
- SQLite is the most used database engine in the world.
- SQLite is built into all mobile phones and most computers and comes bundled inside countless other applications that people use every day.

The Parts Catalog is contained in a single database file DEXPartsCatalog.db. The database file is located in:

*C:\Users**XXX**\AppData\Roaming\AutoTRAX DEX Software\AutoTRAX DEX*

where **XXX** is your Windows user name.

:Find out more...

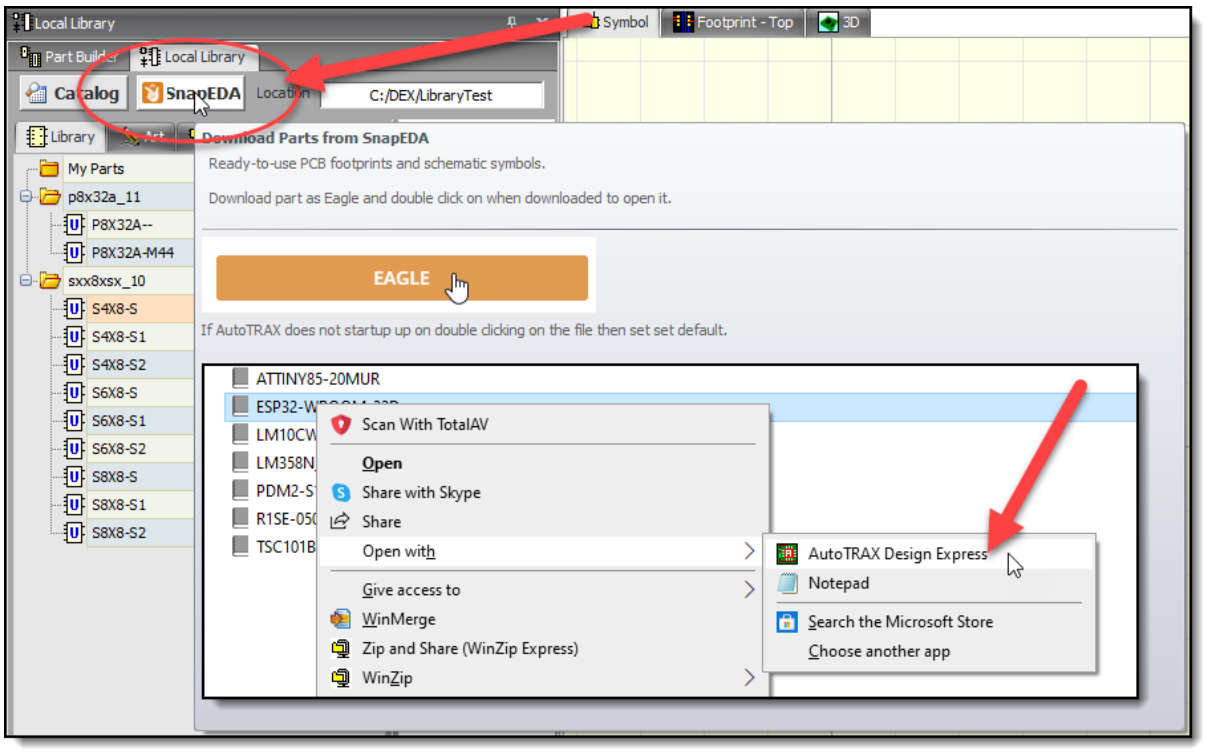
1.2.5.6 Downloading Parts from SnapEDA

Downloading Parts from SnapEDA.com

You can now easily search for parts on SnapEDA.com and quickly download the part into AutoTRAX DEX.

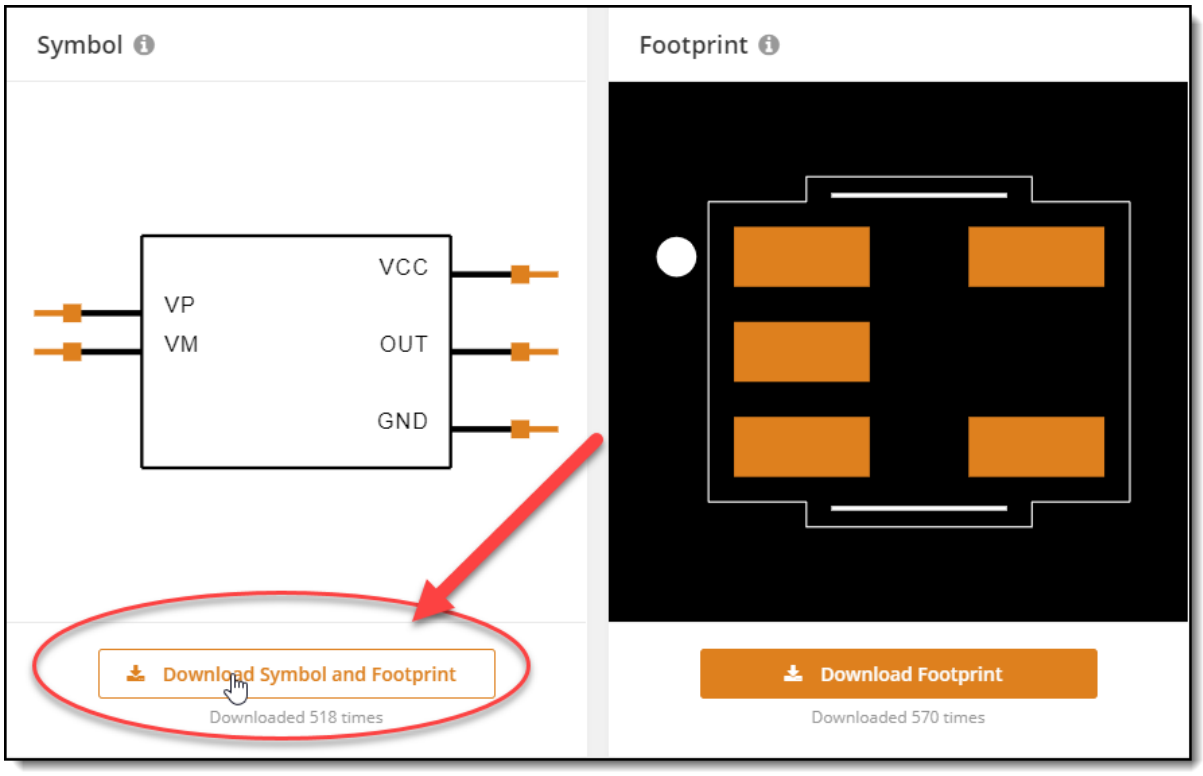
SnapEDA Website

You can go direct to the website by clicking on the SnapEDA button in the Library Panel.



Download File Type

Download both the Symbol and Footprint.

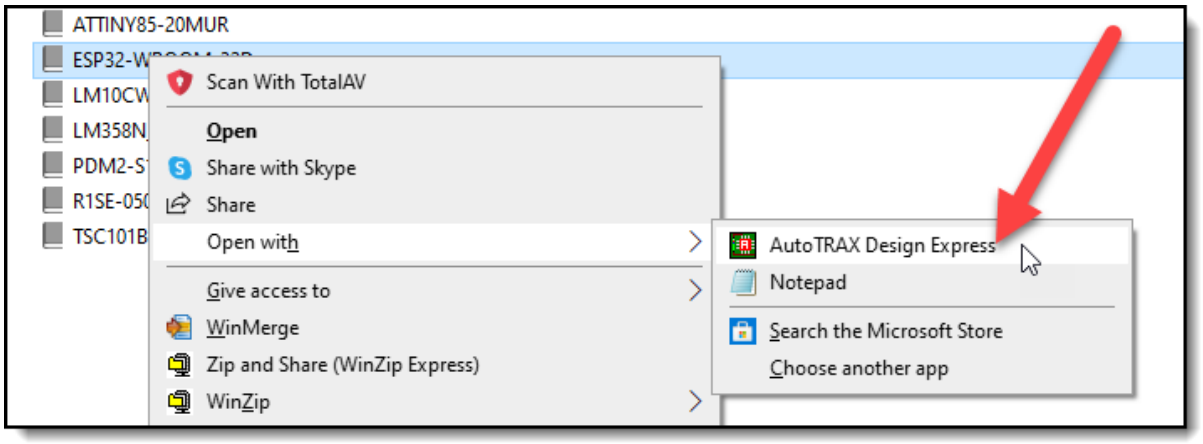


Click on Eagle for the download file type.

Opening the Part

Once downloaded you can open the part by double clicking on it.

If AutoTRAX DEX does not startup the you can set the default app for the file type by right clicking on the file.



Part Requests

You can request any symbol or footprint and have it delivered in under 24 hours.

SnapEDA

We're building the library. You're building the future.

SnapEDA is the Internet's first electronics design library. Our mission is to help engineers build products faster by removing design barriers.

Each year, over 750,000 professional engineers rely on SnapEDA to design faster. They're at small shops, to household names like Google, Facebook, & Samsung. They're building over 192,000 unique products, including medical devices, satellites, drones and so much more.

Since SnapEDA shaves days off development time, engineers can focus more on innovation. They also gain instant transparency into manufacturability with our patented verification technology. You can learn more about our schematic symbol & PCB footprint standards here.

1.2.5.7 Setting The Default Part

You can set the default part template by first creating a part, setting it up exactly as you want all new parts to be and then click the 

in the File Menu. Then, every time you create a new part, it will start off as a mirror image of the default project you saved.

The file is named Default.part and is in the library '___SystemDoNotRemove' directory .

You can always reset the default part to the factory default using the File Settings.

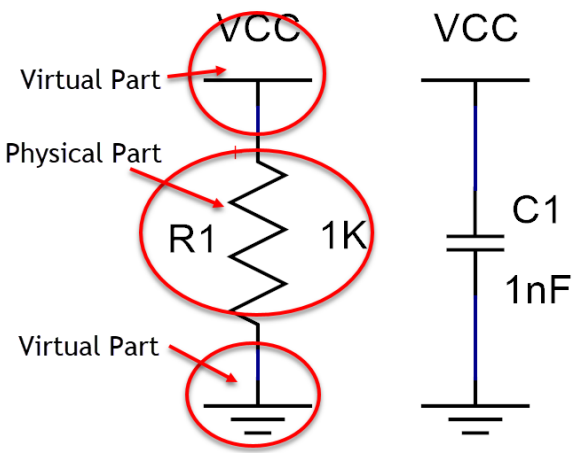
1.2.5.8 Virtual Parts

Virtual parts are parts with no physical package and therefore do not have a footprint. They do not appear on a PCB.

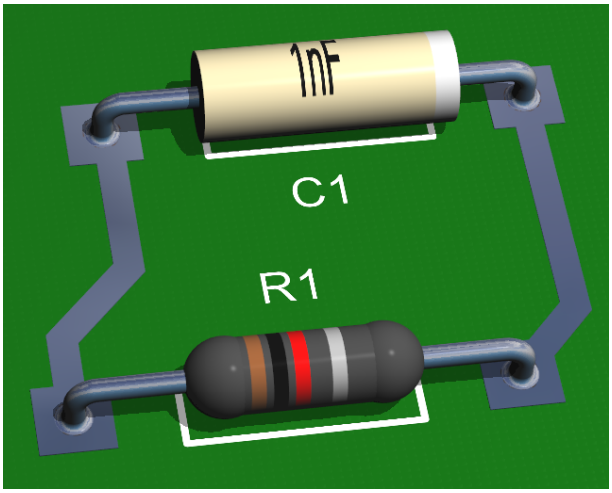
Example of virtual parts are power and ground symbols., See below.

Virtual parts have the following properties...

- You can attach a wire to a terminal in the part. This terminal is usually invisible but it's position is usually at the end of the vertical line at the top or bottom of the part's graphics.
- The terminal has a named node. so that all wires attached to similar symbols will all be electrically connected together e.g. All digital grounds will be connected together using PCB tracks.
- They do not have any electrical pads on the PCB; in fact, nothing is present on the PCB that belongs to a virtual part.
- You connect at least on the them to a terminal on a part that is a physical part. This connection will be to a pad on the parts footprint. Below is a Vcc and an analog ground connected to either end of of resistor.



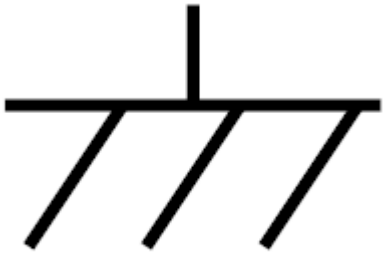
Schematic



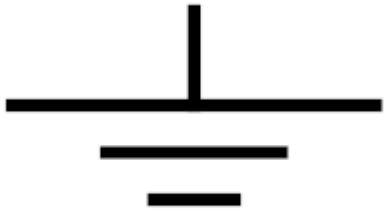
R1 and R2 automatically connected together

Example Virtual Parts

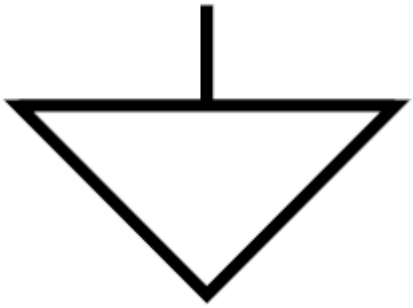
Digital Ground



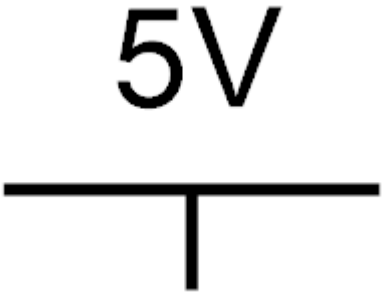
Analog Ground



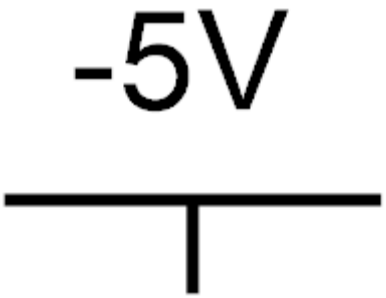
Generic Ground



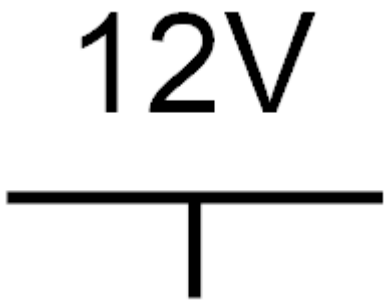
+5V DC



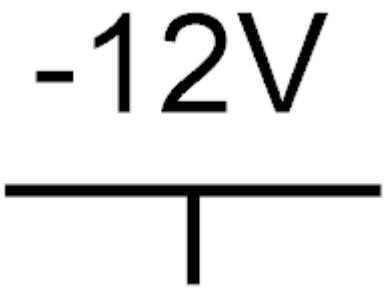
-5V DC



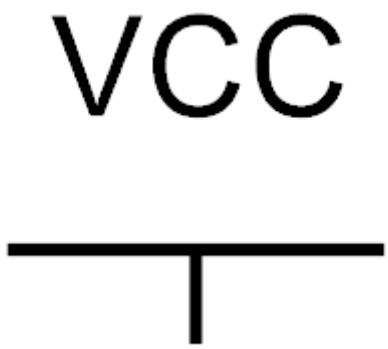
+12V DC



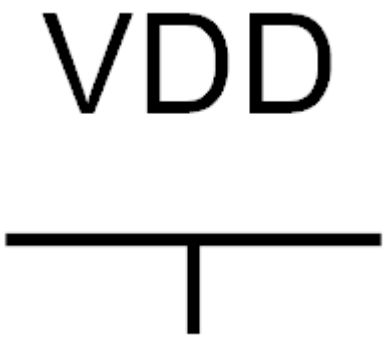
-12V DC



Vcc



Vdd



1.2.5.8.1 Creating a Virtual Part

Compared to creating a physical parts, creating a virtual part.

All you need to do is: (in any order)

- Add 1 or more schematic symbol with named nodes. Usually these are hidden and are placed at then end of straight lines.
- Add graphics to make the purpose of the symbol easy to deduce.

1.2.5.9 Parametric Parts

A parametric part is the encapsulation of all the details needed by AutoTRAX DEX to generate the schematics symbols, footprint and 3D solid model for a real part.

Parametric parts are extremely powerful. From only a small collection of numerical values parametric parts can generate complex graphics including 2-D graphics and 3-D graphics.

To illustrate the concept let's consider a PIC10F200 micro-controller from Microchip.

We will consider the version that comes in an 8-pin PDIP package.

There are 3 parts to consider.

1. The Manufacturer's model and specification of the device.
2. What you actually get.
3. The model used by AutoTRAX DEX to help you incorporate it into the system you are designing.

1.2.5.9.1 The Manufacturers Model

The manufacturer will detail the part they will sell you using a datasheet. The datasheet will describe what the device does and the physical package in which it comes. There may be more than one package for a part.

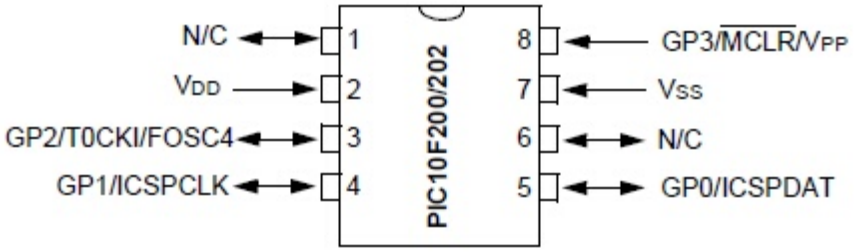
Each package has a set of electrical pins that you must connect to your circuit. The naming, allocation and description of these pins are in the datasheet.

To use a part in AutoTRAX DEX you will need to know its package, its pinout and its recommended footprint.

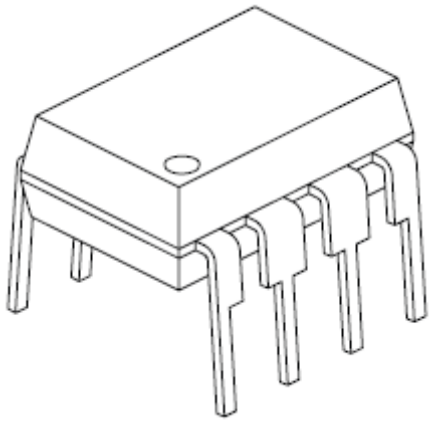
The PIC10F200 micro-controller datasheet is available at https://ww1.microchip.com/downloads/en/DeviceDoc/41239D.pdf

For the Microchip parts there is also an additional packaging specification available at https://ww1.microchip.com/downloads/en/PackagingSpec/00049AU.pdf

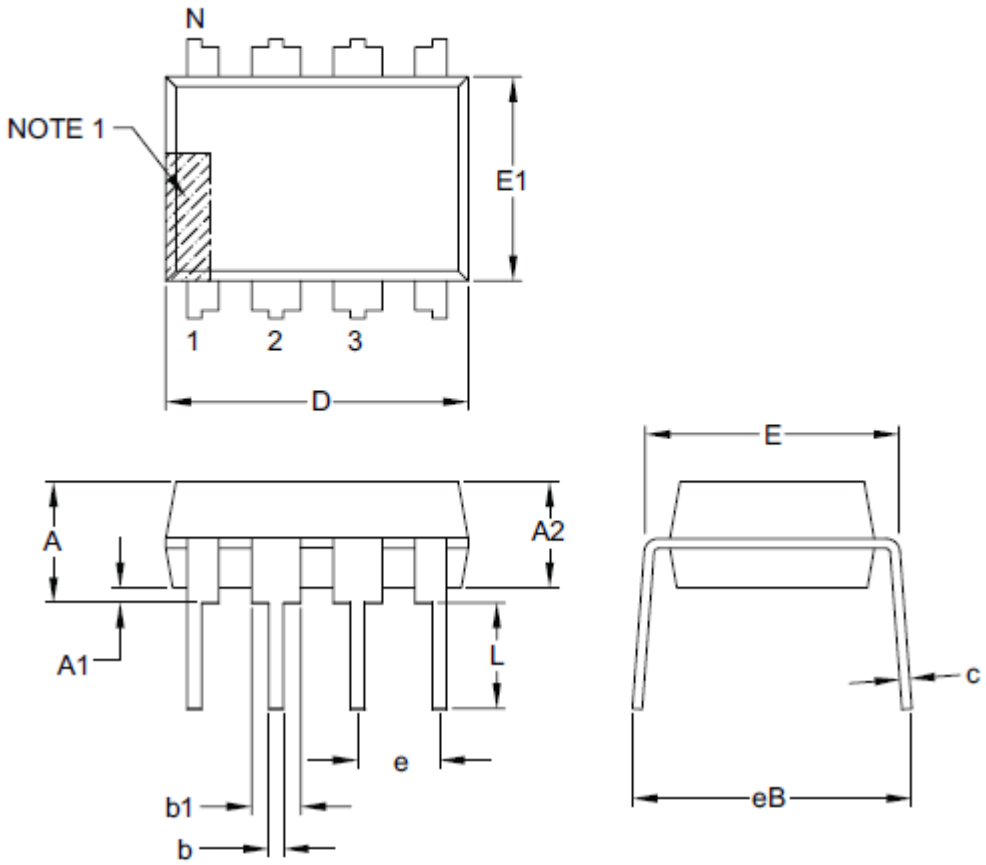
The datasheet has a pinout of the part as shown below.



The physical package is shown below



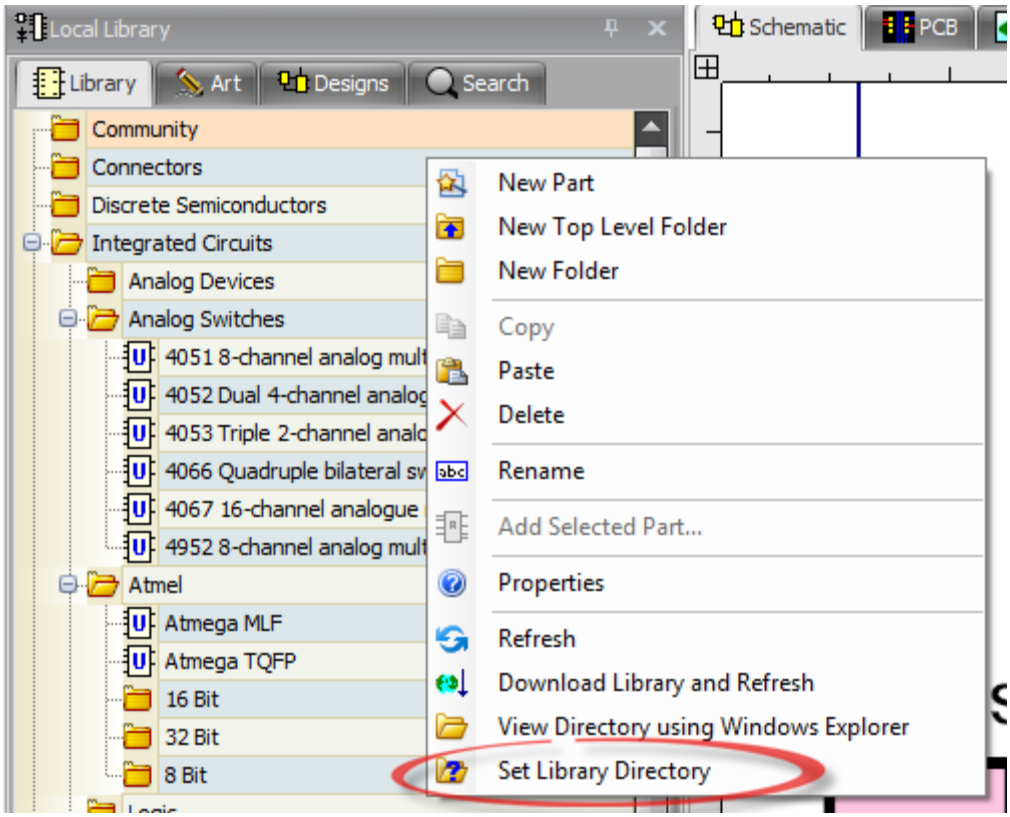
The physical package parameters are defined below.



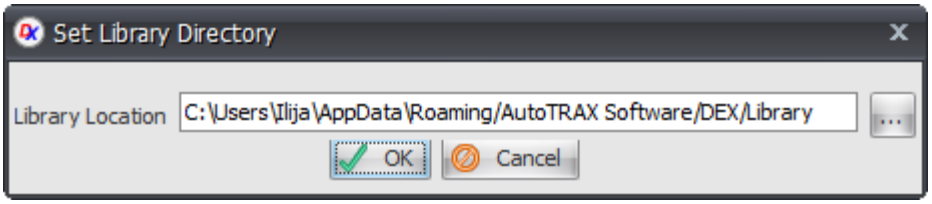
Units		INCHES		
Dimension Limits		MIN	NOM	MAX
Number of Pins	N		8	
Pitch	e		.100 BSC	
Top to Seating Plane	A	–	–	.210
Molded Package Thickness	A2	.115	.130	.195
Base to Seating Plane	A1	.015	–	–
Shoulder to Shoulder Width	E	.290	.310	.325
Molded Package Width	E1	.240	.250	.280
Overall Length	D	.348	.365	.400
Tip to Seating Plane	L	.115	.130	.150
Lead Thickness	c	.008	.010	.015
Upper Lead Width	b1	.040	.060	.070
Lower Lead Width	b	.014	.018	.022
Overall Row Spacing §	eB	–	–	.430

1.2.5.9.1.1 The Parts Library

To set the location of the Parts Library, ***right-click*** in the parts library panel and select **Set Library Directory**.

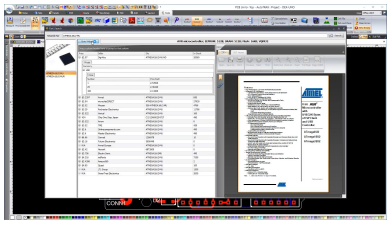


Then enter the location.

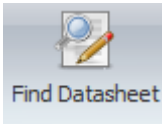


1.2.5.9.1.2 Searching for Part Prices and Datasheets

AutoTRAX DEX now comes with a powerful datasheet search utility. This will search the Internet for electronic parts, their datasheets and the latest prices from several electronic part distributors.

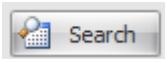


Video - Searching for part prices and datasheets

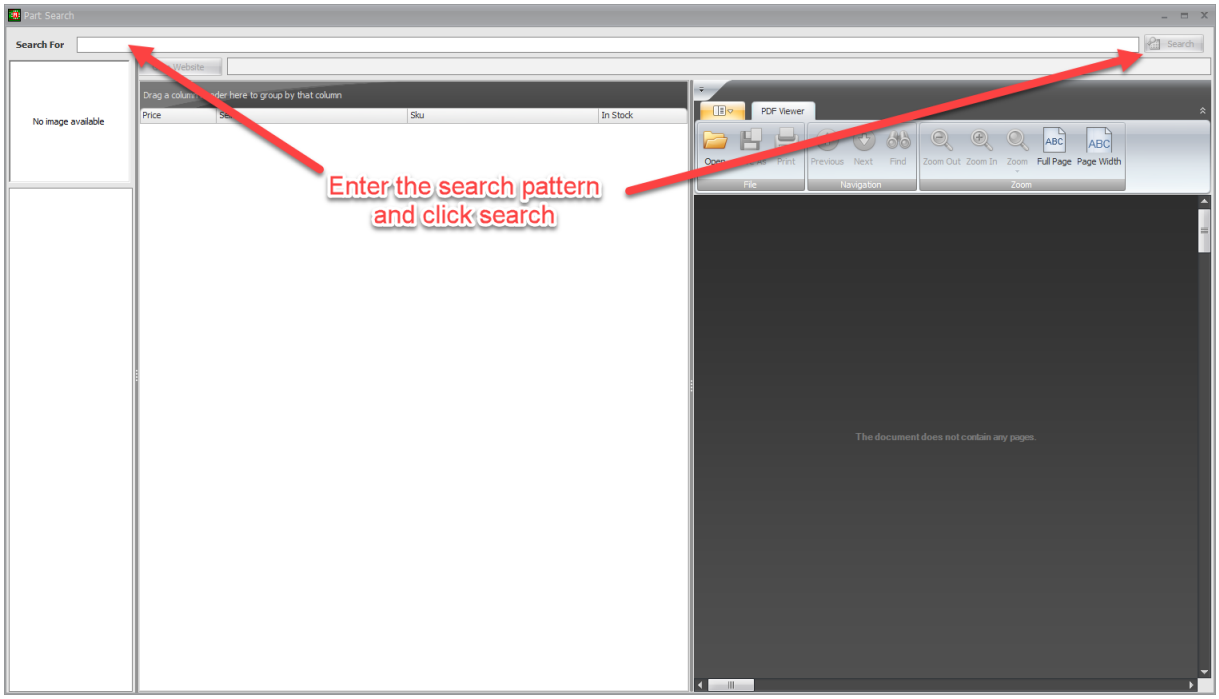
To to start, click on the find datasheet 

in the tools ribbon menu.

You will then see the datasheet search dialog.

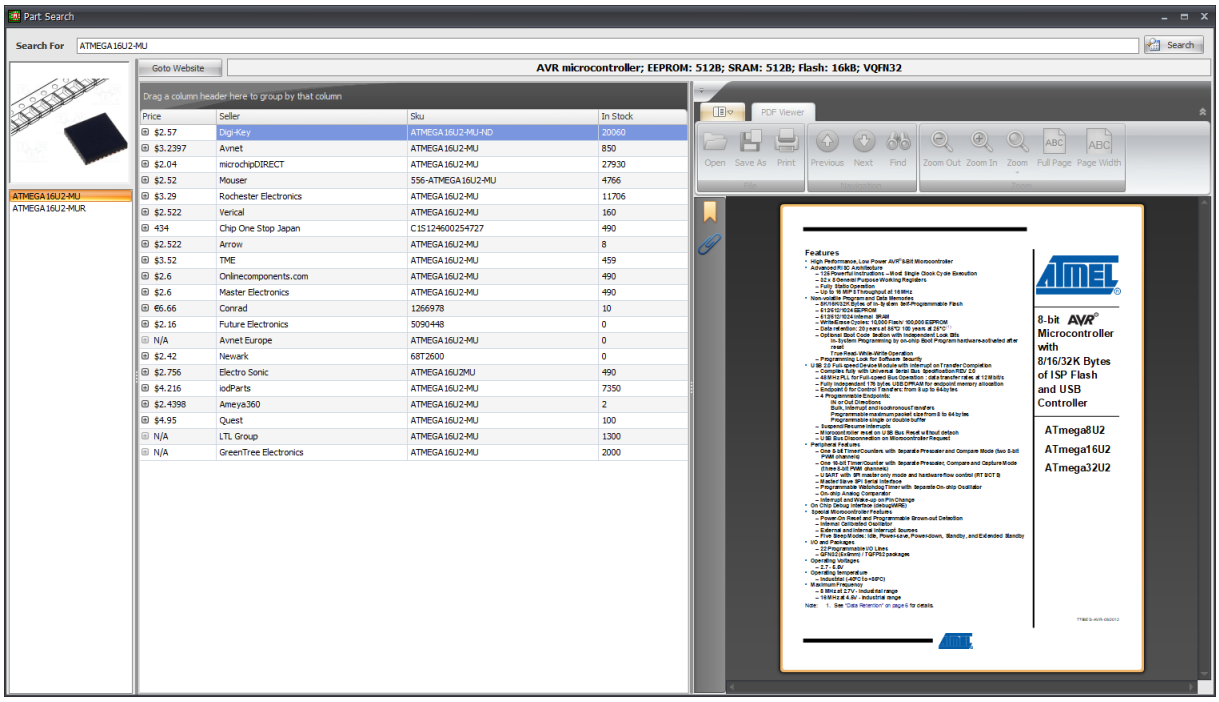
Enter the search pattern and click on the 

to find the part information. You need to be connected to the Internet.

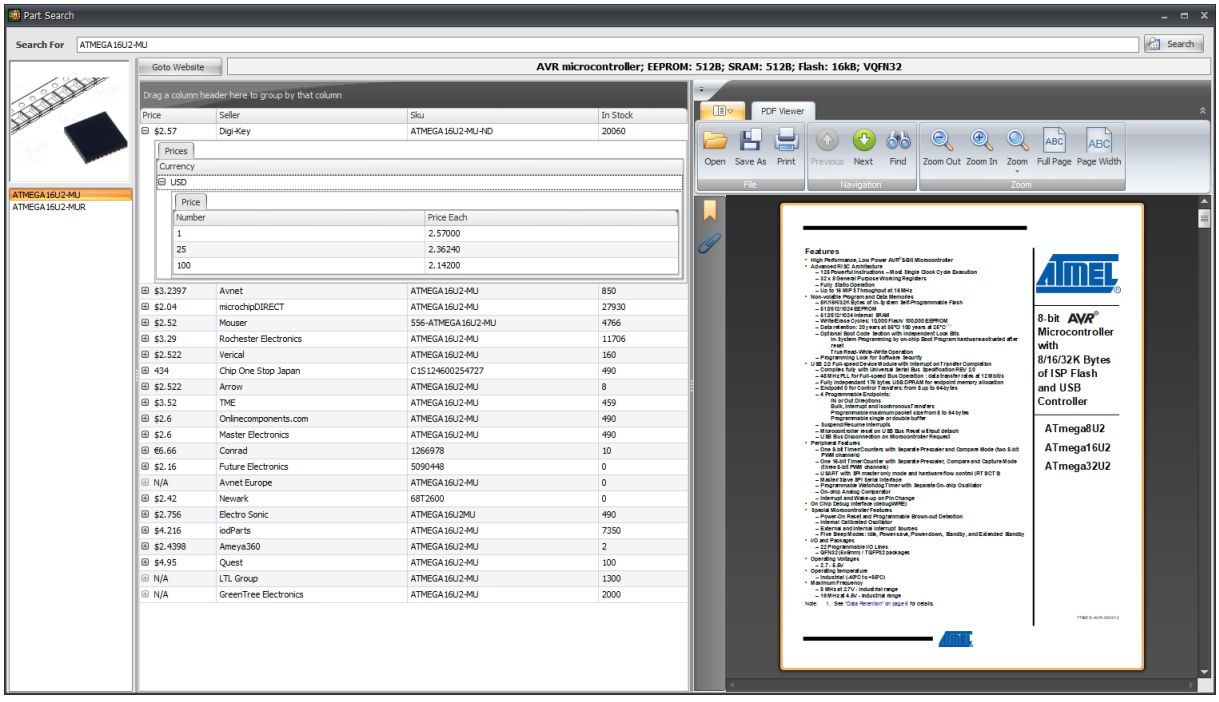


For instance, to search for an ATMEGA16U2-MU device enter ***ATMEGA16U2-MU*** and click search.

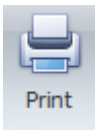
Here is the result obtained. Yours may differ depending on supply.



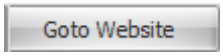
Selecting the first entry for Dig-Key and expanding the Prices you will see a list of today's prices as shown below.



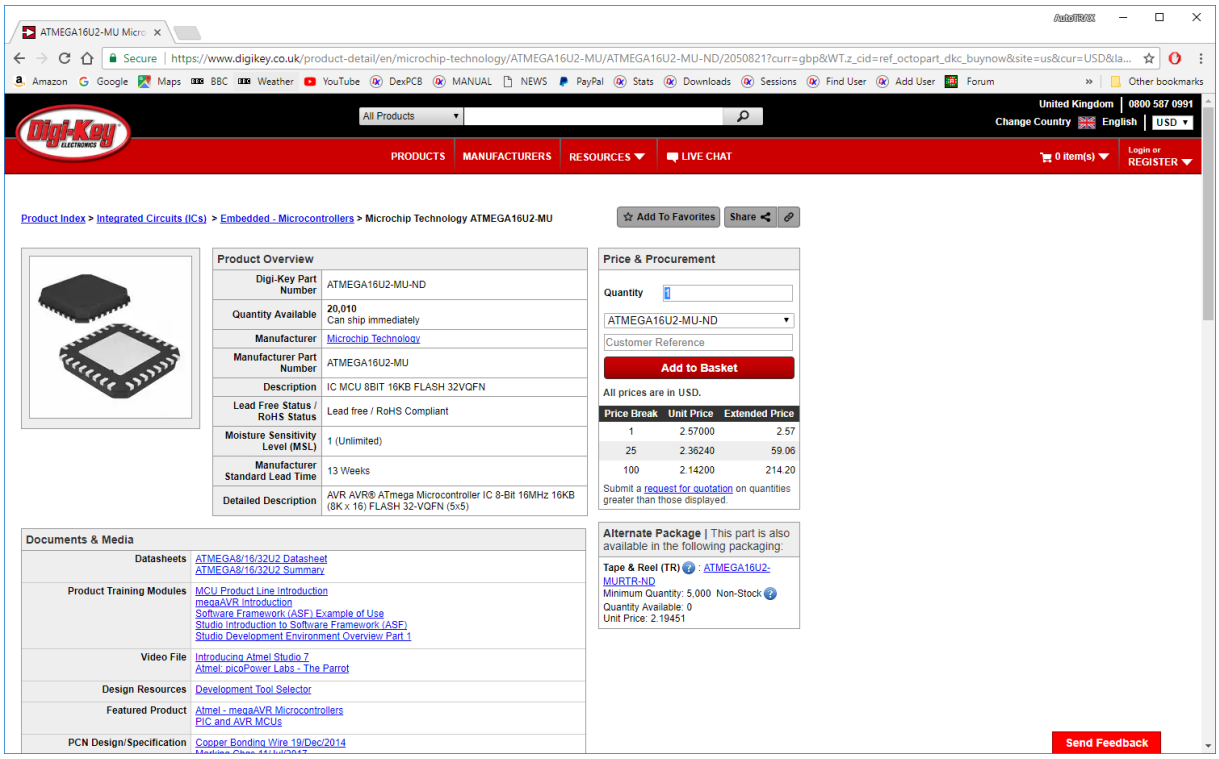
You can click on 

to save the datasheet or click on 

to print it.

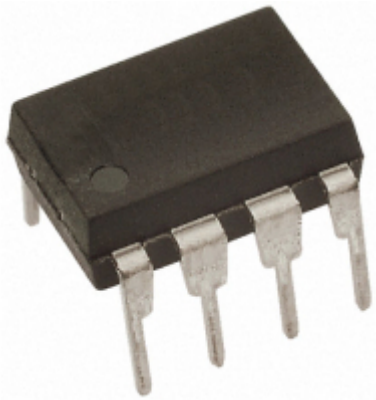
Click on 

to view the suppliers website.



1.2.5.9.2 The Deliverable

What is delivered to you to construct an electronic system is a physical part. In the case of the 8-pin DIP PC its looks like this:



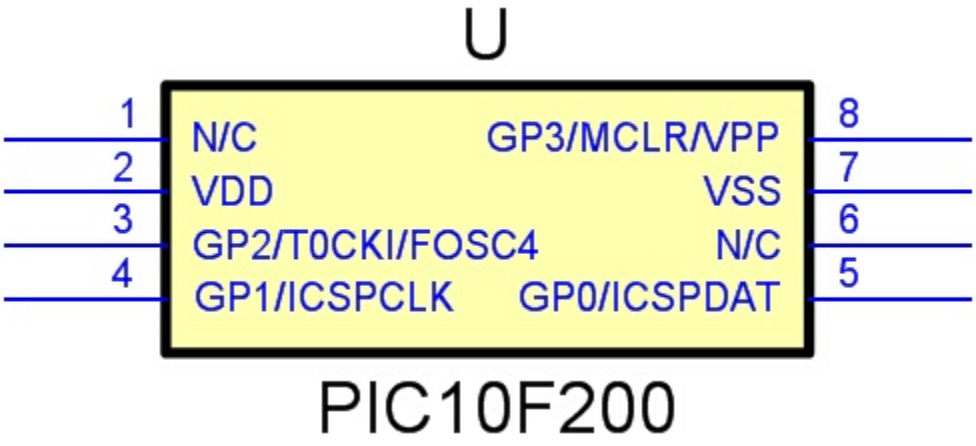
You need to extract the essential data from the supplier's model and make a computer representation that can be used to design the system both logically and physically.

1.2.5.9.3 The AutoTRAX Model

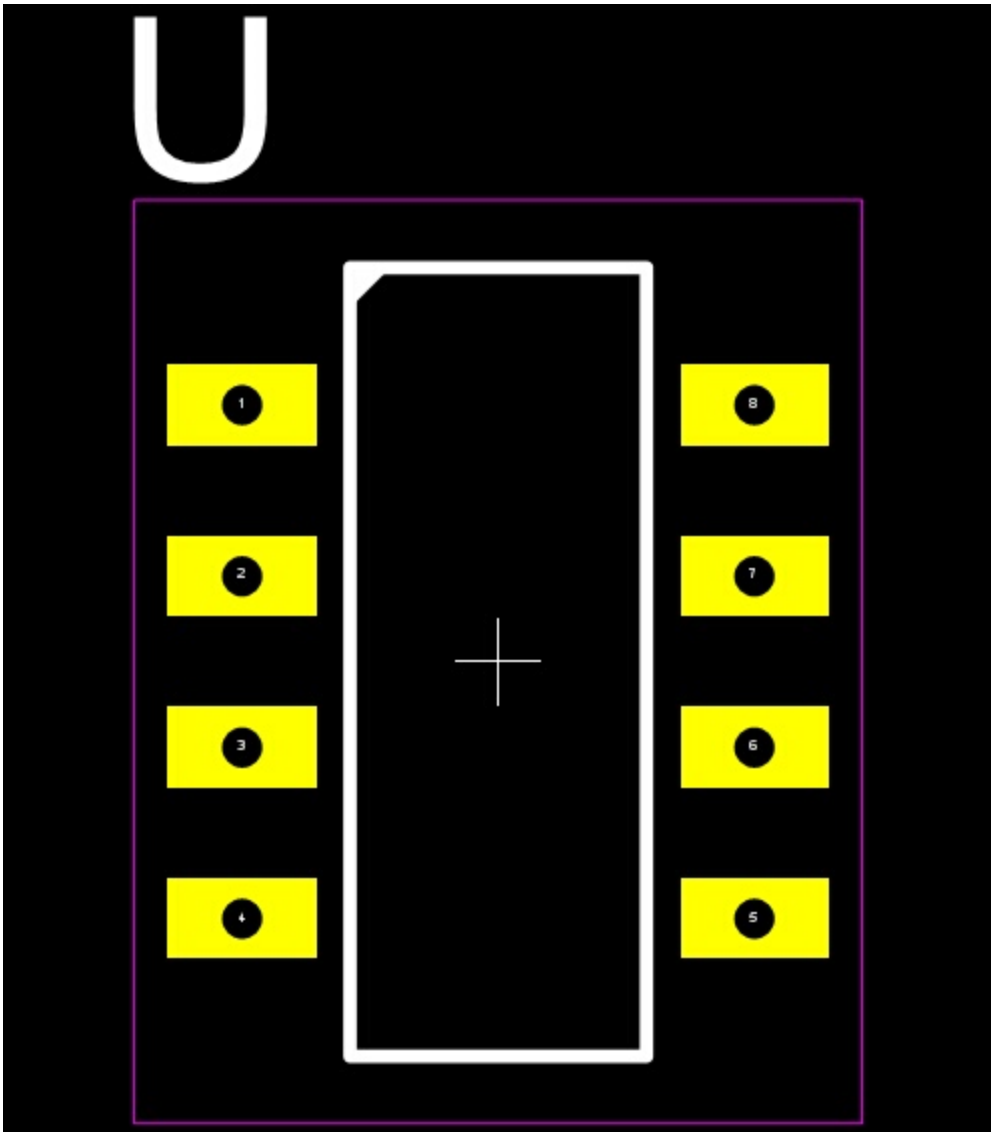
AutoTRAX DEX models the device defined by the above specification as a DIP.

The parametric model of the PIC10F200 is divided into 3 parts.

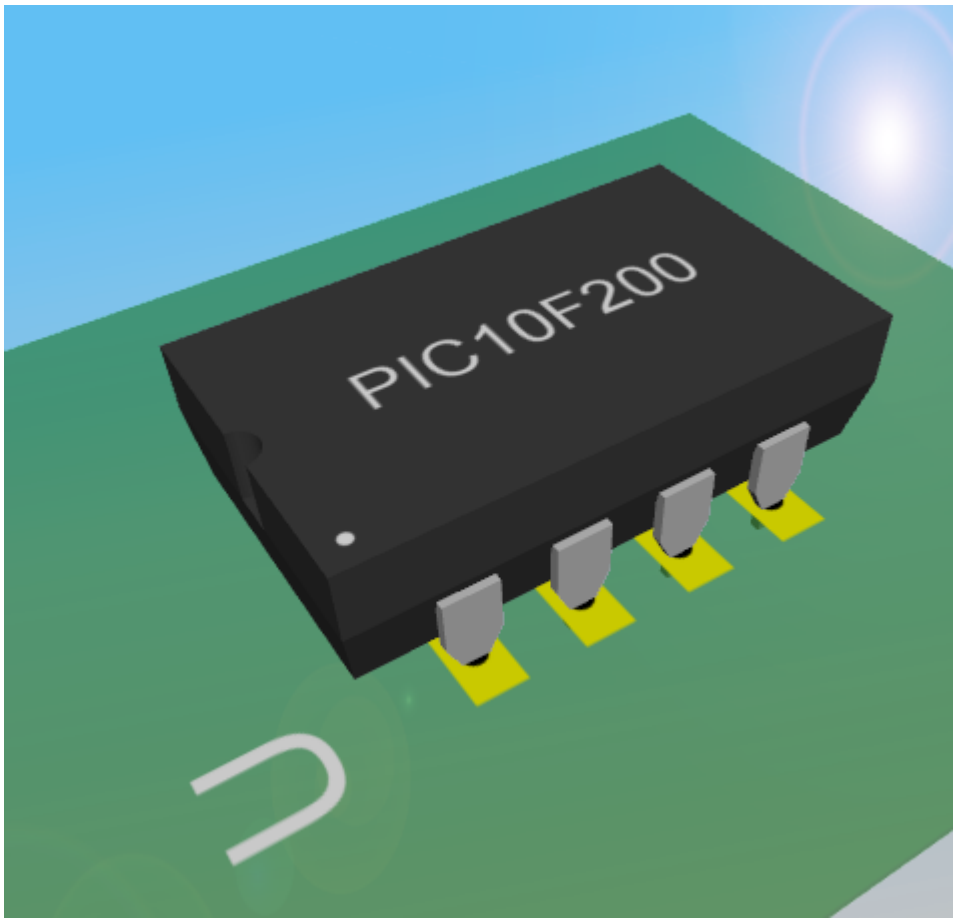
The Schematic Symbol



The Footprint



The 3D Package



1.2.5.9.4 Datasheet

When you create a part, the first thing you must do is find the data sheet for the part and obtain a physical part so you can check the dimensions against the datasheet.

Do not trust part libraries.

Always get the datasheet for all the parts that you use and check each part's symbol(s), footprint and 3D model.

1.2.5.9.5 Schematic Symbol

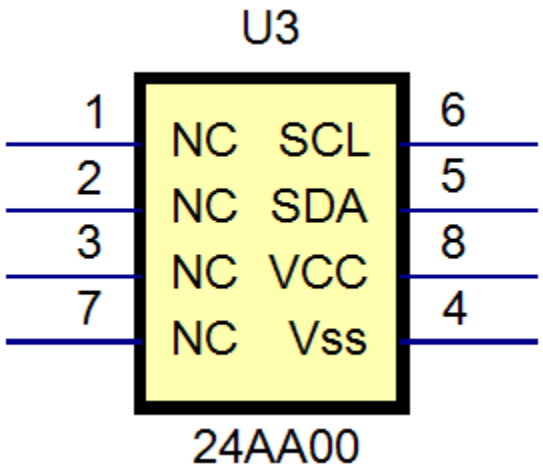
All parametric parts has 1 or more schematic symbols. By default it is a symbol with a rectangular terminal magnet with symbol terminals attached to the left and right edges of the terminal magnet. There will be 1 symbol terminal associated with each pad in the footprint.

Symbols

A part can consist of one or more symbols. Each symbol represents the electrical characteristics of the part and also is the main visualisation of the part.

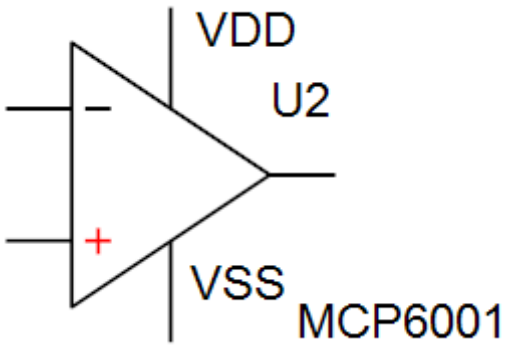
Single Symbol

Below is a single symbol representation of a part using a terminal magnet.



A symbol using a terminal magnet

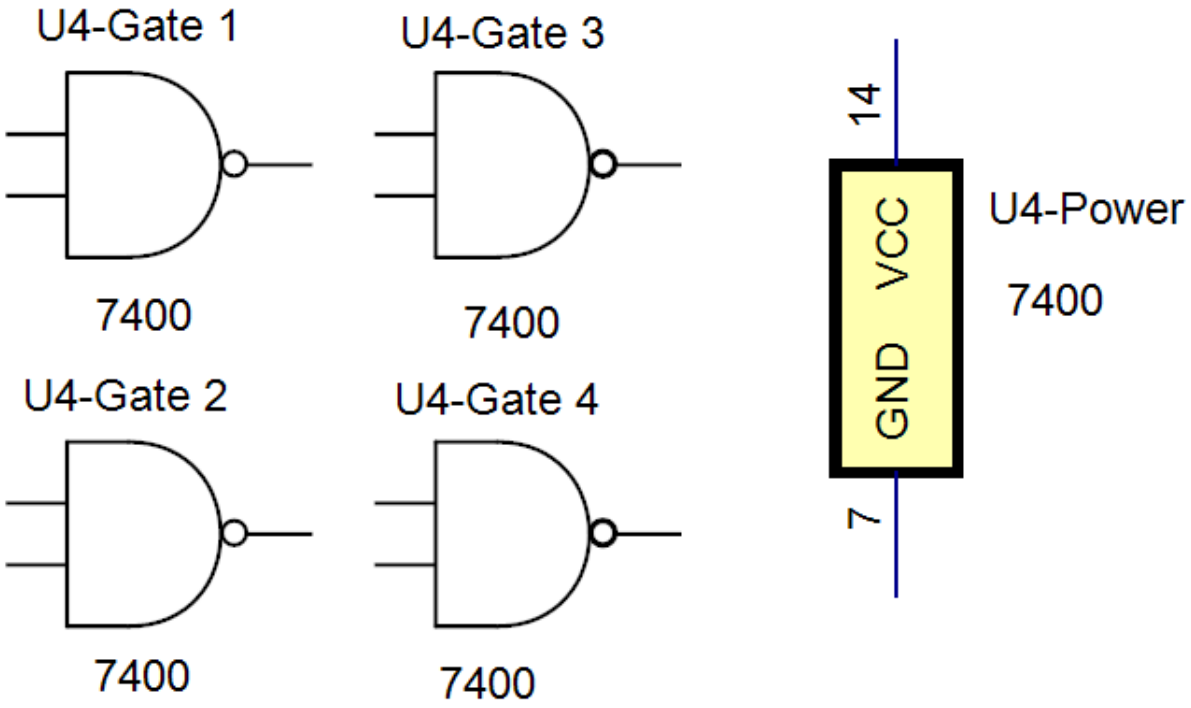
In contrast to the above we have a single symbol representation of a part not using a terminal magnet. Here the symbol terminals can be placed freely. The graphics are designed to make the functionality of the device clearly visible. In this case an operational amplifier. This is in short contrast to a rectangular representation using a terminal magnet where sometimes the functional to is not clearly visible



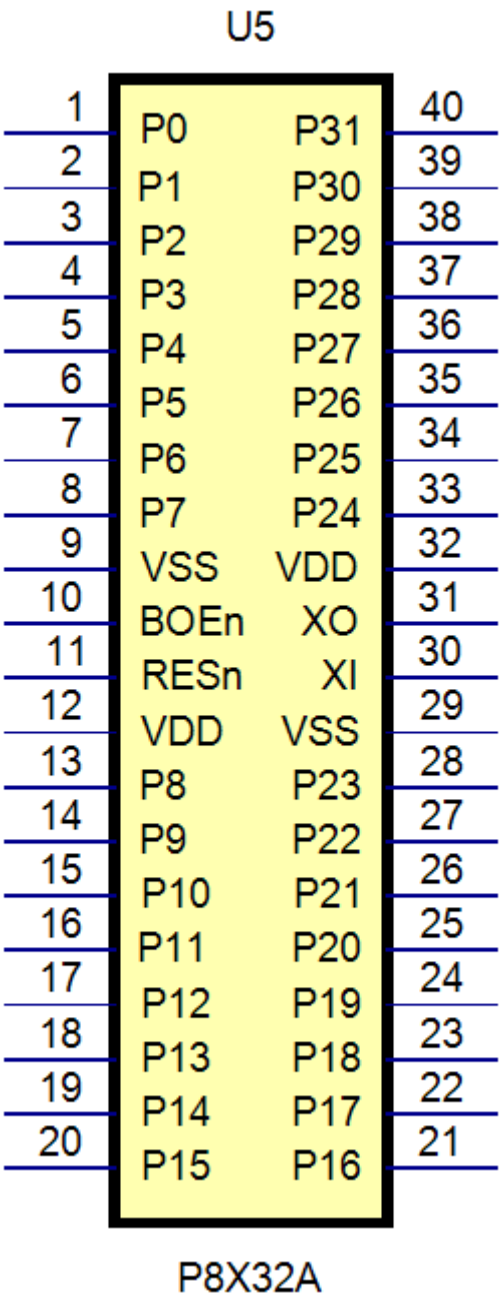
A symbol not using terminal magnet

Multiple Symbols

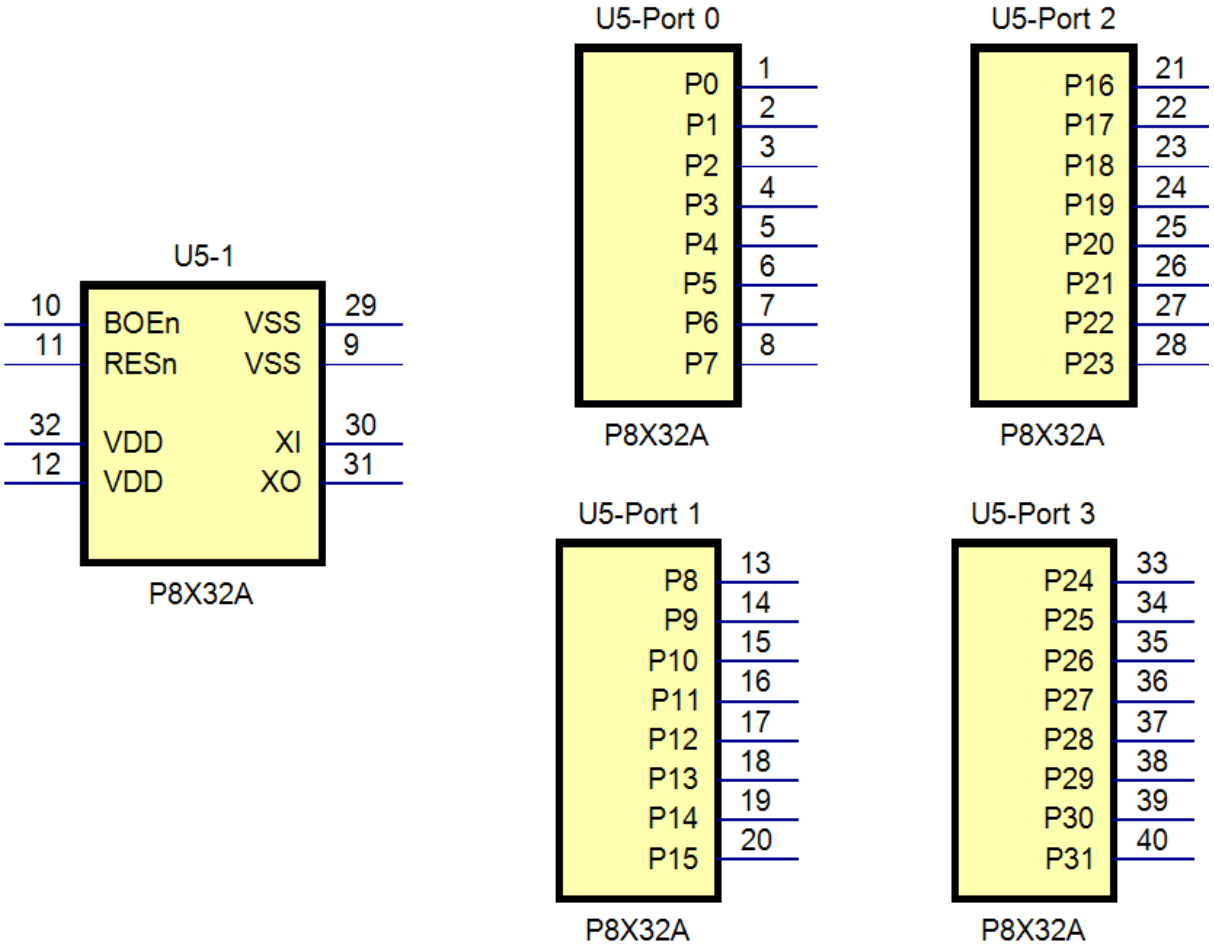
A part can have more than one symbol. Here you have a symbol that has been represented by 4 different logical get symbols and one power connection symbol using a terminal magnet.



Or you can split a symbol as many different symbols as you wish. For instance below the single symbol on the left has been split into five separate symbols whether functionality has been grouped into each symbol. This makes the understanding of the design a lot easier. You can split a symbol into multiple symbols at you have placed a symbol onto a schematic sheet.



The single symbol



1.2.5.9.6 Attributes

Distributor

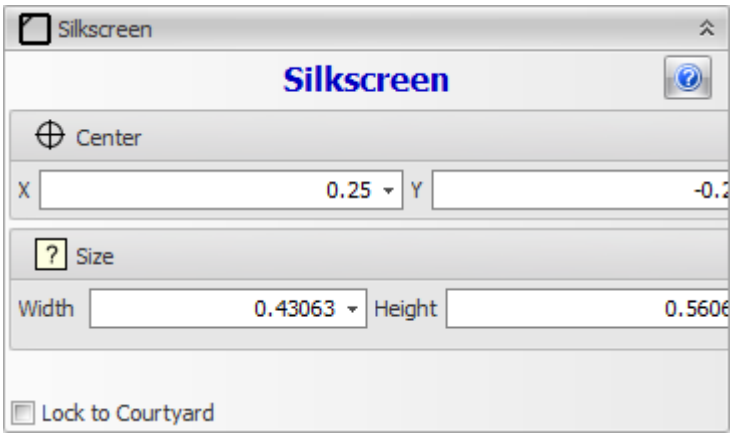
Distributor's Website

Short Description

Full Description

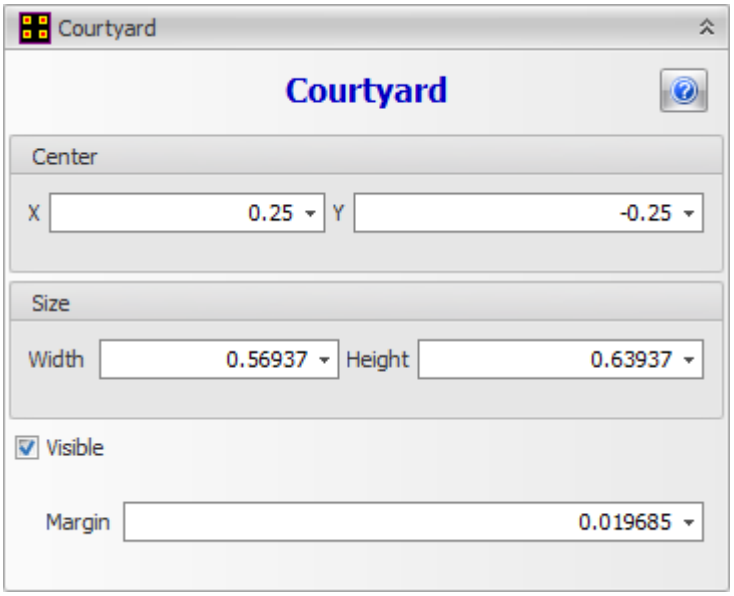
Merge Lines into Paragraphs

1.2.5.9.7 Package Silkscreen



1.2.5.9.8 Package Courtyard

Courtyards

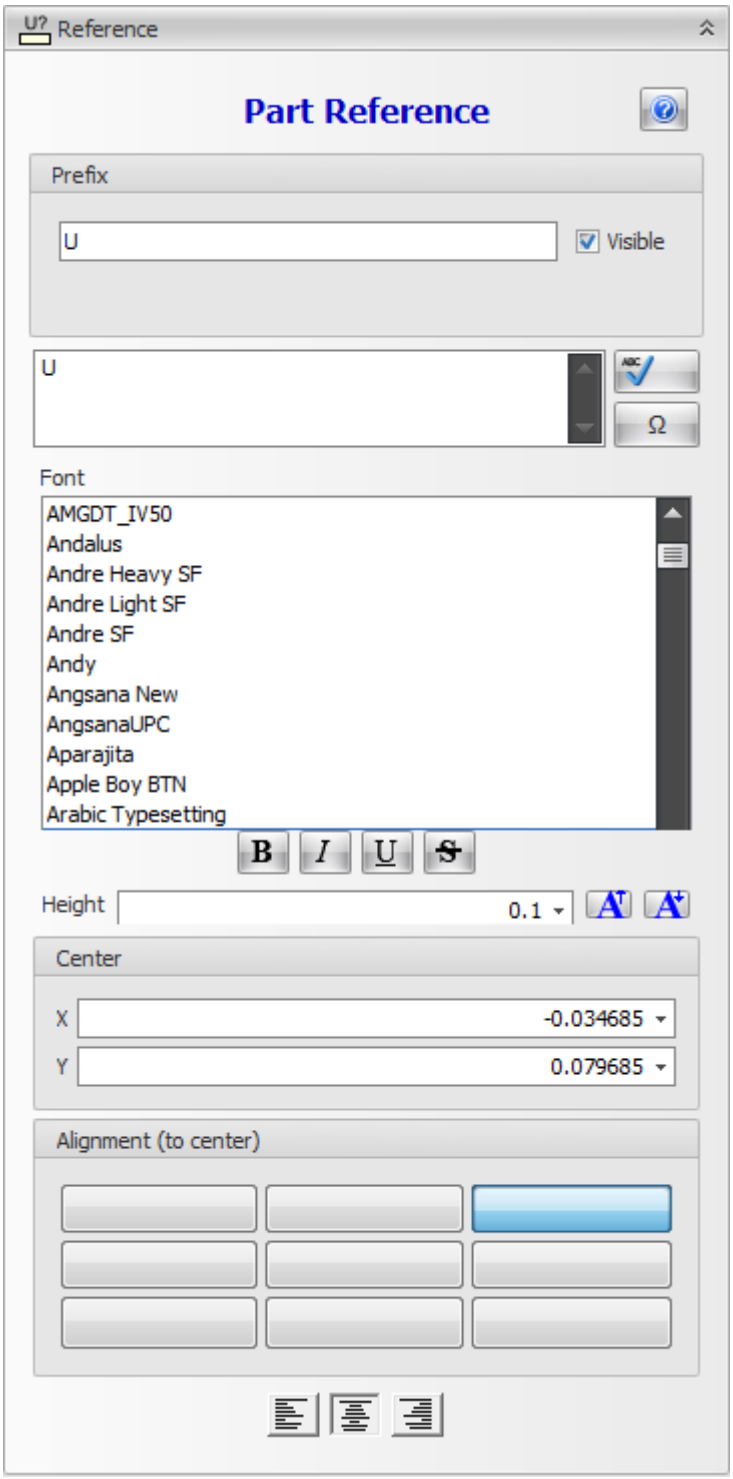


1.2.5.9.9 Package Reference

You can rotate and scale the package value text.

You can also set the font for text.

In the schematic symbol you can set the color for the text.

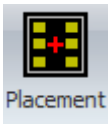


[Symbol References]

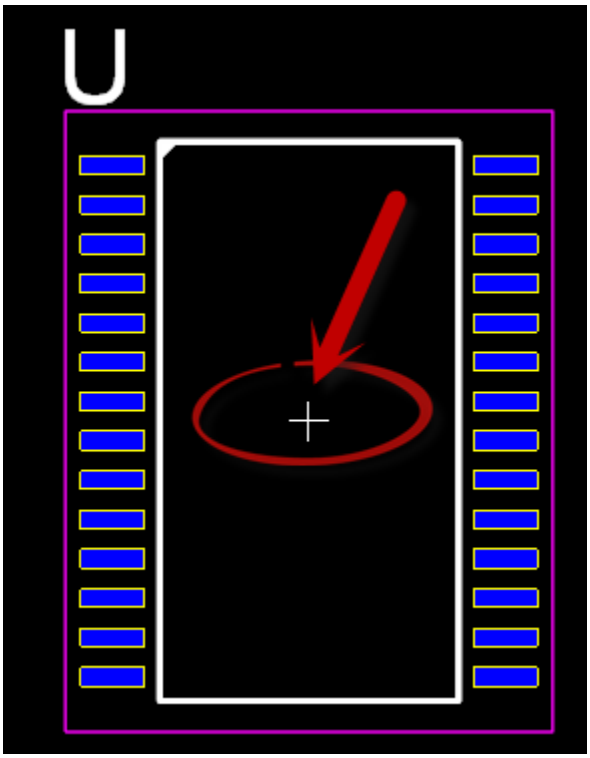
[The Footprint Reference]

1.2.5.9.9.1 Package Placement Point

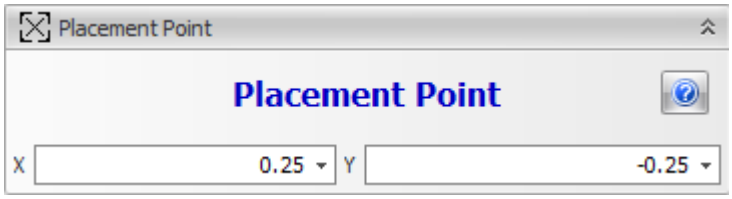
The package placement point defines the placement point for Pick and Place machines.

To hide/show placement points click the View/Edit→Footprints→ 

button.



Placement Point



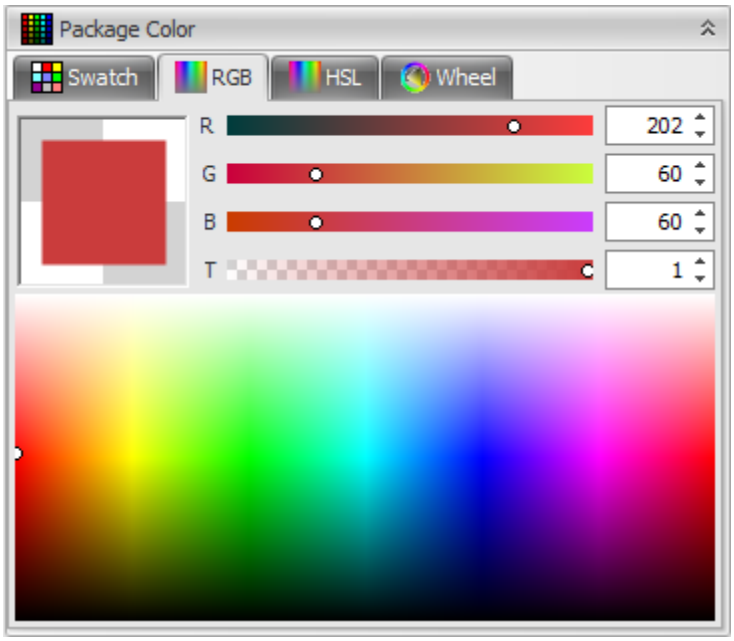
1.2.5.9.10 Package Value

You can rotate and scale the package value text.

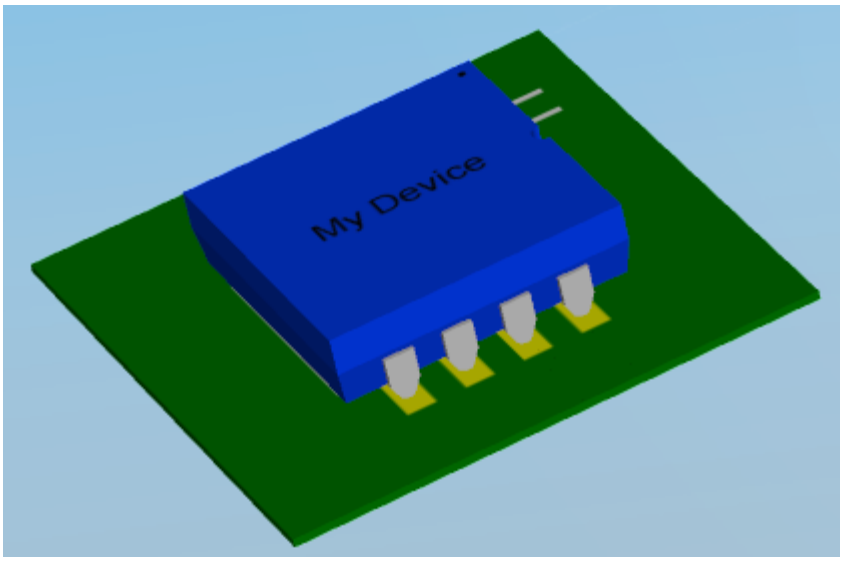
You can also set the font and color for text.

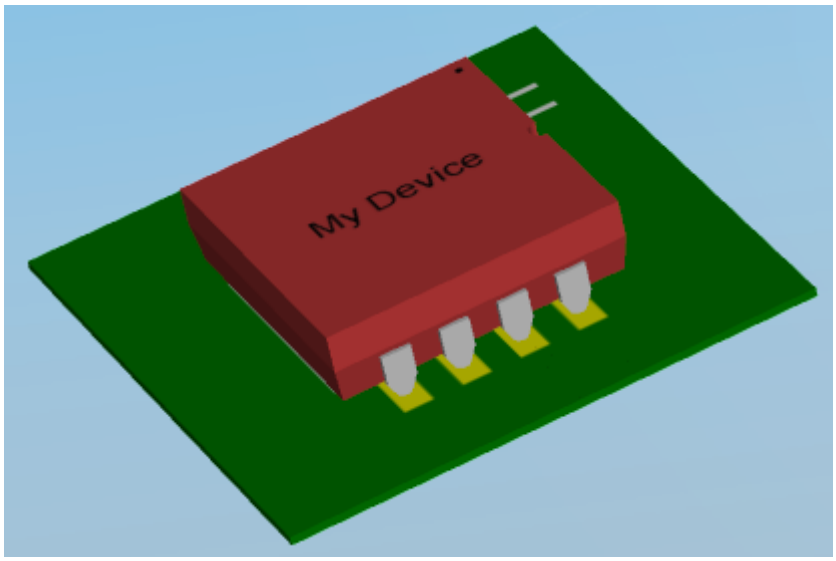
1.2.5.9.11 Package Color

You can set the color of the package using the Package Color expandable control.



The Package Color Expandable Control



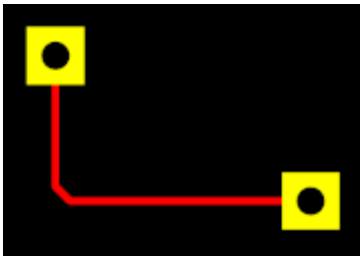


Changing The Package Color

See the Editing Colors topic on how to change colors.

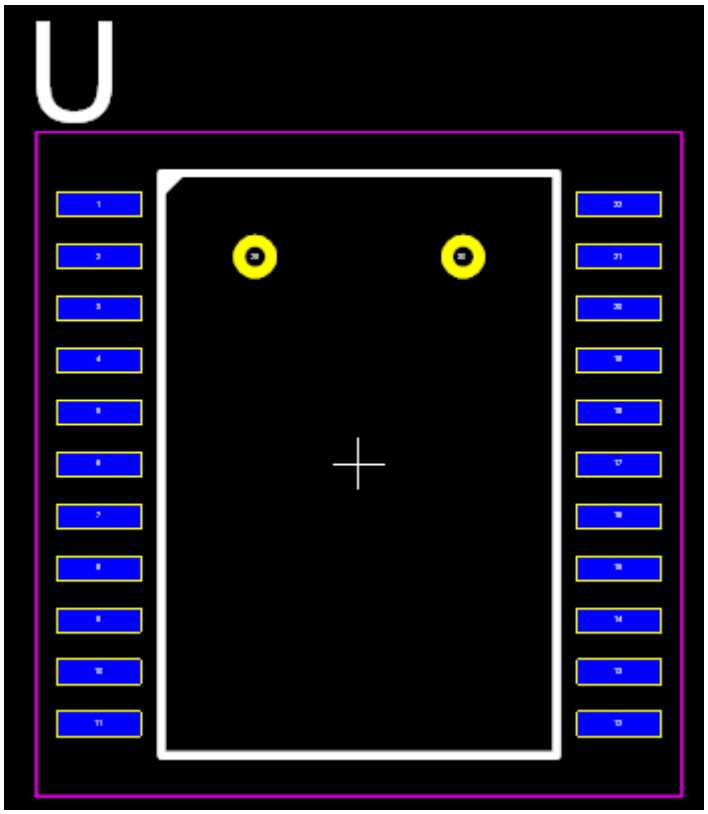
1.2.5.9.12 Pads

Pads are areas of copper on the surface of a PCB with an optional hole that is used to both secure a part to the PCB and also to provide an electrical connection between electrical contact points/areas and pins of a part with other parts using copper tracks.



2 pads connected via a copper track

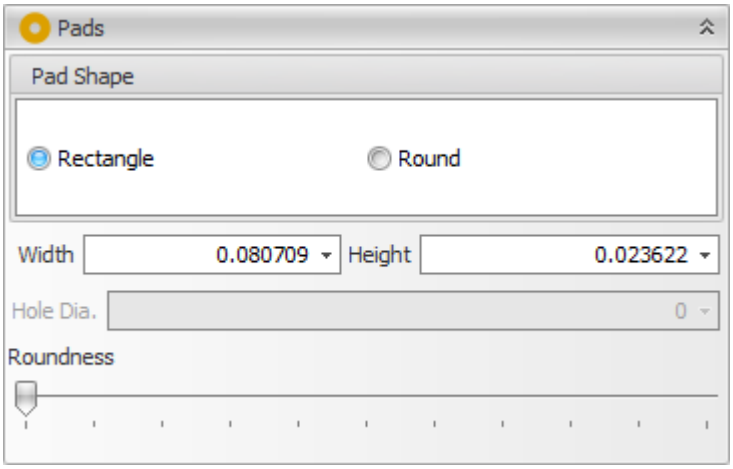
A footprint can have pads automatically generated as for the part type and also have pads that you manually add to the footprint.



Parametric SMT footprint with 2 manually added THT pads

Footprints can have zero or more pads.

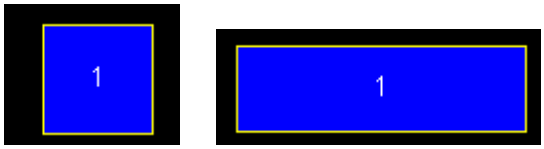
The parametric footprints pads dialog is shown below.



Pad Shape

Pad can have rectangular/square pads with optional rounded corners or circular/elliptical pads.

Check the Rectangle radio button to set make all the pads in the parametric footprint rectangular. This does not affect any additional pads you have added to customize the footprint.



Check the Round radio button to set make all the pads in the parametric footprint round.. This does not affect any additional pads you have added to customize the footprint.

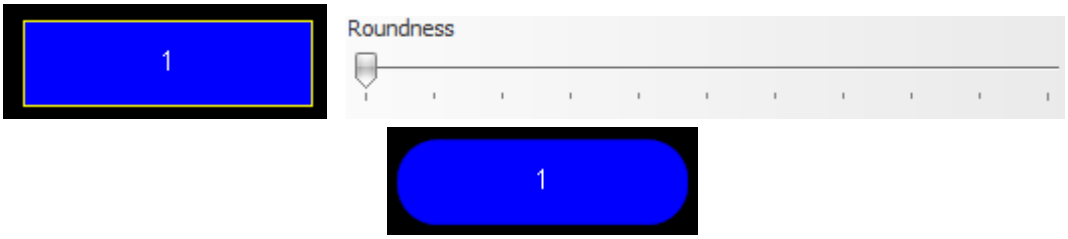
Width - Enter the horizontal width of the pad.

Height - Enter the vertical height of the pad.

Rounding the corners of rectangular pads

Slide the 

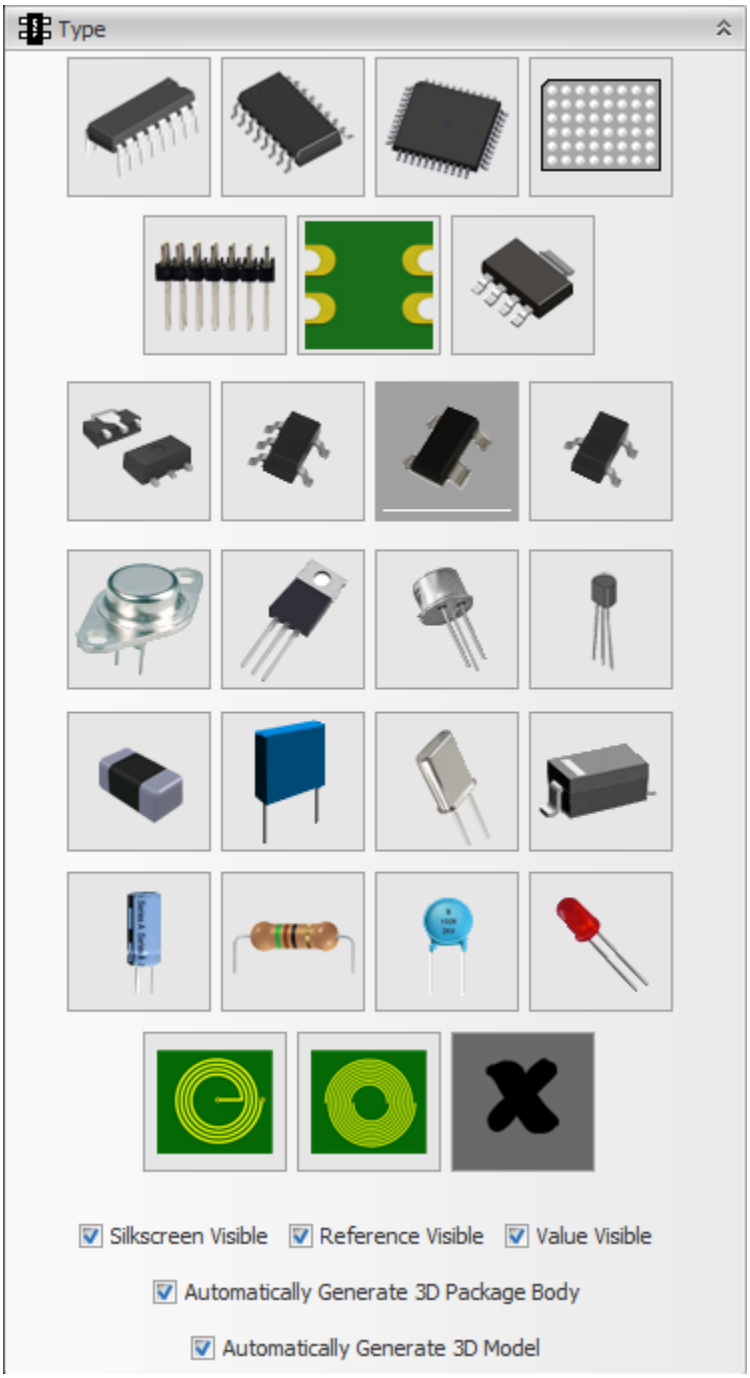
slider to round the 4 corners of rectangular pads.



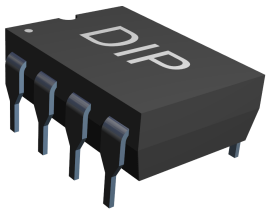
See also:

Pads

1.2.5.9.13 Type Selection

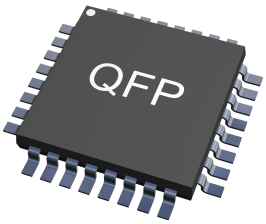


Type Selector



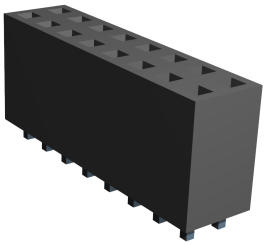
Dual-Inline Package

Small Outline IC



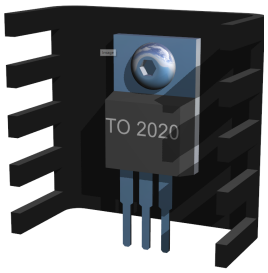
Quad Flat Pack

Ball Grid Array

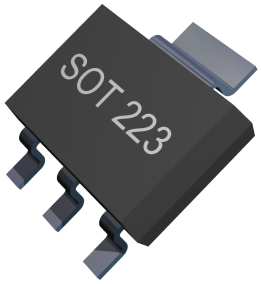


Female Header

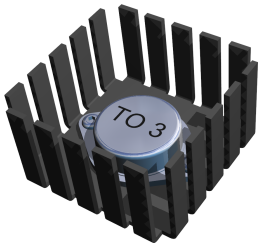
Male Header



TO-220



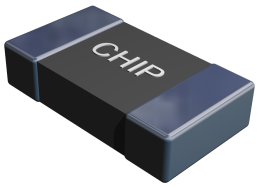
SOT-223



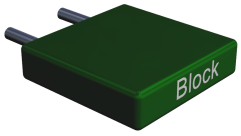
TO-3

TO-66

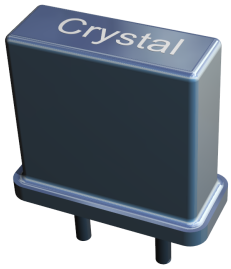
TO-18



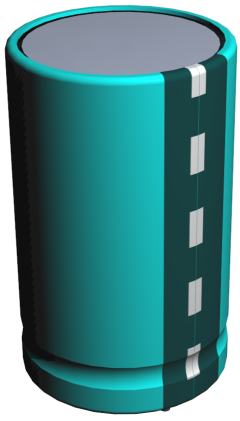
CHIP



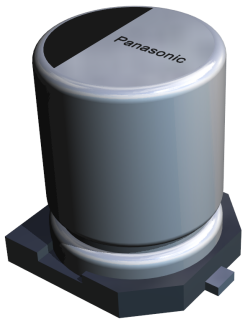
Block



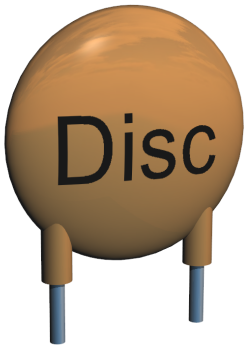
Block/Crystal



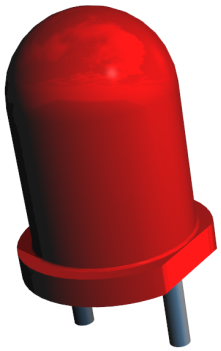
Radial



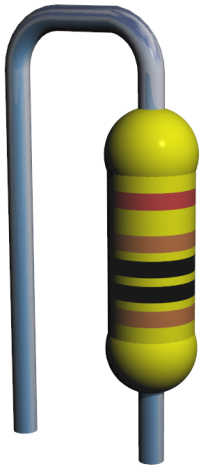
Panasonic Style



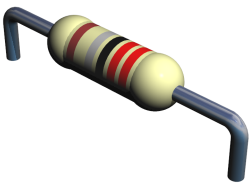
Disc



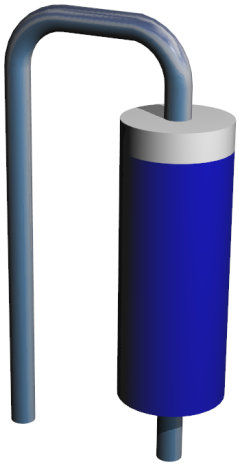
LED



Axial -Vertical-Banded

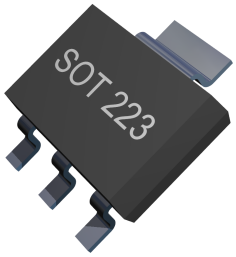


Axial -Horizontal-Banded

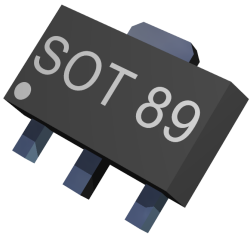


Axial -Vertical

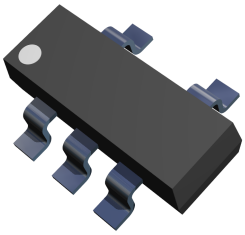
Axial -Horizontal



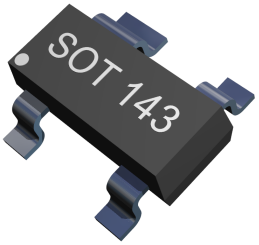
SOT-223



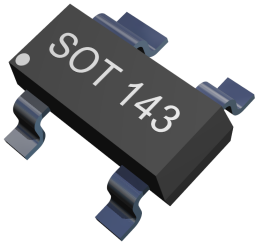
SOT-89



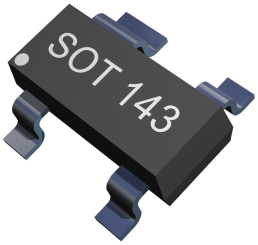
SOT-353



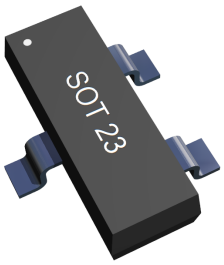
SOT-143



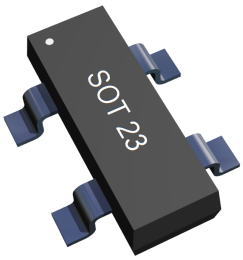
SOT-143



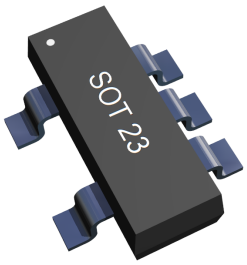
SOT-143



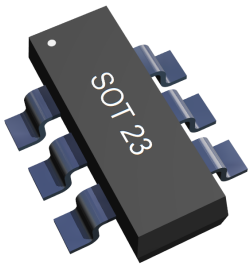
SOT-23-3



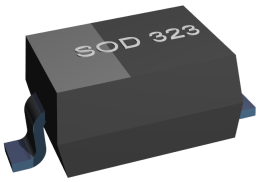
SOT-23-4



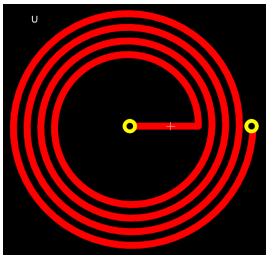
SOT-23-5



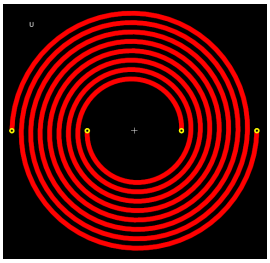
SOT-23-6



SOD-323



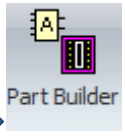
Inductor



Arial

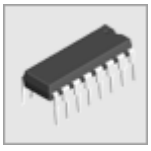
1.2.5.9.13.1 The Part Builder

The semiconductor industry manufactures a very huge variety of integrated circuits that have different packaging requirements.

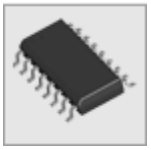
The Part Builder lets you create parts both in-place in a schematic or in a separate part file. Click the Panels→Panels→ 

button to show/hide the Part Builder.

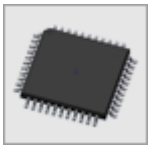
The part builder can create the following devices:



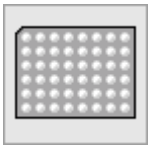
DIP



SOIC



QUAD



BGA

Generic

Header

SOT-223

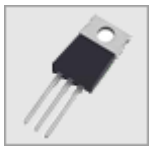
SOT-353

SOT-143

SOT-23

SOT-89

TO-92



TO-220

Metal Can

TO-3 and TO-6

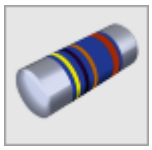
SOD-323

Chip

Block

Crystal

Axial



MELF

Radial

Disc

LED

Inductor

Antenna

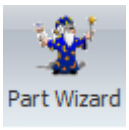
Edge Connector

Castellated PCB

Wizard

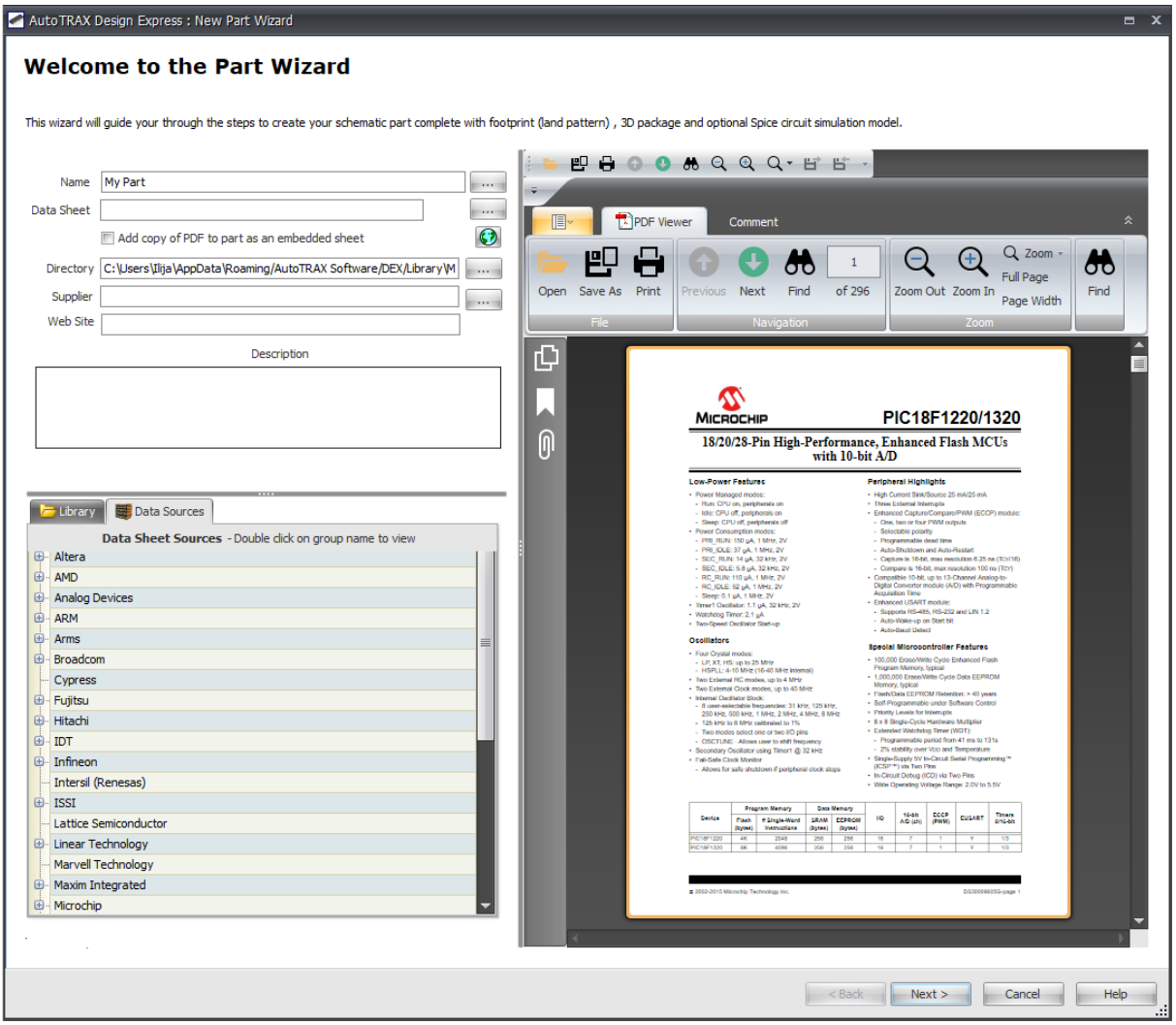
Custom

A quick way to add a part is to use the Parts Wizard.

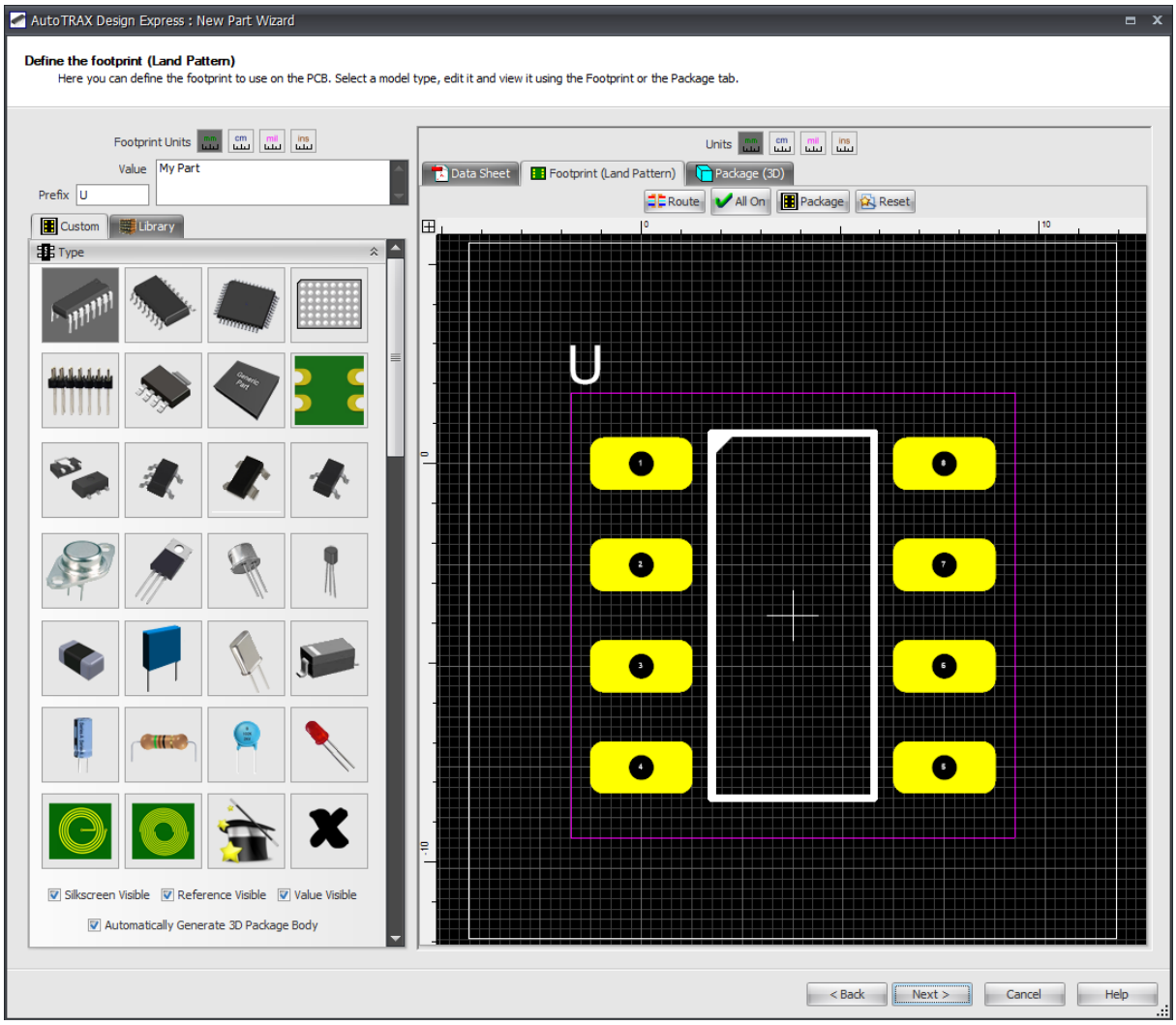
To start the parts Wizard click on the Home->New-> 

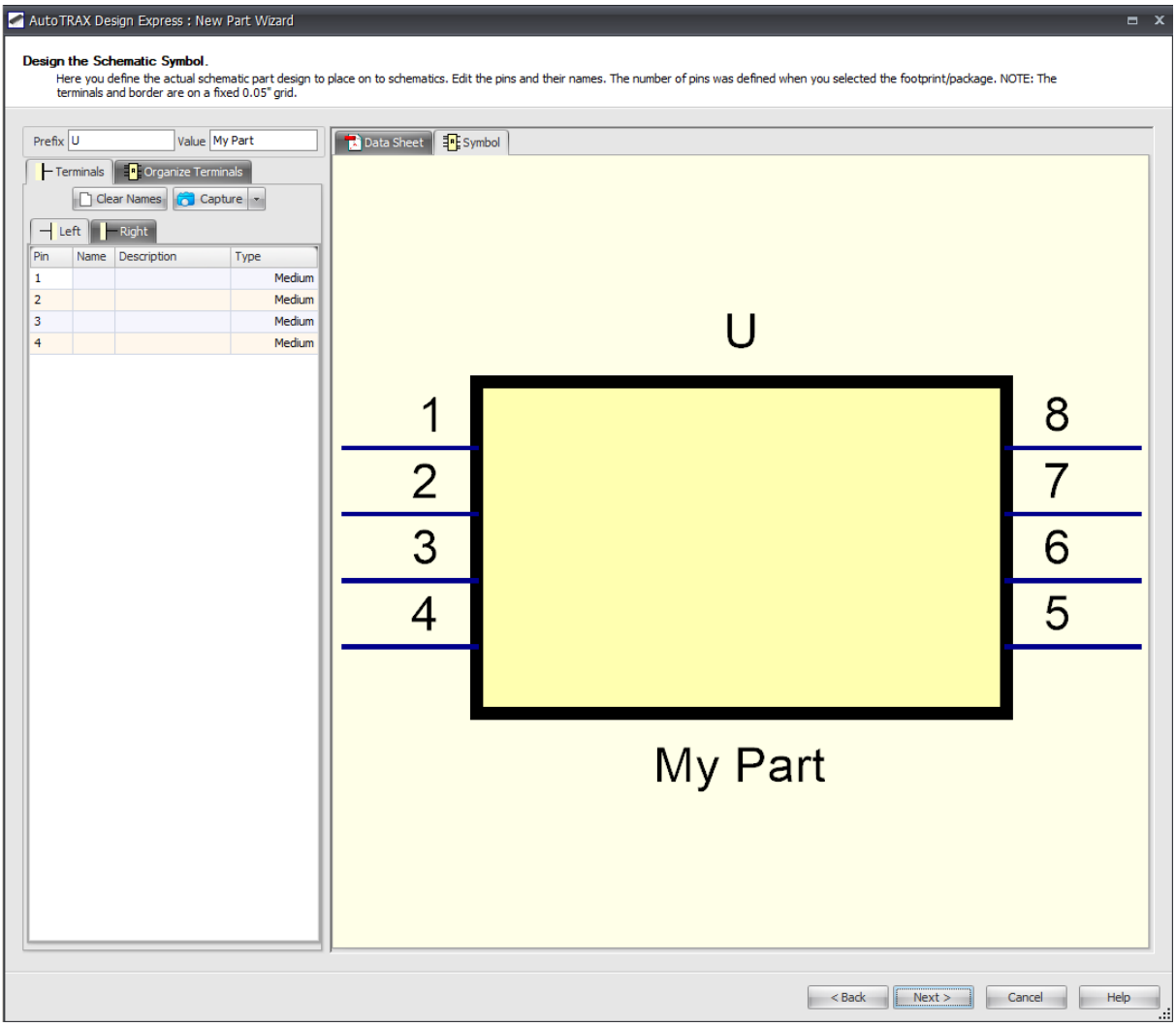
in the ribbon menu. This will display the part Wizard dialogue box shown below.

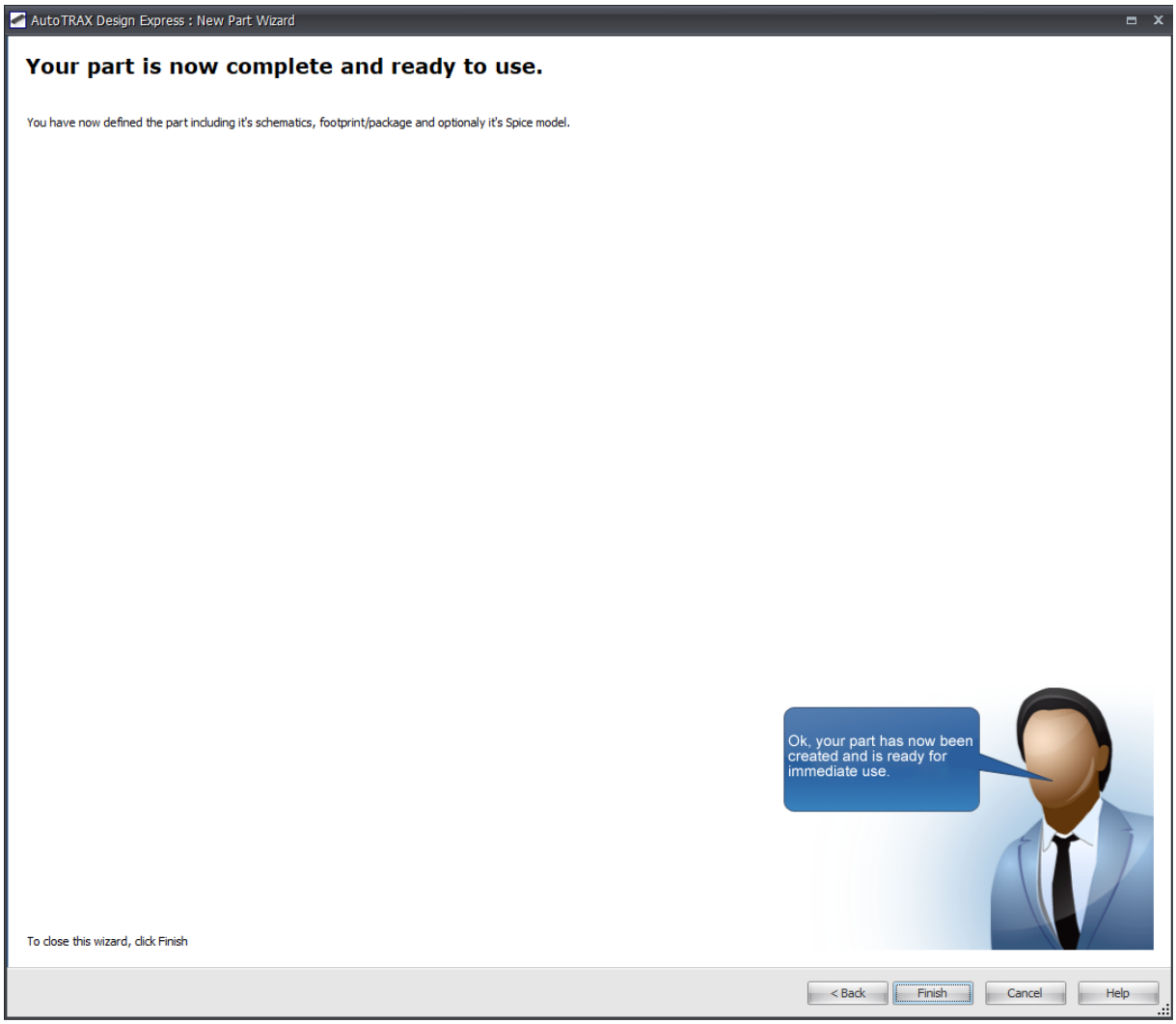
The part Wizard consists of several pages that help you define your new part. It starts with the welcome page shown below



After completing the details required in the welcome page you will be presented with the second page shown below:



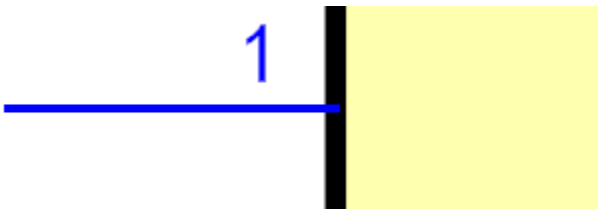




Customizing The Symbol

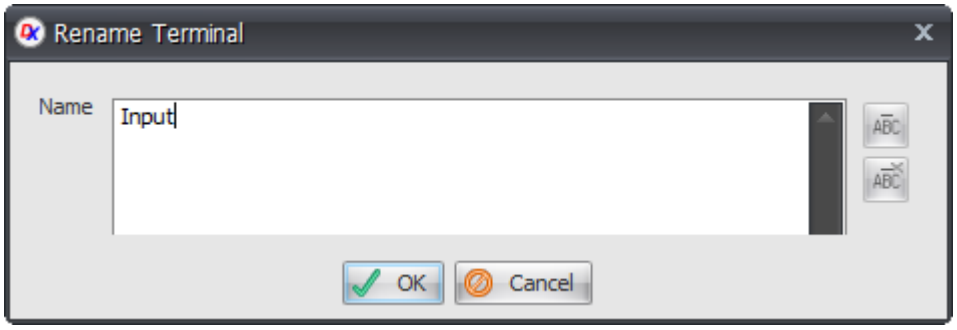
Customizing The Footprint

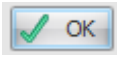
To set the name for a symbol terminal, double click on it.



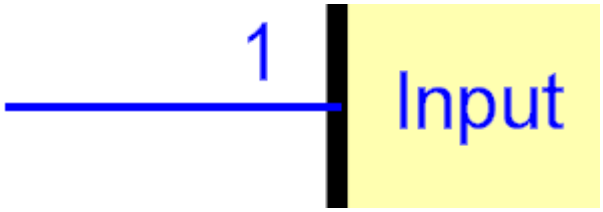
Terminal to Rename

You will see the Rename Terminal dialog shown below.



Enter the new name for the terminal and click the 

button. You can enter more than one line of text.



Renamed Terminal

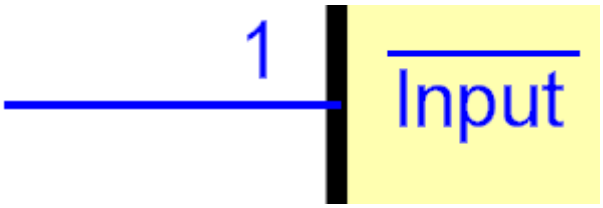
Over-strike Text

To over-strike text in a symbol terminal:

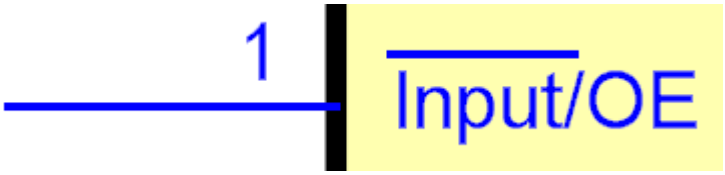
1. Select the text to over-strike in the Rename Terminal dialog.
2. Click the 

button. To remove over-strike text click the 

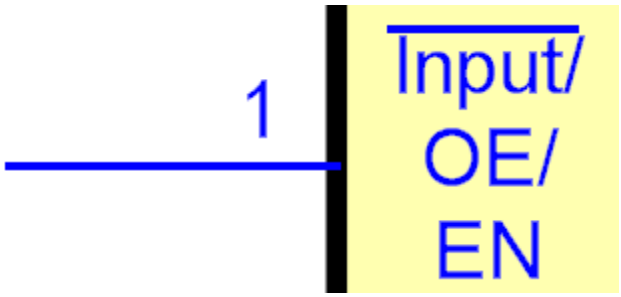
button.



Symbol Terminal with Over-strike



Symbol Terminal with Partial Over-strike



Multi-line Text

In the text, the special character ^ starts over-strike and the special character ^ ends over-strike.

Adding Graphics to Footprint

Adding Pads to Parametric Footprints

Adding Graphics to Parametric Footprints

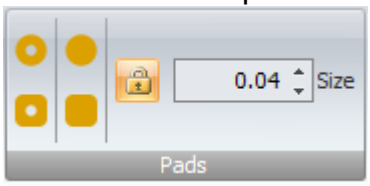
Adding 3D to Parametric Footprints

Adding 3D objects

Adding Copper Areas to Parametric Footprints

Editing the Part Value

Editing the Part Reference

You can add additional non-parametric pads to a parametric part using the Add→Pads 

button group.

See Adding Pads.

You can add vias to footprints by clicking on the add Via button in the add menu.

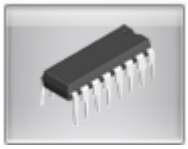
You are free to add graphics to parametric footprints as well as non-parametric footprints.

To add a copper area to a footprint select the layer that you want the copper to be on and add a graphic shape such as a rectangle or circle.

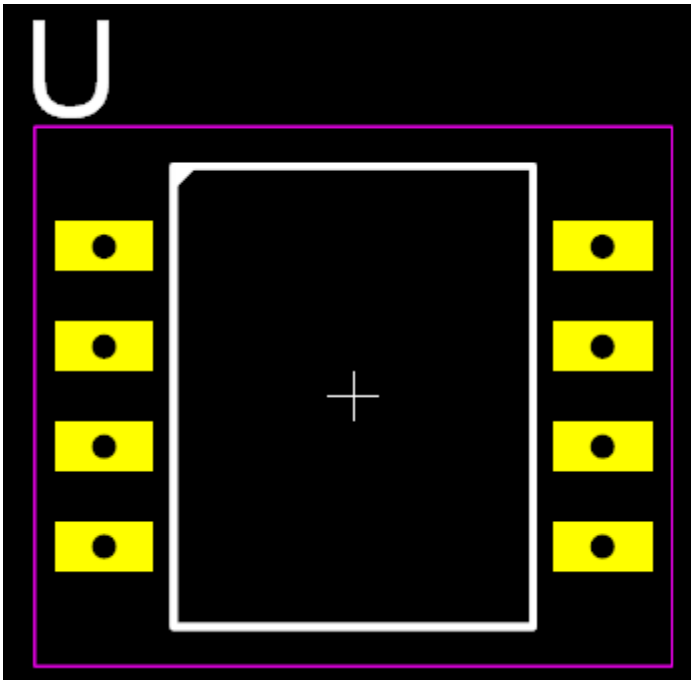
To edit a part value either double-click on the text or select the part value and edit the text and/or other parameters using the properties panel.
.

To edit a part reference either double-click on the text or select the part value and edit the text and/or other parameters using the properties panel.

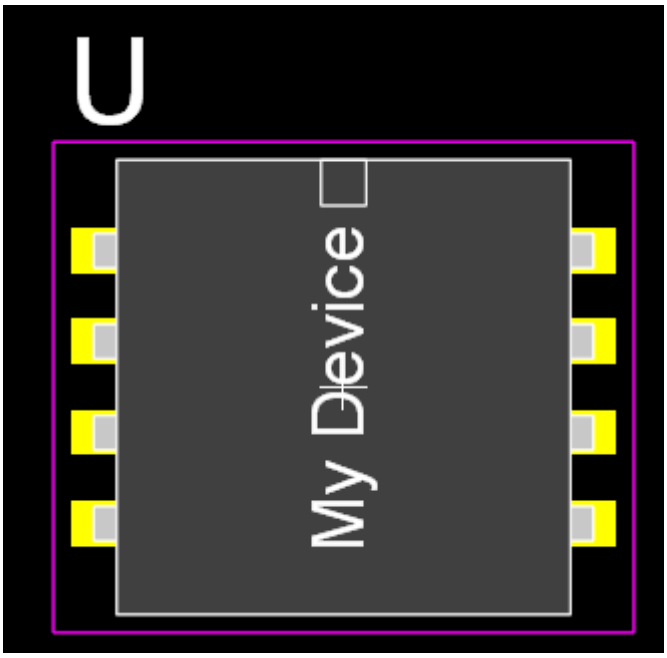
To add additional 3-D to a parametric footprint add any of the 3-D objects in the add menu.

To create a DIP part click the  button in the **The Part Builder**

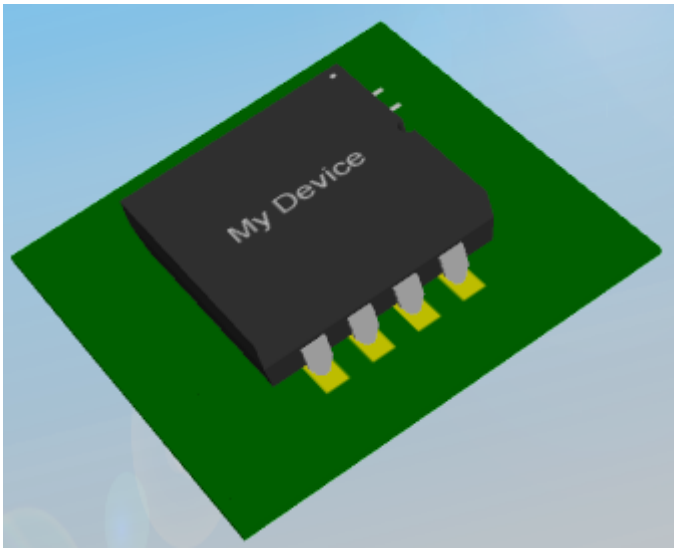
The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.



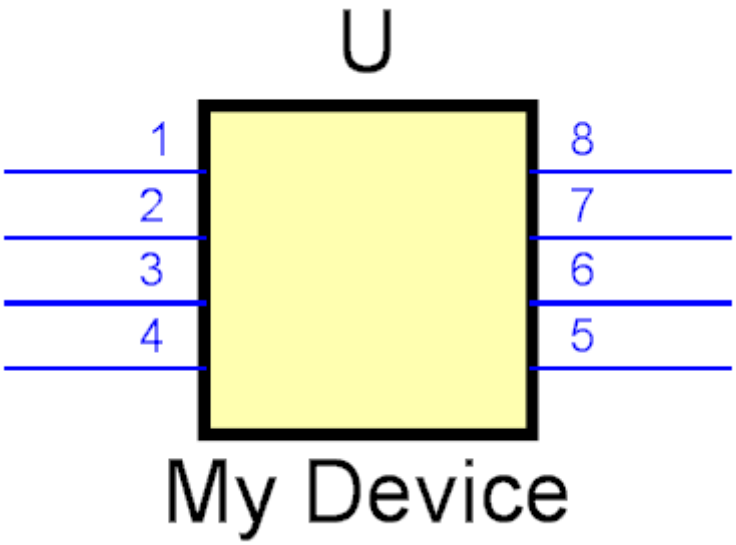
8 pin DIP



8 pin DIP showing package



3D View of 8 pin DIP



Schematic Symbol

[Device Overview]

[Dip Parameters]

[Pads]

[Package Color]

[Package Reference]

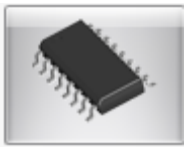
[Package Value]

[Package Silkscreen]

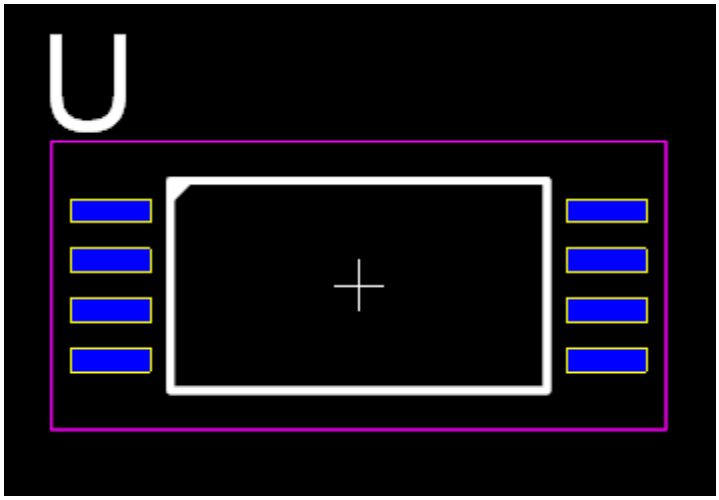
[Package Courtyard]

[Package Placement Point]

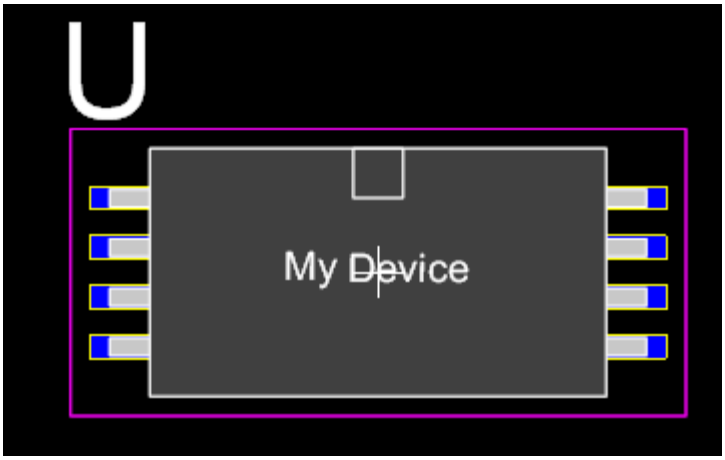
[Package Symbols]

To create a SOIC part click the  button in the [The Part Builder]

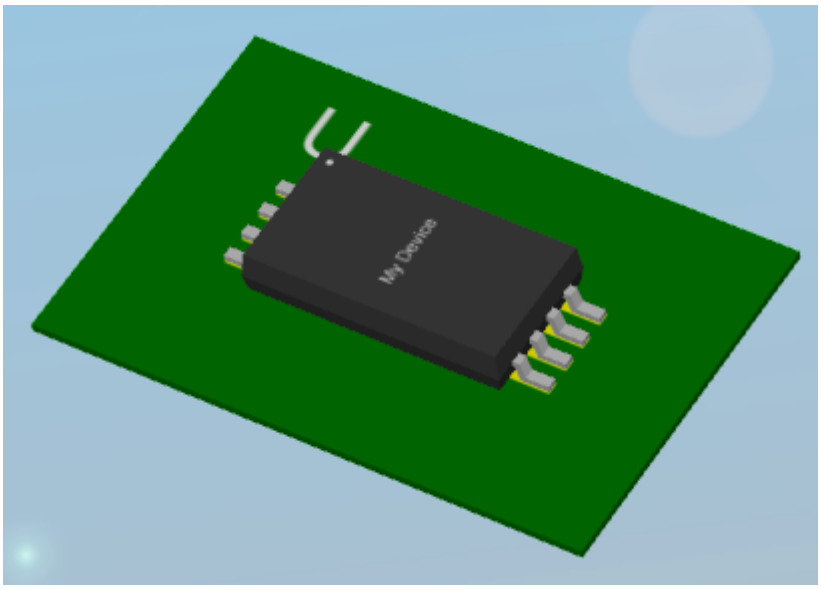
The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.



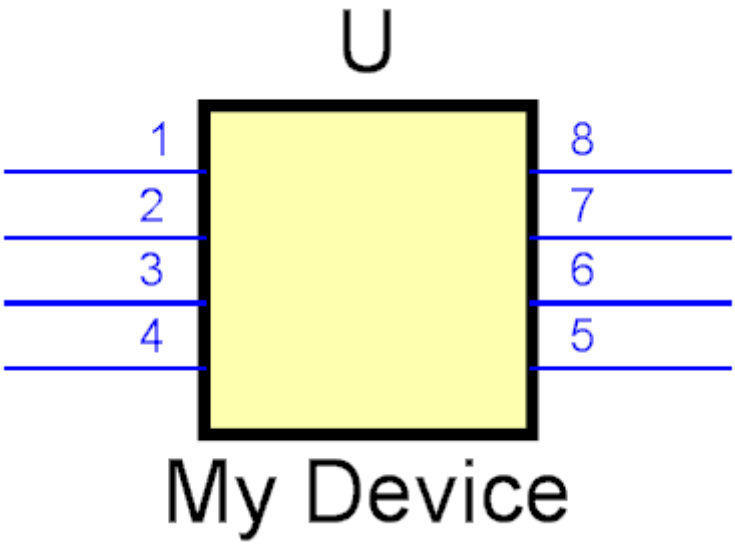
8 pin SOIC



8 pin SOIC showing package



3D View of 8 pin SOIC



Schematic Symbol

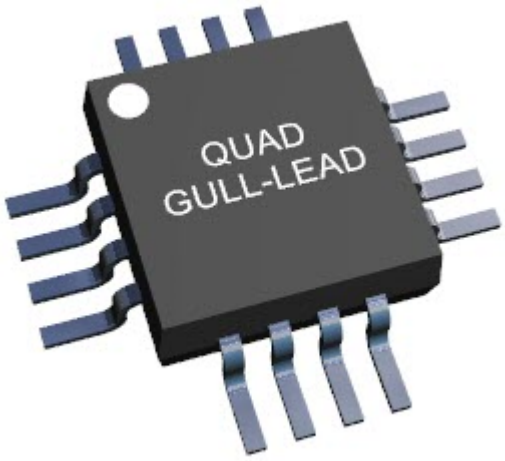
Package Value

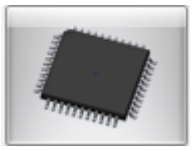
Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols



To create a QUAD part click the  button in the The Part Builder

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

QUAD Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a BGA part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

BGA Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a Header part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Header Parameters

Pads

Package Color

Package Reference

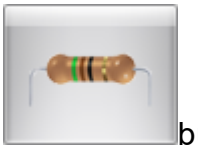
Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create an Axial part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Axial Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a Radial part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Radial Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a Chip part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Chip Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a SOD 323 part click the  button in the The Part Builder

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

SOD 323 Parameters

Pads

Package Color

Package Reference

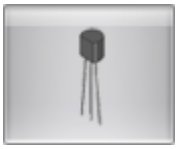
Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a TO-92 click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

TO-92 Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a SOT-23 part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

SOT-23 Parameters

Pads

Package Color

Package Reference

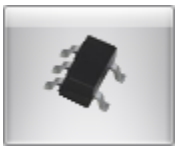
Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a SOT-353 part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

SOT-353 Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a SOT-89 part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

SOT-89 Parameters

Pads

Package Color

Package Reference

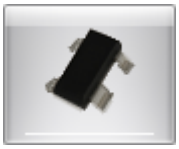
Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a SOT-143 part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

SOT-143 Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a SOT-223 part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

SOT-223 Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

To create a metal can part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Metal Can Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

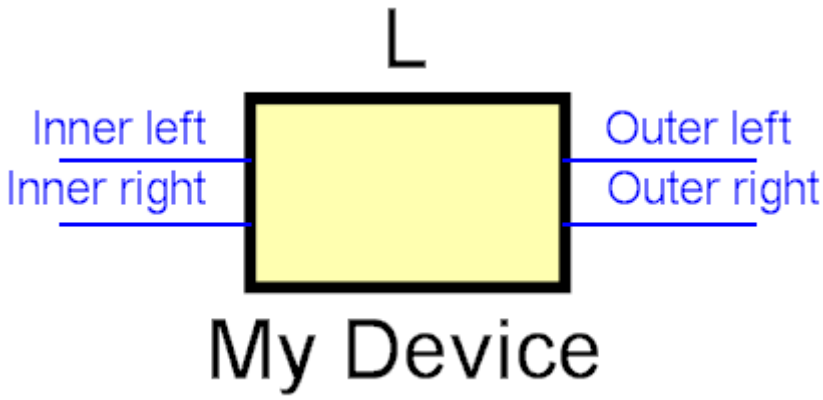
Package Placement Point

Package Symbols

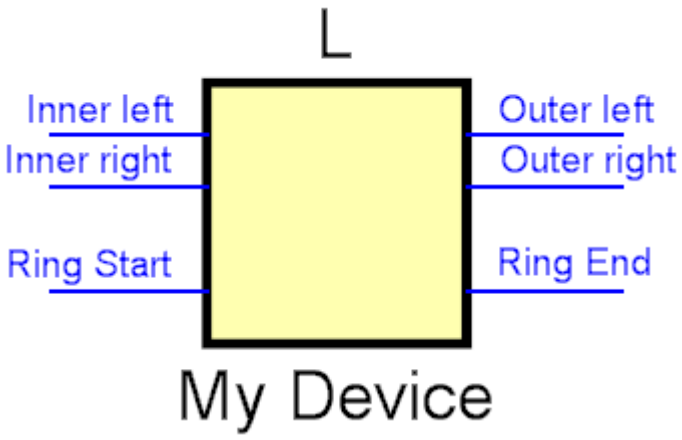
You can create a coil pattern on a copper layer to act as a antenna coil.

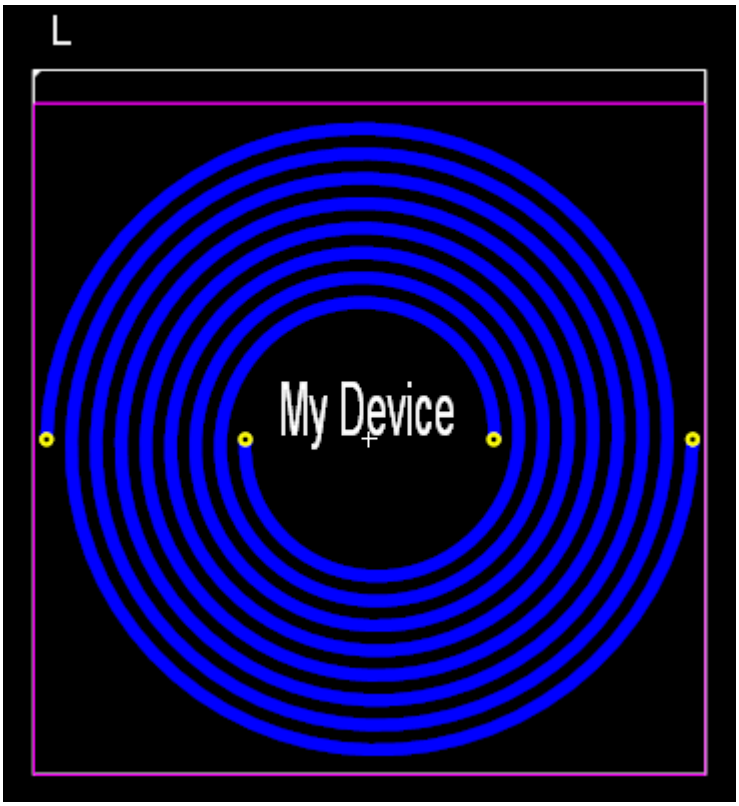
To create an antenna part click the  button in the **The Part Builder**

The part builder will automatically create the copper pattern and pads for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

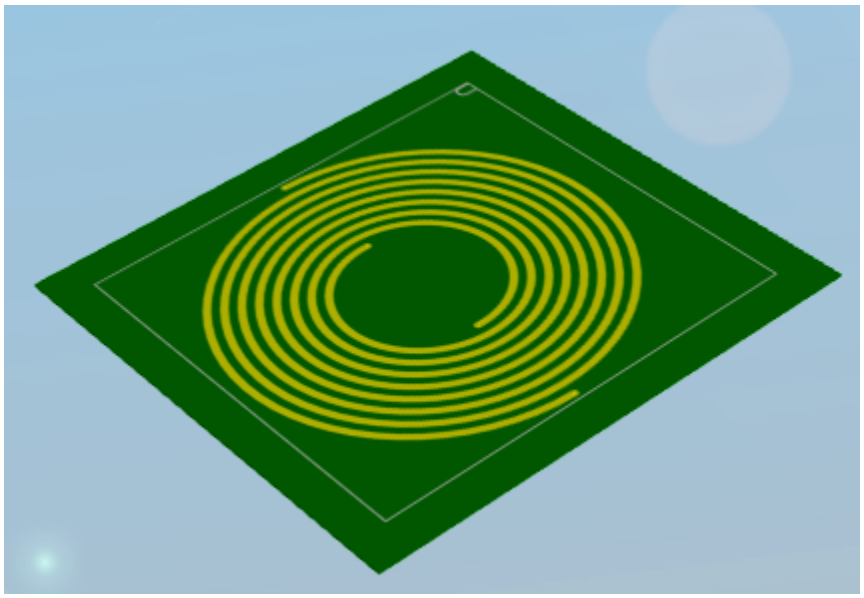


Schematic Symbol without outer ring





Copper pattern and pads



3D View of the antenna

Device Overview

Inductor Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

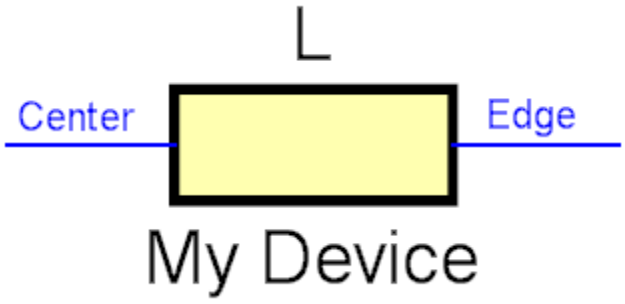
Package Placement Point

Package Symbols

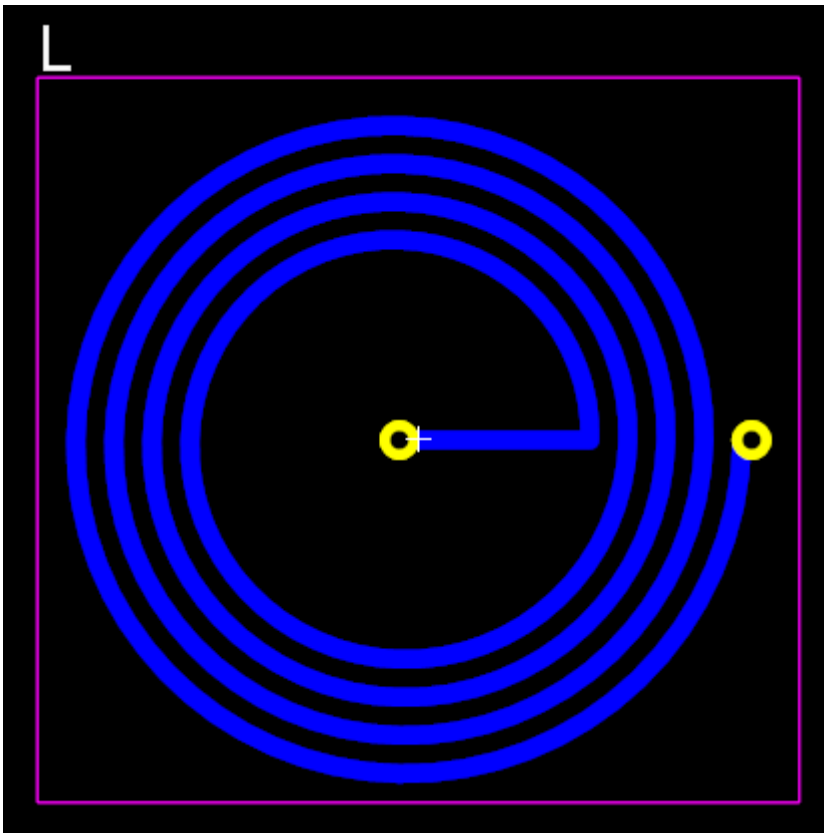
You can create a coil pattern on a copper layer to act as an inductor.

To create a Inductor part click the  button in the **The Part Builder**

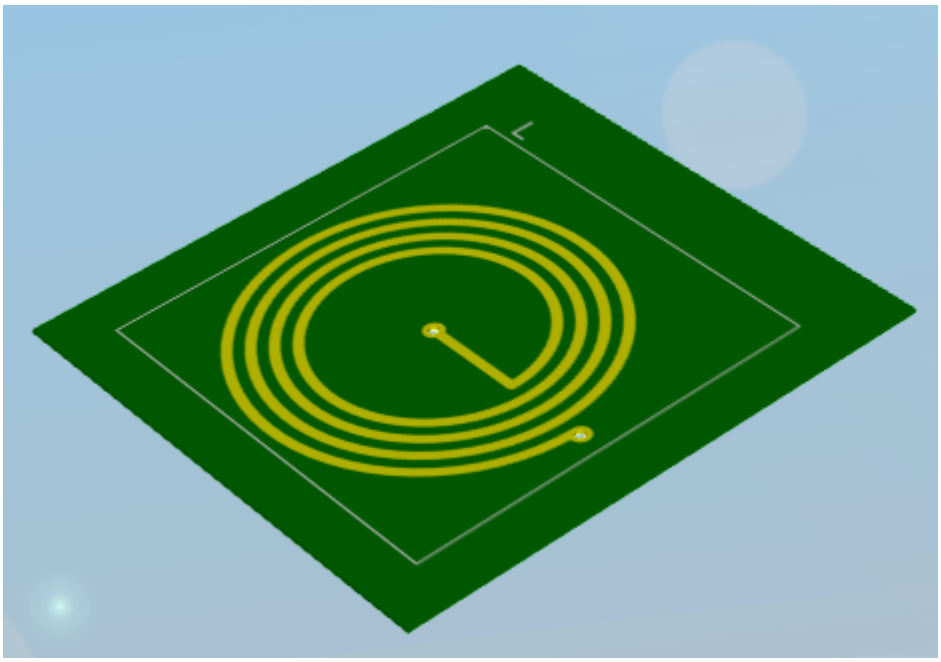
The part builder will automatically create the copper pattern and pads for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.



Schematic Symbol



Copper pattern and pads



3D View of the inductor

Device Overview

Inductor Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

Coming soon...

1.2.5.9.13.2 Antenna

A PCB (Printed Circuit Board) antenna, also known as a trace antenna, is a type of antenna that consists of a trace drawn on a PCB. This can be a simple line or an intricate shape, depending on the specific antenna design and the frequency at which it is intended to operate.

Here are some popular types of PCB antennas:

Monopole Antenna

This is the simplest form of PCB antenna, consisting of a single straight line. These are easy to design and implement, but they require a ground plane to operate correctly.

Dipole Antenna

This antenna consists of two lines or traces of equal length, positioned end to end with a small space in between. This space is where the feed line connects. Dipole antennas do not require a ground plane to operate.

Inverted-F Antenna (IFA)

This type of antenna is a form of monopole antenna that has one end folded back over itself, forming a shape like an inverted "F". It is a popular choice for wireless communication devices because of its good performance and compact size.

Planar Inverted-F Antenna (PIFA)

A PIFA is essentially an IFA with the top section covered by a ground plane. It's a popular antenna for mobile phones due to its compact size and ability to operate at multiple frequencies.

Slot Antenna

This is an antenna made by cutting a slot into a metal surface or layer of the PCB.

Patch Antenna

This is a type of antenna that consists of a flat rectangular or square conductor mounted over a ground plane. They're often used for microwave frequency applications.

Fractal Antenna

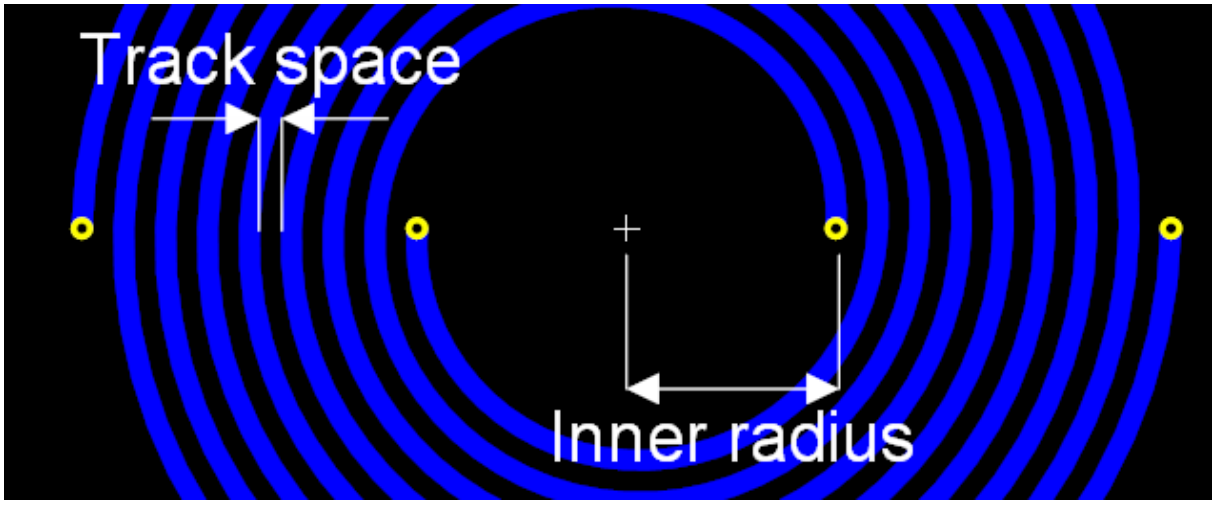
This type of antenna uses a fractal pattern, which is a complex geometric shape that can be split into parts, each of which is a reduced-scale copy of the whole.

The design of a PCB antenna requires careful consideration of the desired frequency, bandwidth, efficiency, and directivity, among other factors. The materials, thickness of the board, and other components placed nearby can also have a significant effect on the antenna's performance.

It's always recommended to simulate the antenna design using appropriate software to predict its performance before fabricating the PCB. Testing the fabricated PCB antenna in a controlled environment can help validate its performance and make any necessary adjustments.

You can create an arial from copper traces on a PCB.

Antenna Parameters



Inner Radius. The inner radius. See the diagram above.

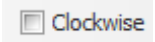
Track Width. The width of the copper track forming the antenna.

Trace Space. The space between copper rings. See the diagram above.

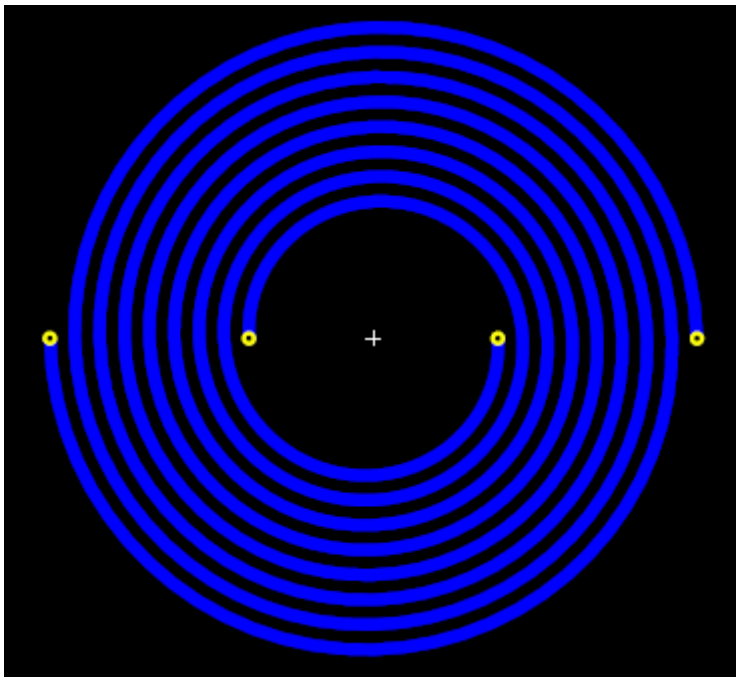
Rings. The number of complete rings.

Check the 

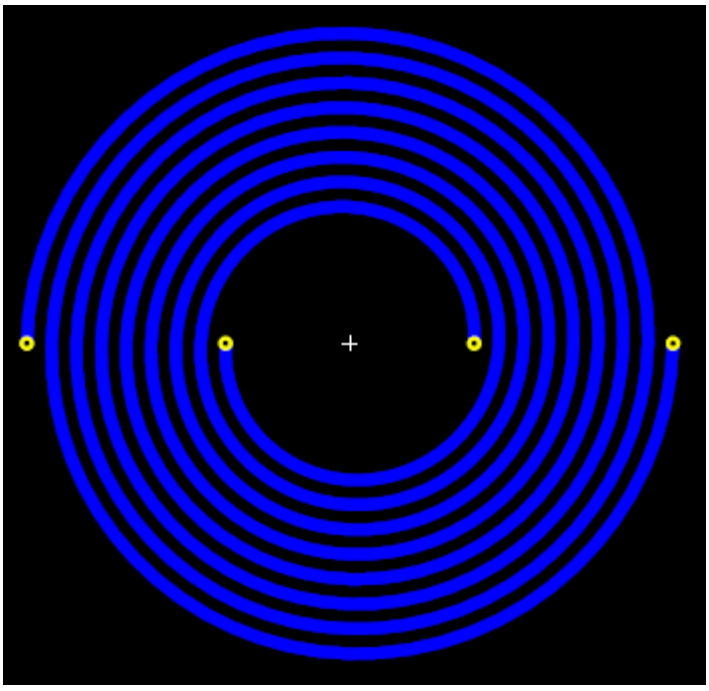
checkbox to use plated through holes for the pads. Uncheck to use SMT pads,

Check the 

checkbox is run the rings clockwise. Uncheck to run the rings in the opposite direction.

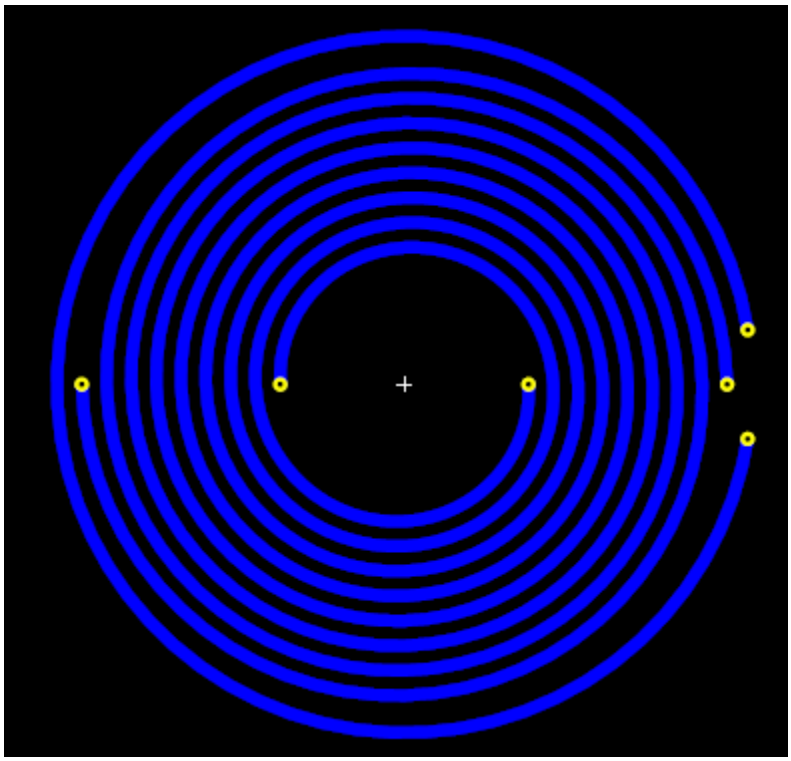


Clockwise



Anticlockwise

Check the Outer Ring checkbox to add an outer ring.



Antenna with outer ring

A monopole antenna is a class of radio antenna that consists of a straight rod or wire, usually mounted perpendicularly over some type of conductive surface, called a ground plane. In the context of PCB (Printed Circuit Board) antennas, the monopole can be a physical wire or component mounted on the board, or it can be a trace etched onto the PCB itself.

Here are some key characteristics of monopole PCB antennas:

- **Design and Implementation:** Monopole antennas are relatively simple in design and implementation. The length of the antenna is typically about one-quarter of the wavelength of the signal it's designed to receive or transmit. The antenna is fed at one end, with the feed line connected to the bottom of the antenna and the ground plane.
- **Ground Plane:** The ground plane is critical for a monopole antenna to function correctly. It serves to reflect the signal, creating an image of the antenna that effectively turns it into a half-wave dipole antenna. The size and shape of the ground plane can significantly affect the performance of the antenna.
- **Directionality and Polarization:** Monopole antennas are omnidirectional in the plane perpendicular to the antenna and have a null in the direction of the antenna's length. They typically produce vertically polarized waves.
- **Applications:** Monopole antennas are often used in mobile and portable devices due to their simple design and omnidirectional radiation pattern. They're used in a

wide range of applications, including mobile communications, Wi-Fi, radio broadcasting, and more.

When designing a monopole PCB antenna, it's essential to consider factors such as the frequency of operation, the size and shape of the ground plane, the materials used in the PCB, and the placement of other components on the board. These factors can all impact the performance of the antenna.

It's generally a good idea to use antenna design software to simulate the antenna's performance before fabrication, and to test the actual PCB in a controlled environment to validate its performance.

A dipole antenna is a common type of radio antenna that consists of two identical conductive elements such as metal wires or rods. These elements are typically bilaterally symmetrical with each other, and the radio signal is fed into the antenna at the center between these two elements.

In the context of a Printed Circuit Board (PCB), a dipole antenna can be designed by etching two lines of equal length on the board, positioned end to end with a small space between where the feed line connects. The dipole antenna is one of the simplest and most widely used antenna setups.

Key characteristics of dipole PCB antennas:

- **Design and Implementation:** In its simplest form, the length of each side of a dipole antenna should be approximately one quarter of the wavelength (λ/4) of the intended operating frequency. The overall length of the antenna then becomes half the wavelength (λ/2).
- **Directionality and Polarization:** Dipole antennas are omnidirectional when viewed from above (from the perspective looking down the axis of the two elements), meaning they radiate radio waves equally in all horizontal directions. They typically produce linearly polarized waves.
- **Balanced Antenna:** A dipole is a balanced antenna, meaning it does not require a ground plane to function. This can be beneficial in situations where the introduction of a ground plane is impractical.
- **Applications:** Due to their simple design, balanced operation, and omnidirectional radiation pattern, dipole antennas are often used in various wireless

communication systems, including Wi-Fi, Bluetooth, and other radio communication systems.

When designing a dipole PCB antenna, it's essential to take into consideration factors such as the frequency of operation, the dielectric constant of the PCB substrate, and the thickness of the copper layer. Other components and traces on the PCB can also impact the performance of the antenna, and these factors should be carefully evaluated and possibly simulated before the final design is decided.

After fabrication, it is typically necessary to measure and tune the antenna in a real-world environment to ensure optimal performance. This process typically involves trimming the length of the antenna elements to resonate at the desired frequency.

An Inverted-F Antenna (IFA) is a type of antenna used in wireless communication, especially in compact devices like smartphones and Wi-Fi devices. It is called an Inverted-F because the antenna diagram looks like an inverted letter 'F'.

The IFA is a variant of the monopole antenna and is designed to be compact, making it ideal for small electronic devices. An IFA can be implemented on a printed circuit board (PCB) either as a physical component or more commonly as a trace etched onto the PCB itself.

Here are some key characteristics of the IFA:

- **Design:** An IFA consists of three main parts: a radiating element, a feed point, and a grounding element. The radiating element is typically a short, straight line, while the grounding element is a longer, bent line that forms the 'F' shape.
- **Size and Resonant Frequency:** The total length of the antenna (the radiating element plus the grounding element) is usually approximately a quarter-wavelength (λ/4) at the resonant frequency.
- **Ground Plane:** Like a monopole antenna, an IFA requires a ground plane to function properly. This ground plane is usually formed by the PCB itself.
- **Impedance Matching:** The IFA is generally easier to match to 50 ohms (standard RF circuit impedance) than a monopole, which can simplify design and improve performance.
- **Directionality and Polarization:** The IFA is omnidirectional in the plane perpendicular to the antenna, and it typically produces vertically polarized waves.

When designing an Inverted-F PCB antenna, it's essential to consider factors like the intended frequency of operation, the size and characteristics of the PCB (which acts as the ground plane), and the placement of other components on the board. It's also crucial to test and potentially adjust the antenna after fabrication to ensure optimal performance in the real-world environment where it will be used.

A Planar Inverted-F Antenna (PIFA) is a type of antenna that's commonly used in wireless communication, particularly for mobile devices due to its compact size and ability to support multiple frequency bands.

The PIFA design is a variation of the Inverted-F Antenna (IFA) and is characterized by a '3D' structure, where the antenna is comprised of a flat rectangular conductive plate (the radiating element) positioned above, and parallel to, a ground plane with a shorting plate or pin at one end. This structure makes the PIFA more compact compared to a simple IFA, allowing it to fit more easily inside small devices.

Key characteristics of the PIFA include:

- **Design:** A typical PIFA includes a radiating patch, a ground plane, and a shorting plate or pin that connects the patch to the ground plane at one end. The signal is usually fed to the radiating patch at a point between the shorting plate and the open end of the patch.
- **Resonant Frequency:** PIFAs are typically designed to be resonant at a quarter-wavelength (λ/4), but because of the shorting plate, they can achieve this resonance at a smaller physical size than a typical quarter-wavelength antenna.
- **Bandwidth and Multiband Operation:** PIFAs can be designed to support wide bandwidth or multiband operation, making them suitable for use in devices that need to operate on different frequency bands. This is achieved by altering the shape and structure of the radiating patch and/or using multiple shorting pins.
- **Radiation Pattern:** PIFAs are generally omnidirectional in the plane perpendicular to the antenna (i.e., in the horizontal plane for a vertically oriented PIFA). This is an advantage in mobile devices, which can be oriented in various ways during use.
- **Polarization:** PIFAs typically produce linearly polarized waves.

Designing a PIFA on a PCB requires careful consideration of various factors such as the size and dielectric properties of the PCB, the desired frequency/frequencies of operation, the placement of the antenna relative to other components, and the overall size constraints. As with any antenna design, testing and potentially adjusting the design after fabrication is critical to ensure optimal real-world performance.

A slot antenna is a type of radio antenna that is used in a wide range of applications, especially at higher frequencies (microwave and above). It consists of a slot or slots cut out from a conductive surface, such as a metal plate. The slot radiates radio waves in a manner similar to a dipole antenna, with the slot acting as the equivalent of a pair of separated dipole elements.

In the context of printed circuit board (PCB) design, a slot antenna can be realized by creating a slot or gap in the conductive ground plane of the PCB.

Here are some key characteristics of a slot PCB antenna:

- **Design and Implementation:** A slot antenna can be of various shapes, but a simple and common form is a rectangular slot, where the length of the slot is approximately half the wavelength (λ/2) of the signal it's intended to receive or transmit.
- **Directionality and Polarization:** A slot antenna emits energy in a pattern that is roughly omnidirectional in the plane of the slot. The polarization of the emitted radiation is perpendicular to the orientation of the slot. So, a vertically oriented slot would produce horizontally polarized radiation.
- **Complementary to Dipole Antennas:** A slot antenna is considered the "dual" of a dipole antenna, meaning it operates under the same basic principles but with the roles of the electric and magnetic fields reversed. Where a dipole antenna is made from a conductive material and operates via the movement of electric charge, a slot antenna is created in a conductive plane (representing an absence of material) and operates via changes in the magnetic field.
- **Applications:** Due to their versatility, slot antennas can be used in a variety of applications, including in radar systems, satellite systems, and in Wi-Fi devices for wireless communication. They are also used in the design of PCBs where the available space is limited.

Designing a slot antenna on a PCB requires a good understanding of the principles of antenna design, including impedance matching, resonant frequency calculation, and the properties of the materials used in the PCB. The antenna should also be tested in a real-world environment to validate its performance and to make any necessary adjustments.

A patch antenna is a type of radio antenna with a low profile, which can be mounted on a flat surface. It consists of a flat rectangular or circular sheet or "patch" of metal, mounted over a larger sheet of metal called a ground plane. Patch antennas are a common type of microstrip antenna, which means they're often fabricated by etching the antenna elements out of a flat sheet of metal, such as the copper layer on a printed circuit board (PCB).

Here are some key characteristics of a patch PCB antenna:

- **Design and Implementation:** A patch antenna is usually designed as a rectangular or circular conductive patch on a ground plane. The antenna is usually fed by a probe (a soldered wire) or by a microstrip line (a thin strip of copper).
- **Resonant Frequency:** The resonant frequency of a patch antenna is determined primarily by its size. For a rectangular patch, the length of the patch is approximately half of the wavelength in the dielectric medium of the frequency it's intended to receive or transmit.
- **Polarization and Directionality:** Patch antennas produce linearly polarized waves, with the electric field oriented perpendicular to the ground plane. They are

usually designed to be directional and emit towards the area above the ground plane, making them useful for surface mounted applications.

- **Applications:** Due to their low profile and ease of fabrication, patch antennas are used in various applications where size, weight, and cost are critical. These include wireless communication devices like mobile phones, Wi-Fi routers, and GPS receivers.

Designing a patch antenna on a PCB requires knowledge of RF circuit design, and careful consideration of factors like the dielectric constant of the substrate, the thickness of the substrate, and the operating frequency. The performance of the antenna should be validated in a real-world environment to ensure optimal performance.

A fractal antenna is a type of antenna that uses a fractal, self-similar design to maximize the effective length, or "load" of the material that can receive or transmit electromagnetic radiation within a given total surface area or volume. Such an antenna takes advantage of the fractal element's ability to resonate at frequencies that are harmonically related, thereby allowing a single antenna structure to receive or transmit effectively over a wide range of frequencies.

In the context of printed circuit boards (PCBs), a fractal antenna can be implemented by etching the fractal pattern onto the PCB itself.

Here are some key characteristics of a fractal PCB antenna:

- **Design and Implementation:** A fractal antenna is made by applying a fractal design (a geometric shape that can be split into parts, each of which is a reduced-scale copy of the whole) to the elements of an antenna. There are many possible fractal shapes, but common examples used in antenna design include the Koch snowflake, the Sierpinski gasket, and the Minkowski island.
- **Multiband and Wideband Capabilities:** Because of their self-similar nature, fractal antennas can resonate at multiple frequencies simultaneously. This makes them useful for wideband and multiband applications.
- **Size Reduction:** Fractal elements allow an antenna to take up less space than a traditional design while maintaining similar performance. This can be especially beneficial in applications where space is at a premium, such as in mobile devices.
- **Applications:** Fractal antennas are used in a wide range of wireless communication systems, including cell phones, Wi-Fi devices, and other wireless networking devices. They're also used in radar and broadcasting systems.

Designing a fractal antenna on a PCB requires careful consideration of the intended operating frequencies, the specific fractal geometry to be used, and the materials and layout of the PCB. As always, the antenna should be thoroughly tested after fabrication to ensure that it performs as expected in its intended application.

1.2.5.9.13.3 Axial

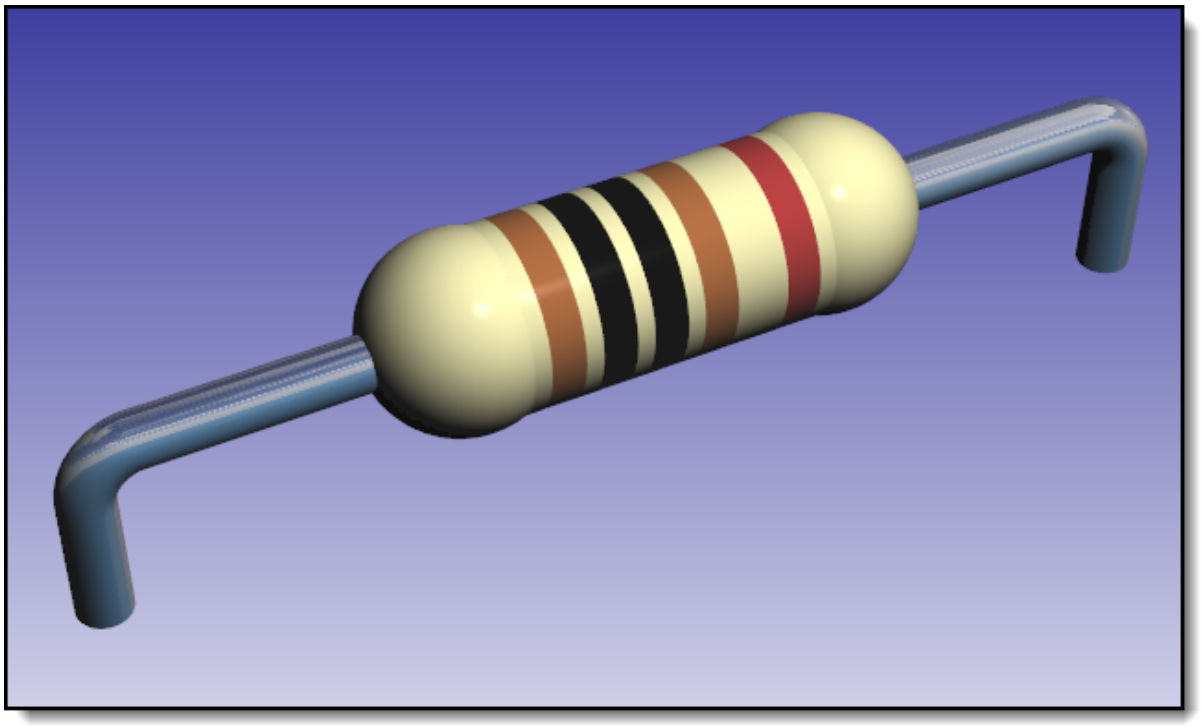
Axial PCB packages are a type of through-hole electronic component package that is commonly used for resistors, capacitors, and diodes. Axial components are named for the axial orientation of the leads (or legs) of the component, which are located at opposite ends of the component and extend outward along a common axis.

In an axial PCB package, the leads are typically inserted into plated through-holes on the PCB and then soldered to the PCB to form a mechanical and electrical connection. The leads can also be bent at a right angle to the component body and inserted into mounting holes on the PCB, where they are secured with a nut.

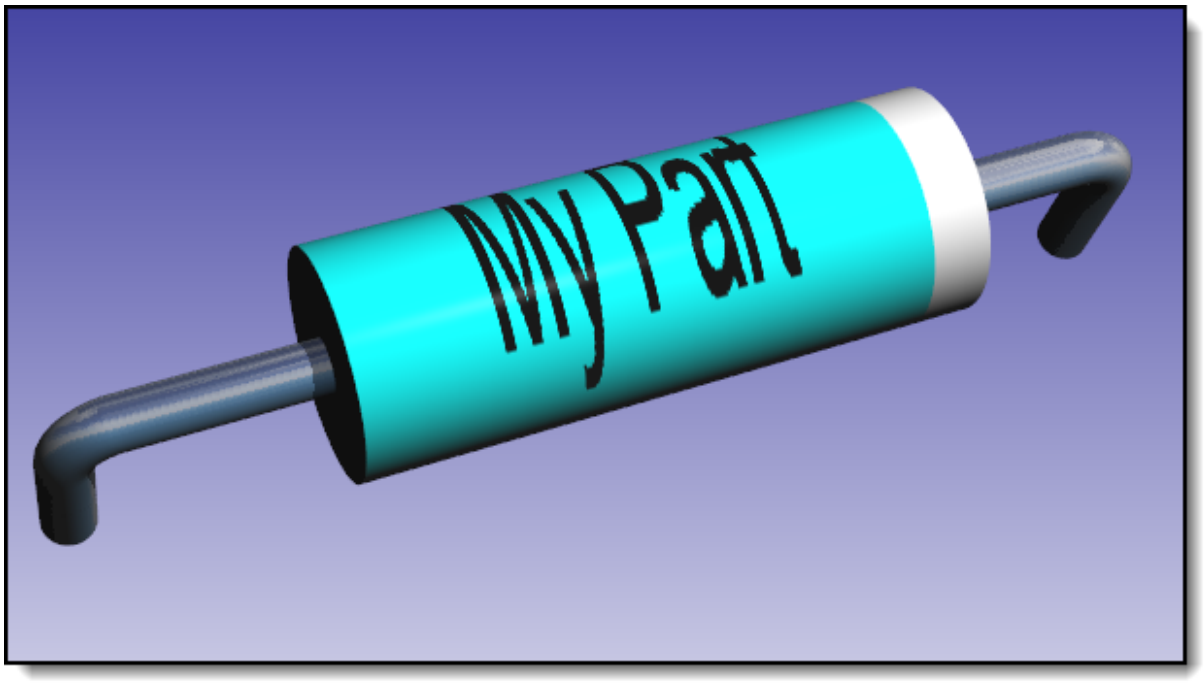
Axial PCB packages come in various shapes and sizes, and their dimensions depend on the specific component they are used for. Some common types of axial components include axial electrolytic capacitors, axial ceramic capacitors, and axial diodes.

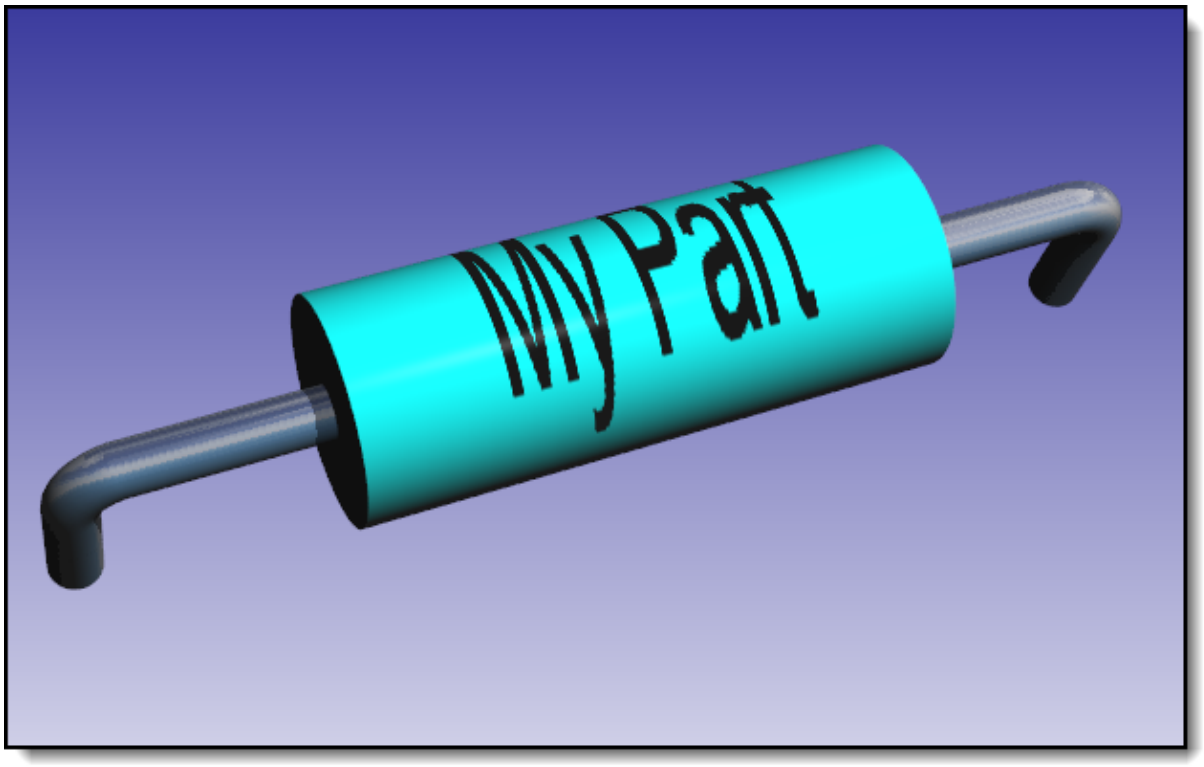
One advantage of axial PCB packages is that they are relatively easy to install and remove, and can be inserted into the PCB by hand. They are also generally more robust than surface-mount components and are less prone to mechanical stress or damage.

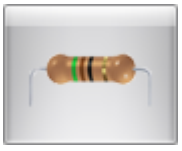
However, axial PCB packages take up more space on the PCB than surface-mount components and can limit the density of other components around them. They are also less suitable for high-frequency applications due to their longer lead lengths, which can introduce unwanted inductance and capacitance into the circuit.



An Axial Part With Banding





To create an Axial part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Axial Parameters

Pads

Package Color

Package Reference

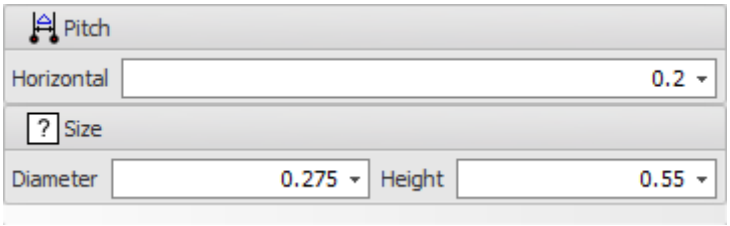
Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols



1.2.5.9.13.4 BGA

Ball Grid Array (BGA) is a type of surface-mount packaging used for integrated circuits (ICs). BGA packages are used to permanently mount devices such as microprocessors. A BGA can provide more interconnection pins than dual in-line or flat packages.

Here are some characteristics of BGA ICs:

- **Design:** The bottom of the BGA is covered in solder balls (hence the name "Ball Grid Array"). These solder balls are placed at the locations where the IC needs to make electrical contact with the PCB. The solder balls are melted to attach the IC to the PCB, making both electrical and mechanical connections.
- **High Density:** Because the entire bottom of the BGA can be used for connections, it can accommodate a larger number of pins within the same space compared to

packages that only have pins around the edge. This high density makes BGA a good choice for complex, high-performance ICs like microprocessors and FPGAs.

- **Better Performance:** The shorter and more direct paths between the IC and the PCB provided by the BGA design can result in better electrical and thermal performance compared to other packages.
- **Soldering and Inspection:** BGA packages can be harder to solder correctly compared to other packages, due to the fact that the solder balls are hidden underneath the IC. Specialized equipment is needed for both the soldering and inspection process.
- **Rework and Repair:** BGA packages are also challenging to rework or repair. If a solder joint fails or the IC needs to be replaced, specialized equipment and skills are required.

BGA ICs are commonly used in applications that require high performance and a large number of interconnections, such as in servers, networking equipment, high-end consumer electronics, and more.

To create a BGA part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

BGA Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

A ball grid array (BGA) is a type of surface-mount packaging used for integrated circuits. BGA packages are used to permanently mount devices such as microprocessors. A BGA can provide more interconnection pins than can be put on a dual in-line or flat package. The whole bottom surface of the device can be used, instead of just the perimeter. The leads are also on average shorter than with a perimeter-only type, leading to better performance at high speeds.

Soldering of BGA devices requires precise control and is usually done by automated processes. A BGA device is never mounted in a socket in use.

Advantages

The BGA is descended from the pin grid array (PGA), which is a package with one face covered (or partly covered) with pins in a grid pattern. These pins conduct electrical signals from the integrated circuit to the printed circuit board (PCB) on which it is placed. In a BGA, the pins are replaced by balls of solder stuck to the bottom of the package. These solder spheres can be placed manually or with automated equipment. The solder spheres are held in place with a tacky flux until soldering occurs. The device is placed on a PCB with copper pads in a pattern that matches the solder balls. The assembly is then heated, either in a reflow oven or by an infrared heater, causing the solder balls to melt. Surface tension causes the molten solder to hold the package in alignment with the circuit board, at the correct separation distance, while the solder cools and solidifies.

The BGA is a solution to the problem of producing a miniature package for an integrated circuit with many hundreds of pins. Pin grid arrays and dual-in-line surface mount (SOIC) packages were being produced with more and more pins, and with decreasing spacing between the pins, but this was causing difficulties for the soldering process. As package pins got closer together, the danger of accidentally bridging adjacent pins with solder grew. BGAs do not have this problem when the solder is factory-applied to the package.

A further advantage of BGA packages over packages with discrete leads (i.e. packages with legs) is the lower thermal resistance between the package and the

PCB. This allows heat generated by the integrated circuit inside the package to flow more easily to the PCB, preventing the chip from overheating.

The shorter an electrical conductor, the lower its inductance, a property which causes unwanted distortion of signals in high-speed electronic circuits. BGAs, with their very short distance between the package and the PCB, have low inductances and therefore have far superior electrical performance to leaded devices.

Disadvantages

A disadvantage of BGAs is that the solder balls cannot flex in the way that longer leads can, so they are not compliant. As with all surface mount devices, bending due to a difference in coefficient of thermal expansion between PCB substrate and BGA (thermal stress) or flexing and vibration (mechanical stress) can cause the solder joints to fracture.

Thermal expansion issues can be overcome by matching the mechanical and thermal characteristics of the PCB to those of the package. Typically, plastic BGA devices more closely match PCB thermal characteristics than ceramic devices.

The predominant use of RoHS compatible lead-free solder alloy assemblies have presented some further challenges to BGAs including "head in pillow" soldering phenomenon, "pad cratering" problems as well as their decreased reliability versus lead-based solder BGAs in extreme operating conditions such as high temperature, high thermal shock and high gravitational force environments, in part due to lower ductility of RoHS-compliant solders.

Mechanical stress issues can be overcome by bonding the devices to the board through a process called "under filling", which injects an epoxy mixture under the device after it is soldered to the PCB, effectively gluing the BGA device to the PCB. There are several types of under fill materials in use with differing properties relative to workability and thermal transfer. An additional advantage of under fill is that it limits tin whisker growth.

Another solution to non-compliant leads is to put a "compliant layer" in the package that allows the balls to physically move in relation to the package. This technique has become standard for packaging DRAMs in BGA packages.

Once the package is soldered down, it may be difficult to look for soldering faults. X-ray machines, Industrial CT Scanning machines, special microscopes as well as endoscopes to look underneath the soldered package have been developed to overcome this problem. If a BGA is found to be badly soldered, it can be removed in a rework station, which is a jig fitted with infrared lamp (or hot air), a thermocouple and a vacuum device for lifting the package. The BGA can be replaced with a new one, or the BGA can be refurbished (or reballed) and re-installed on the circuit board. Pre-configured solder balls matching the array pattern (preforms) can be used to reballing the BGA's when only one or a few need to be reworked.

Due to cost of X-ray BGA inspection, electrical testing is very often used. Very common is boundary scan testing using IEEE 1149.1 JTAG port.

During development it is not practical to solder BGAs into place, and sockets are used instead, but tend to be unreliable. There are two common types of socket: the more reliable type has spring pins that push up under the balls, although it does not allow using BGAs with the balls removed as the spring pins may be too short.

The less reliable type is a ZIF socket, with spring pinchers that grab the balls. This does not work well, especially if the balls are small.

Expensive equipment is required to reliably solder BGA packages; hand-soldering BGA packages is very difficult and unreliable, usable only for the smallest packages in the smallest quantities.

Other BGA package types

CABGA Chip Array Ball Grid Array

CBGA and **PBGA** denote the Ceramic or Plastic substrate material to which the array is attached.

CTBGA Thin Chip Array Ball Grid Array

CVBGA Very Thin Chip Array Ball Grid Array

DSBGA Die-Size Ball Grid Array

FBGA or Fine Ball Grid Array based on ball grid array technology. It has thinner contacts and is mainly used in system-on-a-chip designs. Known as FineLine BGA by Altera.

FCmBGA Flip Chip Molded Ball Grid Array

LBGA Low-profile Ball Grid Array

LFBGA Low-profile Fine-pitch Ball Grid Array

MBGA Micro Ball Grid Array

MCM-PBGA Multi-Chip Module Plastic Ball Grid Array

PBGA Plastic Ball Grid Array

SuperBGA (SBGA) Super Ball Grid Array

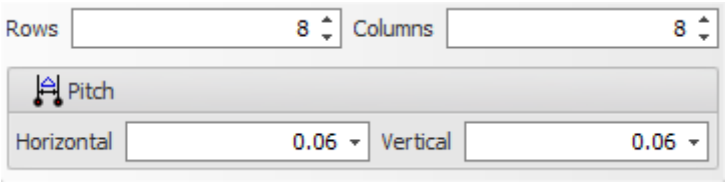
TABGA Tape Array BGA

TBGA Thin BGA

TEPBGA Thermally Enhanced Plastic Ball Grid Array

TFBGA or Thin and Fine Ball Grid Array

UFBGA and **UBGA** and Ultra Fine Ball Grid Array based on pitch ball grid array



1.2.5.9.13.5 Block

1.2.5.9.13.6 Block - Crystal

1.2.5.9.13.7 Castellated PCB

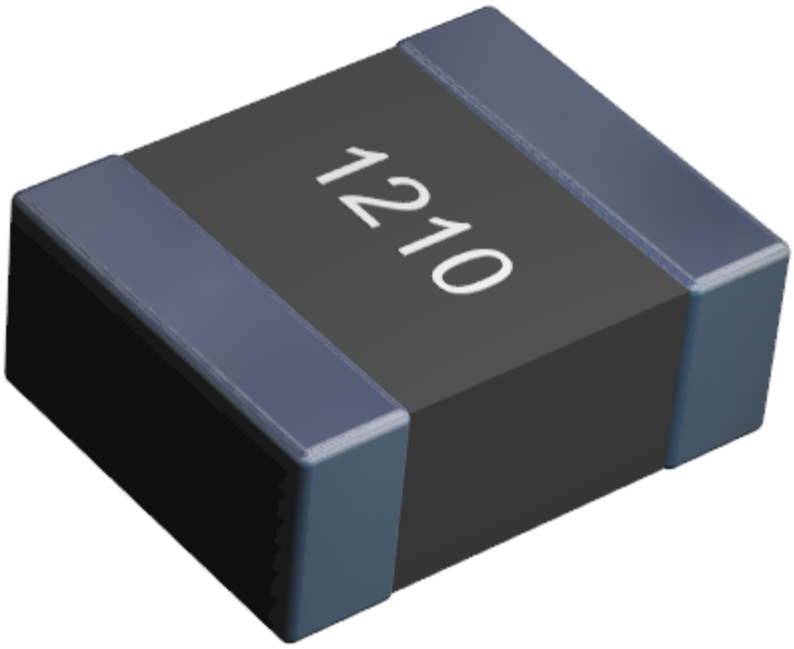
1.2.5.9.13.8 Chip

A two-pin SMT (Surface Mount Technology) chip package is a type of package that is designed to be mounted on the surface of a printed circuit board (PCB), rather than being inserted through holes drilled in the board. This type of package is used for components such as resistors, capacitors, diodes, inductors, and certain types of integrated circuits.

Here are some characteristics of a 2-pin SMT chip package:

- **Design:** The package has two terminals or leads for electrical connection. These terminals are typically located on opposite ends of the package.
- **Installation:** The chip is placed on top of the PCB, and the leads are soldered directly to the board's surface. This is in contrast to through-hole components, which require holes to be drilled into the PCB for mounting.
- **Package Types:** Common types of 2-pin SMT chip packages include chip resistors and capacitors (typically in 0402, 0603, 0805, 1206 sizes, etc., where the numbers refer to dimensions in hundredths of inches), small signal diodes, LEDs, and various types of transistors.
- **Applications:** Two-pin SMT chip packages are used in virtually all modern electronic devices due to their small size, low cost, and ease of manufacturing.
- **Advantages:** Compared to through-hole components, SMT components are smaller and can be placed on both sides of the PCB, allowing for more compact and complex circuit designs. SMT components also typically provide better performance at high frequencies.

For these components to work correctly, they must be carefully designed into the PCB layout, properly placed and soldered onto the PCB, and thoroughly tested to ensure reliable performance.



Chip Device

To create a Chip part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Chip Parameters

Pads

Package Color

Package Reference

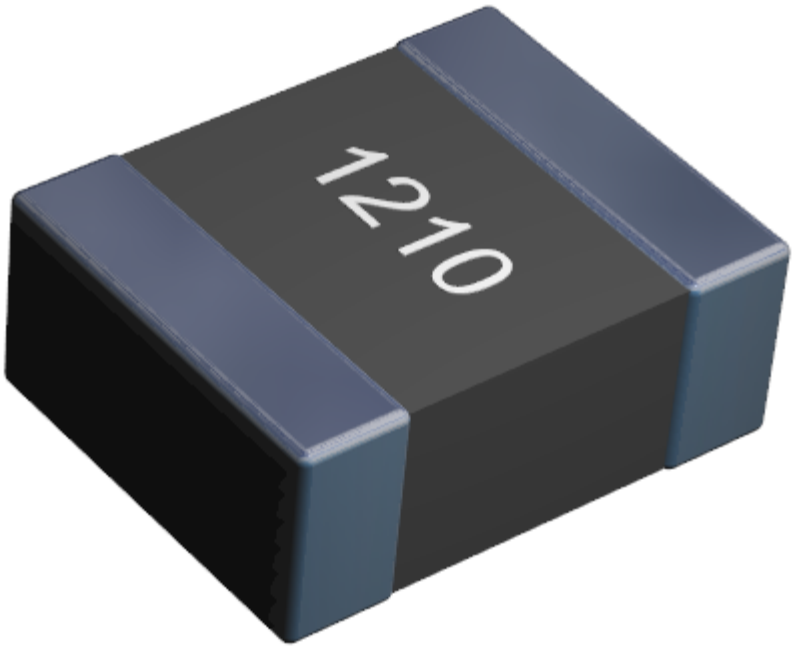
Package Value

Package Silkscreen

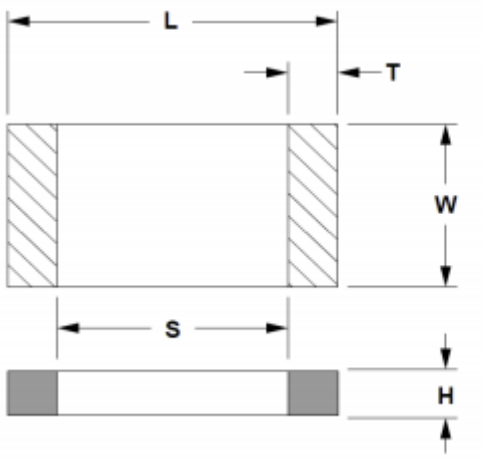
Package Courtyard

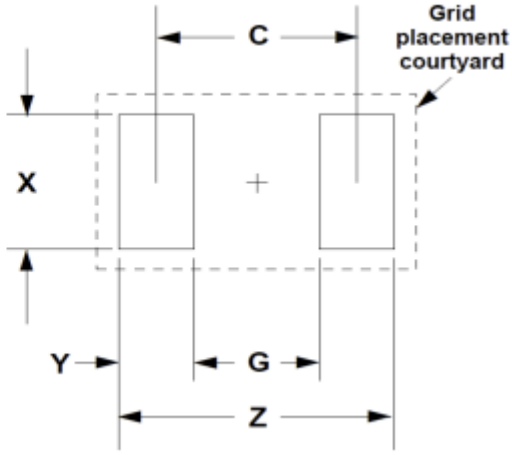
Package Placement Point

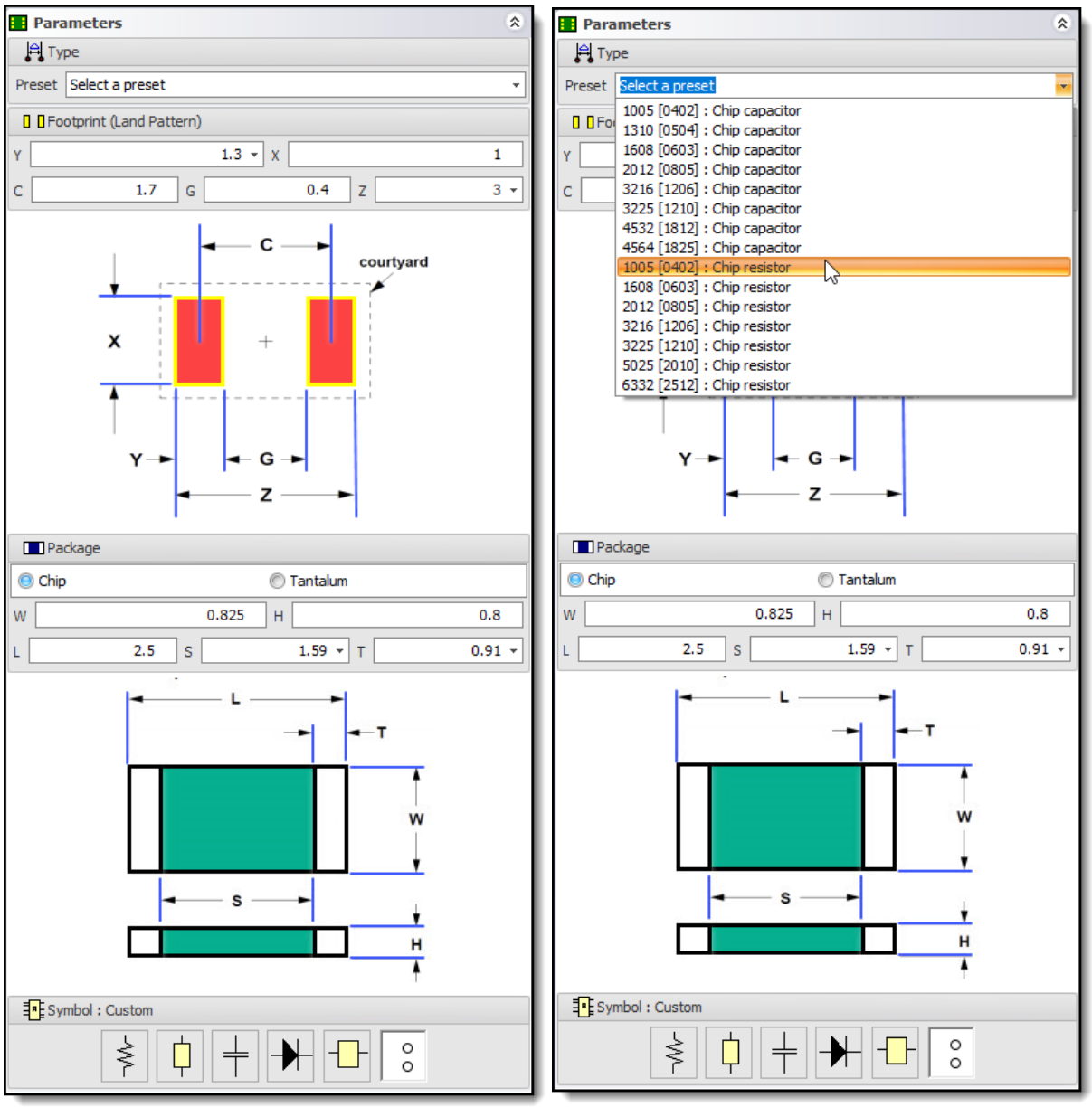
Package Symbols



Chip Device





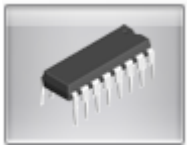


1.2.5.9.13.9 DIP

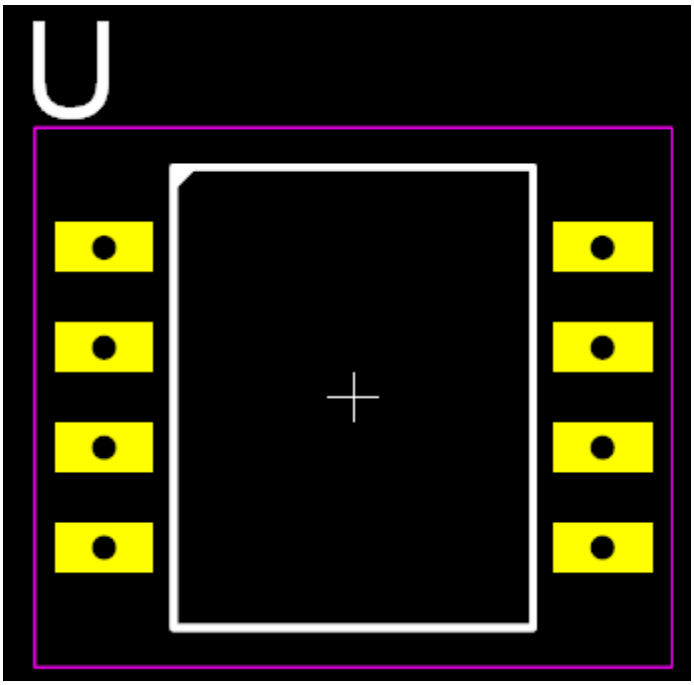
DIP (Dual In-line Package) integrated circuits are a type of electronic packaging used to house integrated circuits (ICs). They are characterized by their two parallel rows of pins, which allow for easy insertion into a socket or breadboard.

DIP ICs were popular in the 1970s and 1980s and were used in a wide range of applications, including computers, calculators, and other electronic devices. They were typically made of ceramic or plastic and could contain anywhere from a few to several hundred individual transistors, resistors, and other components.

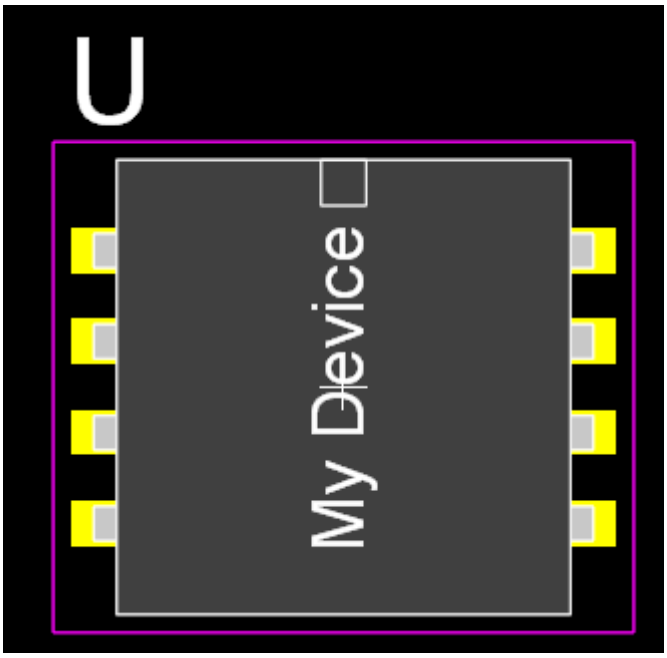
While DIP ICs are still used today in some applications, they have largely been replaced by smaller and more efficient packaging types, such as surface-mount technology (SMT) components. SMT components can be placed directly onto a circuit board, allowing for higher density and greater functionality in smaller spaces.

To create a DIP part click the  button in the **The Part Builder**

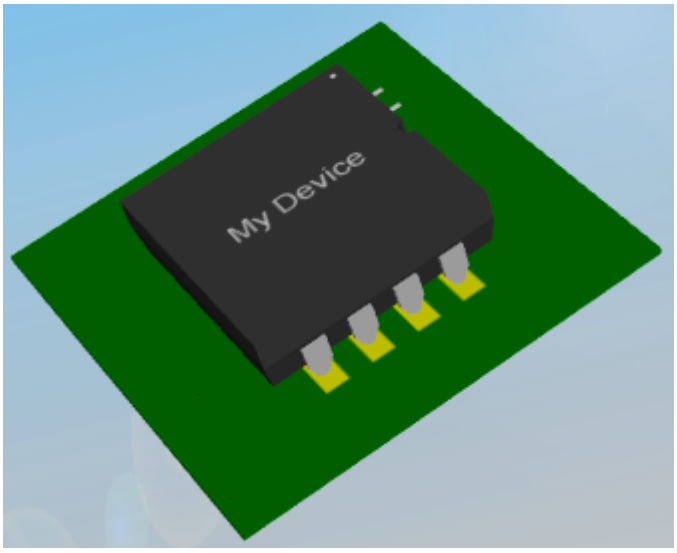
The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.



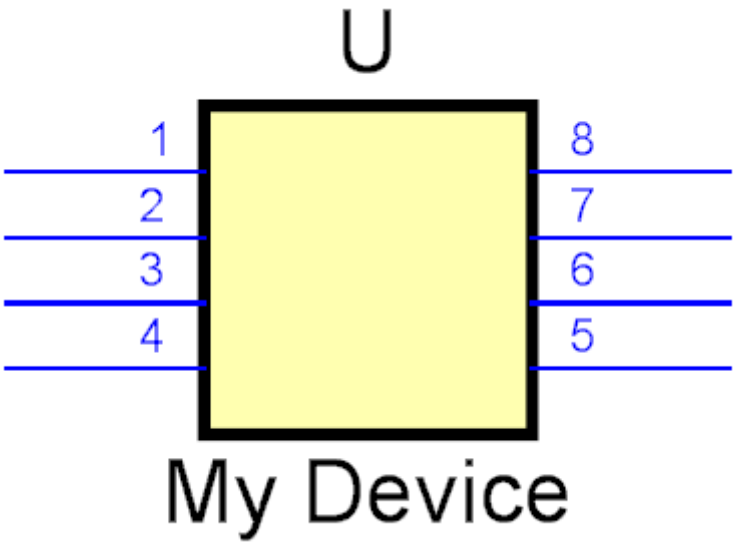
8 pin DIP



8 pin DIP showing package



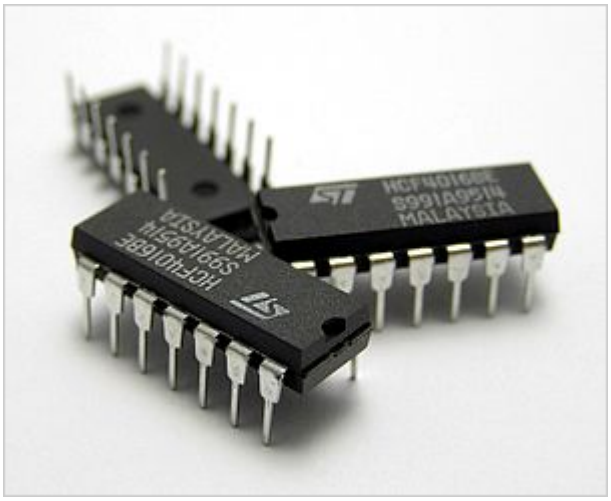
3D View of 8 pin DIP



Schematic Symbol

A dual in-line package (DIP or DIL) is an electronic device package with a rectangular housing and two parallel rows of electrical connecting pins. The package may be through-hole mounted to a printed circuit board or inserted in a socket. Dual-in-line packages were developed in the 1960s when the restricted number of leads available on transistor-style packages became a limitation in the use of integrated circuits. Increasingly complex circuits required more signal and

power supply leads (as observed in Rent's rule); eventually microprocessors and similar complex devices required more leads than could be put on a DIP package, leading to development of higher-density packages.



A DIP is usually referred to as a DIPn, where n is the total number of pins. For example, a microcircuit package with two rows of seven vertical leads would be a DIP14. The above photograph shows three DIP14 ICs. Common packages have as few as four and as many as 64 leads. Many analog and digital integrated circuit types are available in DIP packages, as are arrays of transistors, switches, light emitting diodes, and resistors. DIP plugs for ribbon cables can be used with standard IC sockets.

DIP packages are usually made from an opaque molded epoxy plastic pressed around a tin, silver, or gold-plated lead frame that supports the device die and provides connection pins. Some types of IC are made in ceramic DIP packages, where high temperature or high reliability is required, or where the device has an optical window to the interior of the package. Most DIP packages are secured to a printed circuit board by inserting the pins through holes in the board and soldering them in place. Where frequent replacement of the parts is desired, such as in test fixtures or where programmable devices must be removed for changes, a DIP socket is used. Some sockets include a zero insertion force mechanism.

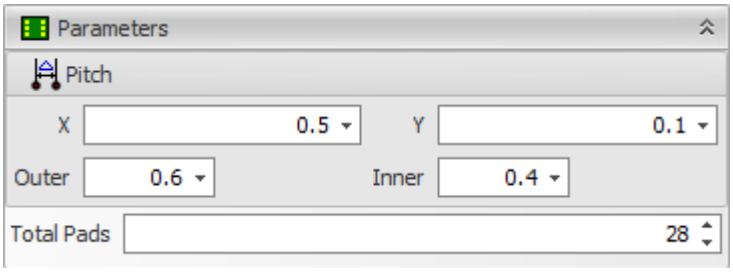
Variations of the DIP package include those with only a single row of pins, possibly including a heat sink tab in place of the second row of pins, and types with four rows of pins, two rows, staggered, on each side of the package. DIP packages have been mostly displaced by surface-mount package types, which avoid the expense of drilling holes in a printed circuit board and which allow higher density of interconnections.

Mounting

DIPs can be mounted either by through-hole soldering or in sockets. Sockets allow easy replacement of a device and eliminate the risk of damage from overheating during soldering. Generally sockets were used for high-value or large ICs, which

cost much more than the socket. Where devices would be frequently inserted and removed, such as in test equipment or EPROM programmers, a zero insertion force socket would be used.

DIPs are also used with breadboards, a temporary mounting arrangement for education, design development or device testing. Some hobbyists, for one-off construction or permanent prototyping, use point-to-point wiring with DIPs, and their appearance when physically inverted as part of this method inspires the informal term "dead bug style" for the method.



DIP Parameter Editor

Pitch

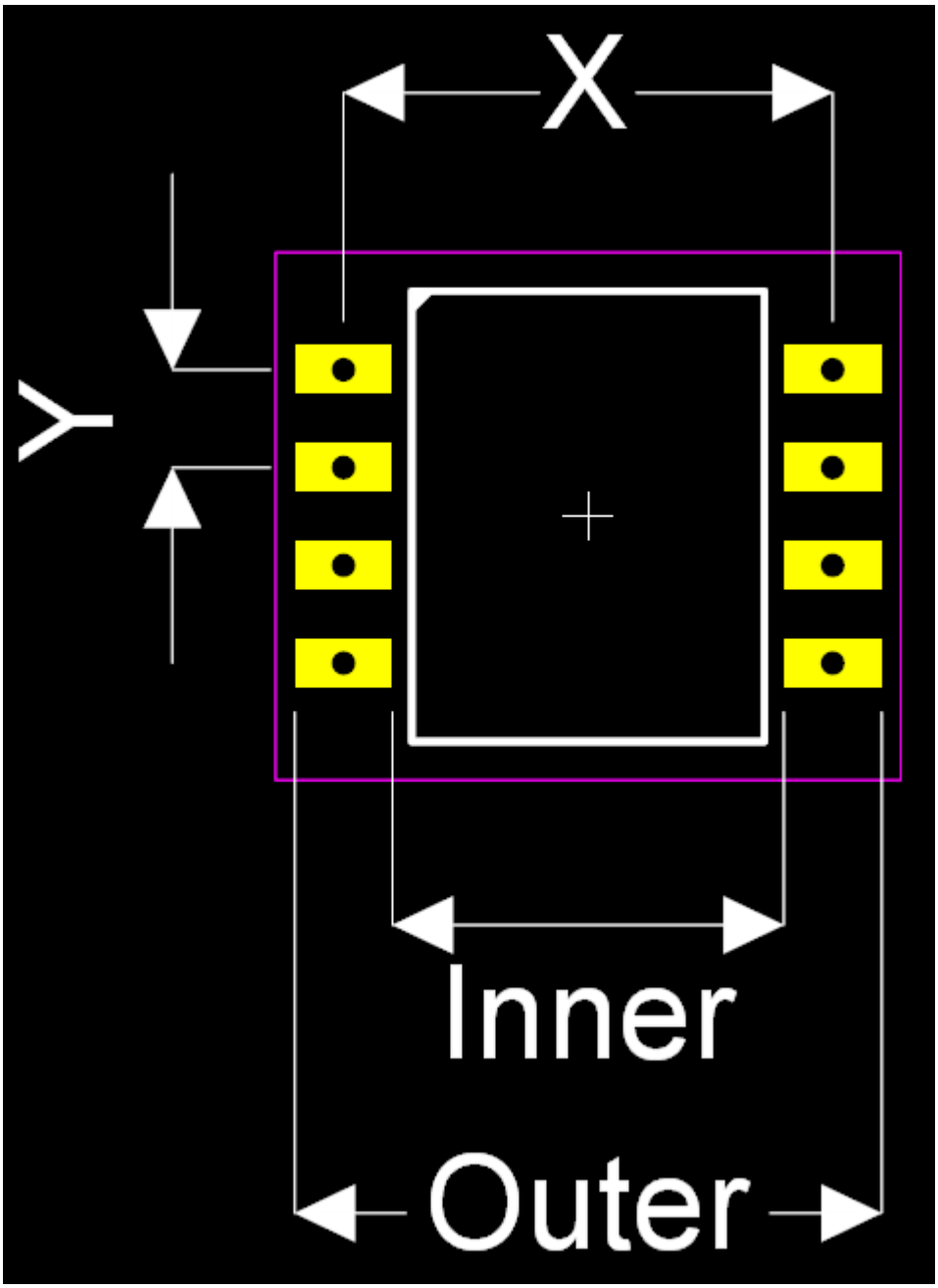
X the horizontal pitch between pad columns centers.

Y the vertical pitch between pad row centers.

Outer The horizontal distance between pad outer edges.

Inner The horizontal distance between pad inner edges.

Total Pads The total number of pads.



DIP Parameters

1.2.5.9.13.10 SIP

SIP (Single In-line Package) integrated circuits are a type of electronic packaging similar to DIP ICs, but with a single row of pins instead of two. SIP ICs are typically used in applications where space is limited, as they take up less board space than DIP ICs.

SIP ICs can contain anywhere from a few to several hundred individual components, depending on their intended use. They are often used in consumer electronics, such as digital cameras, printers, and audio equipment, as well as in industrial and automotive applications.

Like DIP ICs, SIP ICs have largely been replaced by smaller and more efficient packaging types, such as surface-mount technology (SMT) components. However, they are still used in some applications where through-hole components are required for mechanical stability or where the extra space provided by the single row of pins is advantageous.

1.2.5.9.13.11 Disc

A two-pin disc package typically refers to a type of packaging used for certain electronic components, like ceramic disc capacitors or certain types of diodes or transistors. The package essentially includes the disc-shaped component body with two leads or terminals extending from it for electrical connection.

Here are some characteristics of a 2-pin disc package:

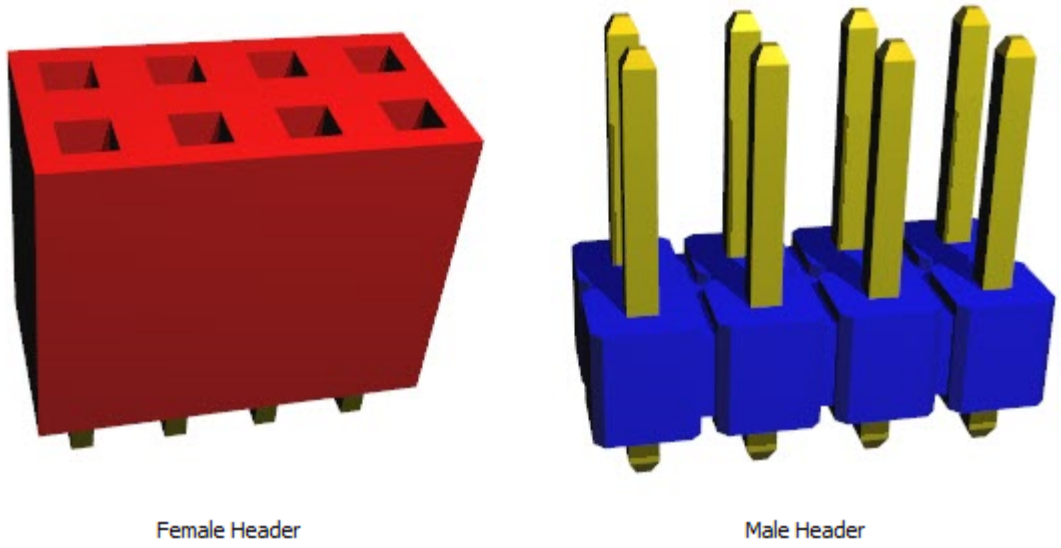
- **Design:** The main body of the component is disc-shaped, usually made of a ceramic material for capacitors. Two metal leads extend from opposite ends of the disc, forming the electrical connections to the circuit.
- **Installation:** These components can be surface-mounted or through-hole mounted depending on the design of the leads and the printed circuit board (PCB). For through-hole mounting, the leads are inserted into drilled holes on the PCB and soldered into place.
- **Package Types:** The most common type of 2-pin disc package is the ceramic disc capacitor. However, some types of transistors or diodes also come in a similar package.
- **Applications:** Two-pin disc packages are used in a variety of electronic devices and circuits. Ceramic disc capacitors are used for a variety of purposes, including bypass, coupling, filtering, and timing applications.

As with all electronic components, these need to be properly incorporated into the circuit design, correctly installed on the PCB, and thoroughly tested to ensure reliable operation.

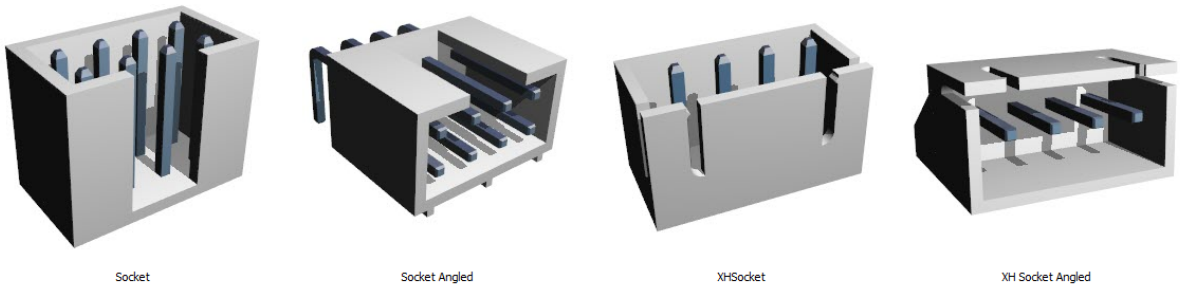
1.2.5.9.13.12 Generic

1.2.5.9.13.13 Header

A PCB header is a type of electrical connector that is used as an interface to connect one circuit board to another, or to connect wires or cables to a printed circuit board (PCB). Headers come in many varieties but generally fall under two categories: male headers (also known as "pin headers") which have protruding pins, and female headers (also known as "socket headers") which have receptacles for accepting pins.



Headers



Sockets

Here are some characteristics of PCB headers:

- **Design:** Headers typically consist of one or more rows of male or female pins that are often spaced 0.1 inches (2.54 mm) apart, although other spacings are also available. The pins are typically housed in a plastic insulator.
- **Types:** There are several types of PCB headers, including straight (vertical) headers, right-angle headers, shrouded headers, unshrouded headers, breakaway headers (which can be cut to the desired length), and others.

- **Number of Pins:** Headers can have any number of pins, ranging from just one to several dozen. The number of pins needed depends on the number of signals that need to be transmitted between the connected devices.
- **Installation:** Headers are typically soldered directly onto the PCB. Wires or other connectors can then be plugged into or onto the header as needed. Some headers are designed to be press-fit into the PCB without the need for solder.
- **Applications:** Headers are used in a wide variety of applications. They are commonly found in electronics equipment where there is a need for an easy method of connecting and disconnecting components, such as connecting a motherboard to peripheral devices, or for programming and debugging purposes.

Selecting the right header for a given application depends on several factors, including the number of connections needed, the required reliability, the operating environment, and cost considerations. As with all electrical connections, headers must be correctly installed and maintained to ensure reliable operation.

To create a Header part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Header Parameters

Pads

Package Color

Package Reference

Package Value

Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

A pin header (or simply header) is a form of electrical connector, often associated with ribbon cable connectors. It consists of one or more rows of pins typically spaced 0.1 inches (2.54 mm) apart, but sometimes 2 millimetres (0.079 in) or 0.05 inches (1.27 mm) is used as well.

In addition to being used to connect to a ribbon cable connector, pin headers often also function as recipients for jumpers. The most common jumper spacing is 0.1 inches (2.54 mm) spacing, though 2 millimetres (0.079 in) is sometimes used in smaller products.

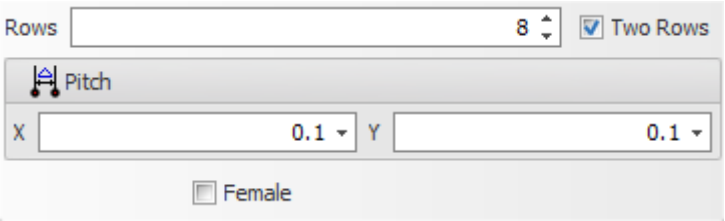
Pin header connectors are thus "male" connectors (female counterparts do exist, but these are normally just called "header connectors", without "pin") and are mostly used inside equipment, rather than being used as a connector on the outside of the device.

Normally pin headers are pin through hole (PTH) devices, but surface-mount technology (SMT) versions of one and two row pin headers also exist. In the latter case the solder sides of the pins are simply bent on a 90 degree angle so as to be soldered to a solder plane. On single row pin headers the pins are bent alternating to one side or the other, on dual row pin headers the pins are simply bent outwards. If pin headers are optional, the PTH variant is often chosen for ease of manual assembly. Pin headers can be either straight or angled. The latter form is often used to connect two boards together.

Pin headers are cost-effective due to their simplicity. Headers are often sold as long strips (typically 40 pins for the dual row versions) which can easily be broken off to the right number of pins.

Pin headers with a plastic guide box around them are known as "box headers" or "shrouded headers" and are normally only used in combination with a ribbon cable connector. A notch (key) in the guide box normally prevents placing the connector (polarised by a "bump" on one side) the wrong way around.

In absence of a pin 1 designation on the header, one or more pins in the header may be removed or clipped to indicate a key for correct orientation. If a designation is missing from the header, the PCB may have a marking indicating orientation (often the solder pad around the hole of pin 1 of a PTH header is square rather than round).

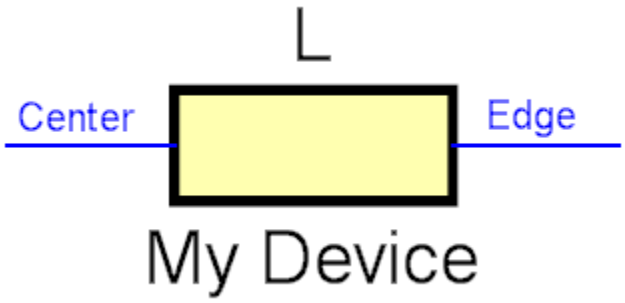


1.2.5.9.13.14 Inductor

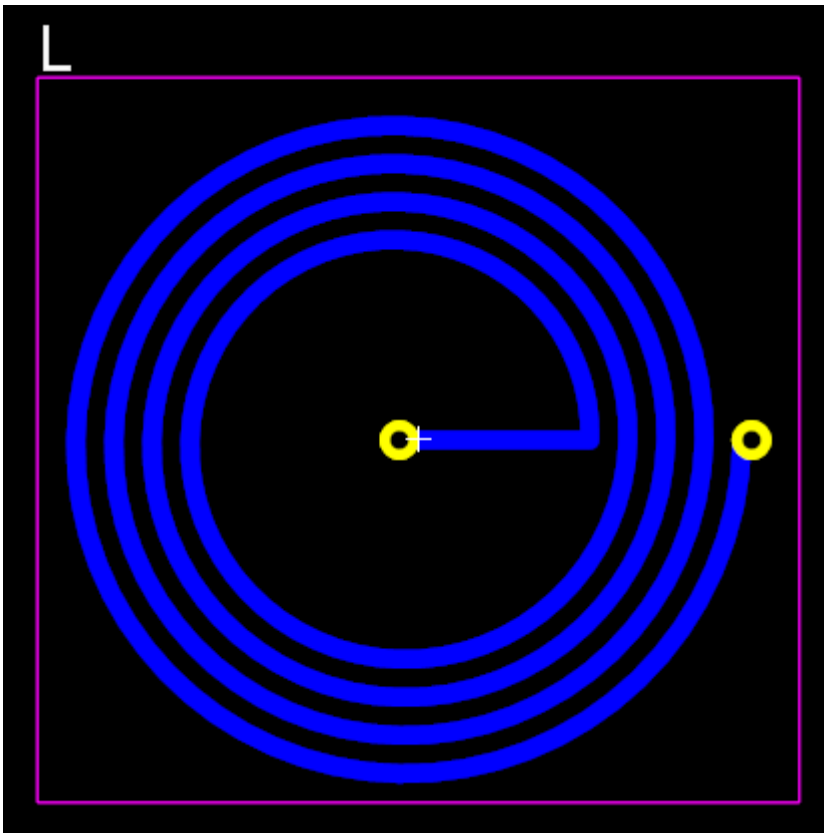
You can create a coil pattern on a copper layer to act as an inductor.

To create a Inductor part click the  button in the **The Part Builder**

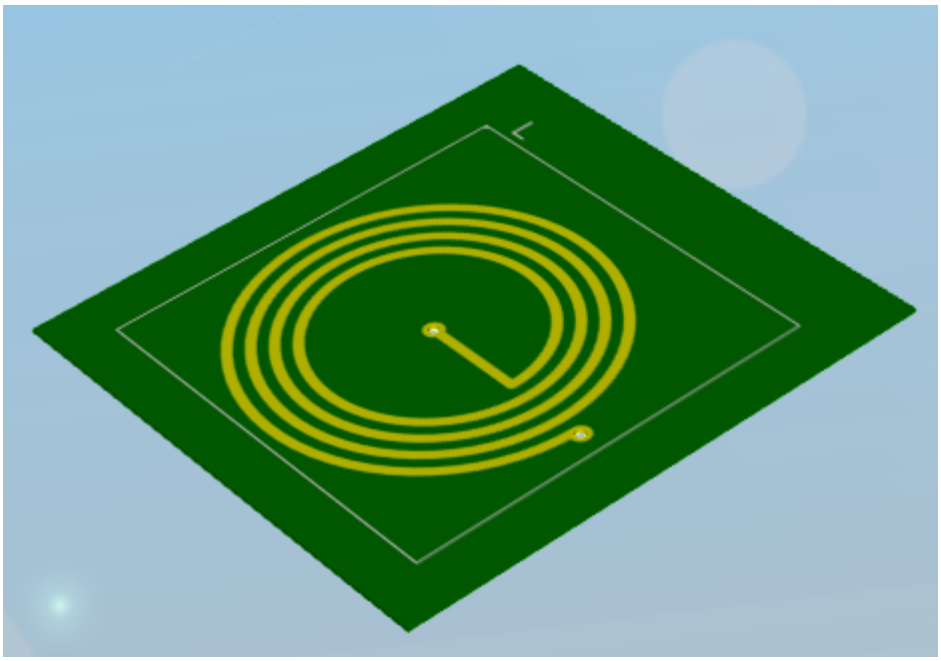
The part builder will automatically create the copper pattern and pads for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.



Schematic Symbol

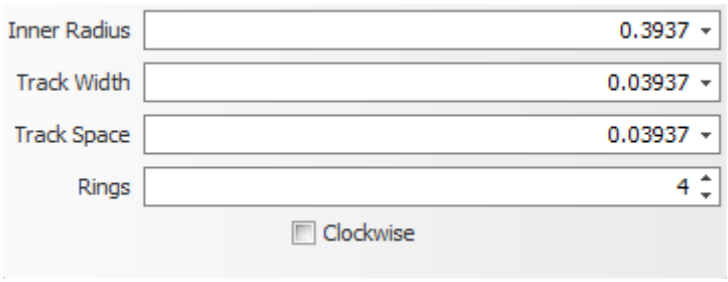


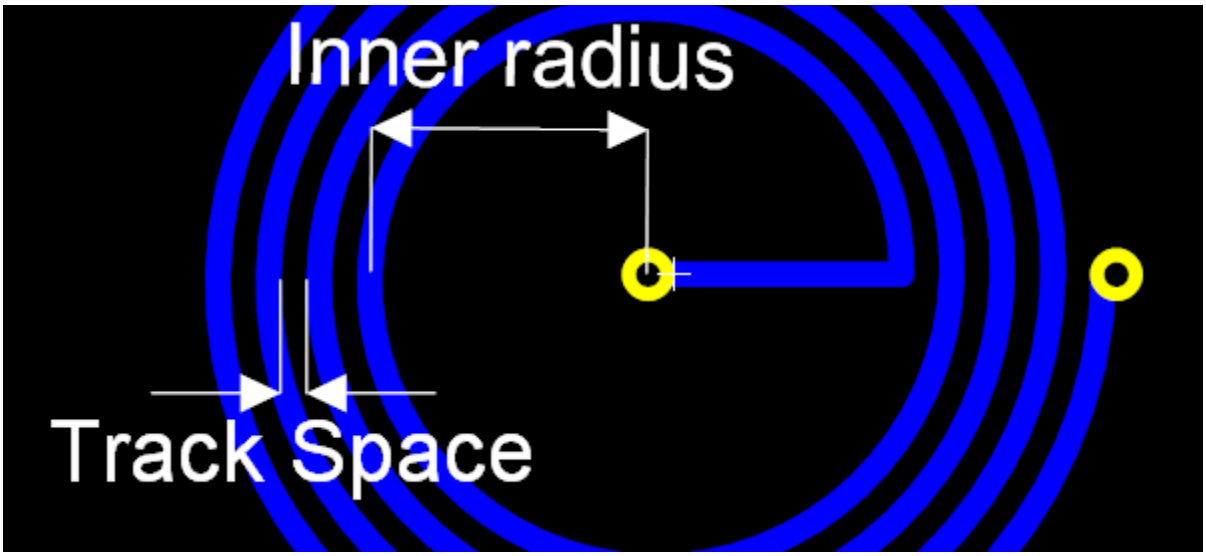
Copper pattern and pads



3D View of the inductor

The inductor parameters editor is shown below.





Inner Radius. The inner radius. See the diagram above.

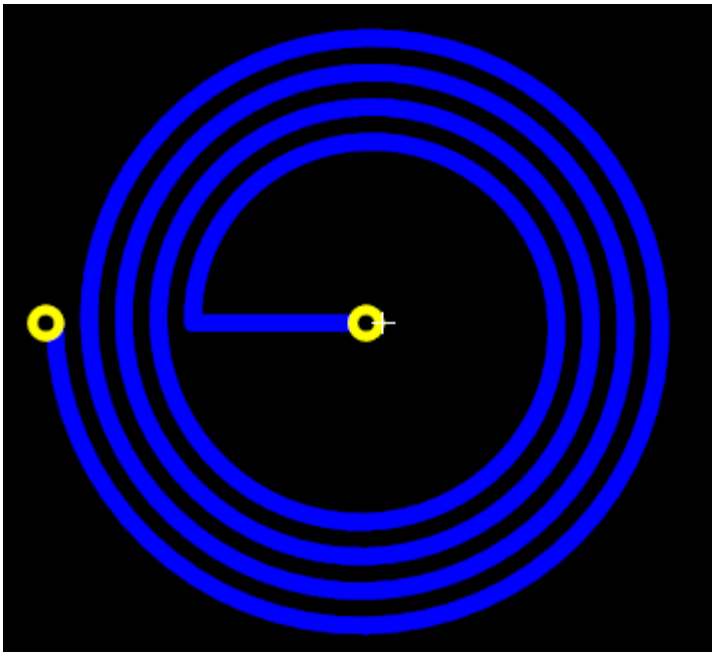
Track Width. The width of the copper track forming the antenna.

Trace Space. The space between copper rings. See the diagram above.

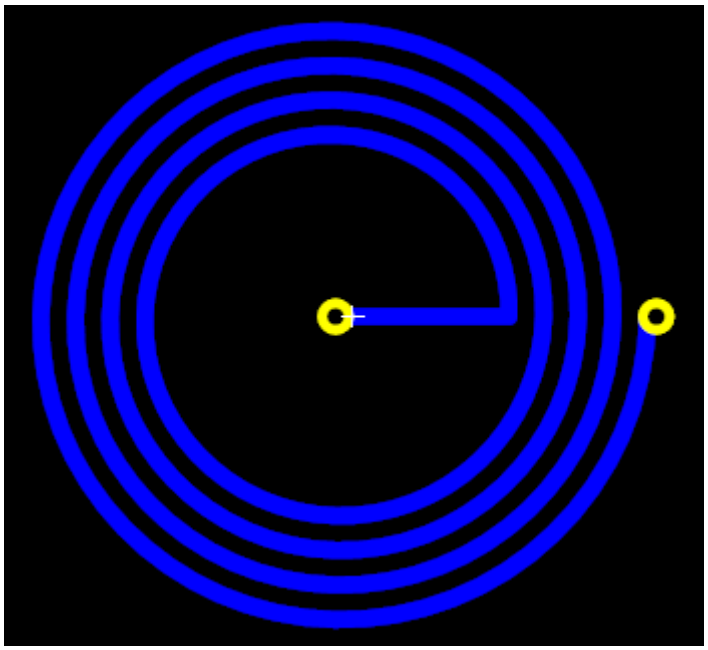
Rings. The number of complete rings.

Check the 

checkbox is run the rings clockwise. Uncheck to run the rings in the opposite direction.



Clockwise



Anticlockwise

You can create an inductor from copper traces on a PCB.

1.2.5.9.13.15 LED

Light Emitting Diodes (LEDs) are a popular choice for providing visual indicators on PCBs (Printed Circuit Boards) due to their efficiency, longevity, and the variety of colors they can produce.

LEDs

There are several different types of LEDs that can be used on PCBs, including:

- **Through-Hole LEDs:** These are the traditional, commonly seen LEDs with two leads that can be inserted into holes drilled into the PCB, and then soldered into place on the other side. They are available in a variety of sizes and colors, and can be used for a wide range of applications.
- **Surface Mount Device (SMD) LEDs:** These LEDs are designed to be soldered directly onto the surface of the PCB. They are much smaller than traditional through-hole LEDs, which makes them ideal for applications where space is at a premium. They also allow for automated assembly, which can lower manufacturing costs for large production runs.
- **Chip-On-Board (COB) LEDs:** This is a method of LED packaging which involves directly mounting the LED semiconductor die onto a PCB, and then wire bonding the die to the electrical contacts. A glob of epoxy or plastic is usually used to cover the die. COB LEDs can be very small and provide high light output.
- **Integrated LEDs:** Some components, such as switches or connectors, may have LEDs integrated directly into them. This can provide a convenient way to add visual indicators without needing to add separate LED components to the PCB.

Each type of LED has its own advantages and considerations, including size, power consumption, brightness, color, and cost. The best choice depends on the specific requirements of your project. As always, refer to the manufacturer's datasheet for detailed information about any specific LED component.

1.2.5.9.13.16 Metal Can

Metal can IC packages, also known as TO (Transistor Outline) can packages, were among the first types of packaging used for early integrated circuits and continue to be used for certain applications today. They are known for their characteristic metal "can" appearance.

The metal can, usually made from aluminum or steel, provides both physical protection and electromagnetic shielding for the integrated circuit inside. The metal leads protrude from the base of the can and are used to connect the IC to the rest of the circuit.

Metal Can Package

Examples of metal can packages include:

- **TO-3:** This is a large, high-power package often used for power transistors and voltage regulators. It typically has two leads and a metal tab for mounting and heat dissipation.
- **TO-5:** This package is smaller than the TO-3 and is commonly used for low-power transistors and integrated circuits. It typically has between 3 and 8 leads.
- **TO-8:** Similar to the TO-5, but typically has a larger number of leads, making it suitable for more complex ICs.
- **TO-18:** This is a smaller package used for low-power transistors and diodes. It typically has three leads.
- **TO-39:** This package is similar in size to the TO-5 but typically has a higher lead count.

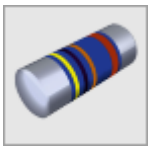
Metal can packages are rugged and offer good heat dissipation, but they are generally more expensive and larger than modern plastic packages, limiting their use in compact, cost-sensitive applications. They are often used in military and aerospace applications due to their reliability and robustness.

As always, the specific properties of an IC, such as its pinout and electrical characteristics, will depend on the specific IC and its manufacturer. It's important to refer to the datasheet for detailed information.

Metal Can

The parameters editor is shown below. As you can see there are currently no editable parameters.

1.2.5.9.13.17 MELF



Metal electrode leadless face (MELF) is a type of leadless cylindrical electronic surface mount device that is metallized at its ends. MELF devices are usually diodes and resistors.

Because of their cylindrical shape and small size, in some cases these components can easily roll off the workbench or circuit board before they have been soldered into place. As such, there is a joke which suggests an alternate meaning for the acronym: **M**ost **E**nd **U**p **L**ying on the Floor. Additionally, MELF components are sometimes called a "roll away" package.

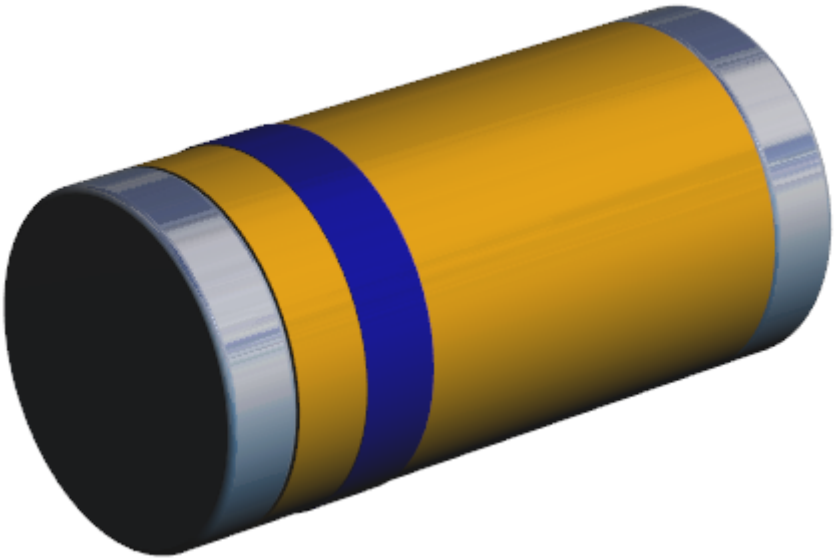
During automated SMT pick-and-place, this happens mostly if the mechanical pressure of the SMD placer nozzle is too low. If the MELF components are placed into the solder paste with enough pressure, then this problem can be minimized. Care must be taken with glass diodes which are less mechanically robust than resistors and other MELF components.

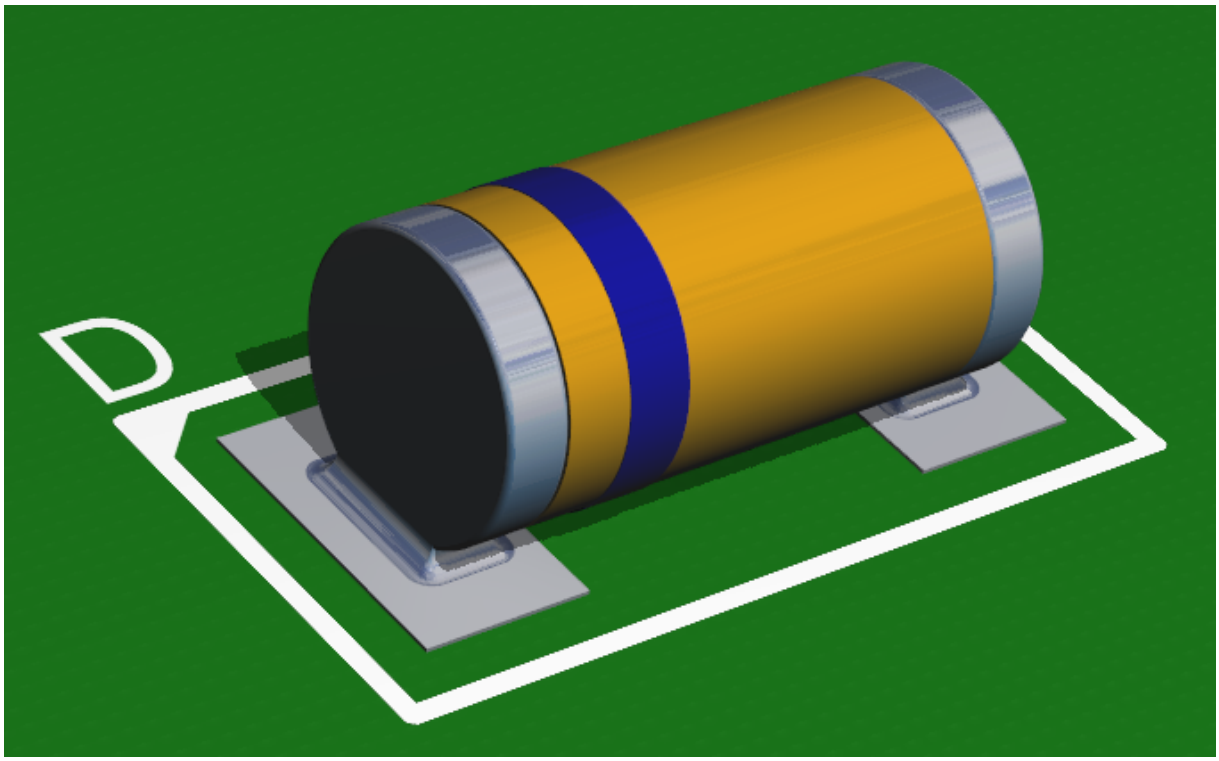
Also, when building up PCBs via manual assembly using tweezers (e.g., for prototyping) then the pressure at the end of tweezers can often cause a MELF component to slip and shoot out the ends, thereby making their placement more difficult, compared to other flat component packages.

Another reason for the nickname of MELF components is that most production engineers do not like to use MELF nozzles on a SMT pick-and-place machine. For them it is waste of time to change from flat nozzles to MELF nozzles. For MICRO-MELF and MINI-MELF most SMD placers are able to use flat chip nozzles if the vacuum is high enough; i.e., higher than for flat chip components. For MELFs with the case size of 0207 or less, it is recommended to use the original MELF nozzle supplied with the SMT machine. Each supplier of such SMD pick-and-place machines offers these types of nozzles.

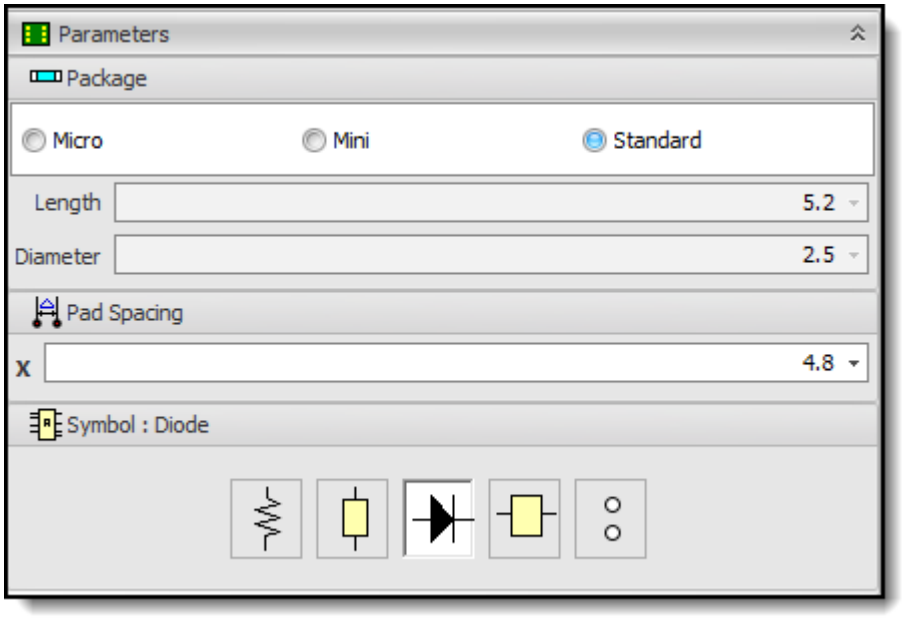
In order to overcome some of the difficulties encountered when mounting these devices, there are also variants with square electrodes (e.g. SQ MELF, QuadroMELF and B-MELF). These variants are mainly used in semiconductor diodes for applications where the high-reliability of hermetically sealed voidless-glass packages is required.

These handling difficulties prompted development of alternative SMT packages for common MELF components (like diodes) where the power handling capability needed to be similar to MELF components (superior to low-power 0805/0603, etc. SMT components) but with improved automated pick-and-place handling characteristics. This resulted in various squared-off packages with fold-over contacts, similar to rectangular inductor/tantalum capacitor packages





Standard MELF on PCB with Solder

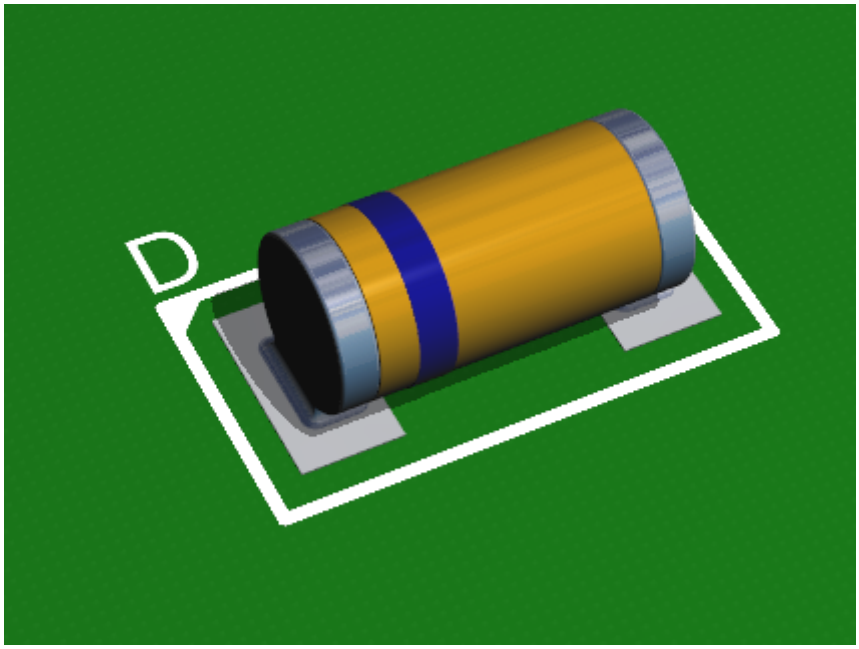


MELF Parameter Editor

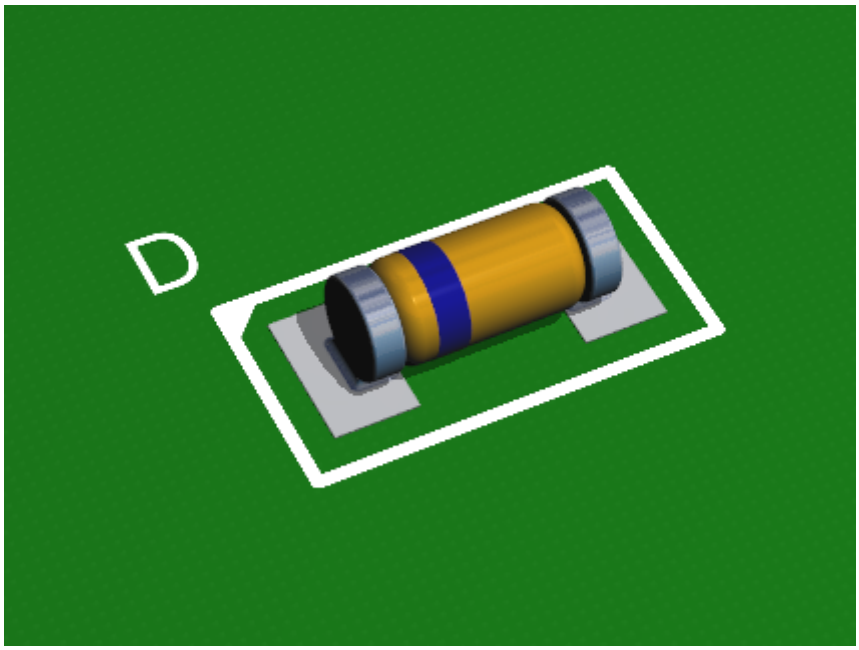
Type

Micro Mini Standard

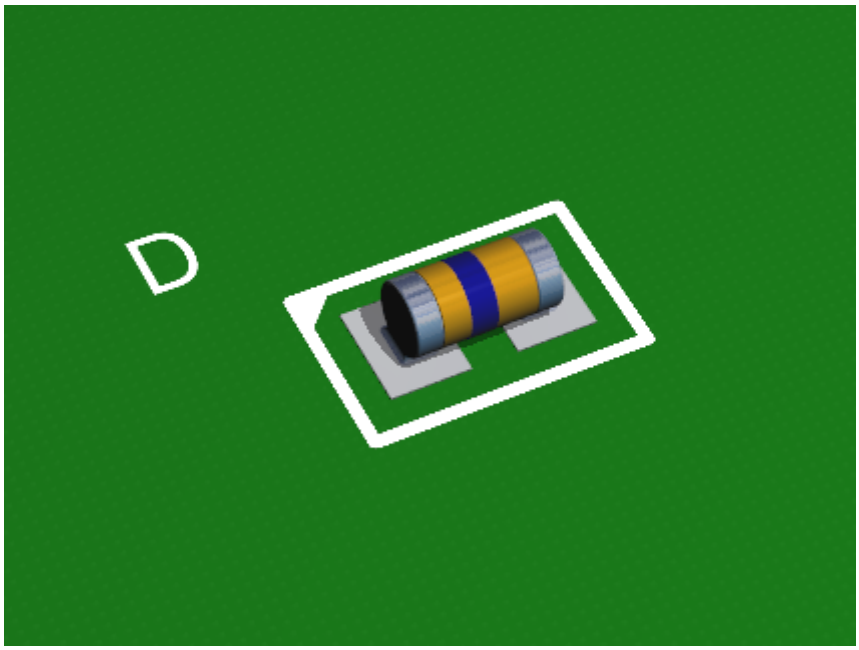
Selects one of 3 different package types.



Standard



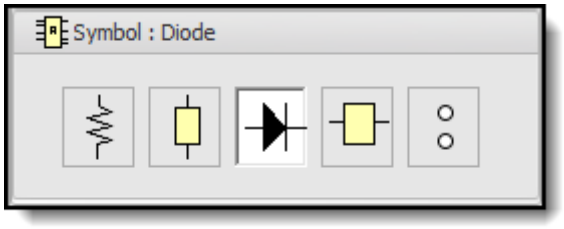
Mini



Micro

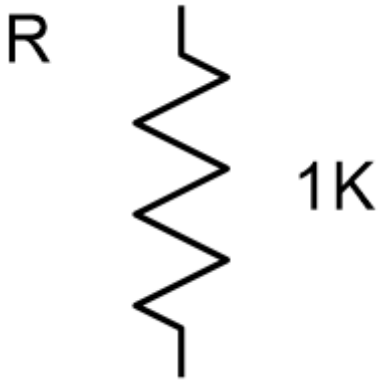
Schematic Symbol

A schematic symbol will be automatically created for you by clicking on one of the following buttons.

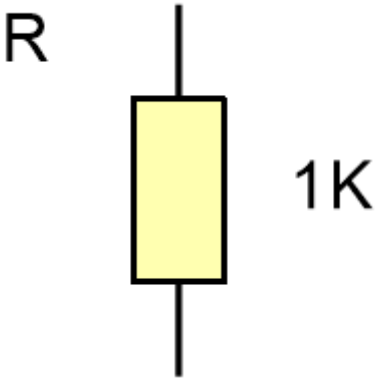


Symbol Selector

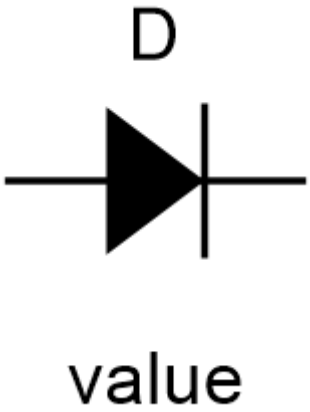
IEEE Resistor



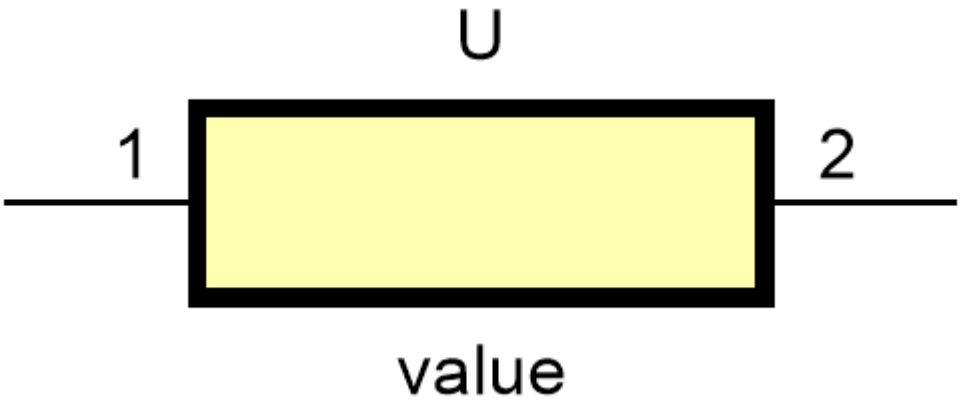
IEC Resistor



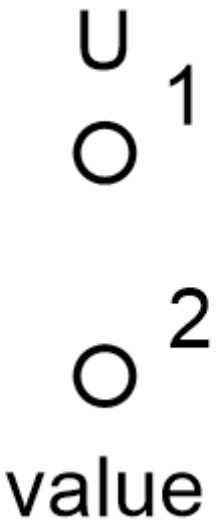
Diode



Terminal Magnet with 2 Terminals



Custom with 2 Terminals



1.2.5.9.13.18 QUAD

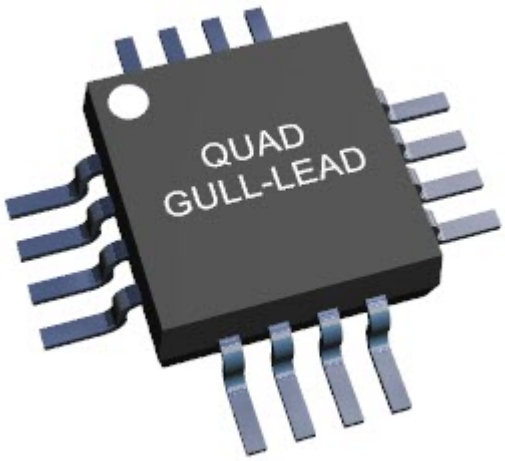
"Quad" in IC packages typically refers to a configuration where the IC (Integrated Circuit) has pins on all four sides. Some examples of quad packages include:

- **Quad Flat Packages (QFP):** These are flat, square or rectangular surface-mount packages with pins on all four sides. They can have a varying number of pins depending on the specific package, ranging from as few as 32 to over 200. QFPs

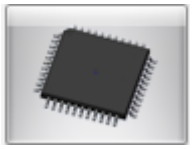
are commonly used for microprocessors, microcontrollers, and other integrated circuits.

- **Quad Flat No-Lead (QFN):** This package is similar to the QFP but the leads do not extend from the package, making it even more compact. The electrical connection is made through conductive pads on the bottom of the IC. This package is often used for ICs that require good thermal performance in a small size.
- **Quad In-Line Package (QIP):** This is a type of through-hole mount IC package with two rows of pins on each side, for a total of four rows. This is less common than the dual in-line package (DIP) which has two rows of pins on opposite sides.
- **Quad J-Lead (SOJ):** This is a type of surface-mount package that uses J-shaped leads on four sides of the component.
- **Plastic Leaded Chip Carrier (PLCC):** This is a square surface-mount package with rounded corners, and leads on all four sides.

These packages provide a high pin count in a small area, making them useful for complex ICs. As always, the specific properties of an IC, such as its pinout and electrical characteristics, will depend on the specific IC and its manufacturer. It's important to refer to the datasheet for detailed information.



Quad IC Package

To create a QUAD part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

QUAD Parameters

Pads

Package Color

Package Reference

Package Value

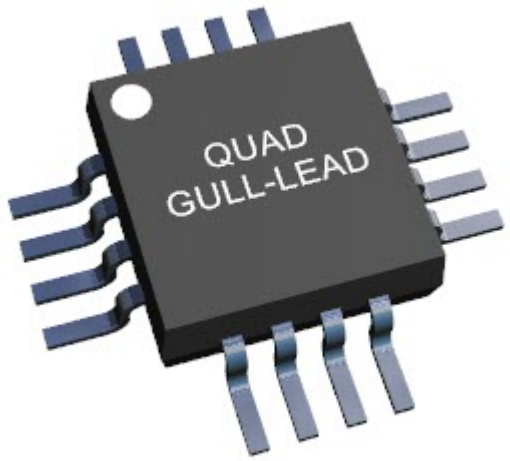
Package Silkscreen

Package Courtyard

Package Placement Point

Package Symbols

A QFP or Quad Flat Package is a surface mount integrated circuit package with "gull wing" leads extending from each of the four sides. Socketing such packages is rare and through-hole mounting is not possible. Versions ranging from 32 to 304 pins with a pitch ranging from 0.4 to 1.0 mm are common. Other special variants include low profile QFP and thin QFP.



The QFP component package type became common in Europe and United States during the early nineties, even though it has been used in Japanese consumer electronics since the seventies. It is often mixed with hole mounted, and sometimes socketed, components on the same printed circuit board.

A package related to QFP is PLCC which is similar but has pins with larger pitch, 1.27 mm (or 1/20 inch), curved up underneath a thicker body to simplify socketing (soldering is also possible). It is commonly used for NOR Flash memories and other programmable components.

The quad flat-pack has connections only around the periphery of the package. To increase the number of pins, the spacing was decreased from 50 mil (thousandths of an inch) (as found on small outline packages) to 20 and later 12 mil. However, this close lead spacing made solder bridges more likely and put higher demands on the soldering process and alignment of parts during assembly. The later pin grid array and ball grid array packages, by allowing connections to be made over the area of the package and not just around the edges, allowed for higher pin counts with similar package sizes, and reduced the problems with close lead spacing.

The basic form is a flat rectangular (often square) body with leads at four sides but with numerous variations in the design. These differ usually only in lead number,

pitch, dimensions, and materials used (usually to improve thermal characteristics). A clear variation is Bumpered Quad Flat Package with extensions at the four corners to protect the leads against mechanical damage before the unit is soldered.

Heat sink Quad Flat Package, Heatsink Very-thin Quad Flat-pack No-leads (HVQFN) is a package with no component leads extending from the IC. Pads are spaced along the sides of the IC with an exposed DIE that can be used as ground. Spacing between pins can vary.

A thin quad flat pack (TQFP) provides the same benefits of the metric QFP, but is thinner. Regular QFP are 2.0 to 3.8 mm thick depending on size. TQFP packages range from 32 pins with a 0.8 mm lead pitch, in a package 5 mm by 5 mm by 1 mm thick, to 256 pins, 28 mm square, 1.4 mm thick and a lead pitch of 0.4 mm.

TQFPs help solve issues such as increasing board density, die shrink programs, thin end-product profile and portability. Lead counts range from 32 to 176. Body sizes range from 5 mm x 5 mm to 20 x 20 mm. Copper lead-frames are used in TQFPs. Lead pitches available for TQFPs are 0.4 mm, 0.5 mm, 0.65 mm, 0.8 mm, and 1.0mm. PQFP, or plastic quad flat pack, is a type of QFP, as is the thinner TQFP package. PQFP packages can vary in thickness from 2.0 mm to 3.8 mm. A Low-profile Quad Flat Package is a surface mount integrated circuit package format with component leads extending from each of the four sides. Pins are numbered counter-clockwise from the index dot. Spacing between pins can vary; common spacings are 0.4, 0.5, 0.65 and 0.80 mm intervals.

Some QFP packages have an Exposed Pad. The exposed pad is an extra pad underneath or on top of the QFP that may act as a ground connection and/or as a heat sink for the package. The pad is typically 10 or more mm², and with the pad soldered down onto the ground plane heat is passed into the PCB. This exposed pad also gives a solid ground connection. These type of QFP packages often have a -EP suffix (e.g. a LQFP-EP 64), or they have an odd number of leads, (e.g. a TQFP-101).

Other quad flat package types

BQFP Bumpered Quad Flat Package

BQFPH Bumpered Quad Flat Package with heat spreader

CQFP Ceramic Quad Flat Package

EQFP Plastic Enhanced Quad Flat Package

FQFP Fine Pitch Quad Flat Package

LQFP Low Profile Quad Flat Package

MQFP Metric Quad Flat Package

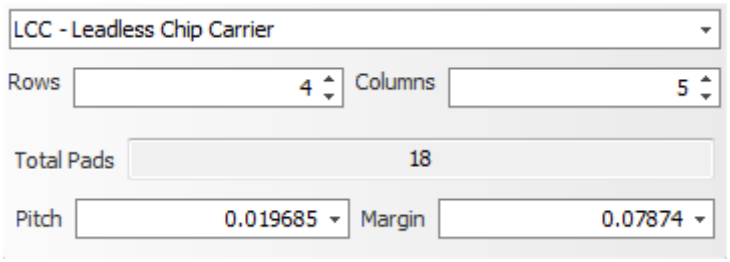
NQFP Near chip-scale Quad Flat Package.

SQFP Small Quad Flat Package

TQFP Thin Quad Flat Package

VQFP Very small Quad Flat Package

VTQFP Very Thin Quad Flat Package



1.2.5.9.13.19 Radial

"Radial" parts in the context of electronics typically refer to through-hole components that have leads protruding from the bottom or one side of the component, generally aligned along a single axis. These components are often mounted vertically on the PCB (Printed Circuit Board).

The term "radial" comes from the way the leads radiate out from the body of the component.

Common examples of radial parts include:

- **Radial Electrolytic Capacitors:** These are often used in power supply circuits due to their high capacitance and voltage ratings. They have a cylindrical body with two leads coming out from one end.
- **Radial Inductors:** These are commonly used in filter circuits and power electronics. Like capacitors, they have a cylindrical body with two leads coming out from one end.
- **Radial Leaded Ceramic Disc Capacitors:** These have a disc-shaped body with two leads coming out from one end.

One of the benefits of radial parts is that they can help save space on the PCB because they can be mounted vertically. However, they do increase the overall height of the board, so design considerations need to be made if space is a concern.

As always, for specific information such as lead spacing, physical dimensions, or electrical characteristics, refer to the manufacturer's datasheet for the specific component in question.

Electrolytic

Panasonic

Radial

To create a Radial part click the  button in the **The Part Builder**

The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.

Device Overview

Radial Parameters

Pads

Package Color

Package Reference

Package Value

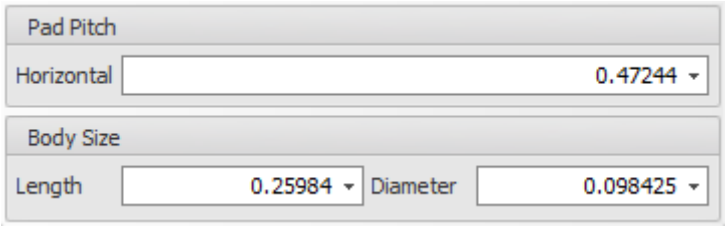
Package Silkscreen

Package Courtyard

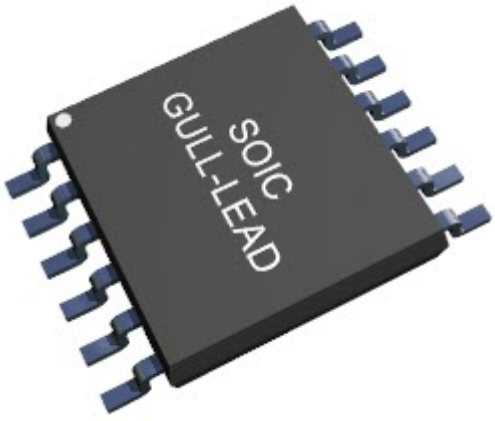
Package Placement Point

Package Symbols

Coming soon...



1.2.5.9.13.20 SOIC

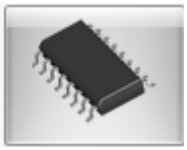


SOIC (Small Outline Integrated Circuit) is a type of surface-mount technology (SMT) packaging used for integrated circuits (ICs). SOIC ICs are characterized by their small size and low profile, which make them suitable for use in compact electronic devices.

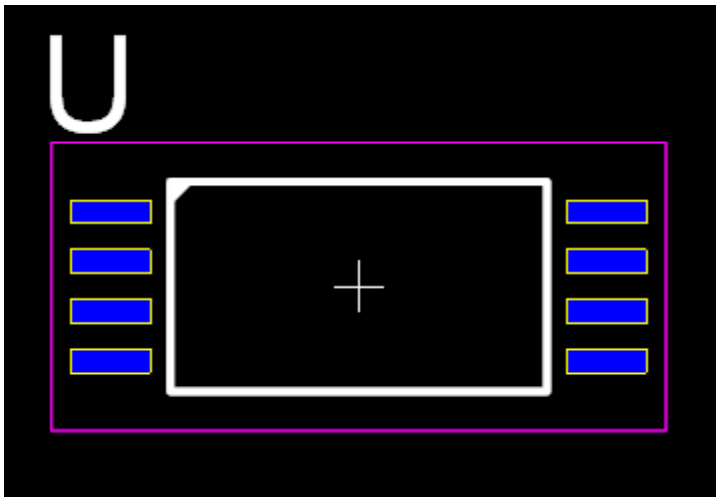
SOIC ICs come in a range of sizes, with different numbers of pins, from 8 to 32. They can contain a variety of components, including transistors, resistors, and capacitors, and are often used in digital circuits, such as microcontrollers and memory devices.

SOIC ICs are mounted directly onto a printed circuit board (PCB) using solder, without the need for a socket or through-hole mounting. This allows for higher density and greater functionality in smaller spaces, making SOIC ICs a popular choice for modern electronic devices.

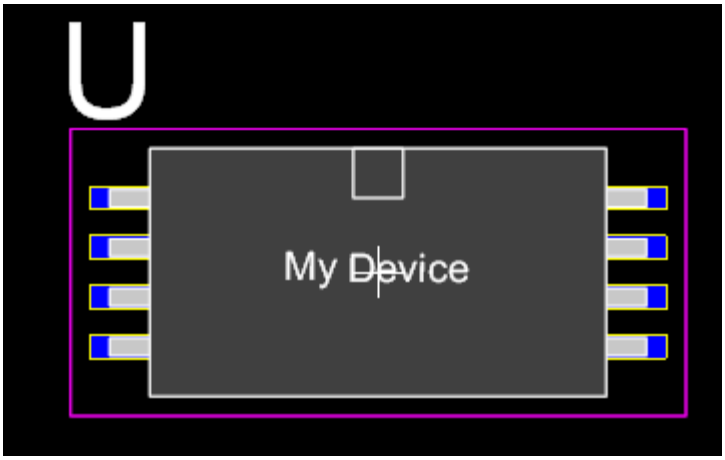
There are several variations of SOIC ICs, including narrow-body, wide-body, and shrink small-outline package (SSOP) versions, each with their own specific characteristics and advantages.

To create a SOIC part click the  button in the **The Part Builder**

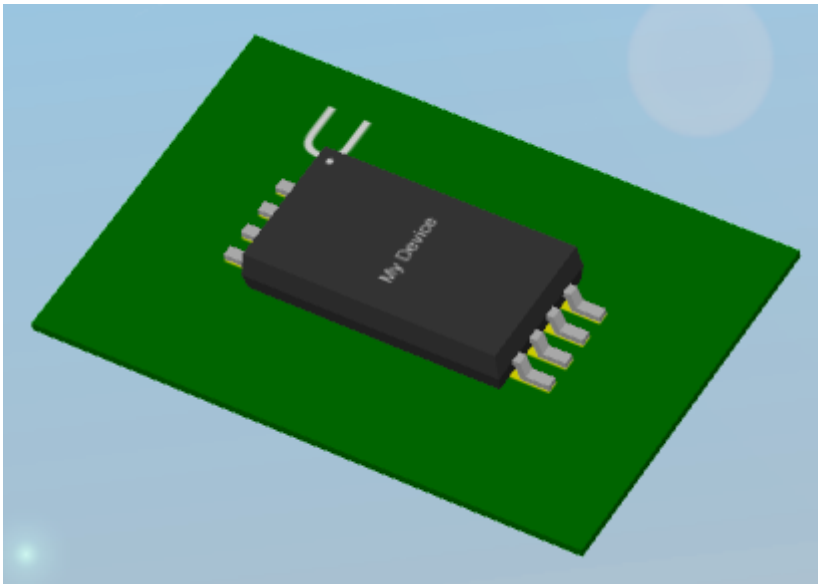
The part builder will automatically create the footprint for the device and also a suitable schematic symbol with the pin names left blank for you to fill in.



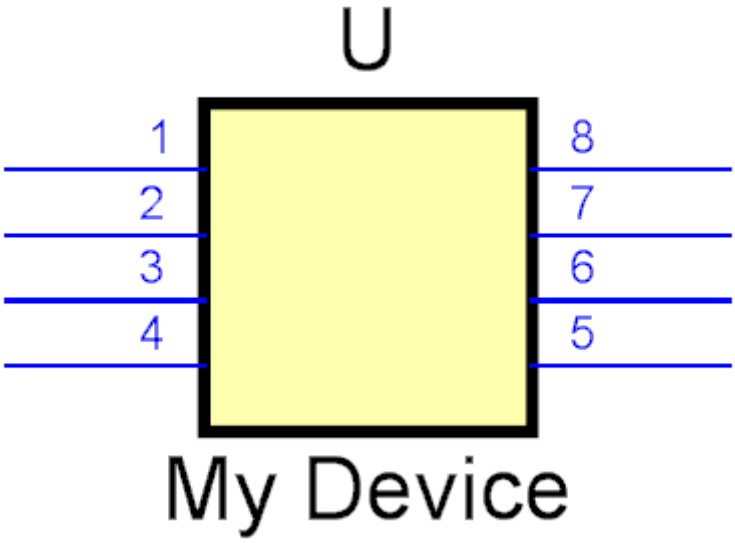
8 pin SOIC



8 pin SOIC showing package



3D View of 8 pin SOIC



Schematic Symbol

A small-outline integrated circuit (SOIC) is a surface-mounted integrated circuit (IC) package which occupies an area about 30–50% less than an equivalent DIP, with a typical thickness that is 70% less. They are generally available in the same pinouts as their counterpart DIP ICs. The convention for naming the package is SOIC or sometimes just SO followed by the number of pins. For example, a 14-pin 4011 would be housed in an SOIC-14 or SO-14 package.

Small-outline J-leaded package (SOJ) is a version of SOIC with J-type leads instead of gull-wing leads.

JEDEC and EIAJ standards

SOIC actually refers to at least two different package standards. The EIAJ SOIC body is approximately 5.3 mm (0.21 in) wide while the JEDEC SOIC body is approximately 3.8 mm (0.15 in) wide. The EIAJ packages are also thicker and slightly longer. Otherwise the packages are similar.

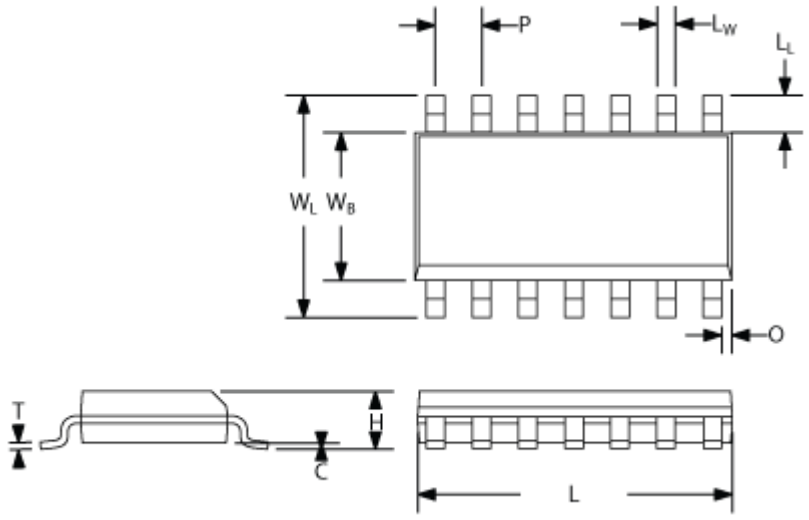
Note that because of this, SOIC is not specific enough of a term to describe parts which are interchangeable. Many electronic retailers will list parts in either package as SOIC whether they are referring to the JEDEC or EIAJ standards. The wider EIAJ packages are more common with higher pin count ICs, but there is no guarantee that an SOIC package with any number of pins will be either one or the other.

General package characteristics

This package is shorter and narrower than DIPs, the side-to-side pitch being 6 mm for an SOIC-14 (from lead tip to lead tip) and the body width being 3.9 mm. These dimensions differ depending on the SOIC in question, and there are several variants. This package has "gull wing" leads protruding from the two long sides and a lead spacing of 0.050 in (1.27 mm).

Narrow (JEDEC) SOIC package

The picture below shows the general shape of an SOIC narrow package, with major dimensions. The values of these dimensions (in mm) for common SOICs are shown in the table.



C Clearance between IC body and PCB

H Total carrier height

T Lead thickness

L Total carrier length

LW Lead width

LL Lead length

P Pitch

WB IC body width

WL Lead-to-lead width

O End overhang

Package	WB	WL	H	C	L	P	LL	T	LW	O
SOIC-8-N	3.8–4.0	5.8–6.2	1.35–1.75	0.10–0.25	4.8–5.0	1.27	0.41 (1.04)	0.19–0.25	0.35–0.51	0.33
SOIC-14-N	3.8–4.0	5.8–6.2	1.35–1.75	0.10–0.25	8.55–8.75	1.27	1.05	0.19–0.25	0.39–0.46	0.3–0.7
SOIC-16-N	3.8–4.0	5.8–6.2	1.35–1.75	0.10–0.25	9.8–10.0	1.27	1.05	0.19–0.25	0.39–0.46	0.3–0.7
SOIC-16-W	7.4–7.6	10.0–10.65	2.35–2.65	0.10–0.30	10.1–10.5	1.27	0.40–1.27	0.20–0.33	0.31–0.51	0.4–0.9

Wide (or extended) SOIC package

Next to the narrow SOIC package (commonly represented as SOx_N or SOICx_N, where x is the number of pins), there's also the wide (or sometimes called extended) version. This package is commonly represented as SOICX_W or SOICx_W.

The difference is mainly related to the parameters WB and WL.

As an example, the values WB and WL are given for an 8-pins wide (extended) SOIC package

Package	WB	WL
SOIC-8	5.41 (5.16)	8.07 (7.67)

Mini- or micro-SOIC package

Another SOIC variant, available only for 8-pins and 10-pins ICs, is the mini-SOIC, also called micro-SOIC. This case is much smaller with a pitch of only 0.5mm. See the following table for the 10-pin model:

Package	WB	WL	H	C	L	P	LL	T	LW
miniSOIC-10	3.0	4.9	1.09	0.10–0.25	3.0	0.5	0..095	0.19	0.23

SOP

After SOIC came a family of smaller form factors, small outline package (SOP), with a pin spacing of 0.635 mm:

- Plastic small-outline package (PSOP)
- Thin small-outline package (TSOP)
- Thin-shrink small-outline package (TSSOP)

Shrink small-outline package (SSOP) chips have "gull wing" leads protruding from the two long sides, and a lead spacing of 0.025 inches (0.635mm). 0.5mm lead spacing is less common, but not rare.

The body size of a SOP was compressed and the lead pitch tightened to obtain a smaller version SOP. This yields an IC package which is a significant reduction in the size (compared to standard package). All IC assembly processes remain the same as with standard SOPs.

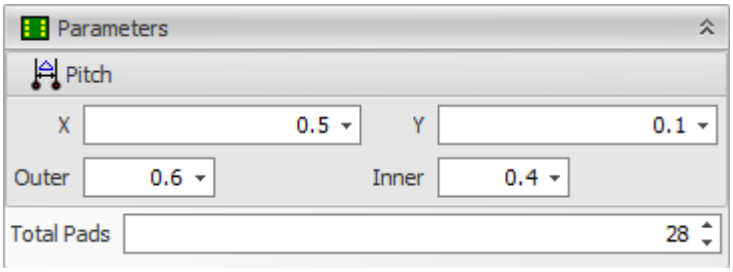
Applications for a SSOP enable end-products (pagers, portable audio/video, disc drives, radio, RF devices/components, telecom) to be reduced in size and weight. Semiconductor families such as operational amplifiers, drivers, optoelectronics, controllers, logic, analog, memory, comparators and more using BiCMOS, CMOS or other silicon / GaAs technologies are well addressed by the SSOP product family.

TSSOP

TSSOPs are particularly suited for gate drivers, controllers, wireless / RF, op-amps, logic, analog, ASICs, memory (EPROM, E2PROM), comparators and optoelectronics. Memory modules, disk drives, recordable optical disks, telephone handsets, speed dialers, video / audio and consumer electronics / appliances are suggested uses for TSSOP packaging.

The ExposedPad variant of small-outline packages can increase heat dissipation by as much as 1.5 times over a standard TSSOP, thereby expanding the margin of operating parameters. Additionally, the ExposedPad can be connected to ground, thereby reducing loop inductance for high frequency applications. The ExposedPad should be soldered directly to the PCB to realize the thermal and electrical benefits.

The ICs on DRAM memory modules were usually TSOPs until they were replaced by ball grid array (BGA).



SOIC Parameter Editor

Pitch

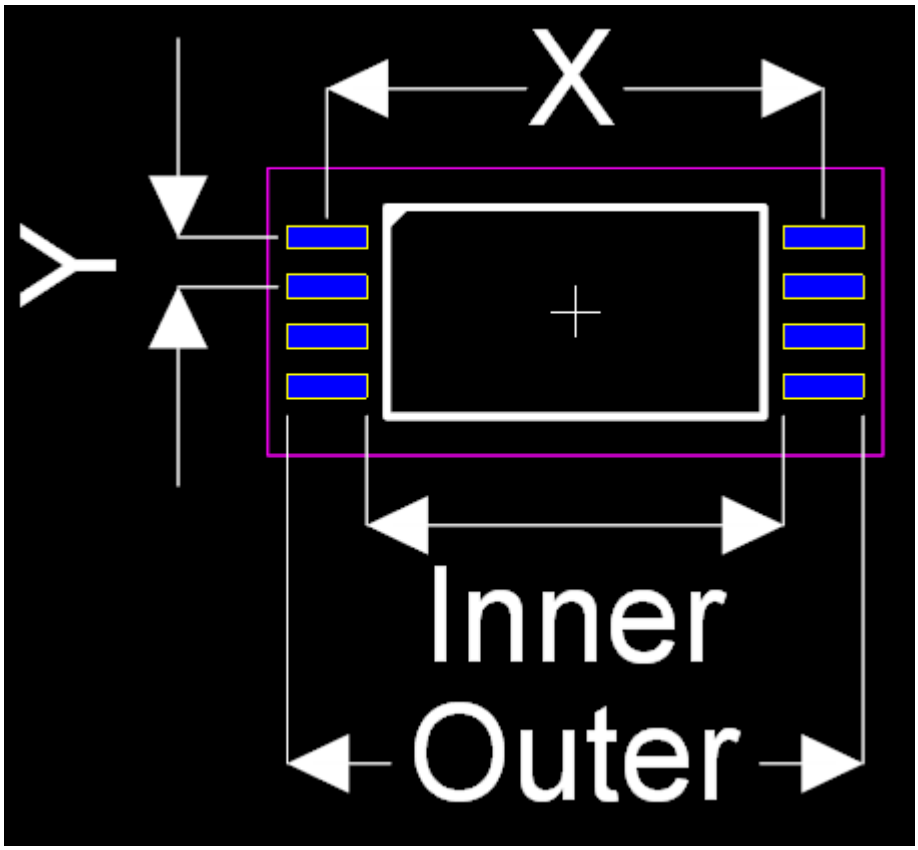
X the horizontal pitch between pad columns centers.

Y the vertical pitch between pad row centers.

Outer The horizontal distance between pad outer edges.

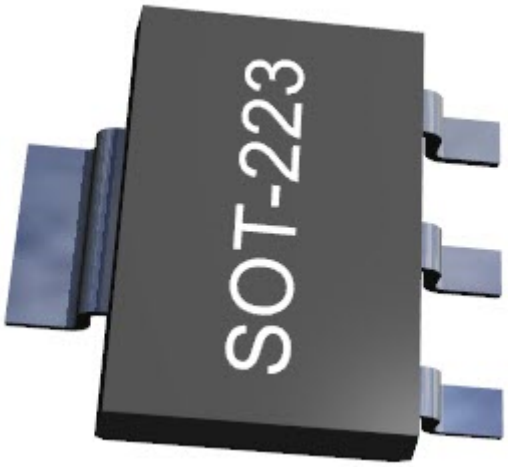
Inner The horizontal distance between pad inner edges.

Total Pads The total number of pads.



SOIC Parameters

1.2.5.9.13.21 SOT-223



SOT-223

SOT-223 is a type of package used for semiconductors, such as transistors and integrated circuits. The acronym "SOT" stands for Small Outline Transistor, and the number "223" refers to the specific form factor or physical dimensions of the package.

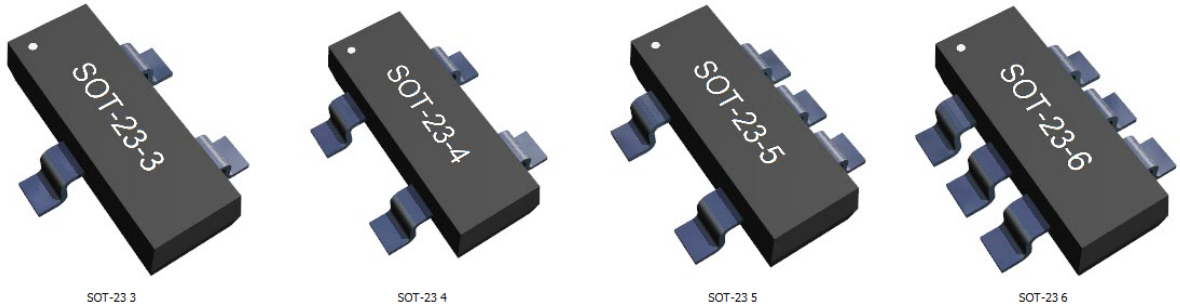
The package has three leads coming out from the bottom and an additional heat sink tab for better thermal dissipation. This is useful in applications where the device needs to dissipate a considerable amount of heat, such as power transistors.

SOT-223 packages are commonly used in surface-mount technology, which allows for high-density PCB (printed circuit board) assembly and automation-friendly manufacturing processes. Their design makes them compact and efficient, fitting well into modern electronic device design philosophies that prioritize space efficiency and power management.

For specific measurements or pin configuration of the SOT-223 package, you would need to refer to the datasheet of the particular component you are using, as these details can vary between manufacturers or even between different devices from the same manufacturer.

1.2.5.9.13.22 SOT 23

The SOT-23 is another popular semiconductor package type. It stands for Small Outline Transistor-23. It's typically used for low power, surface mount devices such as transistors and small-scale integrated circuits.



The SOT-23

The package is quite small, often measuring about 3mm x 1.7mm. It generally has three leads, making it suitable for devices such as single gate logic, op-amps, and low-power transistors.

Despite its small size, the SOT-23 package is designed to be easily handled and placed by automated assembly equipment, making it an excellent choice for high volume manufacturing processes.

As with the SOT-223 package, for specific pin configurations and other details of a particular SOT-23 packaged component, it's best to refer to the specific datasheet provided by the manufacturer.

A SOT is a small outline transistor.

An SOT

The parameters editor is shown below. As you can see there are currently no editable parameters.

1.2.5.9.13.23 SOD 323

SOD 323

The SOD-323 package is a type of surface-mount packaging for semiconductor devices, typically for small signal diodes. The abbreviation "SOD" stands for Small Outline Diode.

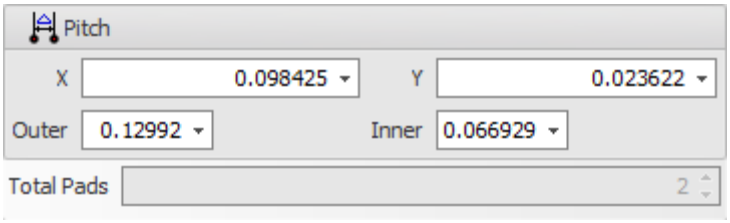
The SOD-323 package is characterized by its very small size. It typically measures approximately 2.5 mm long by 1.25 mm wide, though the exact dimensions may vary slightly between different manufacturers.

This package type typically has two leads or terminals, and is designed for surface mount technology, which allows for direct mounting and soldering onto printed circuit boards (PCBs). This makes it well suited for use in high-density mounting designs.

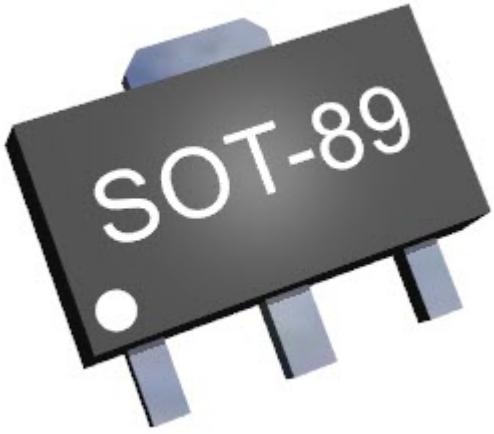
As with any specific component, for detailed information such as pin assignment or electrical characteristics, you should refer to the manufacturer's datasheet for the specific device in question.

A SOD a small outline diode.

An SOD device



1.2.5.9.13.24 SOT 89



SOT 89

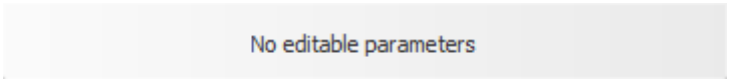
The SOT-89 is a type of semiconductor package often used for low power, surface mount devices such as transistors and small-scale integrated circuits. The abbreviation "SOT" stands for "Small Outline Transistor," and "89" refers to the specific package style.

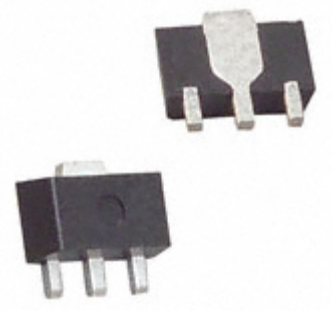
Compared to the SOT-23 and SOT-223 packages, the SOT-89 package is somewhat larger, allowing for better heat dissipation. It typically measures about 4.5 mm long by 2.5 mm wide, though exact dimensions can vary somewhat between manufacturers.

The SOT-89 package typically has three leads and is designed for surface mount technology, which allows for direct mounting and soldering onto printed circuit boards (PCBs).

As with any specific device, for detailed information such as pin assignment, electrical characteristics, or exact physical dimensions, you should refer to the manufacturer's datasheet for the specific device in question.

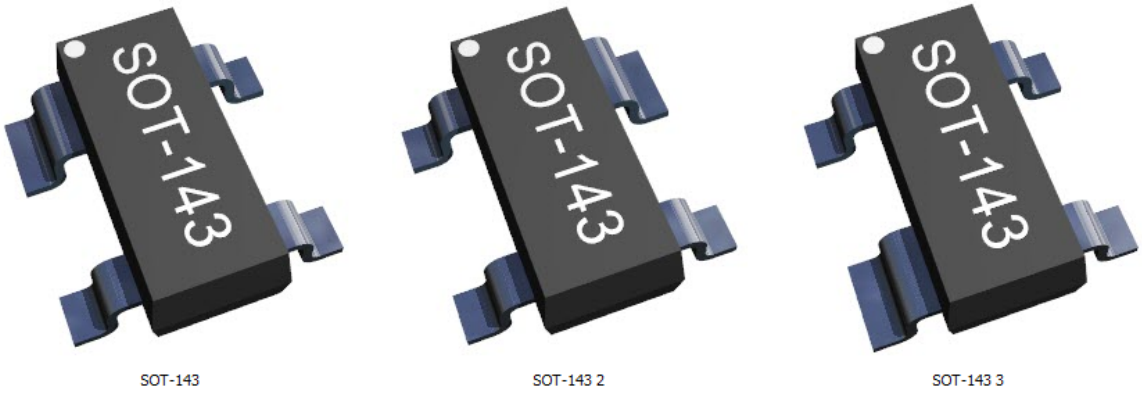
The parameters editor is shown below. As you can see there are currently no editable parameters.





1.2.5.9.13.25 SOT 143

The SOT-143 package is a type of semiconductor package used for small outline transistors and integrated circuits. The abbreviation "SOT" stands for "Small Outline Transistor," while "143" identifies the specific package style.



SOT 143

The SOT-143 package is designed for surface-mount technology, which allows for direct mounting and soldering onto printed circuit boards (PCBs). This makes it well suited for use in high-density mounting designs.

The SOT-143 package typically has four leads (or pins), although the configuration of these pins can vary depending on the specific device and manufacturer.

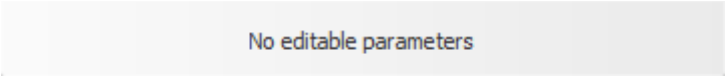
In terms of size, the SOT-143 package is relatively small, which makes it suitable for compact electronic devices. However, the exact dimensions can vary somewhat between manufacturers.

As always, for detailed information such as pin assignment, electrical characteristics, or exact physical dimensions, you should refer to the manufacturer's datasheet for the specific device in question.

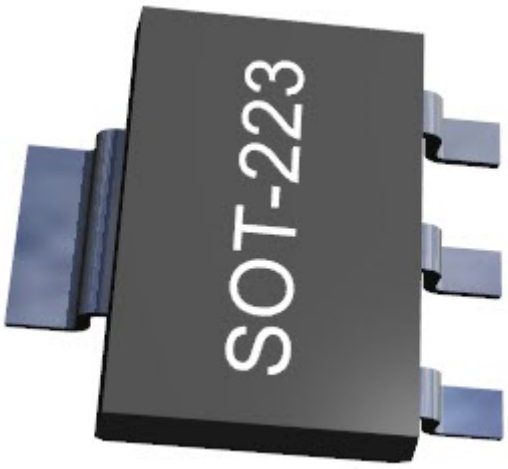
A SOT is a small outline transistor.

SOT

The parameters editor is shown below. As you can see there are currently no editable parameters.



1.2.5.9.13.26 SOT 223



SOT 223

The SOT-223 package is a type of semiconductor package that's commonly used for power transistors and voltage regulators. "SOT" stands for "Small Outline Transistor," and "223" refers to the specific style of the package.

The SOT-223 package typically has three leads (or pins) plus a large tab for heat sinking. This tab can either be electrically connected or not, depending on the design of the specific component.

One of the main benefits of the SOT-223 package is that it's capable of dissipating more heat than some other small outline transistor packages due to this tab. This makes it well-suited for components that generate a considerable amount of heat, such as power transistors or voltage regulators.

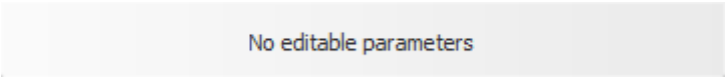
This package is designed for surface-mount technology, allowing for direct mounting and soldering onto printed circuit boards (PCBs).

For detailed information like pin assignment, electrical characteristics, or exact dimensions, you should refer to the manufacturer's datasheet for the specific device in question.

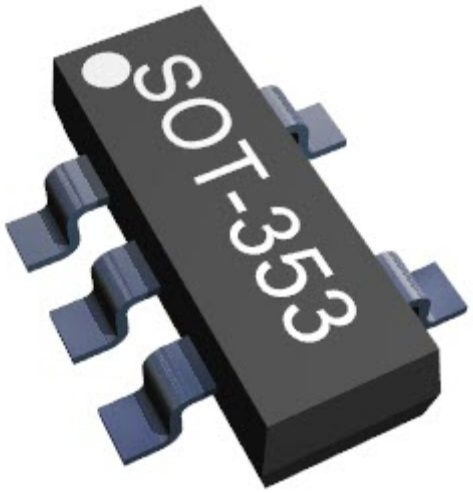
A SOT is a small outline transistor.

SOT

The parameters editor is shown below. As you can see there are currently no editable parameters.



1.2.5.9.13.27 SOT 353



SOT 353

The SOT-353 package, also known as SC-88A in some contexts, is a type of semiconductor package used for small outline transistors and integrated circuits. The acronym "SOT" stands for "Small Outline Transistor," and "353" refers to a specific package style.

The SOT-353 package is designed for surface-mount technology, which allows it to be directly mounted and soldered onto printed circuit boards (PCBs). It is used in applications where space efficiency is critical due to its small size.

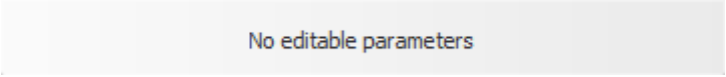
This package typically has five leads (or pins). Despite its small size, it is designed in a way that allows it to be easily handled by automated assembly equipment, making it suitable for high volume manufacturing processes.

As with any specific device, for detailed information such as pin assignment, electrical characteristics, or exact physical dimensions, you should refer to the manufacturer's datasheet for the specific device in question.

A SOT is a small outline transistor.

SOT

The parameters editor is shown below. As you can see there are currently no editable parameters.



1.2.5.9.13.28 TO 3 / TO 66

TO 3

The TO-3 package is a type of semiconductor package that is often used for power transistors and some integrated circuits. The acronym "TO" stands for "Transistor Outline," and "3" refers to the specific package style as per the JEDEC standard.

TO 3

Unlike the SOT packages (Small Outline Transistor) which are typically used for surface mount devices, the TO-3 is a through-hole package. This means that it is designed to be inserted into holes drilled in printed circuit boards (PCBs) and then soldered into place.

The TO-3 package has two leads and a metal tab which can serve as a third electrical connection. This metal tab is also used as a heat sink to dissipate heat away from the device, which makes TO-3 packages ideal for high-power transistors that generate a lot of heat.

One of the characteristic features of the TO-3 package is its large, metal, can-like shape. While this makes it less suitable for compact, high-density electronic devices, it provides excellent heat dissipation for high-power applications.

TO-66

TO-66 TO-66 Large Heatsink TO-66 Small Heatsink

TO-66

The TO-66 package is a type of semiconductor package, often used for high power transistors. The acronym "TO" stands for "Transistor Outline," and "66" refers to the specific package style.

The TO-66 is a through-hole package, which means it's designed to be inserted into drilled holes in printed circuit boards (PCBs) and then soldered into place.

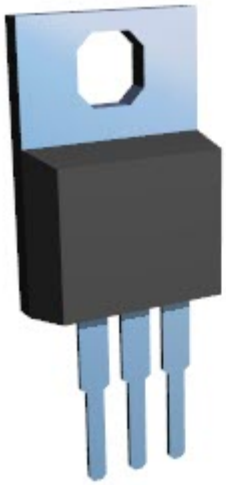
The TO-66 package typically has two leads and a metal tab. This metal tab, like the one found on the TO-3 package, is usually used for both electrical connection and as a heat sink to dissipate heat away from the device. This characteristic makes the TO-66 suitable for components that generate considerable heat, such as power transistors.

The TO-66 is smaller and more compact than the TO-3 package but can still handle high-power applications due to its design.

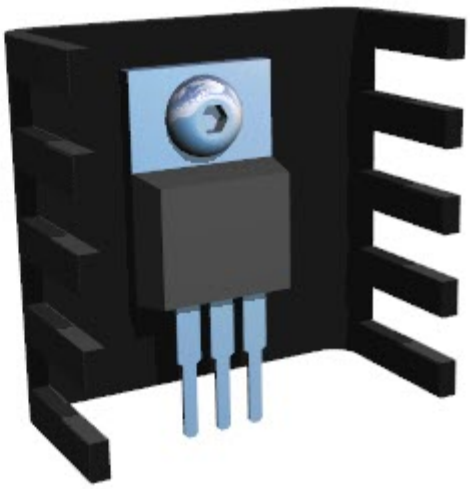
As always, for specific information such as pin assignment, electrical characteristics, or precise physical dimensions, refer to the manufacturer's datasheet for the specific device in question.

1.2.5.9.13.29 TO 220

The TO-220 is a type of semiconductor package that is commonly used for medium to high-power transistors, voltage regulators, and integrated circuits. The acronym "TO" stands for "Transistor Outline," and "220" refers to the specific package style as per the JEDEC standard.



TO-220



TO-220 with Heatsink

TO 220

The TO-220 package is a through-hole package, which means it's designed to be inserted into drilled holes in printed circuit boards (PCBs) and then soldered into place. It typically has three leads and a metal tab, which is often used as a heat sink to help dissipate heat away from the device.

This heat sink capability makes the TO-220 package suitable for components that generate considerable heat, such as power transistors and voltage regulators. It's widely used in power management applications because of its good heat dissipation properties.

As always, for specific information such as pin assignment, electrical characteristics, or precise physical dimensions, refer to the manufacturer's datasheet for the specific device in question.

1.2.5.9.13.30 Edge Connector Parts

Coming soon...

1.2.5.9.13.31 Wizard

Coming soon...

1.2.5.9.13.32 Custom Parts

Coming soon...

1.2.5.9.13.33 Castellated PCB Parts

A castellated PCB (Printed Circuit Board) is a type of PCB that has its edges modified with plated holes or half-holes. These half-holes appear like the

battlements, or "castellations," of a castle when viewed from the side, which is how the PCB gets its name.

Castellations are typically used to allow one PCB to be soldered to another. This is often used in the creation of modules, where a small PCB with a specific functionality is soldered onto a larger main PCB. These small castellated PCBs are usually called breakout boards or modules.

In terms of manufacturing, a castellated PCB is made by first creating a standard PCB with vias placed close to the edge, then milling the PCB edge, halving the vias. The half-vias are then plated to provide a good surface for soldering.

Common examples of this include Wi-Fi modules like the ESP8266 or ESP32, or sensor modules, that can be directly soldered onto another PCB, effectively making them a component of a larger system.

As with any PCB design, the specific layout, dimensions, and specifications of a castellated PCB will depend on its intended use and the requirements of the overall device it's a part of.

1.2.5.9.13.34 TO 92

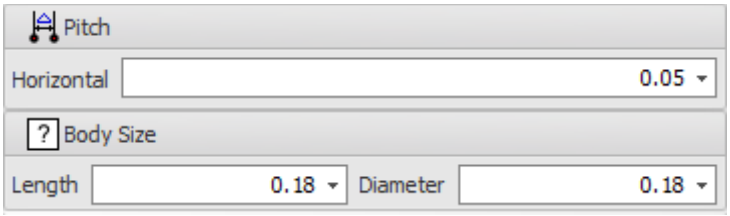
The TO-92 is a widely used style of semiconductor package mainly used for transistors. The "TO" stands for "Transistor Outline," and "92" refers to the JEDEC standard package number.

Unlike the larger TO-220 or TO-3 packages, which are typically used for power transistors and regulators, the TO-92 package is generally used for low-power devices like small signal transistors.

The TO-92 is a through-hole package, meaning it's designed to be inserted into drilled holes on a printed circuit board (PCB) and then soldered into place. This package typically has three leads, which often come out of the bottom of the package in a triangular arrangement or in-line, depending on the manufacturer and specific device.

The TO-92 package is small, lightweight, and inexpensive, making it suitable for a wide range of applications.

As always, for specific information such as pin assignment, electrical characteristics, or precise physical dimensions, refer to the manufacturer's datasheet for the specific device in question.



The TO-92 is a widely used style of semiconductor package mainly used for transistors. The case is often made of epoxy or plastic, and offers compact size at a very low cost. The JEDEC TO-92 descriptor is derived from the original full name for the package: Transistor Outline Package, Case Style 92.

Today, the industry prefers the new TO-226 nomenclature for the TO-92 three-leaded enclosure.

There are three TO-206 variants: TO-206AA with three straight leads; TO-206AB with the lead 2 formed to create a triangular pattern between the leads and the TO-206AC with the lead 2 omitted. (the latter is in fact a diode package and is mainly used for Varactor tuning diodes and shunt voltage references).

Construction and orientation

The case is molded around the transistor elements in two parts; the face is flat, bearing a machine-printed part number. The back is semi-circularly-shaped. A line of moulding flash from the injection-moulding process can be seen around the case.

The leads protrude from the bottom of the case. When looking at the face of the transistor, the leads are commonly configured from left-to-right as the emitter, base, and collector for 2N series (JEDEC) transistors, however, other configurations are possible, such as emitter, collector, and base commonly used for 2S series (Japanese) transistors.

If the face has a part name made up of only one letter and a few numbers, it is usually assumed to be a Japanese part number (with the base on the end rather than in the center). Thus, "C1234" would likely be a 2SC1234 device.

The leads coming out of the case are spaced 0.05" (1.27 mm) apart. It is often convenient to bend them outward to a 0.10" (2.54 mm) spacing to make more room for wiring. Units with their leads pre-bent may be ordered to fit specific board layouts, depending on the application. Otherwise, the leads may be bent manually; however, care must be taken as they can break easily.

The physical dimensions of the TO-92 housing may vary sightly depending of the manufacturer, however, the 1.27mm lead spacing must be respected.

Advantages

- Transistors of this type can be made very inexpensively and take up very little board space. Most models are readily available in large quantities from wholesale distributors.
- They are easy to find in small electronics stores because of their wide usefulness, making them a popular choice for hobby work and prototyping.

Disadvantages

The main disadvantage of this style of case is the lack of heat sinking.

- Transistors and ICs of these types cannot handle as much power as higher-power equivalents, such as the TO-220 and can burn out quickly if they dissipate excessive power.
- There is no standard pinout for the TO-92. The American BJT's use the E-B-C pinout while their Japanese counterparts use the E-C-B pinout and some RF devices use the B-E-C pinout.

More often than not, however, the disadvantages concerning accidental destruction are greatly outweighed by the small size and low cost of producing these units.

Although TO-92 devices are mainly used in low-voltage / low-current (<30 V; <1 A) applications, high-voltage (600 Volt Vce) and high-current (5 A Ic) devices are available. Nominal maximum power dissipation is less than one watt (600 mW).

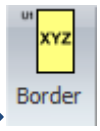
1.2.5.10 Creating Parts

Parts are the cities of your design. They are where the business is done. They are connected together with copper tracks that carry the goods from one part to another.

You can create them in one of 3 ways.

- Create a new part.
- Use the part wizard.
- Create the in place in schematics.

1.2.5.10.1 Creating Parts on Schematics

To create a part on a schematic click the Add→Symbol→ 

button.

Add a terminal magnet and then add symbol terminals.

Finally use the Part Wizard to set the footprint.

1.2.5.10.2 Editing Parts on Schematics

You can change part symbols in schematics by dragging the symbol terminal, the symbol reference or the symbol value.

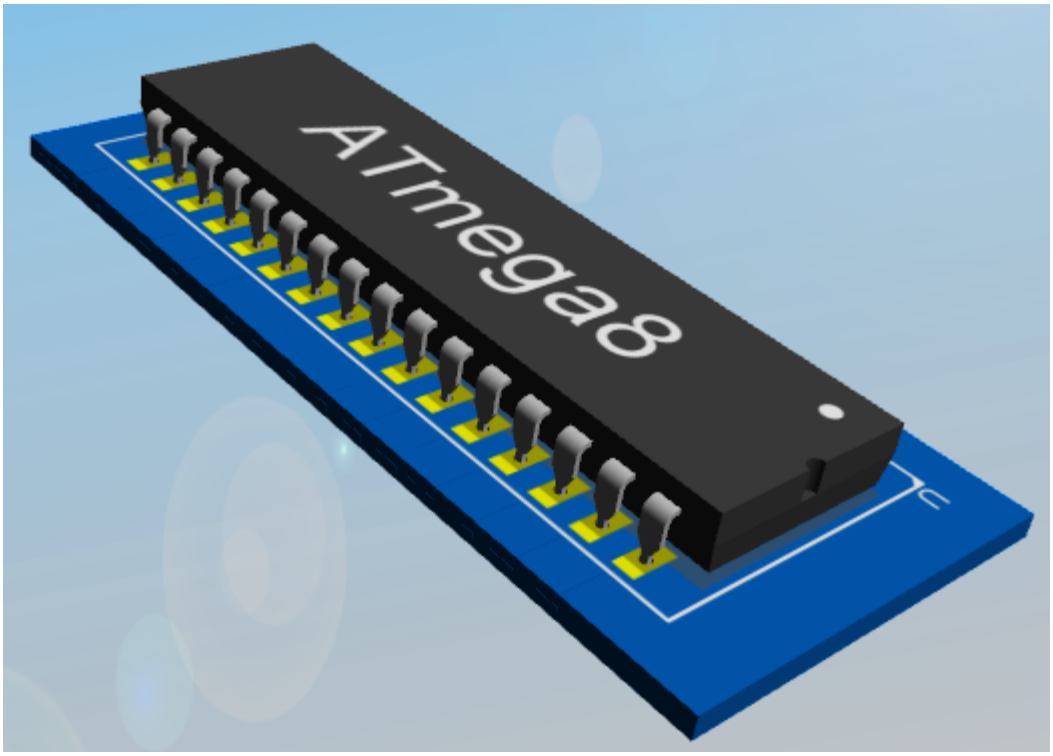
Use sub-pick to select the terminal magnet and you can then drag and re-size it.

You can also use Part Builder

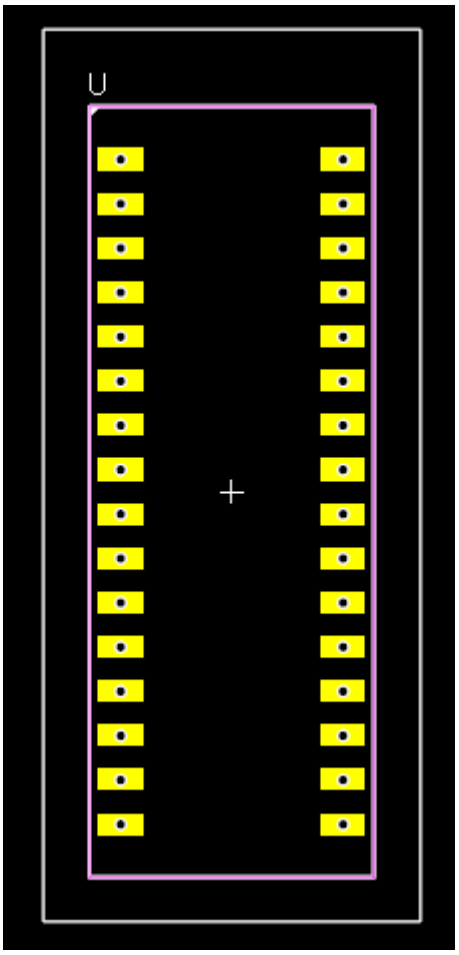
1.2.5.10.3 Parts

Parts consist of 3 distinct representation:

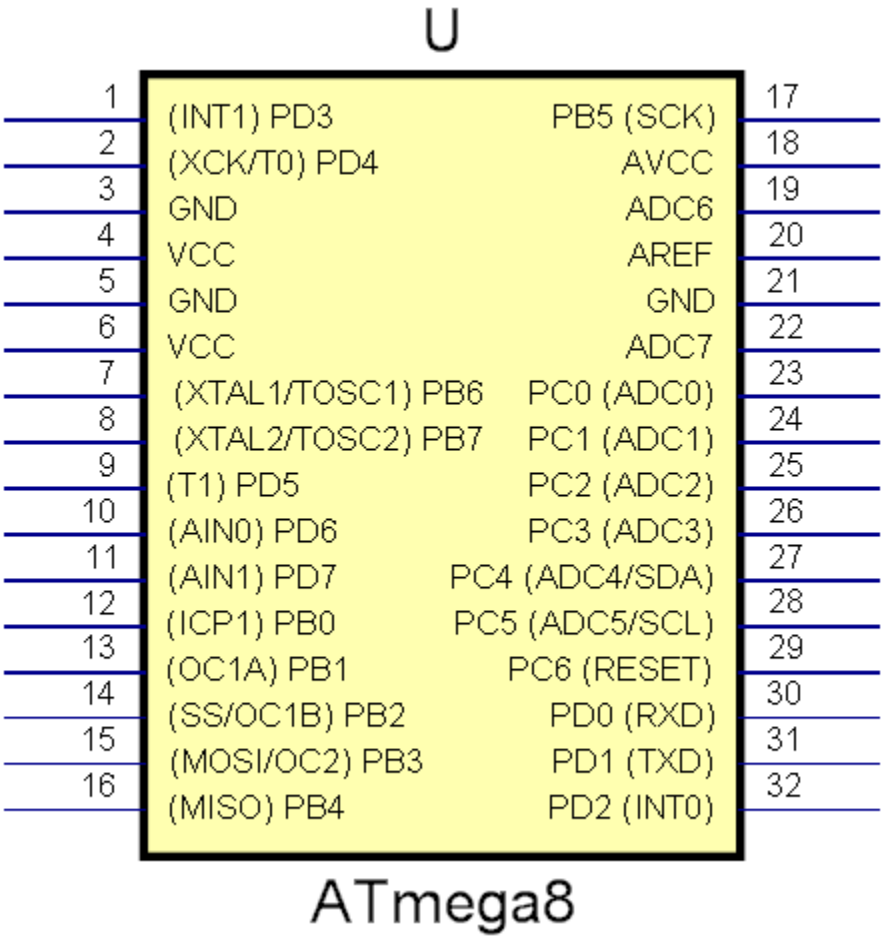
1. The physical device, This is shown in the 3-D view.
2. The footprint or land pattern which defines how the part is Attached to the PCB using pads, silkscreen graphics, placement points (which serve as device coordinates for pick and place machines) and courtyard rectangles (which define the area used by the part). on the PCB.
3. The schematic symbol(s). A part can have the logical device split up into more than one symbols. This is common in logic devices that contain more than one logical gate.



The Physical Part (3D)



The PCB Footprint or Land Pattern for the part (2D for PCB layout)



The Schematic Symbol for the part (for schematic diagrams)

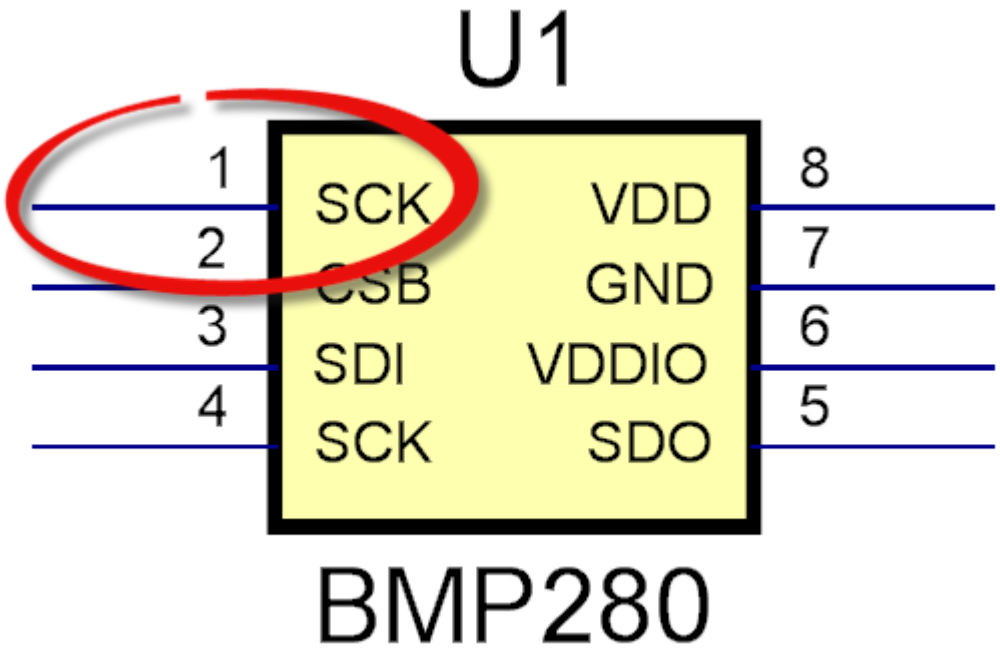
You place one or more parts on your schematic(s).

1.2.5.10.4 Symbols

The parts are electrically connected together using schematic wires.

Parts are shown on schematics using schematic part symbols.

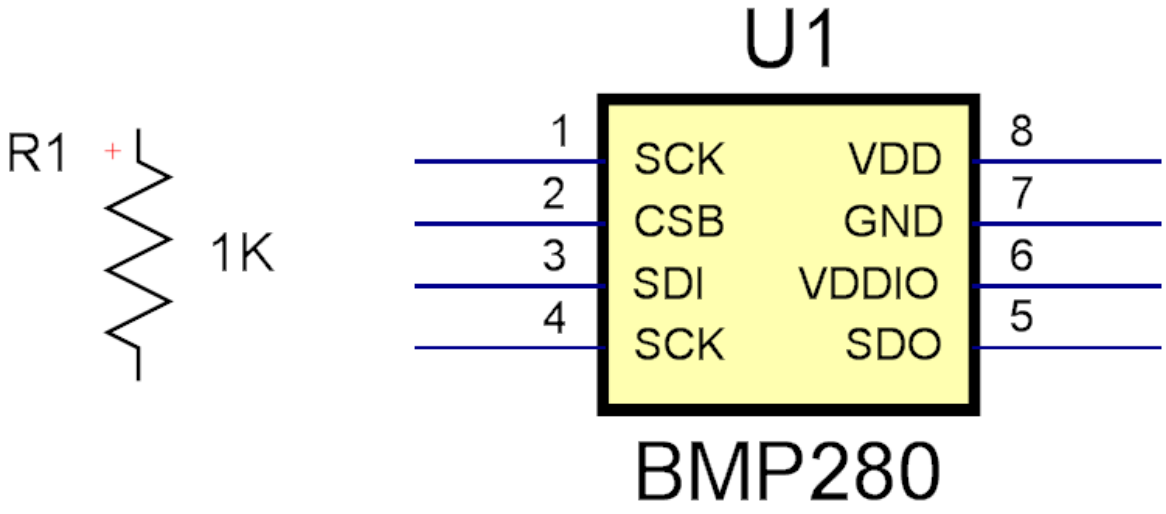
A part symbol consists of one or more symbol terminals.



Symbol terminal

The symbol terminals serve as the electrical connection points for the part and schematic wires start and end on symbol terminals.

A symbol can optionally have a terminal magnet that is rectangular and serves as the anchor for all the symbol terminals in a part. If you move the symbol terminal is always stays attached to the terminal magnet.

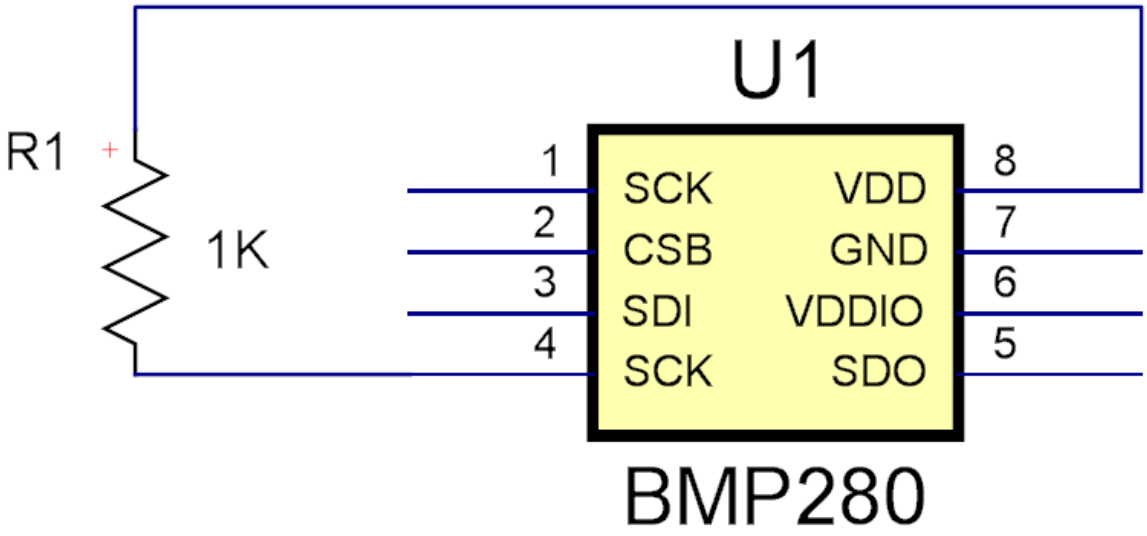


R1 has no terminal magnet while U1 does.

Symbol terminals are always snapped to a 0.05 grid. Since schematics are logical the coordinates are not tied to a measuring system such as inches or cm. However, when it comes to printing the grid is mapped to inches.

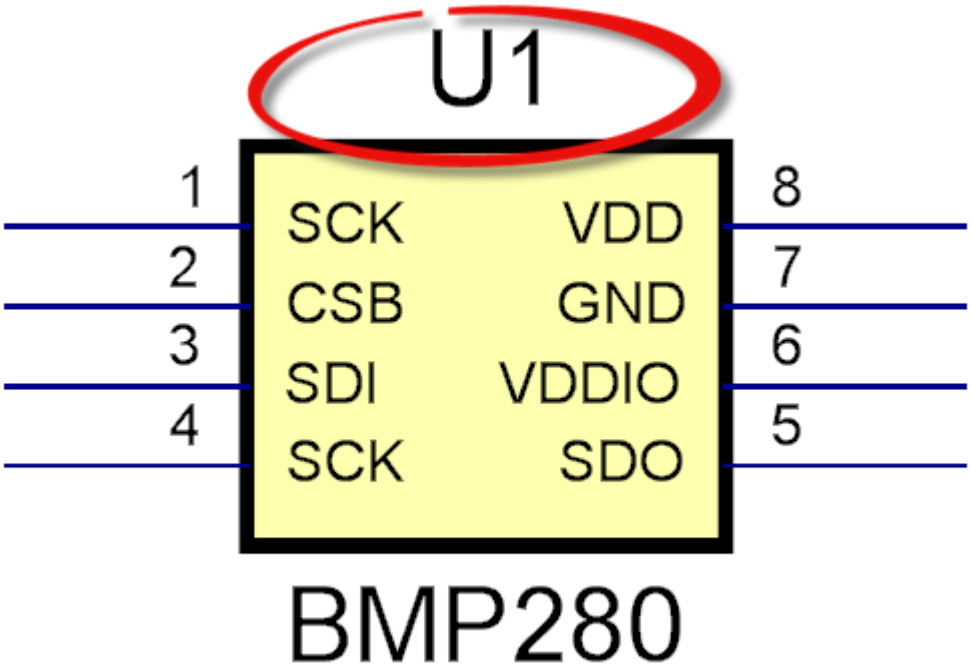
A symbol terminal has a name. The name can be set to invisible when it is on a project schematic. It is always visible in the part designer.

Wire connect terminals to each other.



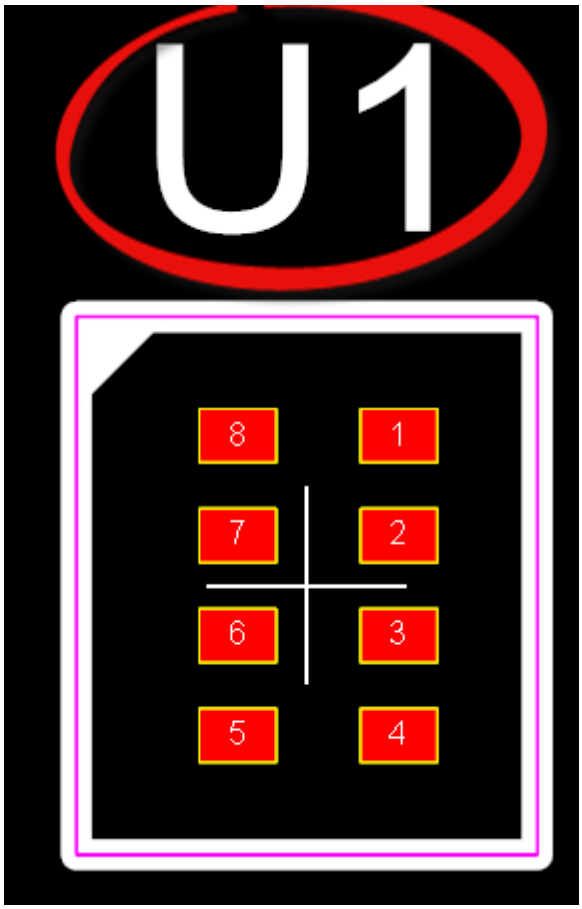
2 parts wired together

All symbols have a symbol reference. This serves to identify the part on the schematic and the PCB and is used in the Bill of Materials (BOM) which lists all the parts used in your design.



Symbol Reference

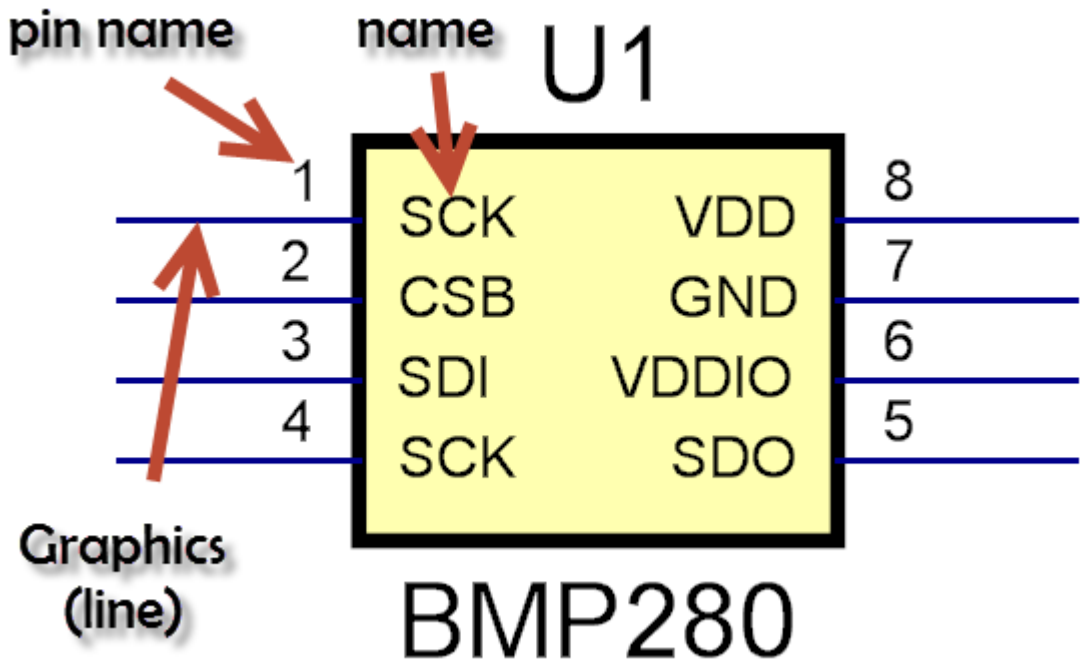
There is also Footprint reference on the PCB for the part.



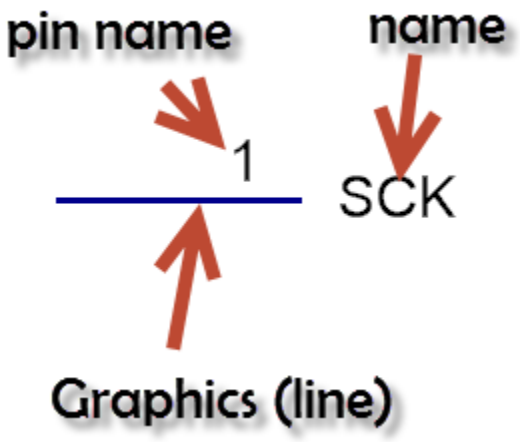
1.2.5.10.4.1 Terminals

Terminals consist of 3 parts.

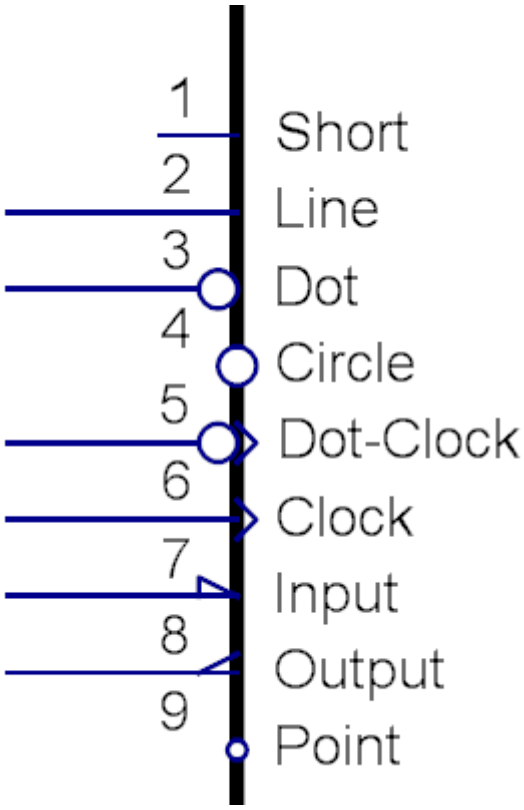
1. Graphical shapes such as a straight line or circle.
2. A name. This usually show the signal name such as Vcc or CLK.
3. A pin name. Normally this is just a number but in AutoTRAX DEX it can be text with spaces. This is the pin name of the device.



Symbol Terminal

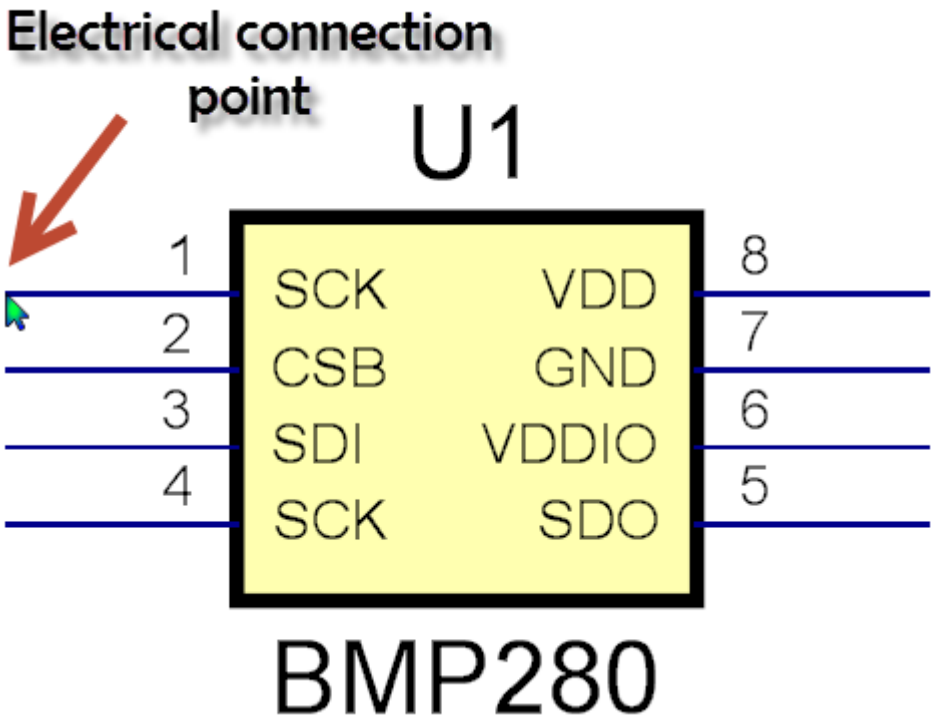


Symbol Terminal showing components

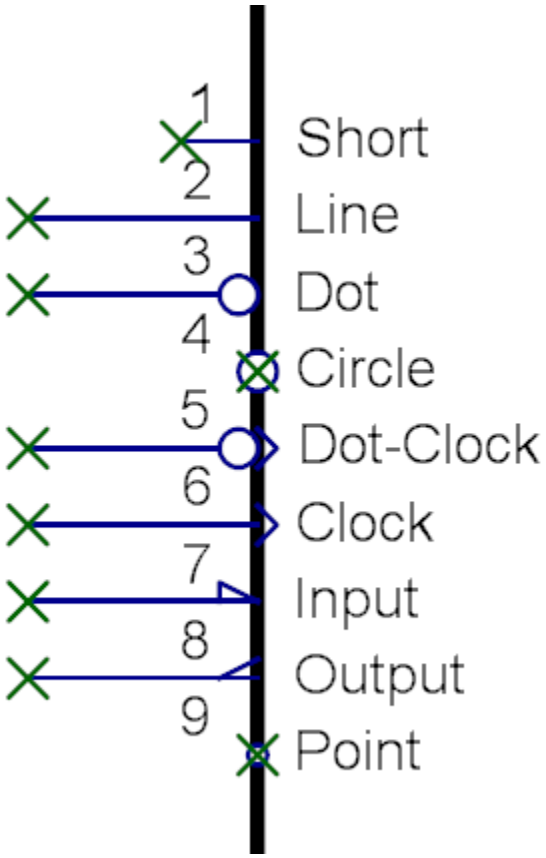


The 9 different terminal types (Graphics)

All terminals have an electrical connection point for schematic wires.

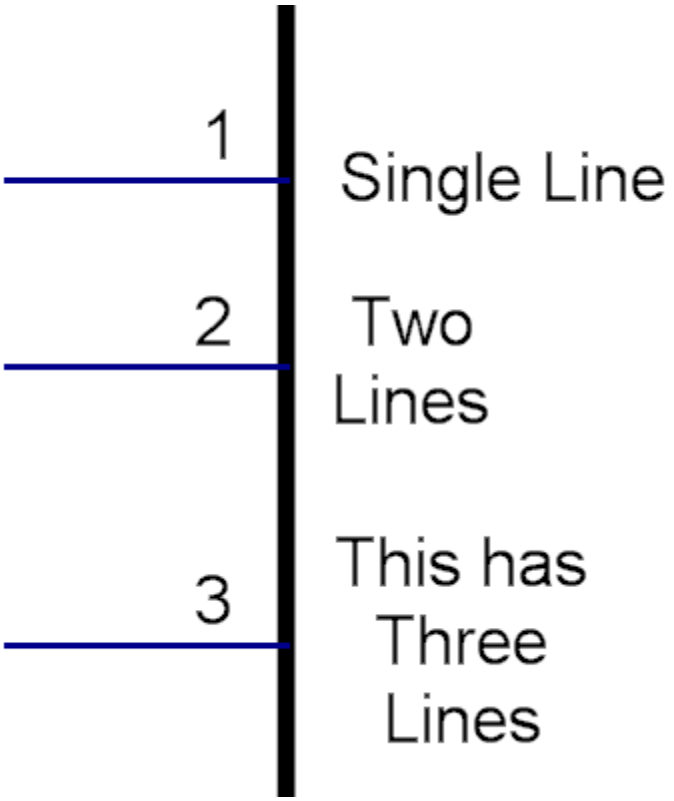


Electrical connection point

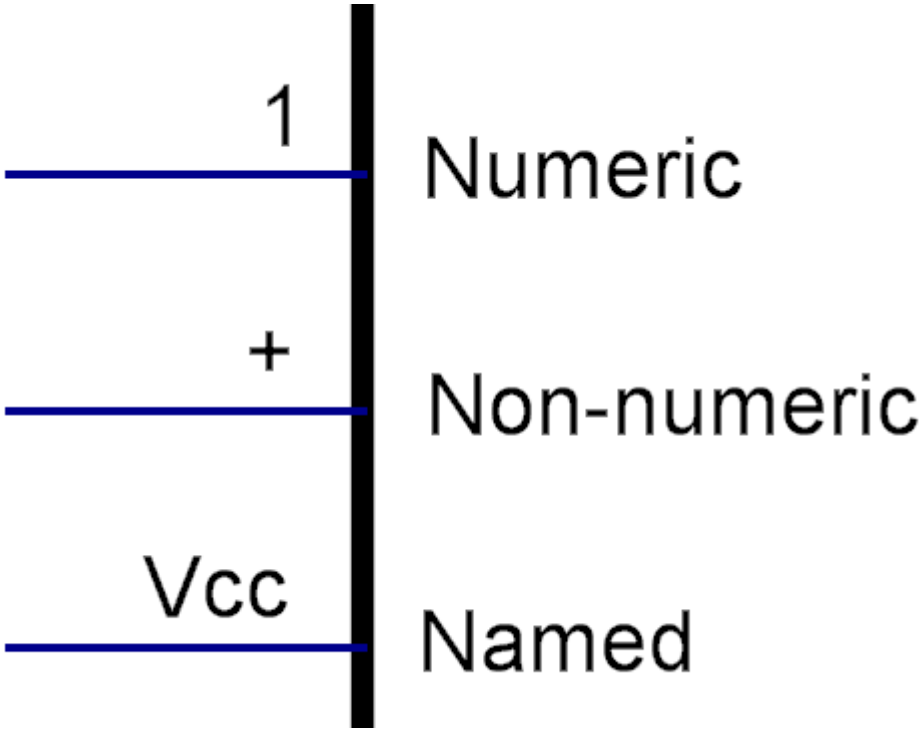


Electrical connection points marked as green crosses

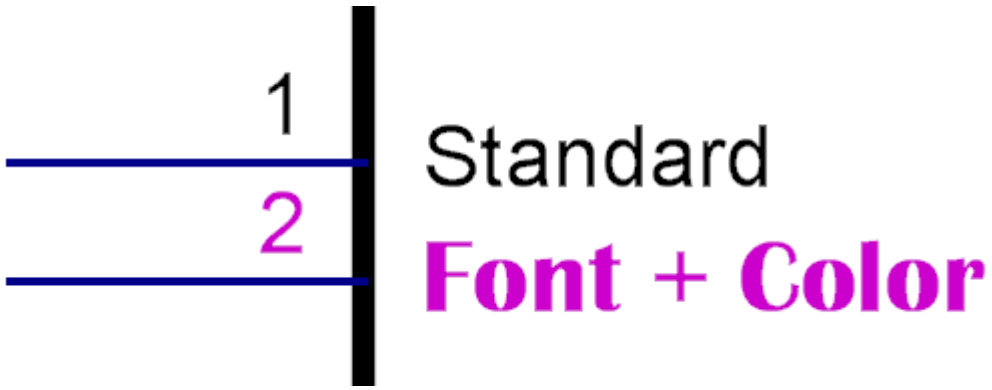
.



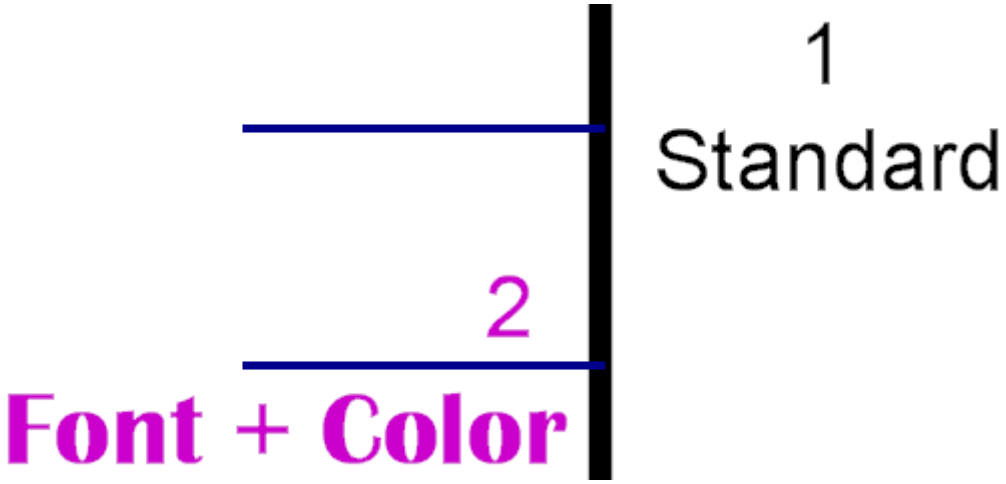
Multi-line terminal names



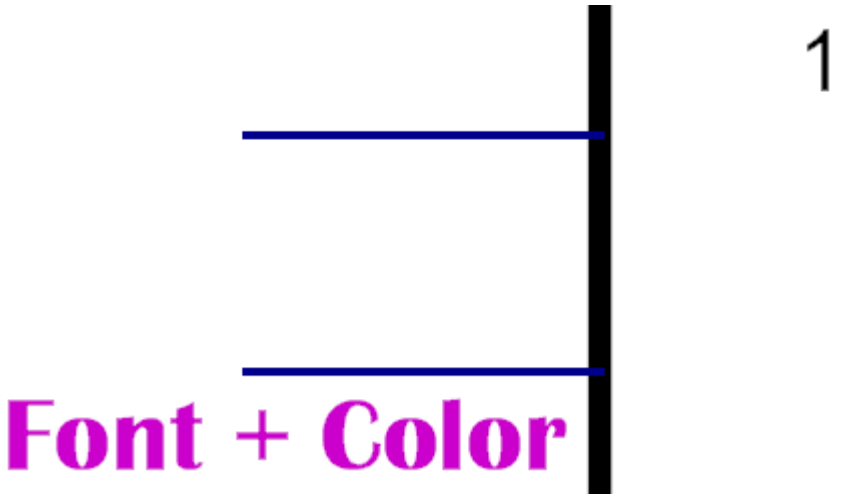
Pin names - numeric and non-numeric



You can set the font, size and color for each terminal

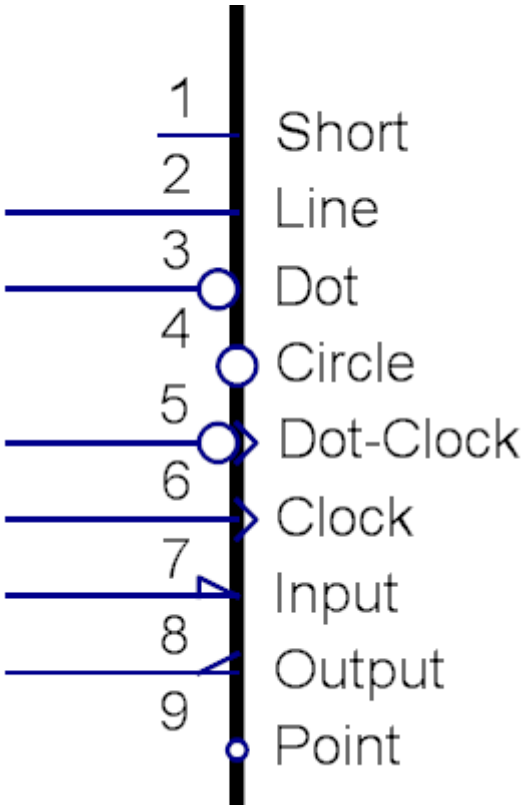


Names and pin names can be moved independently



The name and pin name can be independently hidden

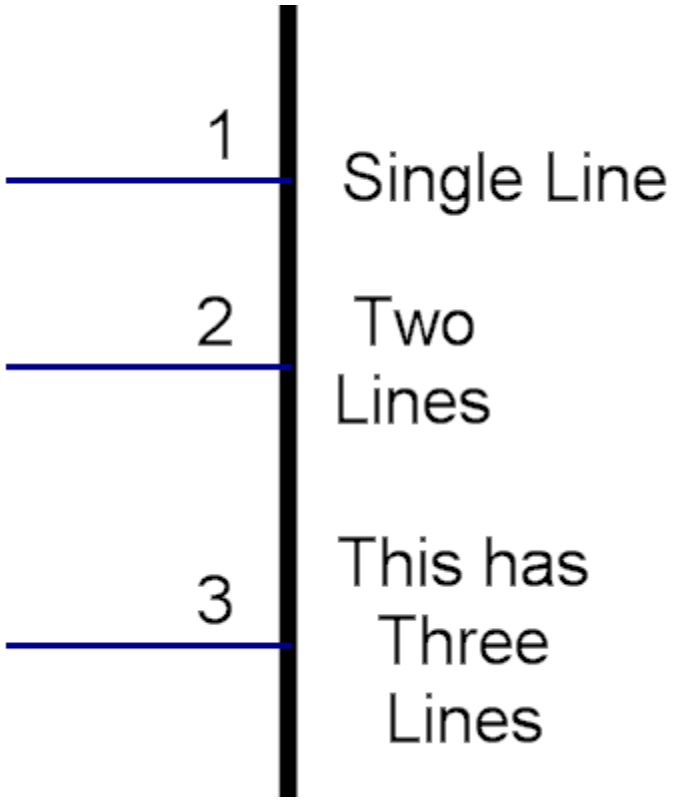
There are 9 different types of terminal graphics.



Terminal Graphics

Each terminal can have an name. The name has no electrical significance when it comes to PCB layout. It is only for you to make sense of the design in schematics.

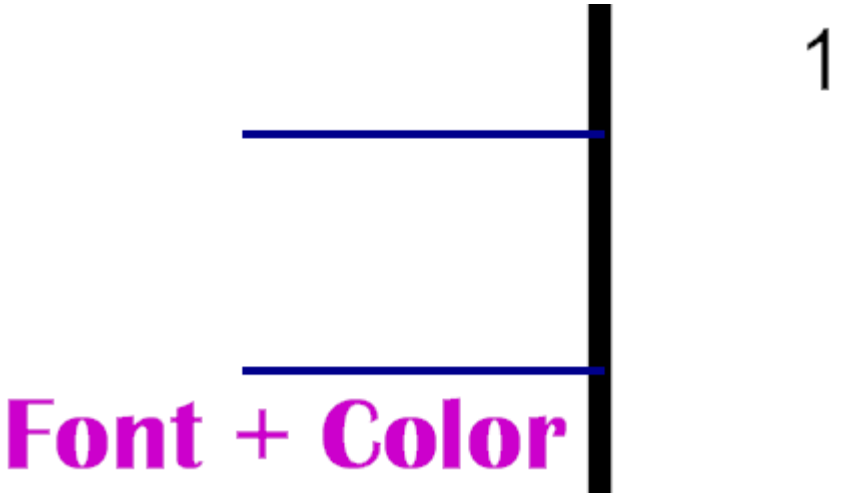
The names can be more than one line.



Multi-line terminal names

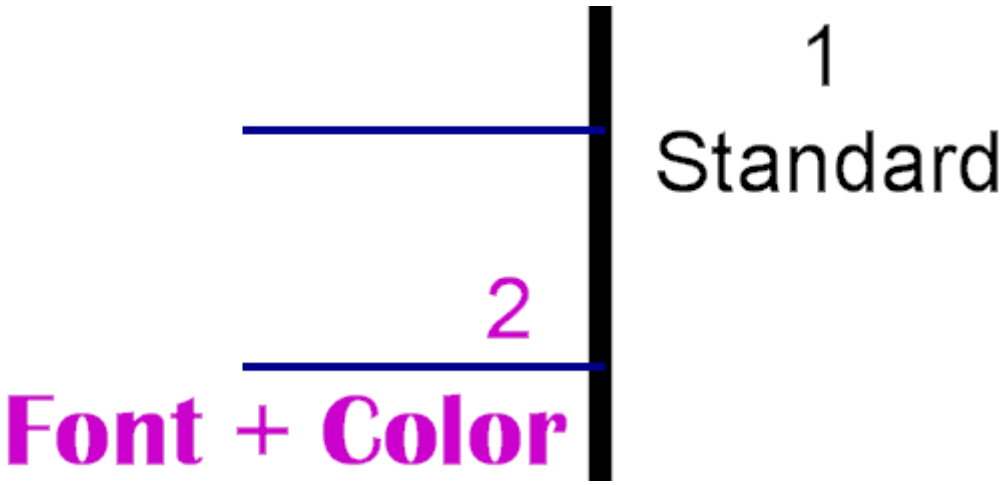
The name is normally set to a signal name such as Vcc or D0.

Terminal names can be hidden.



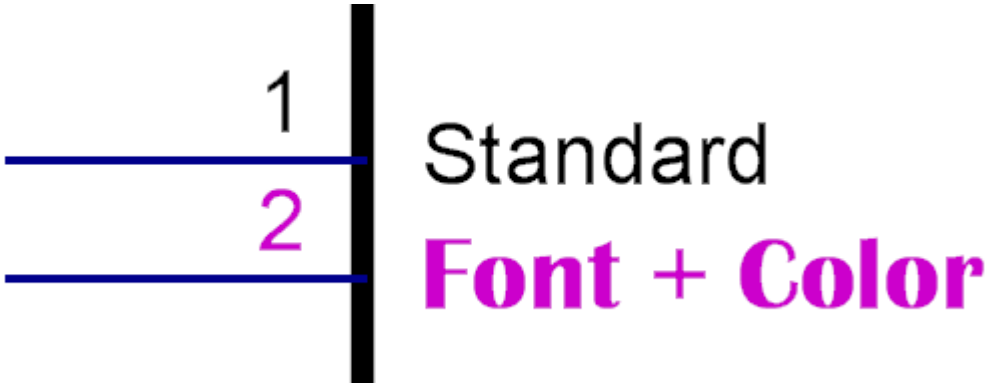
The name and pin name can be independently hidden

You can move a terminal name from its default position.



Names and pin names can be moved independently

You can set the font, text size and color.



You can set the font, size and color for each terminal

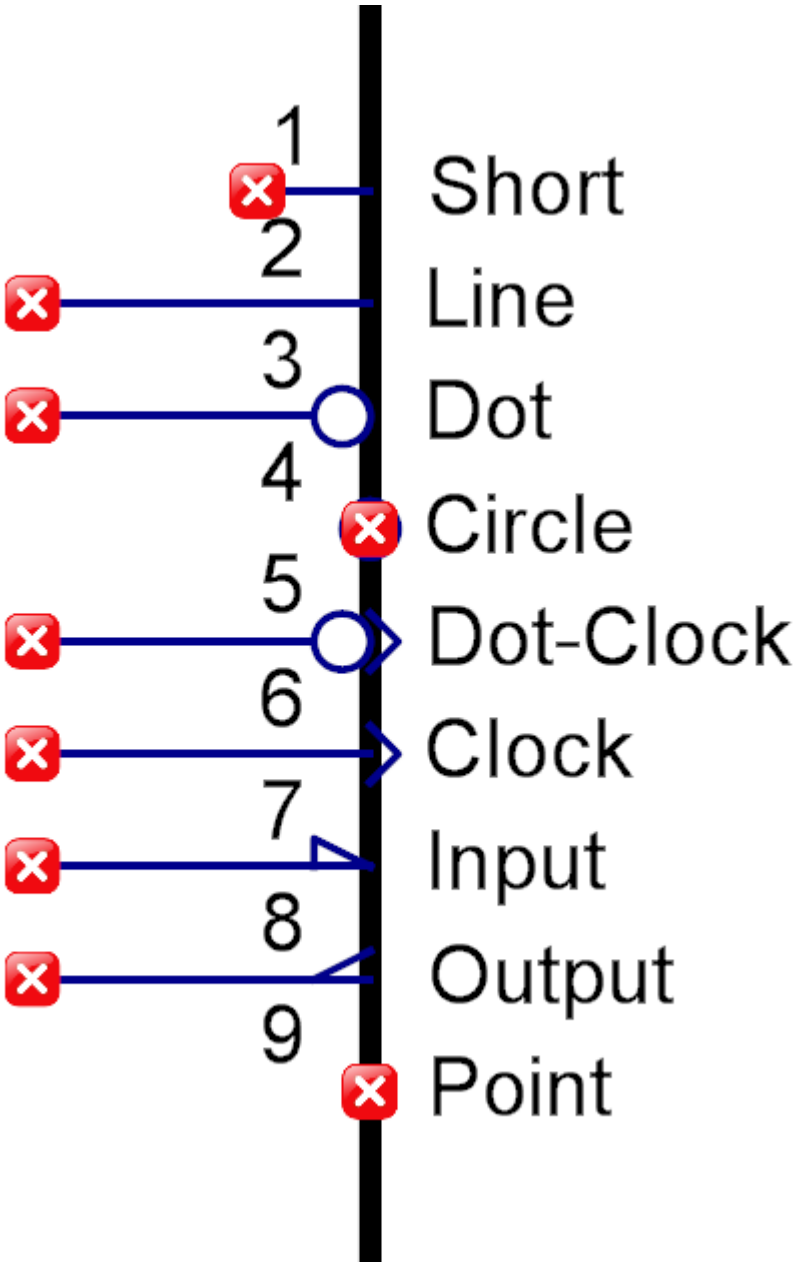
The terminal name is the name you give to the physical connection of the device the part represents. It is generally the device pin leads. Conventionally you will use the name in the devices data sheet. Often it is a number but sometimes it is a character and a number. With AutoTRAX DEX you can name your pins with any combination or number of characters and digits.

AutoTRAX DEX does not use the pin name electrically. It uses an internal unique id for each electrical connection on the device.

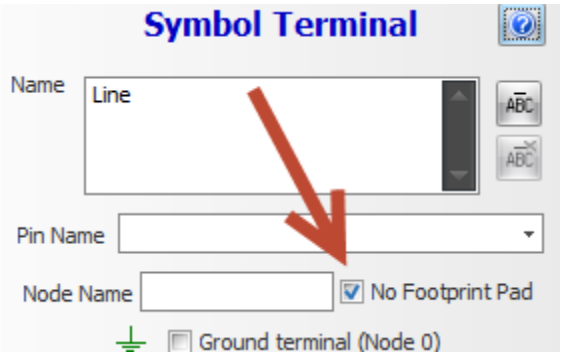
A pad in the part's footprint is the embodiment of the connection.

AutoTRAX DEX records the pin name in the pad. If you rename it in the symbol, the name in the pad is changed.

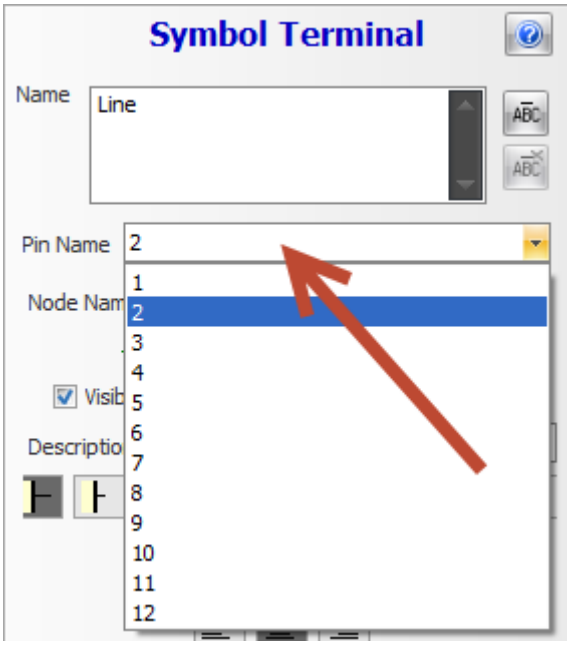
Terminals with no associated pads do not display a pin name. Instead they display a no connection symbol.



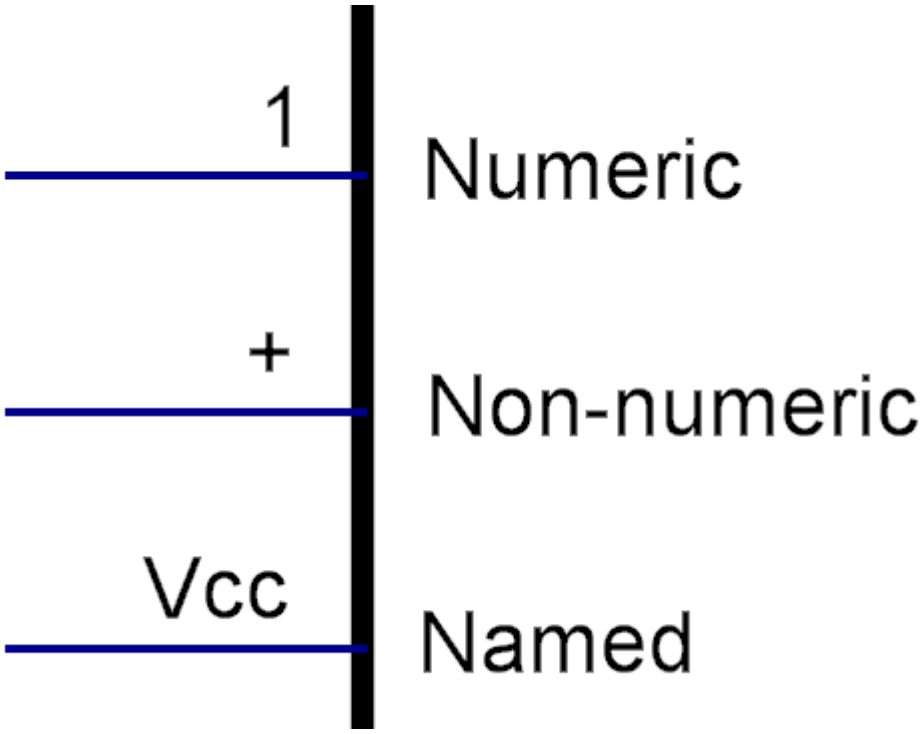
Symbol terminals with no associate pad.



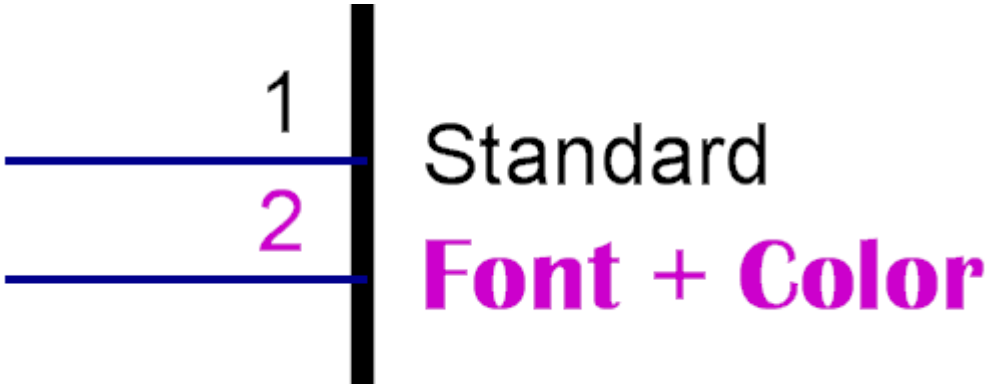
You can mark a terminal as not having an associated pad by checking the No Footprint Pad in the terminals properties panel



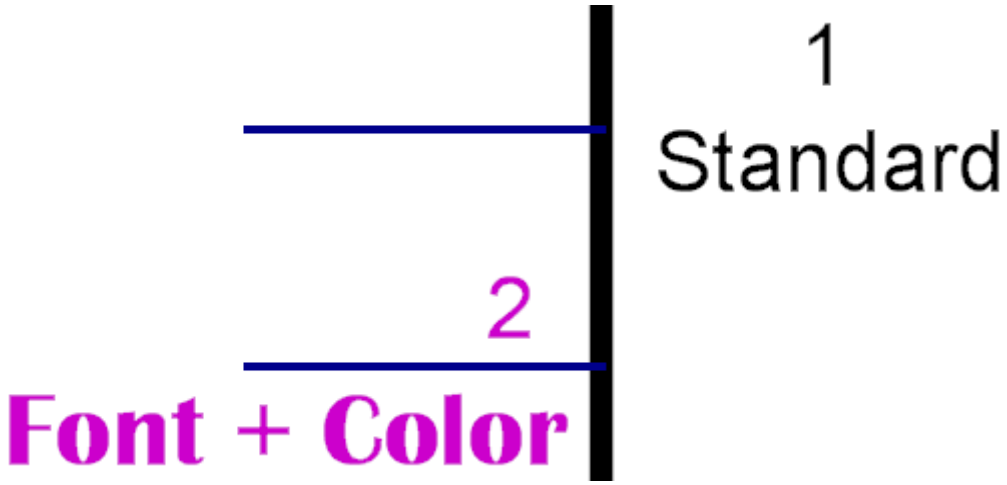
You can change the associated pad for a terminal by selecting the pin name in the terminals properties panel



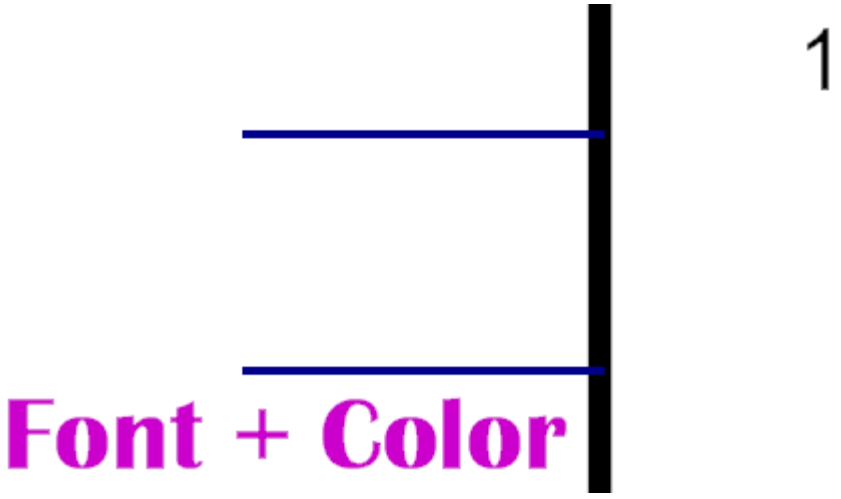
Pin names - numeric and non-numeric



You can set the font, size and color for each terminal



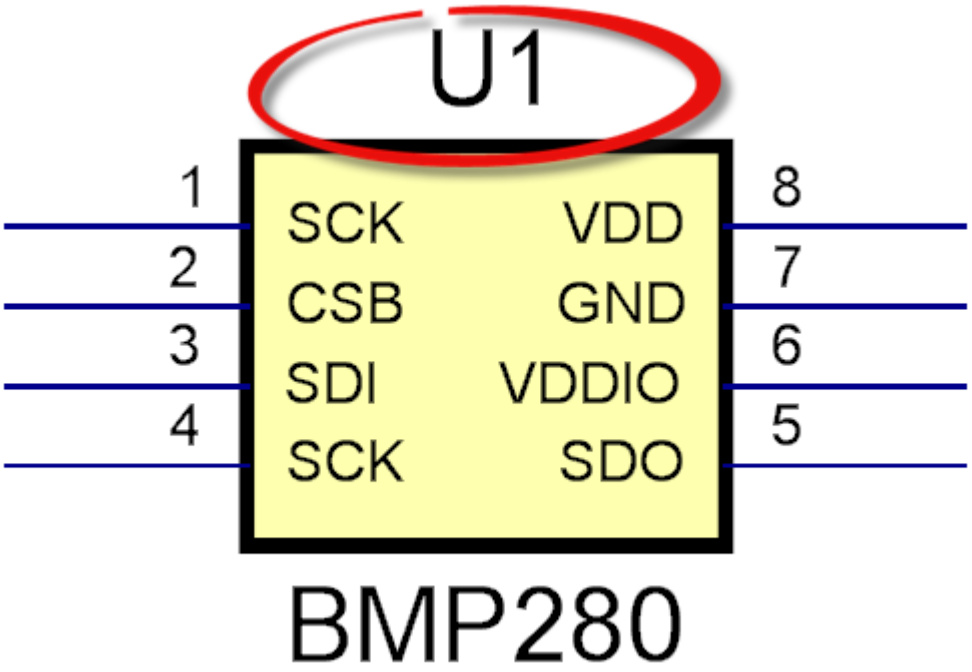
Names and pin names can be moved independently



The name and pin name can be independently hidden

1.2.5.10.4.2 Reference

All symbols have a symbol reference. This serves to identify the part on the schematic and the PCB and is used in the Bill of Materials (BOM) which lists all the parts used in your design.

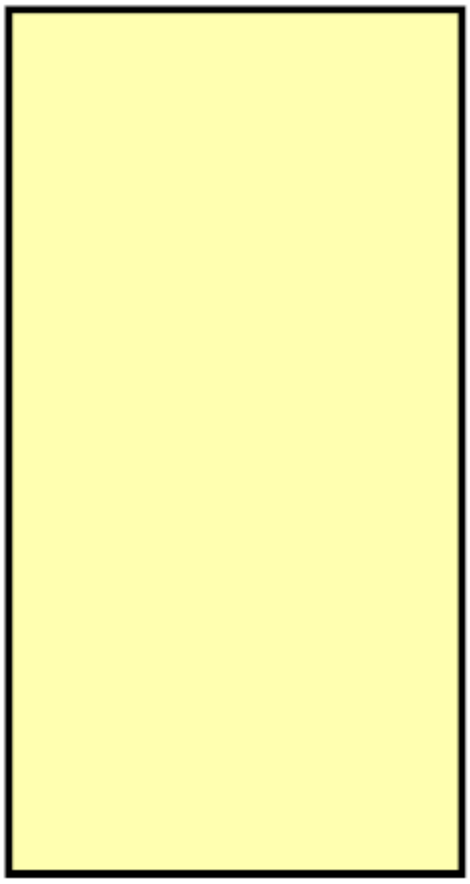


1.2.5.10.4.3 Border

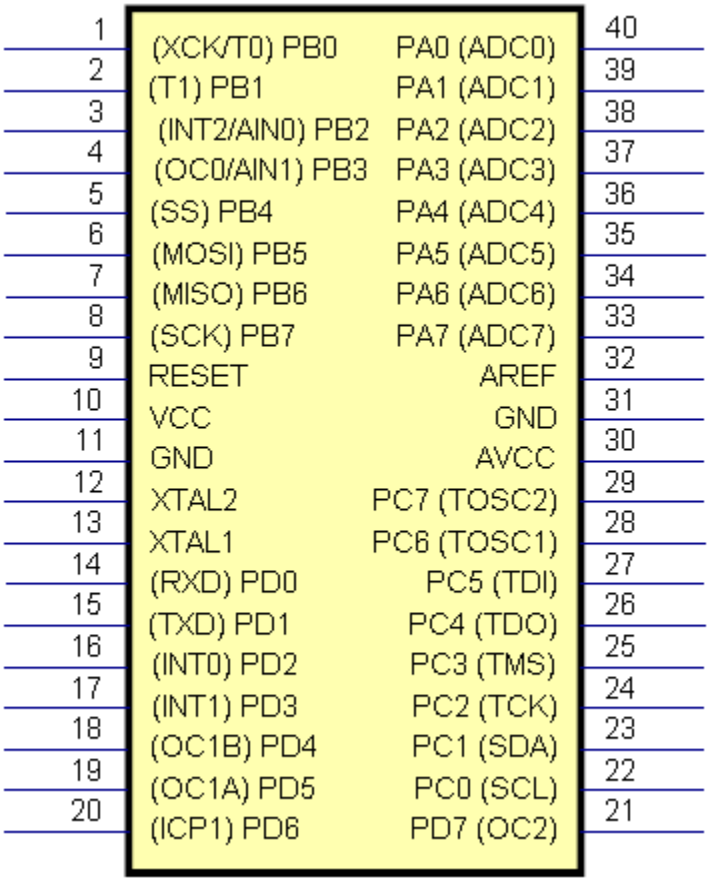
Symbols optionally have a terminal magnet. Terminal magnets server as anchor edges forces all symbol terminals to stick to the edge. Terminals cannot be dragged of the edge. As you drag a terminal the terminal will try to follow but it will always remain attached to the border.

Terminals always stick to a grid of 0.1 units. This gives you a good separation of terminals. Schematic wires snap to a 0.05 unit grid.

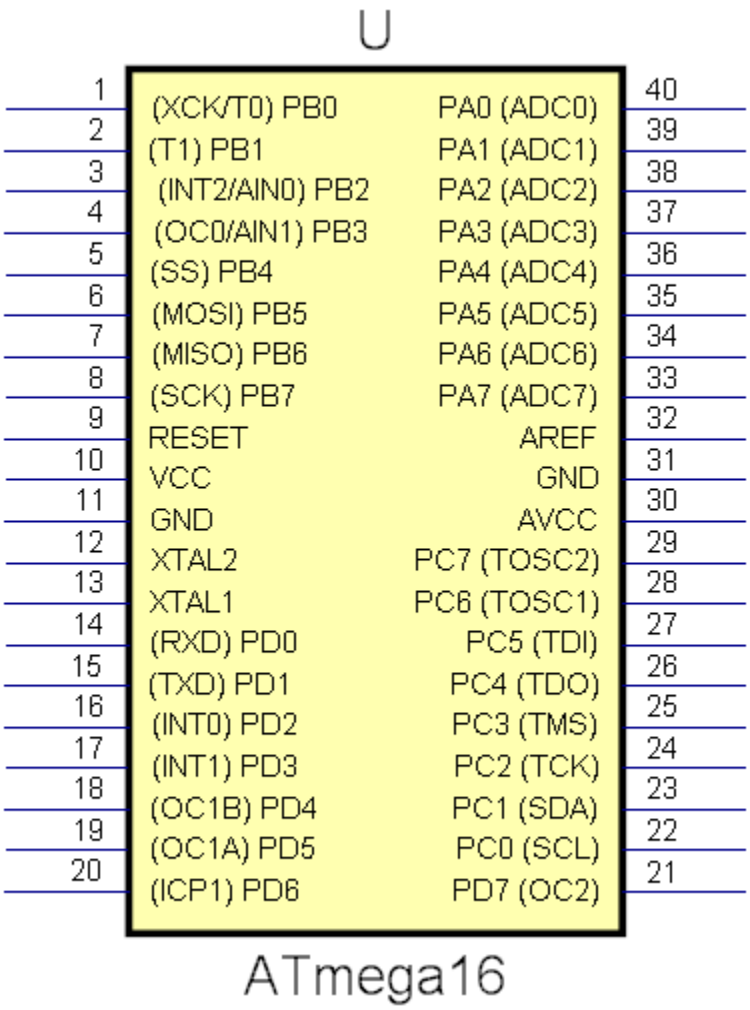
You can delete the border by selecting it and pressing the DEL key.



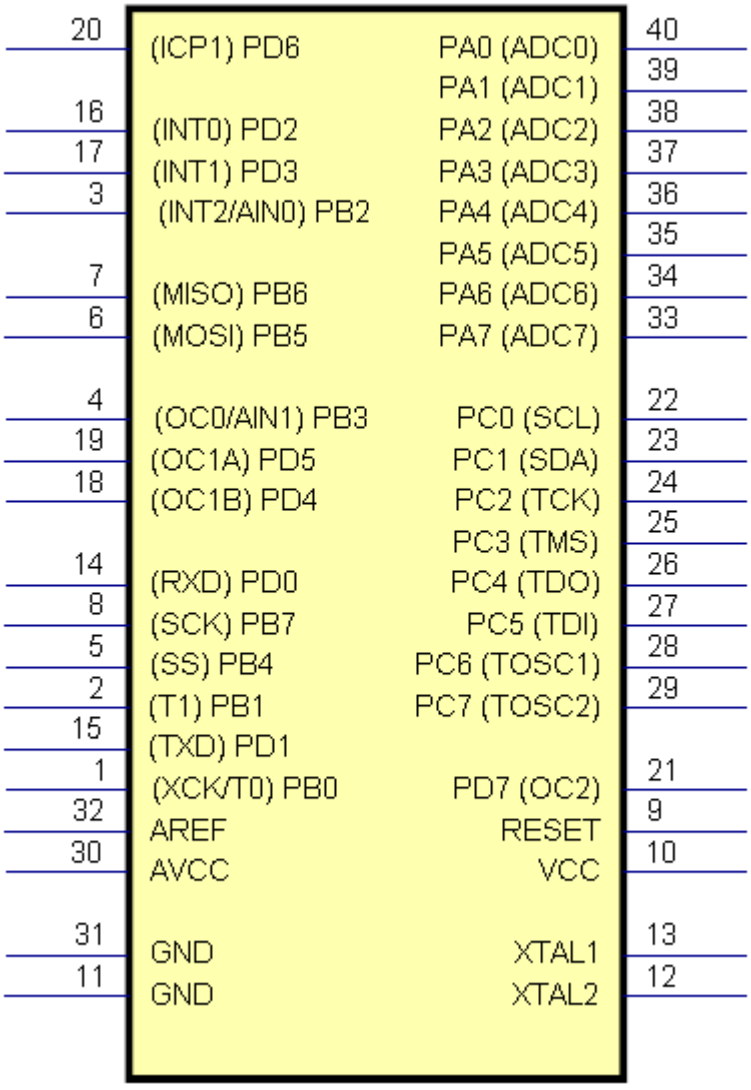
A terminal magnet with no terminals



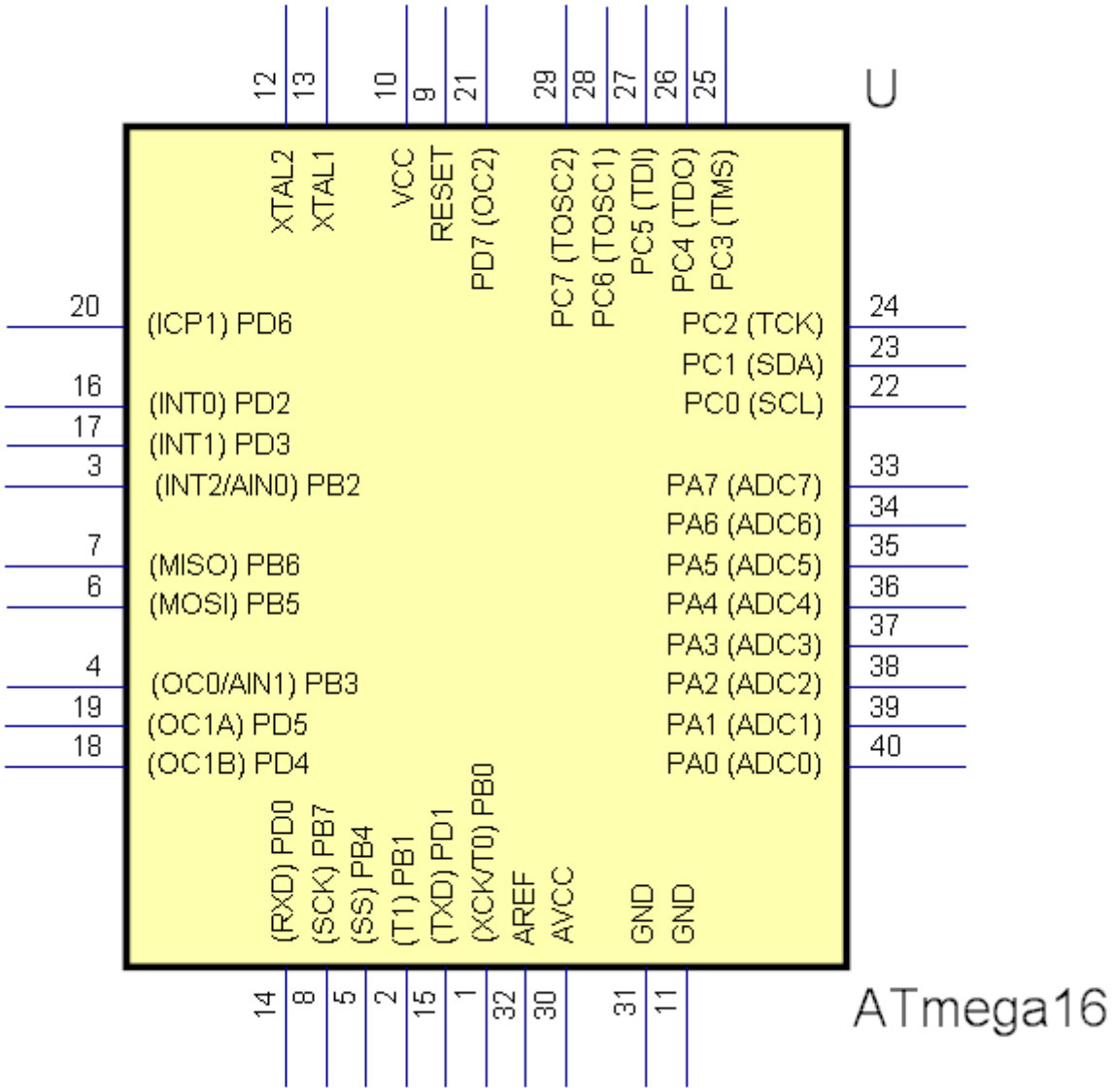
A terminal magnet with terminals



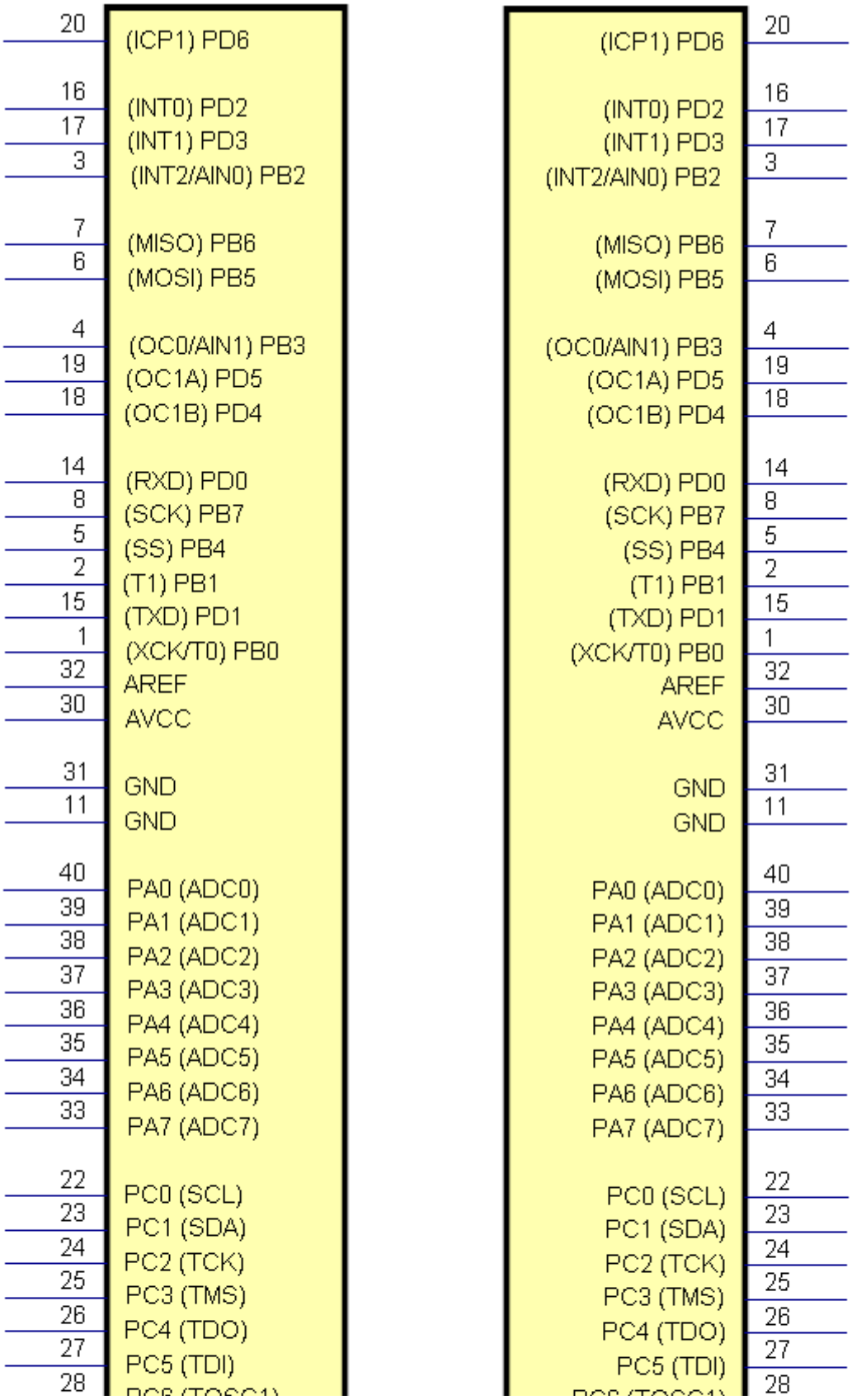
Organized by pin name



Organized by name with grouping



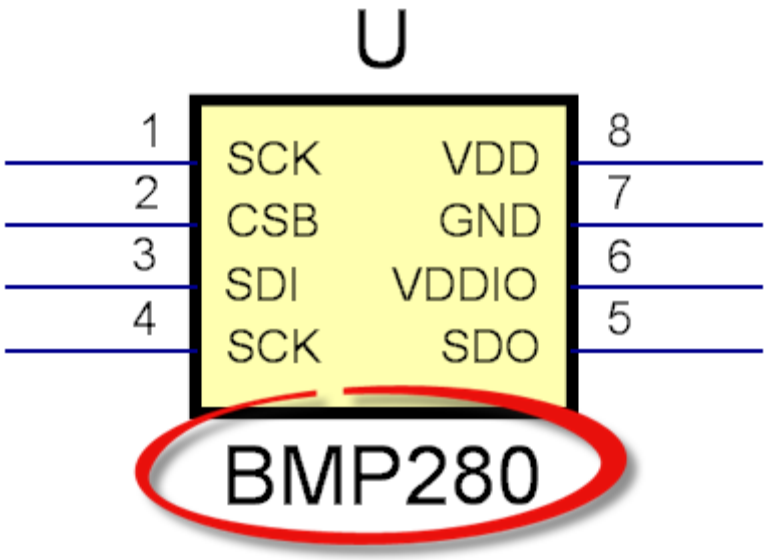
Pins on all 4 sides



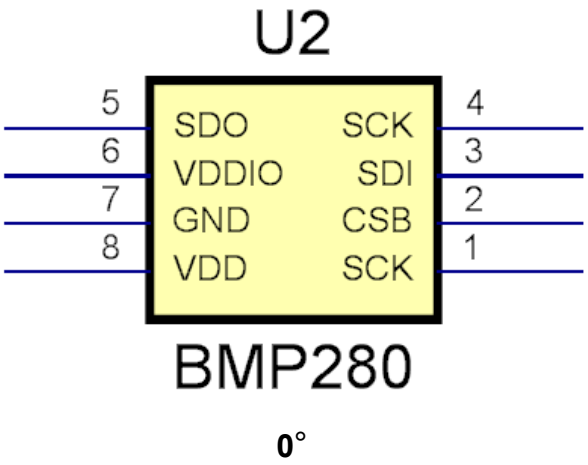
Pins on left **Pins on right**

1.2.5.10.4.4 Value

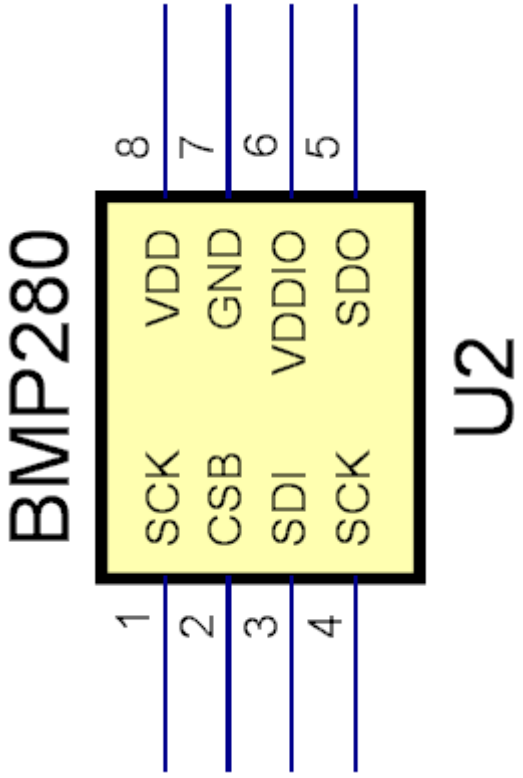
Symbol have a symbol value. It usually represents the part's name. It can be hidden.



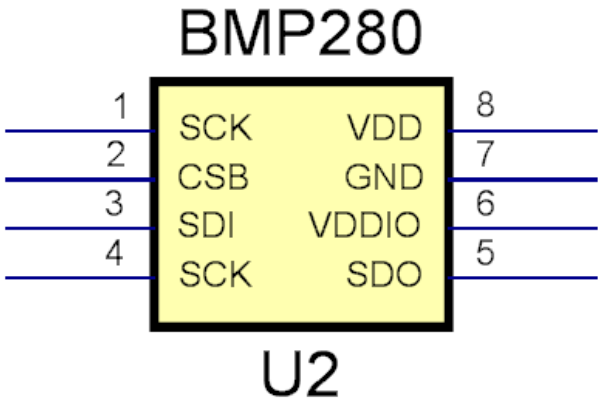
Symbol Value



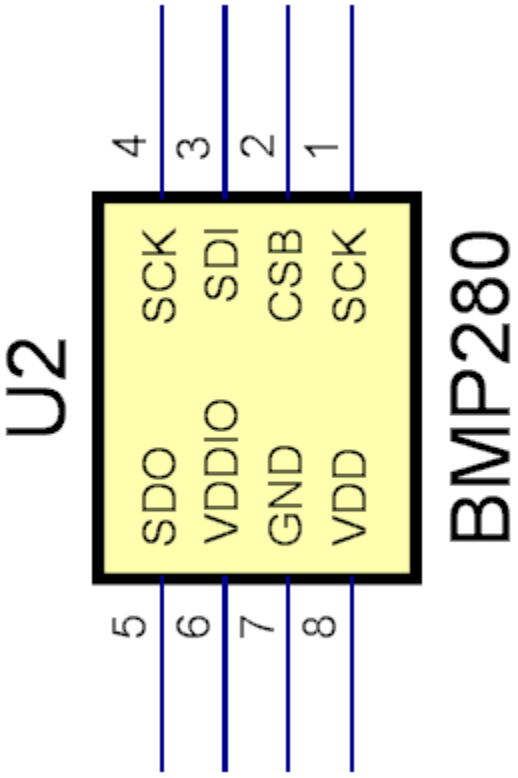
0°



90°



180°



270°

Rotating symbols - Symbol reference and value remain right reading

1.2.5.10.5 Footprints/Land Patterns

The Footprint or Land Pattern defines what is needed on the PCB for manufacture.

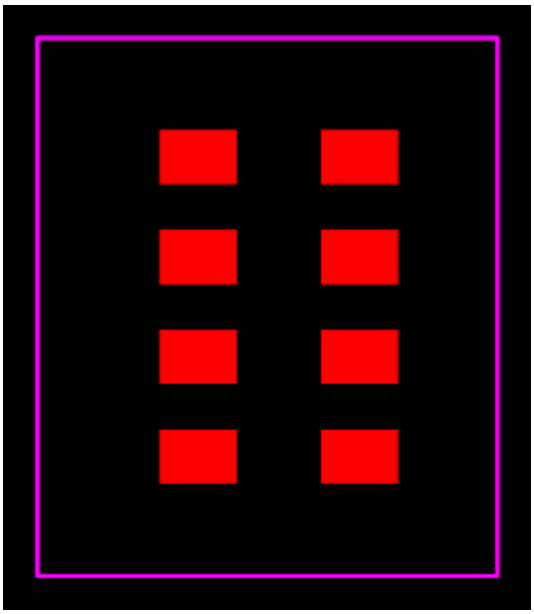
This includes the footprint reference, pads, silkscreen pattern, the courtyard and the placement point.

the Footprint reference. This is drawn on the top or bottom silkscreen layer has no electrical significance.

The placement point used for pick and place devices when you board is made.

1.2.5.10.5.1 Courtyard

The courtyard rectangle. This defines the area of the PCB used by the part. Courtyard rectangles must not overlap each other.



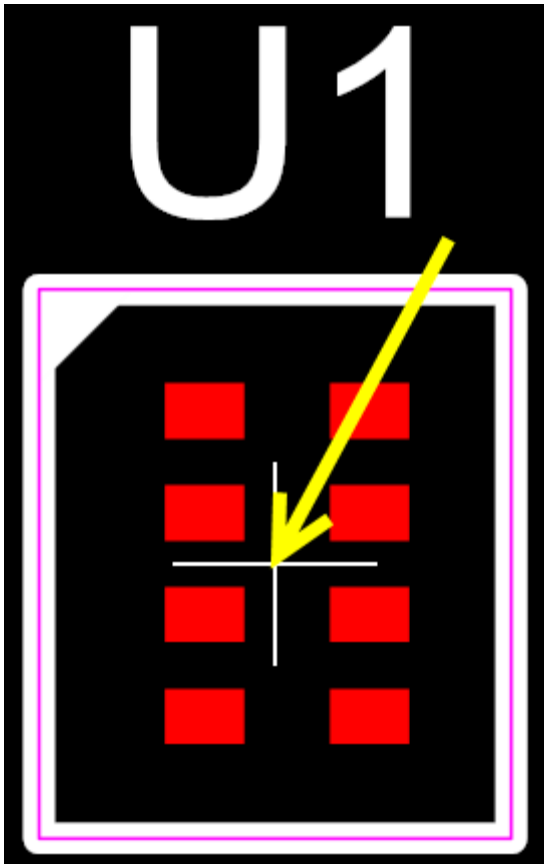
Courtyard rectangle (magenta)

A **footprints courtyard**

1.2.5.10.5.2 Placement Point

The placement point used for pick and place devices when you board is made.

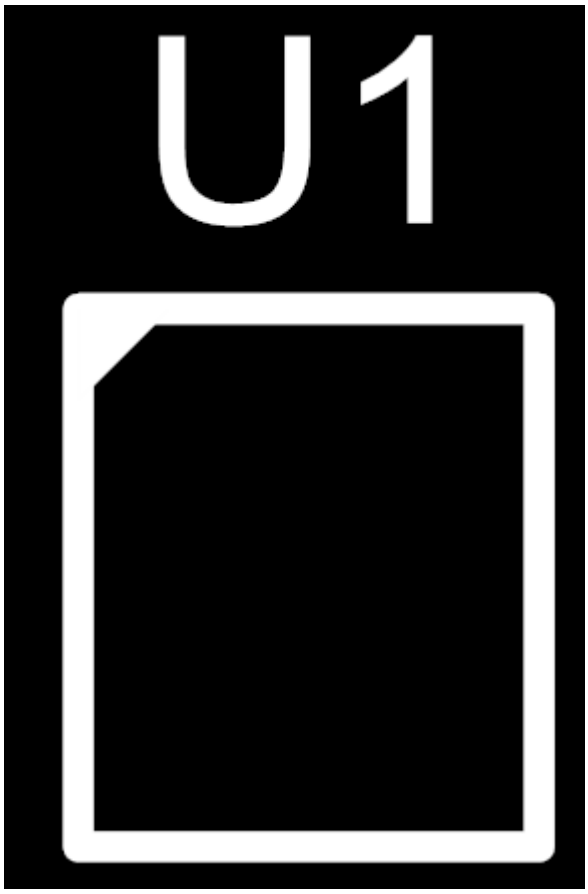
You can move it by dragging it or setting it's position in the placement points properties panel.



The footprint placement point

1.2.5.10.5.3 Silkscreen Pattern

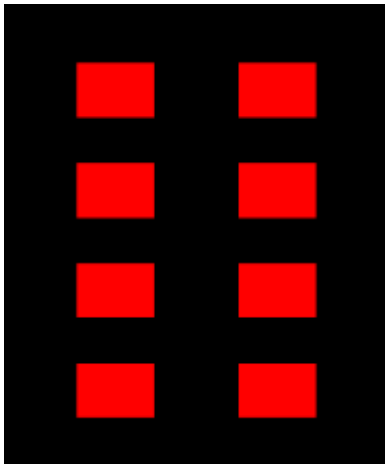
The screen pattern to show humans where the part is. In this case it is the white rectangle. Like the footprint reference has no electrical significance and could be left out for cost conscious board at manufacture time.



The silkscreen layer showing only the footprint reference and silkscreen pattern

1.2.5.10.5.4 Pads

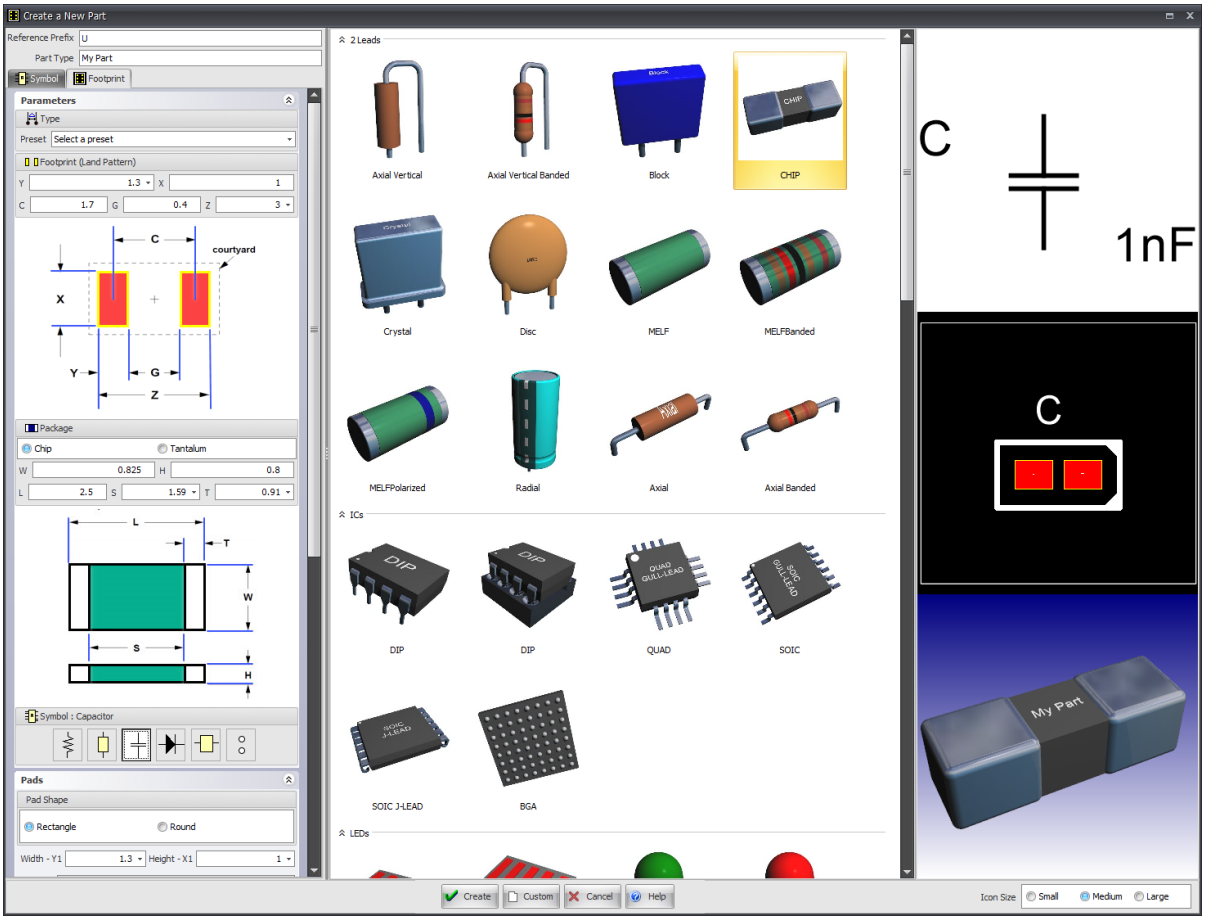
The electrical pads and holes for Through Plated Technology (TPT) devices.



Electrical pads (SMT)

1.2.5.10.6 Creating a NewPart

Creating a Parametric Part



1.2.5.11 Symbols

A symbol is a graphical representation of a part that is placed on a schematic sheet.

A part can have one or more symbols.

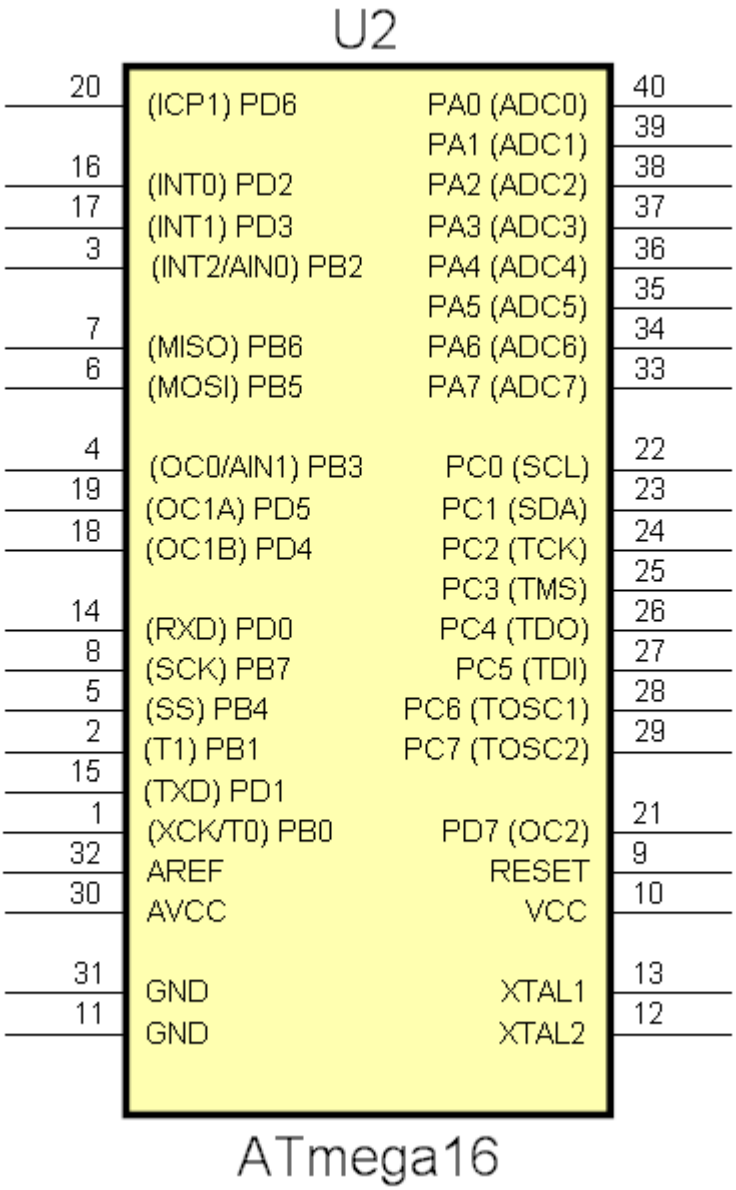
A symbol can have several sibling symbols that optionally represent a part. (Its footprint or land pattern).

However it is possible for symbols to not be related to a part.

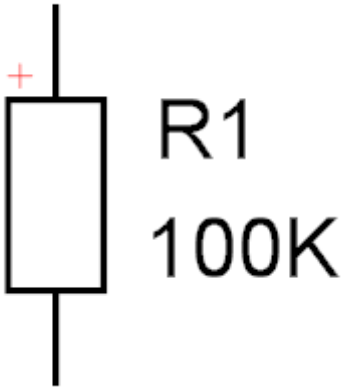
A symbol consists of:

- An optional Terminal Magnet
- Symbol Terminals
- An optional Symbol Reference

- An optional Symbol Value
- Optional Graphics

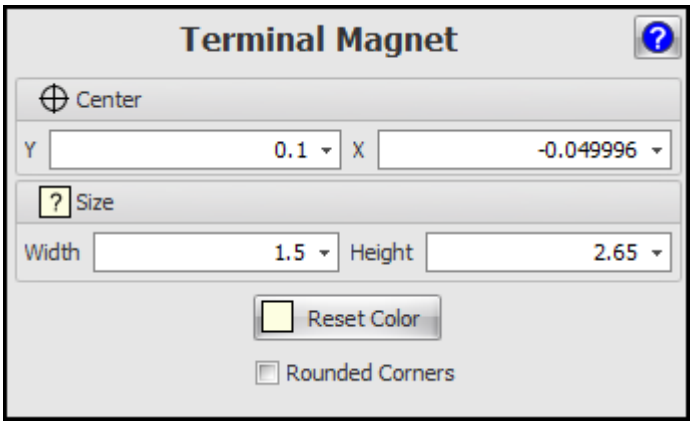


Symbol with border



Symbol without border

1.2.5.11.1 Symbol Terminal Magnet Editor



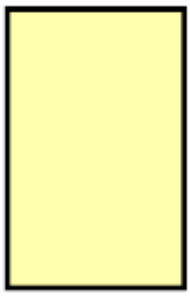
1.2.5.11.2 Symbol Borders (Terminal Magnets)

A symbol border is like a rectangular bar magnet where symbol terminals stick to the sides. This is a great aid in laying out and moving symbol terminals around a rectangular shape. It also facilitates the reorganization of terminals based either on pin names or symbol names. This reorganization is done by right clicking on the symbol's magnet and selecting the appropriate reorganize command from the pop-up context menu.

Using the magnet metaphor you can regard the terminal magnet (symbol border) as a magnetic monopole, say a north pole, [yes, finally one has been discovered]. Now you can regard symbol terminals as regular magnets with the end that connects to the terminal magnet (symbol border) being the South Pole and the end that faces away from the terminal magnet being the North Pole. The point on the terminal that is furthest away from the terminal magnet is the electrical connection point. You can regard point and dot terminals as zero length magnets.

It's a bit like when you pick up iron filings with magnetic bar, all the filings stick out as spikes.

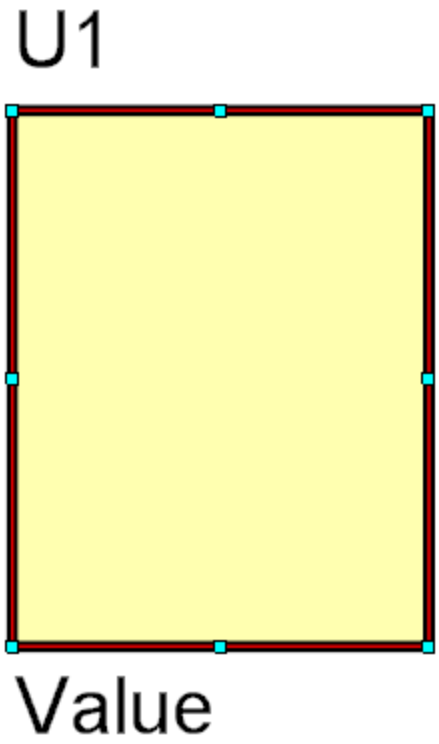
A terminal magnet, shown below, serves as anchor points for symbol terminals



Terminal Magnet

1.2.5.11.2.1 Editing Symbol Borders (Terminal Magnets)

To edit a terminal magnet first select it. If the terminal magnet is on a schematic (project) then you will need to use sub-pick to select it.



A selected terminal magnet

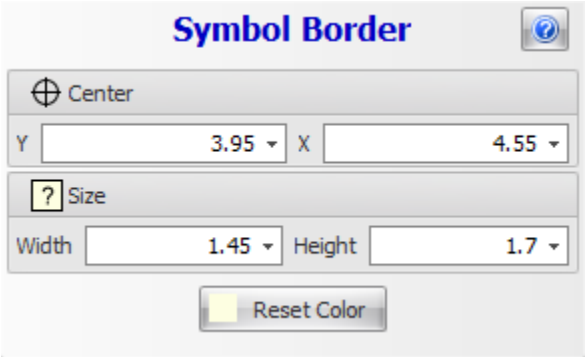
Drag either of the border's corner manipulators( ) to change the corners or ends.

Drag any scale manipulator to re-size the border.

Drag the border to move it.

Terminal Magnet Properties Dialog

You can also edit the parameters of the border using its properties panel as shown below. To view the properties panel first select it.Then ***right-click*** on it, then select Properties Panel from the context menu that opens.



Terminal Magnet Properties Dialog

X

The X coordinate of the center of the border.

Y

The Y coordinate of the center of the border

Width

The width of the border.

Height

The height of the border

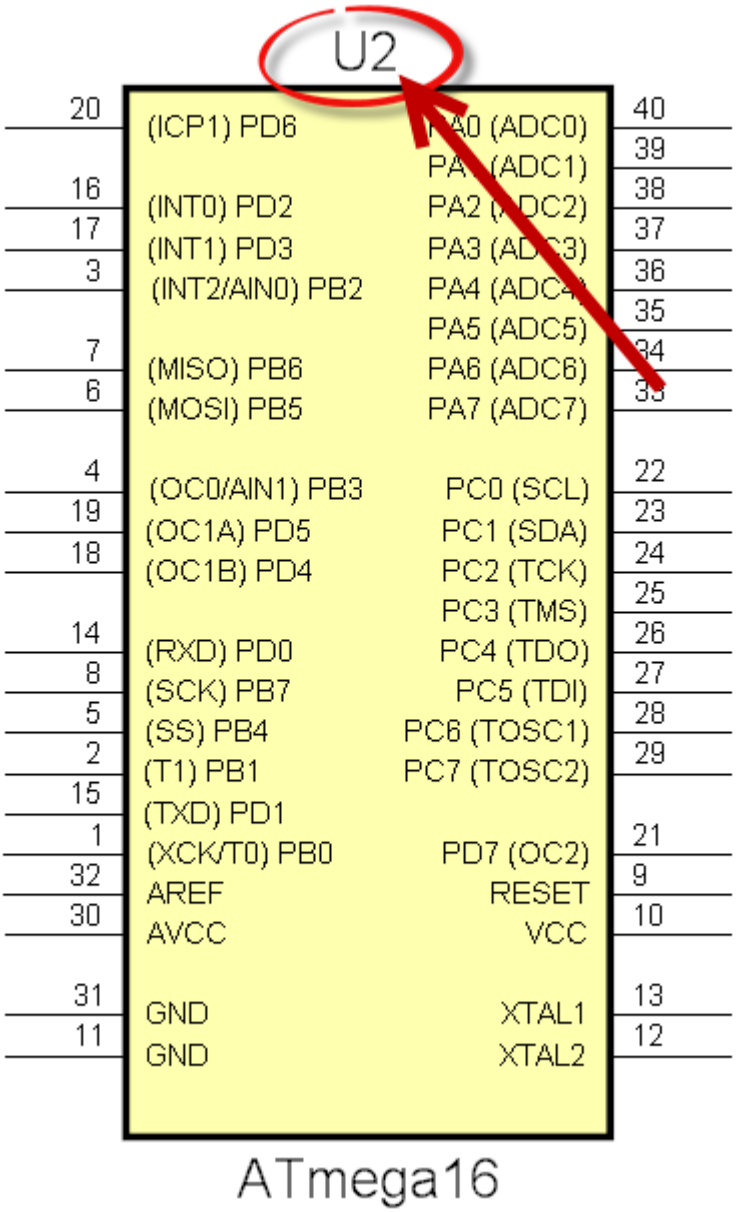
Reset Color

Click the 

button to reset the fill color.

1.2.5.11.3 Symbol References

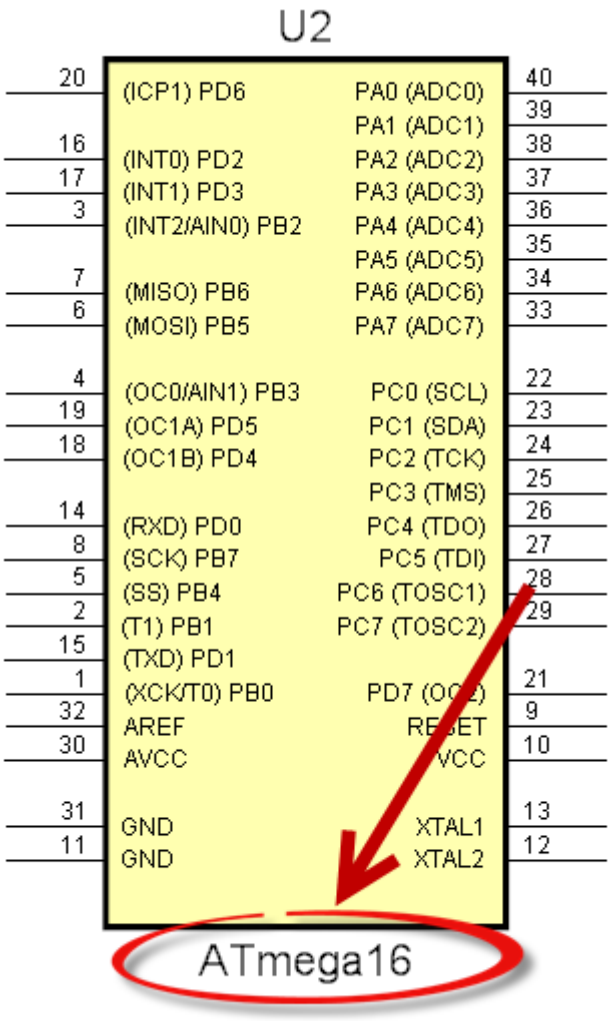
A symbol reference is a text marker to identify a part. A symbol reference is also known as a symbol/part designator.



Symbol Reference (pointed out by red arrow)

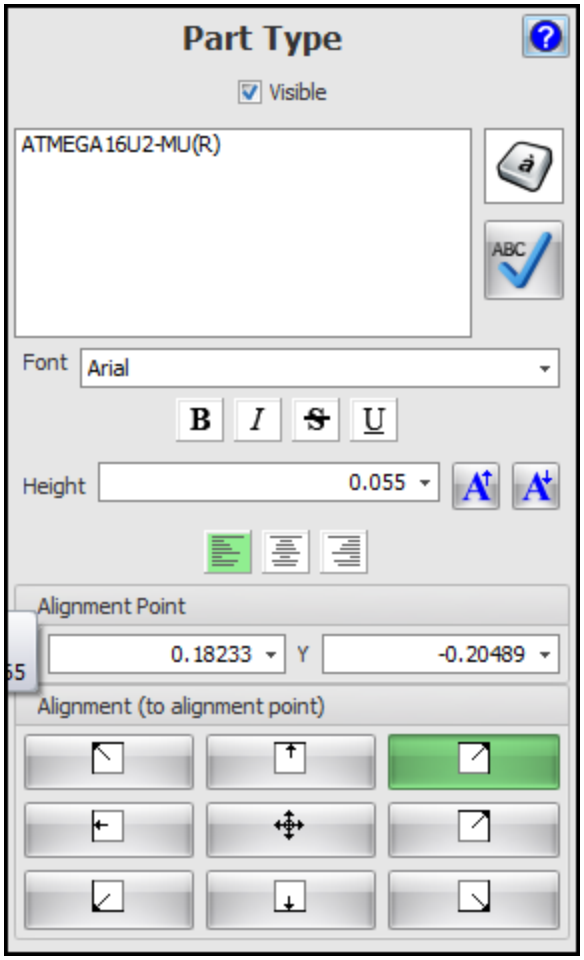
1.2.5.11.4 Symbol Values

A symbol value is a text marker to identify a part type.



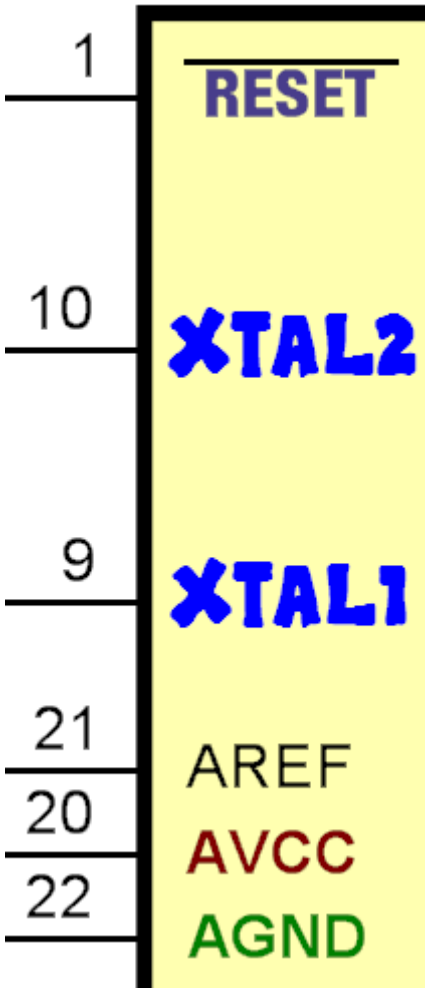
Symbol Value (pointed out by red arrow)

1.2.5.11.4.1 Symbol Value Editor



1.2.5.11.5 Symbol Terminals

A symbol terminal is an electrical connection point for a symbol. You can set the font, height and color for each terminal.



A symbol terminal is made up of

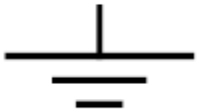
1. Lines/graphics (You can set the color for the text in a symbol terminal).
2. A pin number.
3. A pin value (Text description)
4. An electrical connection point that you attach to which you connect electrical wires.

Terminal can be invisible. (as in a ground symbol which is a collection of lines and an invisible terminal, symbol value and symbol reference.

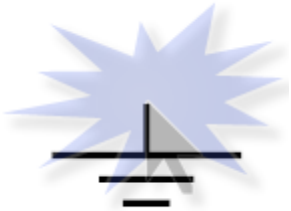
Moving the pin number or pin value

You can move the pin number or pin value (relative to the terminal connection point/terminal line) by holding down the mouse cursor over the pin number and dragging the mouse. The pin number will follow. If the pin number will not move then hold down the control key and drag (sub- picking). This displacement will be maintained when you drag the terminal around the terminal magnet (symbol border).

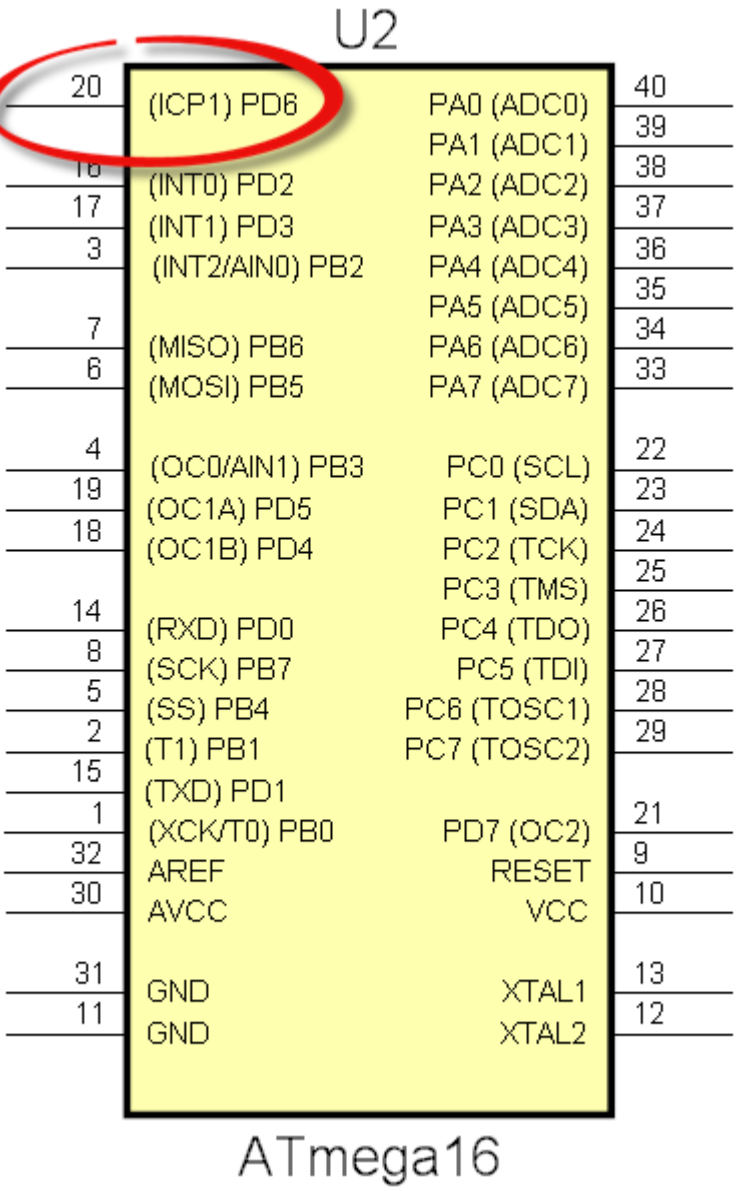
Pin numbers are actually be text and can contain any character including Unicode characters. Pin values/names again are text and can actually be multi-line text. So a pin value can extend over more than one line.



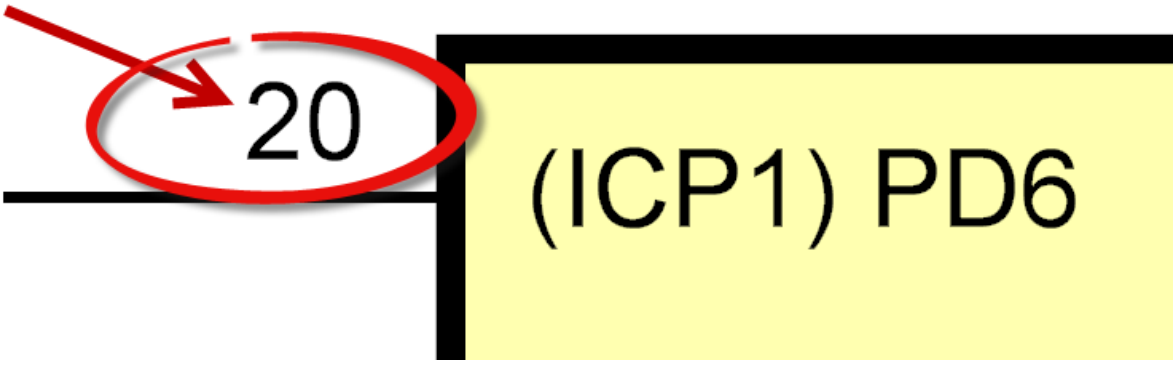
Ground Symbol



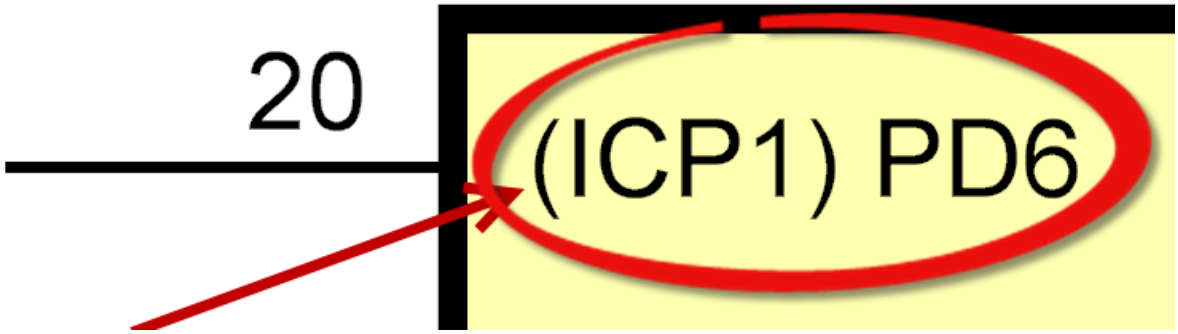
Connection Point



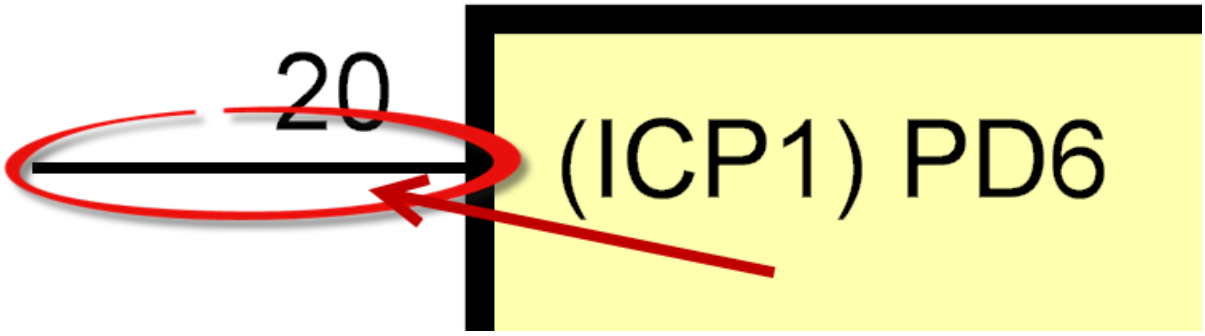
Symbol Terminal



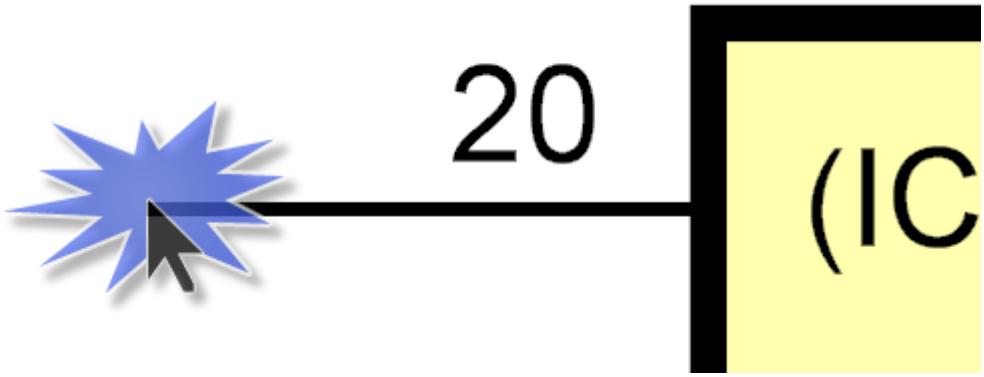
Pin number (Drag to move relative to the connection point)



Pin Value (Drag to move relative to the connection point)



Line graphics

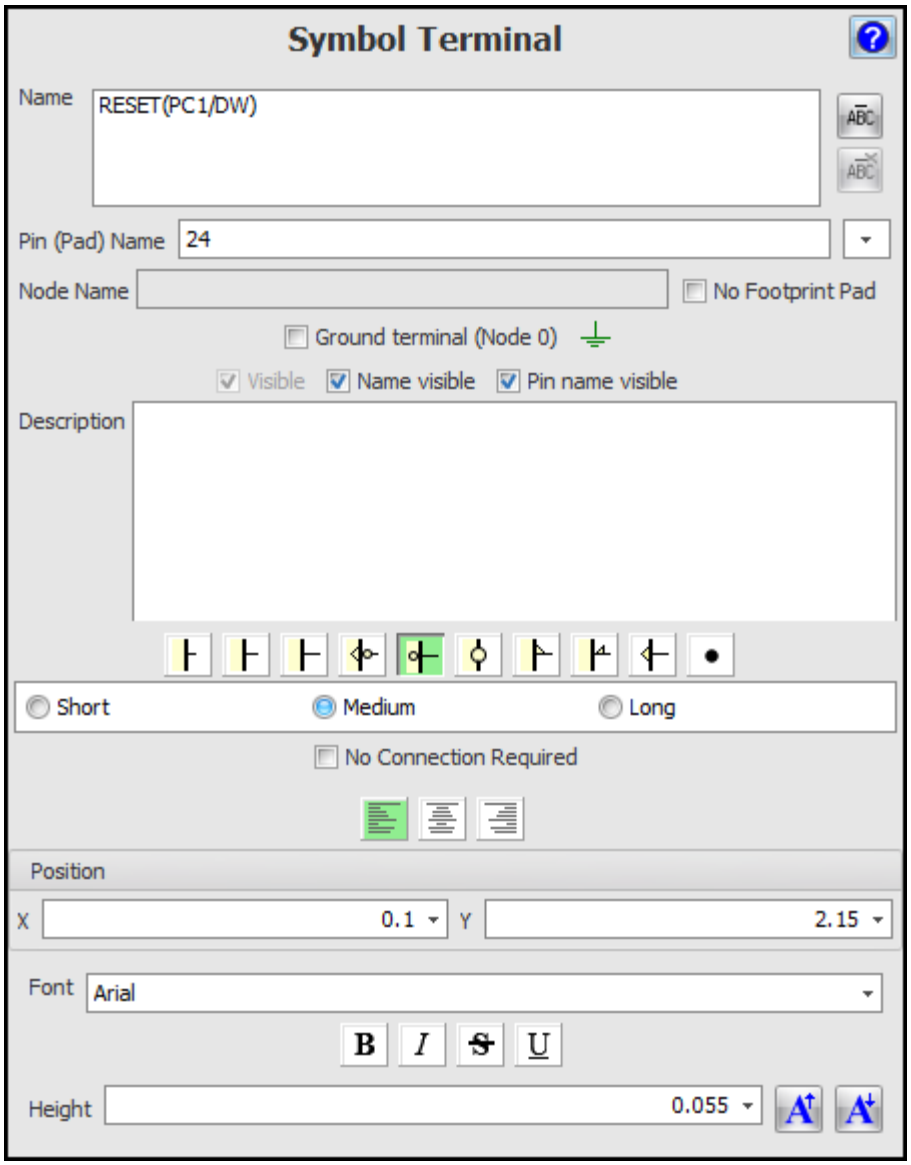


Electrical connection point

1.2.5.11.5.1 Terminal Magnets

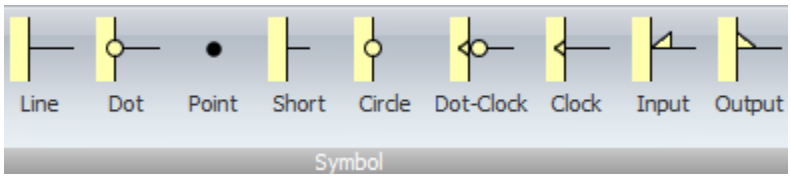
Enter topic text here.

1.2.5.11.5.2 Symbol Terminal Editor



1.2.5.11.5.3 Adding Terminals

To add a terminal click on one of buttons in the **Add→Symbol** button group.



1.2.5.11.5.4 Terminal Graphics

Coming soon...

1.2.5.11.5.5 Terminal Names

Coming soon...

1.2.5.11.5.6 Terminal Pin Names/Numbers

Coming soon...

1.2.5.11.5.7 Terminal Connection Points

Coming soon...

1.2.5.11.5.8 EditingTerminals

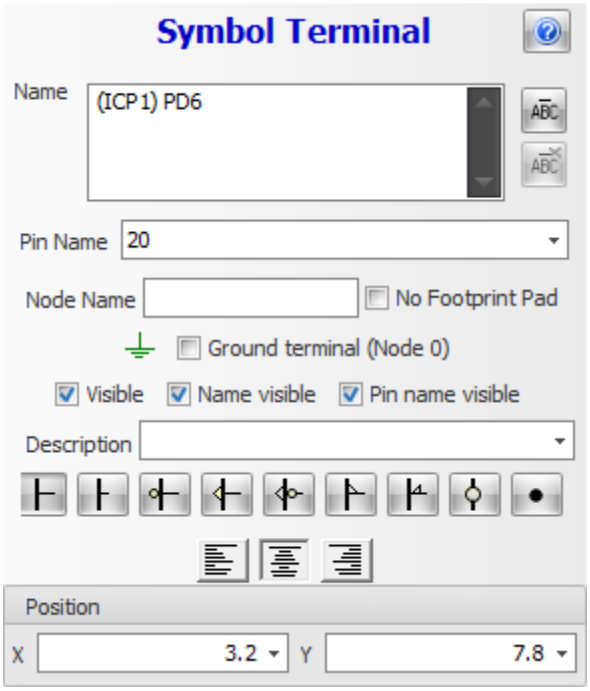
To edit a terminal you can drag it.

If a terminal magnet is present it will drag around the edges of the terminal magnet.

Double-click on it to rename it.

Terminal Rename Dialog

You can also set its parameters using the Symbol Terminal properties dialog.



Symbol Terminal properties dialog

1.2.5.11.6 Editing Symbols in Schematics

You can reordering terminals in place and add graphics to symbols

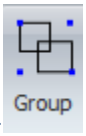
1.2.5.11.6.1 Reordering Terminals

When you place a part's symbol in a schematic you are not left with a fixed immutable symbol. With AutoTRAX DEX you can drag terminals, their textural names and the pin numbers to customize the symbol to how you to your liking. This is a powerful feature that helps you to keep tidy schematics.

1.2.5.11.6.2 Adding Graphics to Symbols

To add graphics to a symbol in a schematic:

Select the symbol and the graphics to add.

Click the **Edit→Group→** 

button.

The graphics will then be added to the internal data for the Symbol.

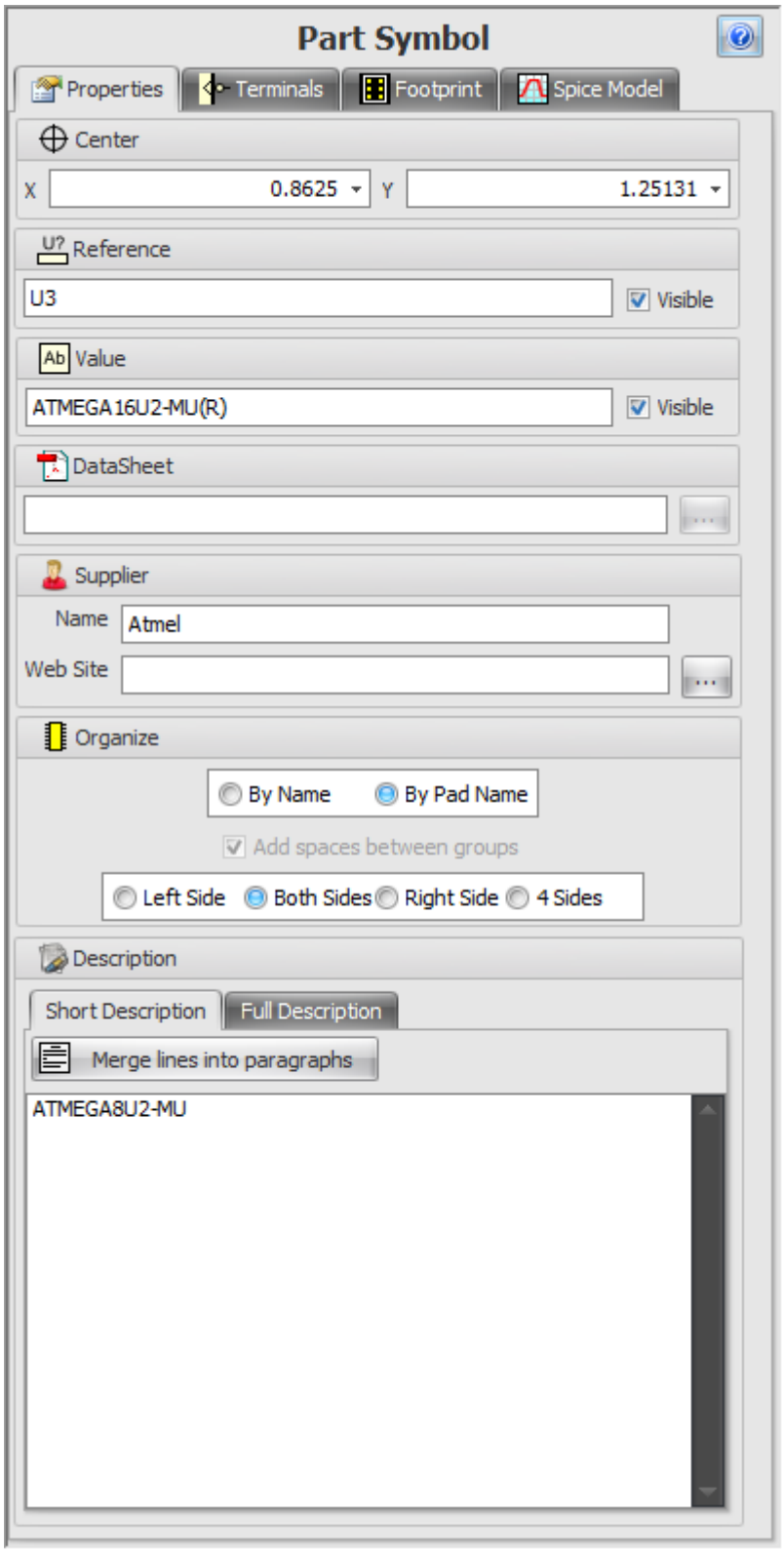
1.2.5.11.6.3 Symbol Editor

The properties editor for selected symbols is shown below.

It consists of 4 tabs: Properties, Terminals,Footprint,and Spice model.

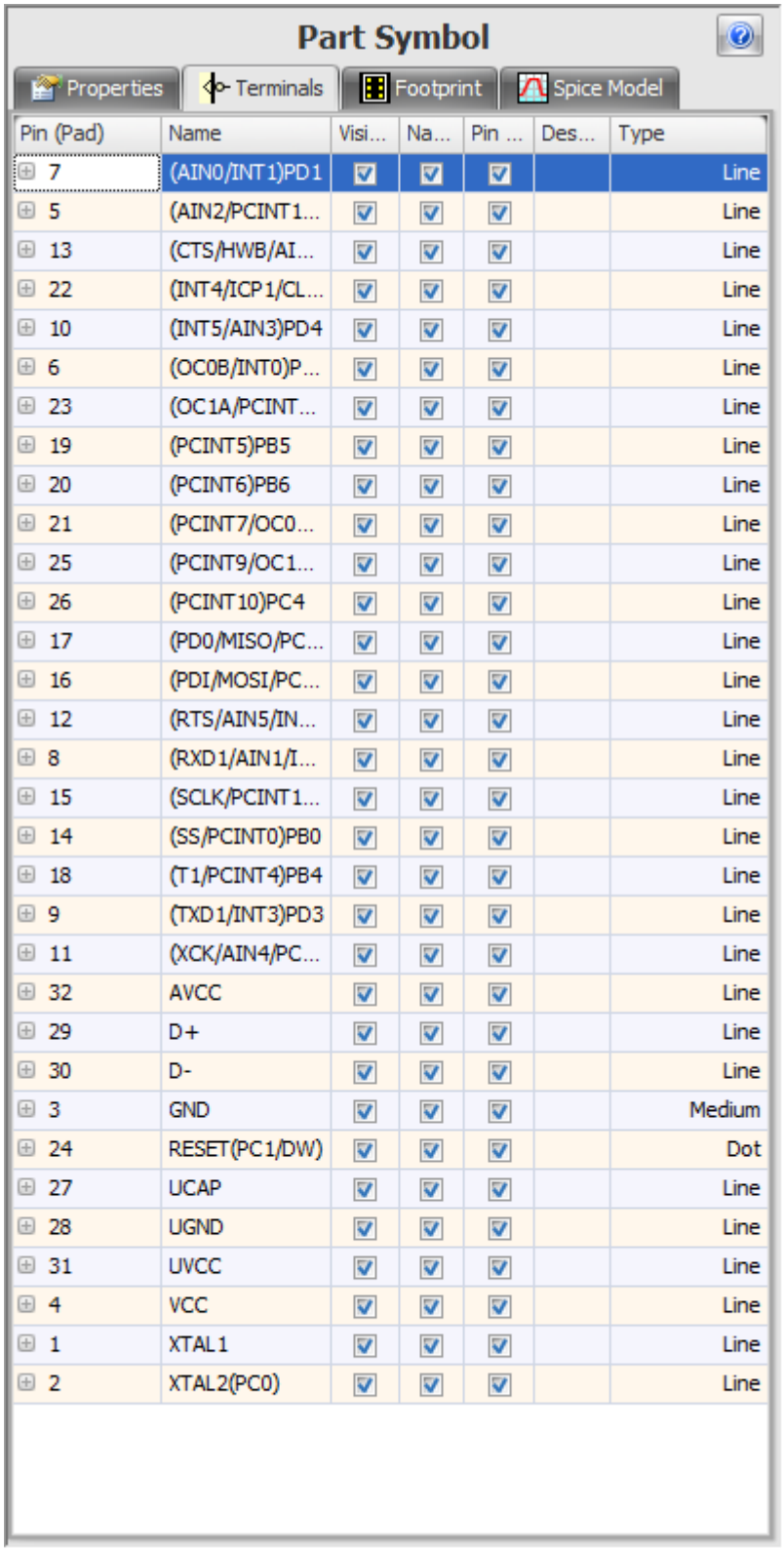
The Properties Tab

The Properties Tab lets you set general properties for the symbol/part.



The Terminals Tab

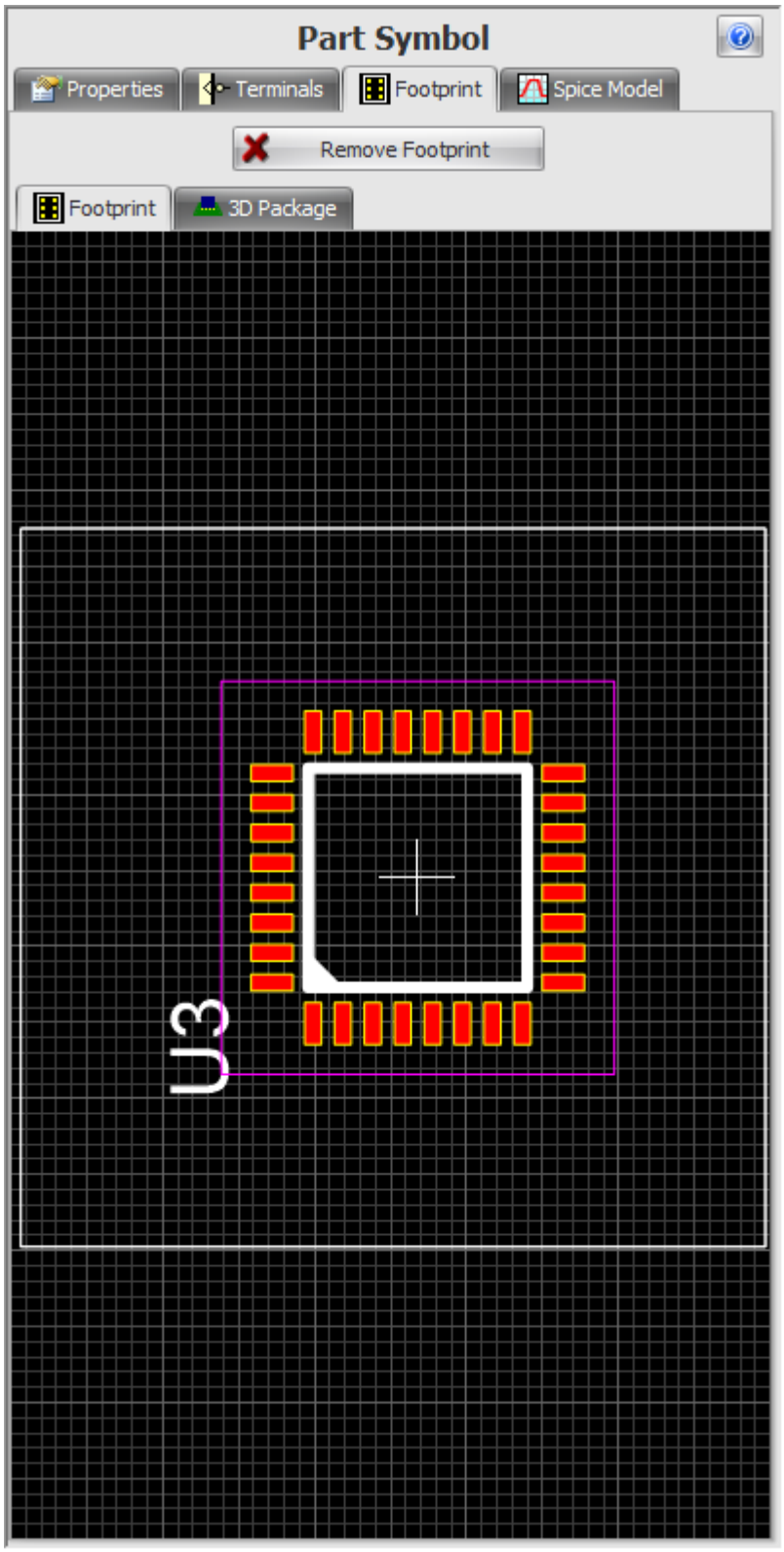
The Terminals Tab lists all the symbol terminals and their properties.



Pin (Pad)	Name	Visi...	Na...	Pin ...	Des...	Type
7	(AIN0/INT1)PD1	☑	☑	☑		Line
5	(AIN2/PCINT1...	☑	☑	☑		Line
13	(CTS/HWB/AI...	☑	☑	☑		Line
22	(INT4/ICP1/CL...	☑	☑	☑		Line
10	(INT5/AIN3)PD4	☑	☑	☑		Line
6	(OC0B/INT0)P...	☑	☑	☑		Line
23	(OC1A/PCINT...	☑	☑	☑		Line
19	(PCINT5)PB5	☑	☑	☑		Line
20	(PCINT6)PB6	☑	☑	☑		Line
21	(PCINT7/OC0...	☑	☑	☑		Line
25	(PCINT9/OC1...	☑	☑	☑		Line
26	(PCINT10)PC4	☑	☑	☑		Line
17	(PD0/MISO/PC...	☑	☑	☑		Line
16	(PDI/MOSI/PC...	☑	☑	☑		Line
12	(RTS/AIN5/IN...	☑	☑	☑		Line
8	(RXD1/AIN1/I...	☑	☑	☑		Line
15	(SCLK/PCINT1...	☑	☑	☑		Line
14	(SS/PCINT0)PB0	☑	☑	☑		Line
18	(T1/PCINT4)PB4	☑	☑	☑		Line
9	(TXD1/INT3)PD3	☑	☑	☑		Line
11	(XCK/AIN4/PC...	☑	☑	☑		Line
32	AVCC	☑	☑	☑		Line
29	D+	☑	☑	☑		Line
30	D-	☑	☑	☑		Line
3	GND	☑	☑	☑		Medium
24	RESET(PC1/DW)	☑	☑	☑		Dot
27	UCAP	☑	☑	☑		Line
28	UGND	☑	☑	☑		Line
31	UVCC	☑	☑	☑		Line
4	VCC	☑	☑	☑		Line
1	XTAL1	☑	☑	☑		Line
2	XTAL2(PC0)	☑	☑	☑		Line

The Footprint Tab

The Footprint Tab displays the properties of the footprint for the selected part.



The Spice Model Tab

The Spice Model Tab displays the editable Spice simulation model for the part.

1.2.5.11.7 Multiple Symbols

Parts have one more symbols with each symbol representing a different aspect of the parts behavior.

1.2.5.11.8 Splitting Symbols

Enter topic text here.

1.2.5.11.9 Symbol Graphics

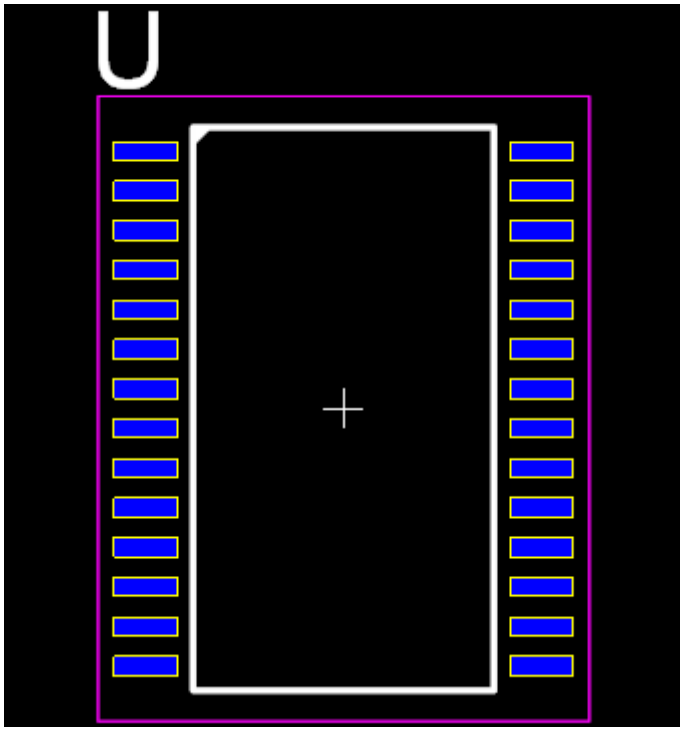
Enter topic text here.

1.2.5.11.10 Capturing Terminal Names From a Datasheet

Capturing Schematic Symbol Pin Details from a Datasheet

1.2.5.12 Footprints

Footprints are the copper and silkscreen patterns (also called **Land Patterns**) that define the part on a PCB.



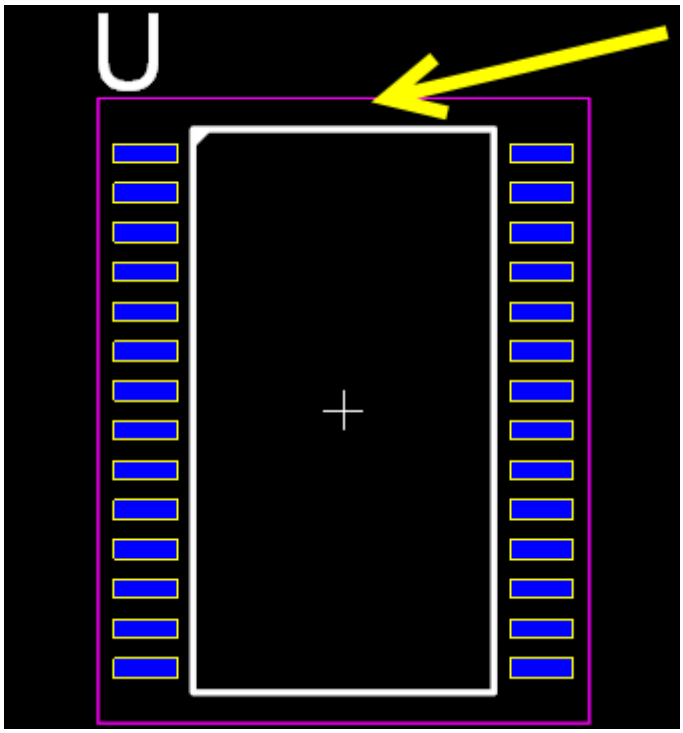
Footprint

A footprint consists of:

- Pads
- The Footprint Reference
- A Courtyard
- A Silkscreen Rectangle
- The Footprint Placement Point

1.2.5.12.1 Courtyards

Courtyards are rectangles that define the area used by a part. They have no electrical significance and can be omitted to save costs. They are used during the auto-layout. Pick and Place machines do not need them.



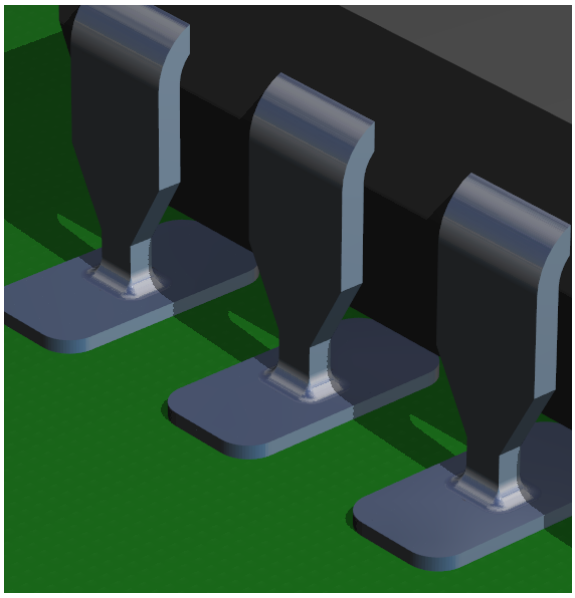
Courtyard (magenta rectangle)

1.2.5.12.1.1 Editing Courtyards

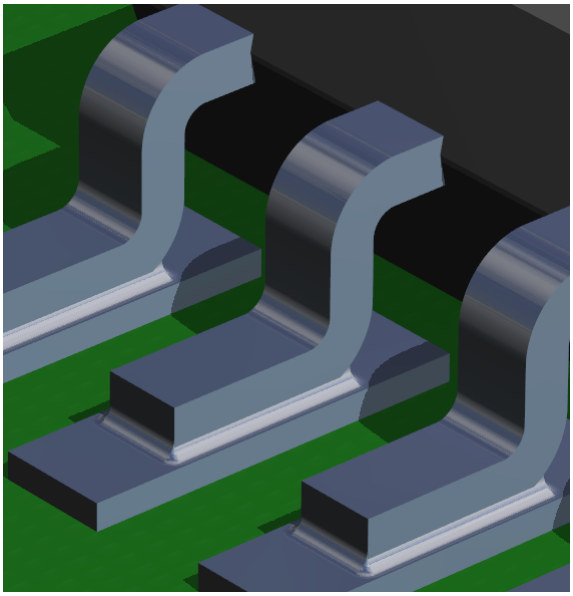
You edit courtyards like editing keep pout regions.

1.2.5.12.2 Pads

Electrical parts are 'glued' on the PCB using solder between the part's electrical terminals and copper pads on the PCB. The solder is usually the only way the part is mechanically affixed to the PCB.

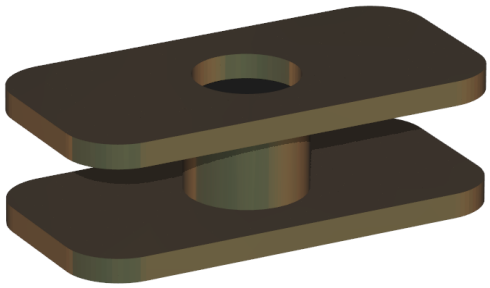


TPH Pads Soldered (glued) to PCB

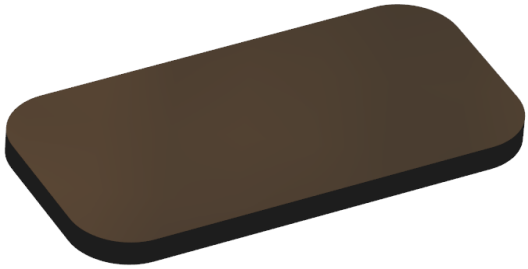


SMT Pads Soldered (glued) to PCB

Pad are copper areas on the top/bottom or both sides of the PCB.



TPH Pad (PCB invisible)

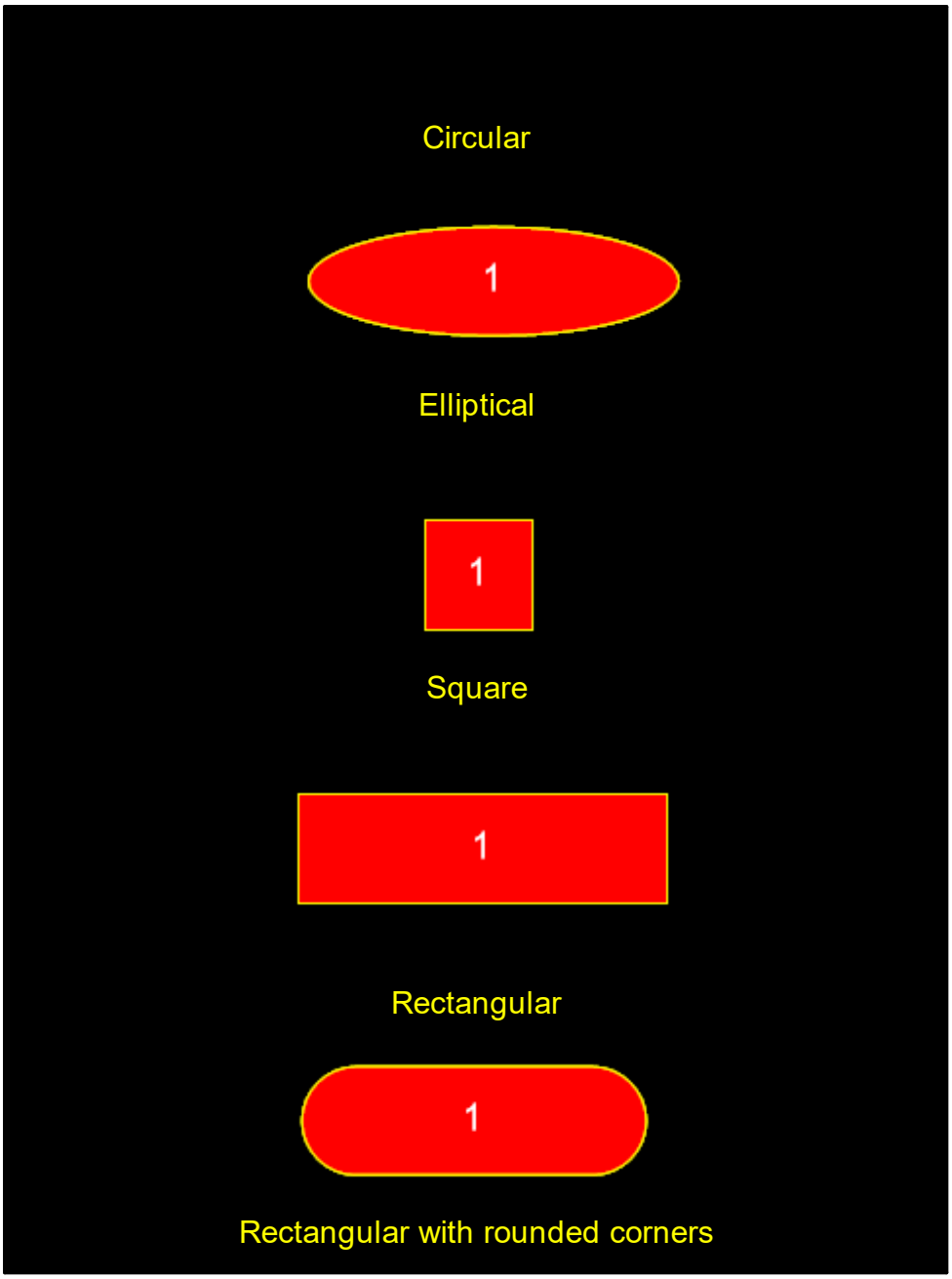


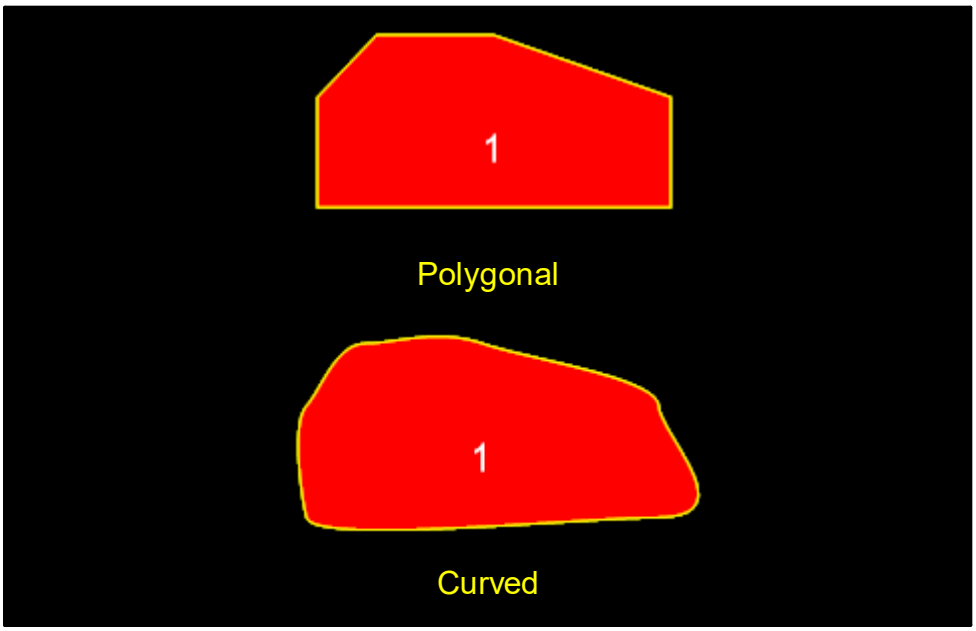
SMT Pad (PCB invisible)

If pads are on both sides then there is a hole drilled trough both the top and the bottom side of PCB and is usually plated with copper to provide an electrical connection between both sides.

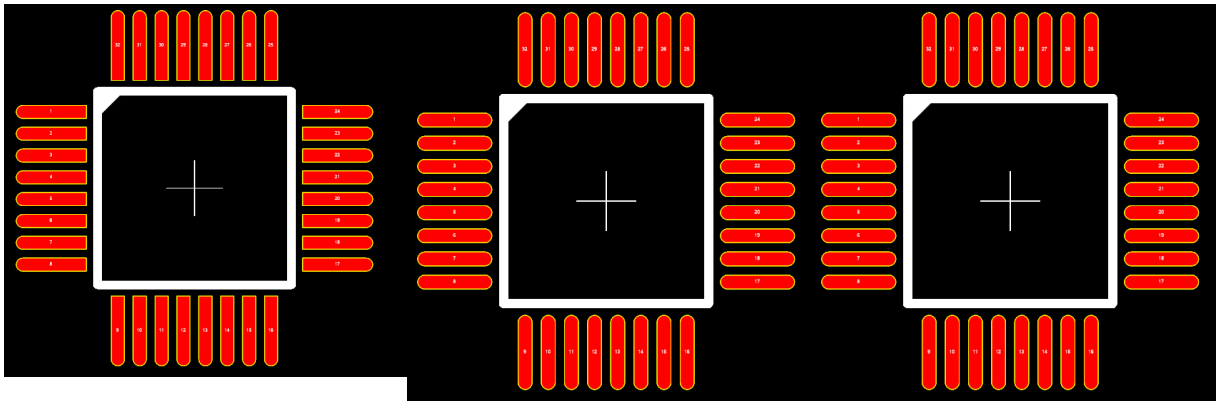
1.2.5.12.2.1 Pad Shapes

Pads can be circular, elliptical, rectangular (with optional rounded corners) polygonal or a complex curve.



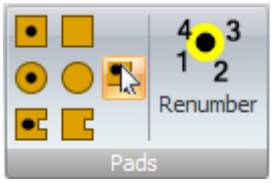


In addition Pads in rectangular parametric footprints can have rounded corners set to sides.



1.2.5.12.2.2 Creating Pads

For parametric parts, pads are automatically created. However, once you have fully created a parametric part you can add additional pads.

You can add a pad by clicking a pad in the 

group in the Add menu.

Click  to toggle between and

TPH Pads

Click  to add a rectangular TPH pad.

Click  to add a circular/elliptical TPH pad.

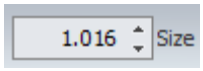
Click  to add a polygonal/curved TPH Pad.

SMT Pads

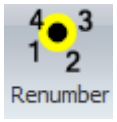
Click  to add a rectangular SMT pad.

Click  to add a circular/elliptical SMT pad.

Click  to add a polygonal/curved SMT Pad.

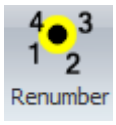
If 

is visible it sets the size of pads added. So when pads are added they have a fixed size. If it is not visible then you define the pad sized by dragging the mouse to define the pad size.



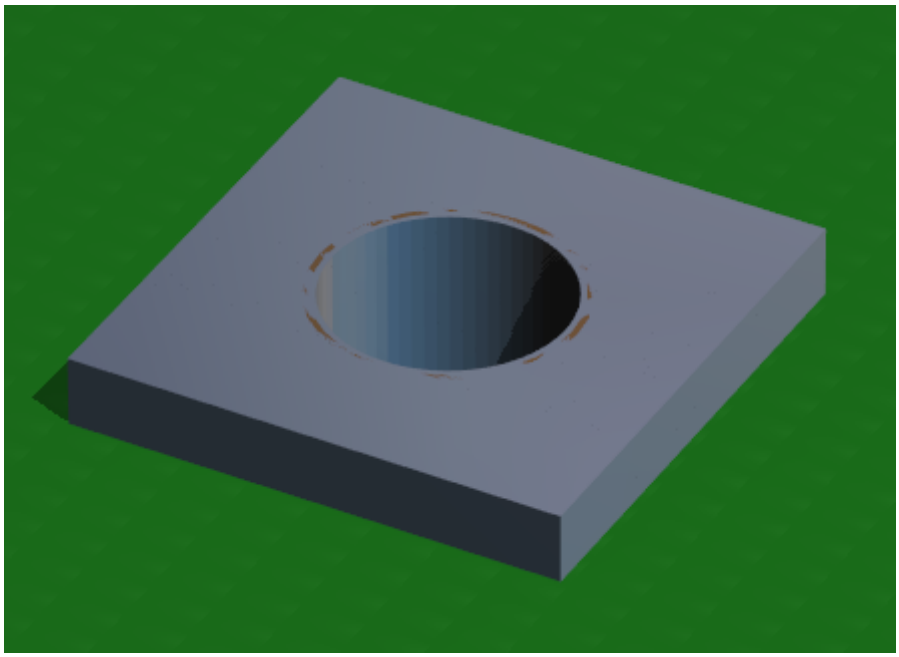
is used for setting pad number.

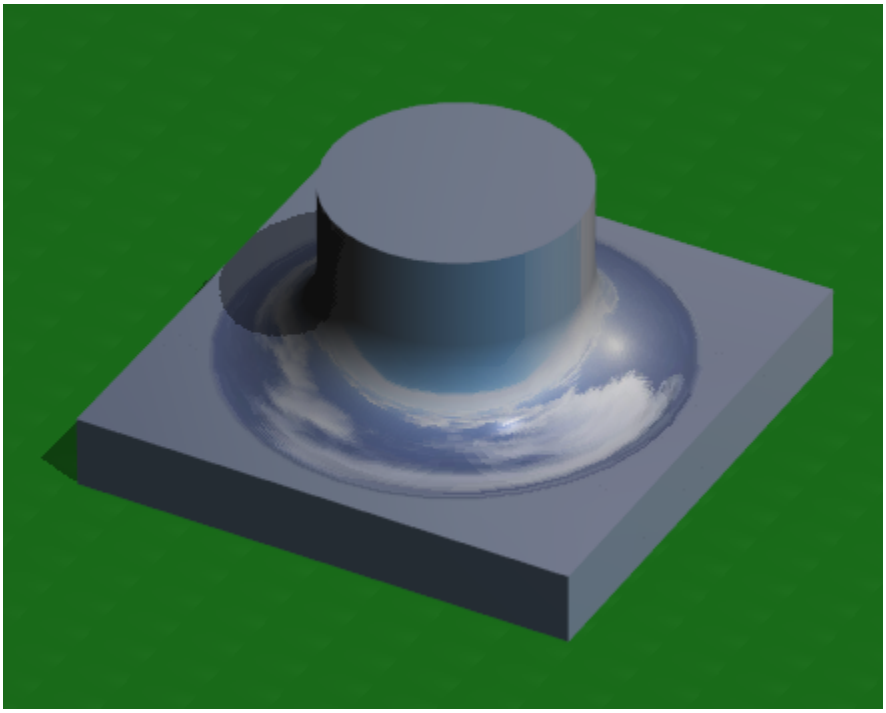
1.2.5.12.2.3 Setting Pad Numbers

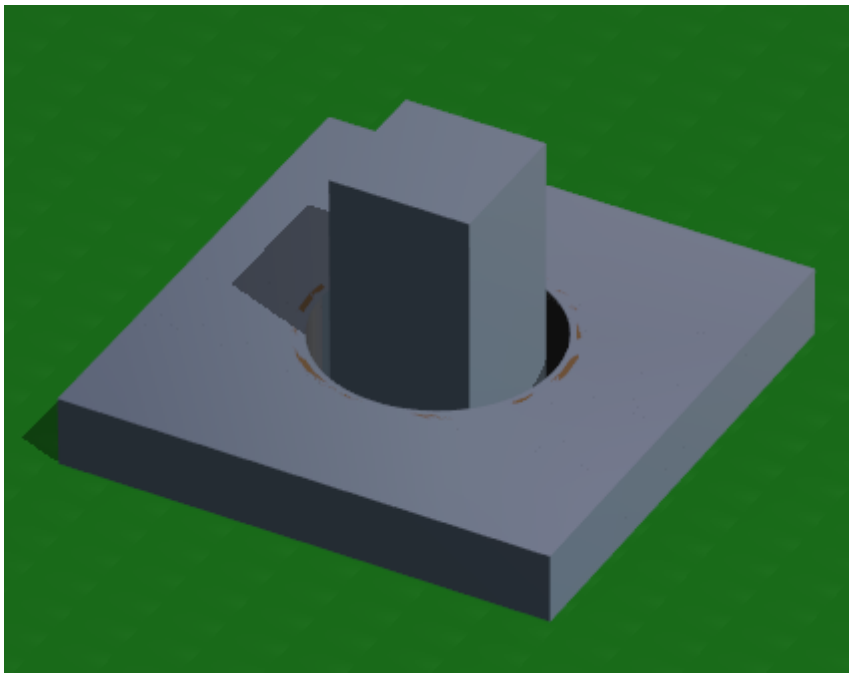
Click 

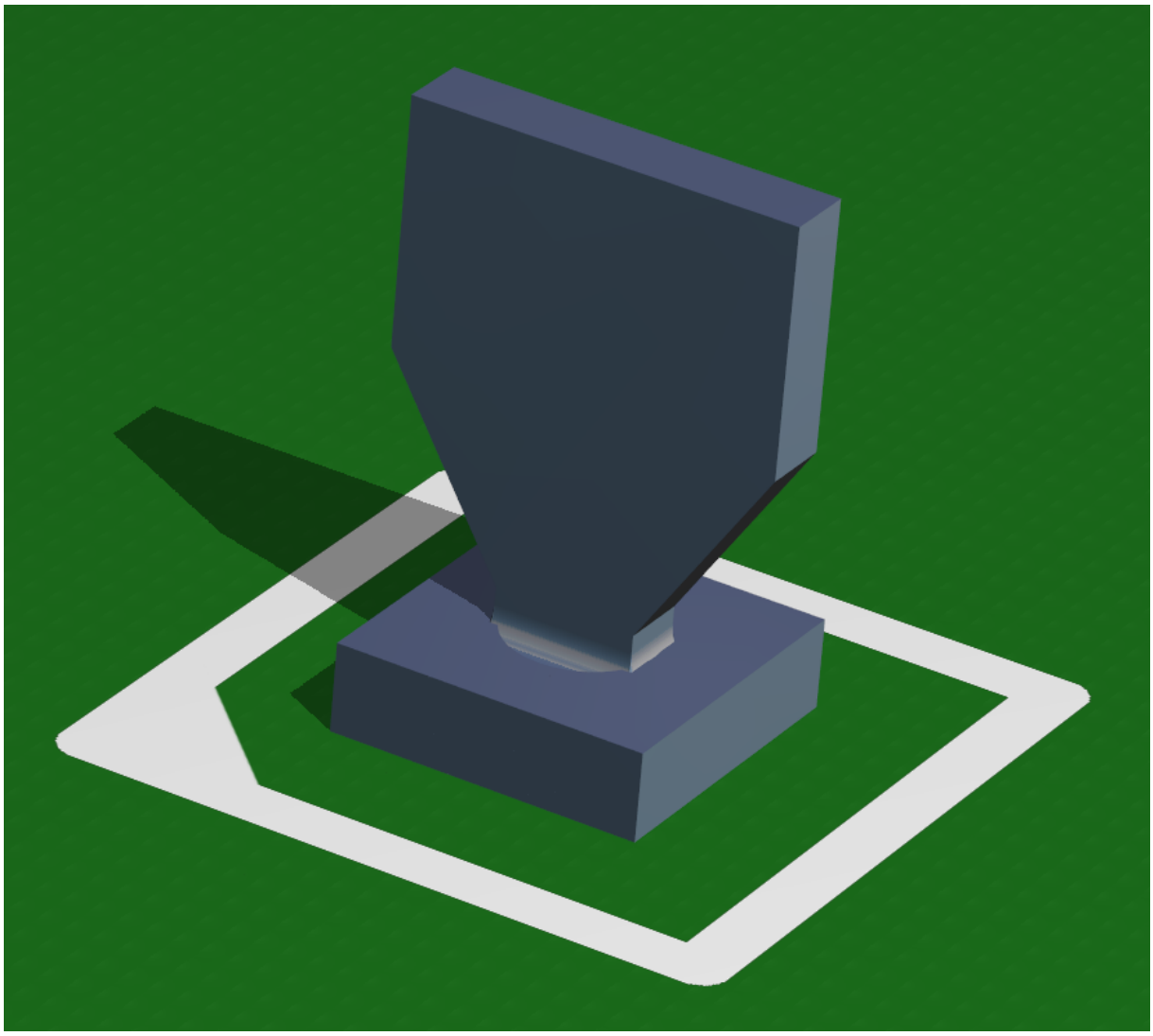
in the Add menu to set pad numbers. You would normally do this after you have added all your pads.

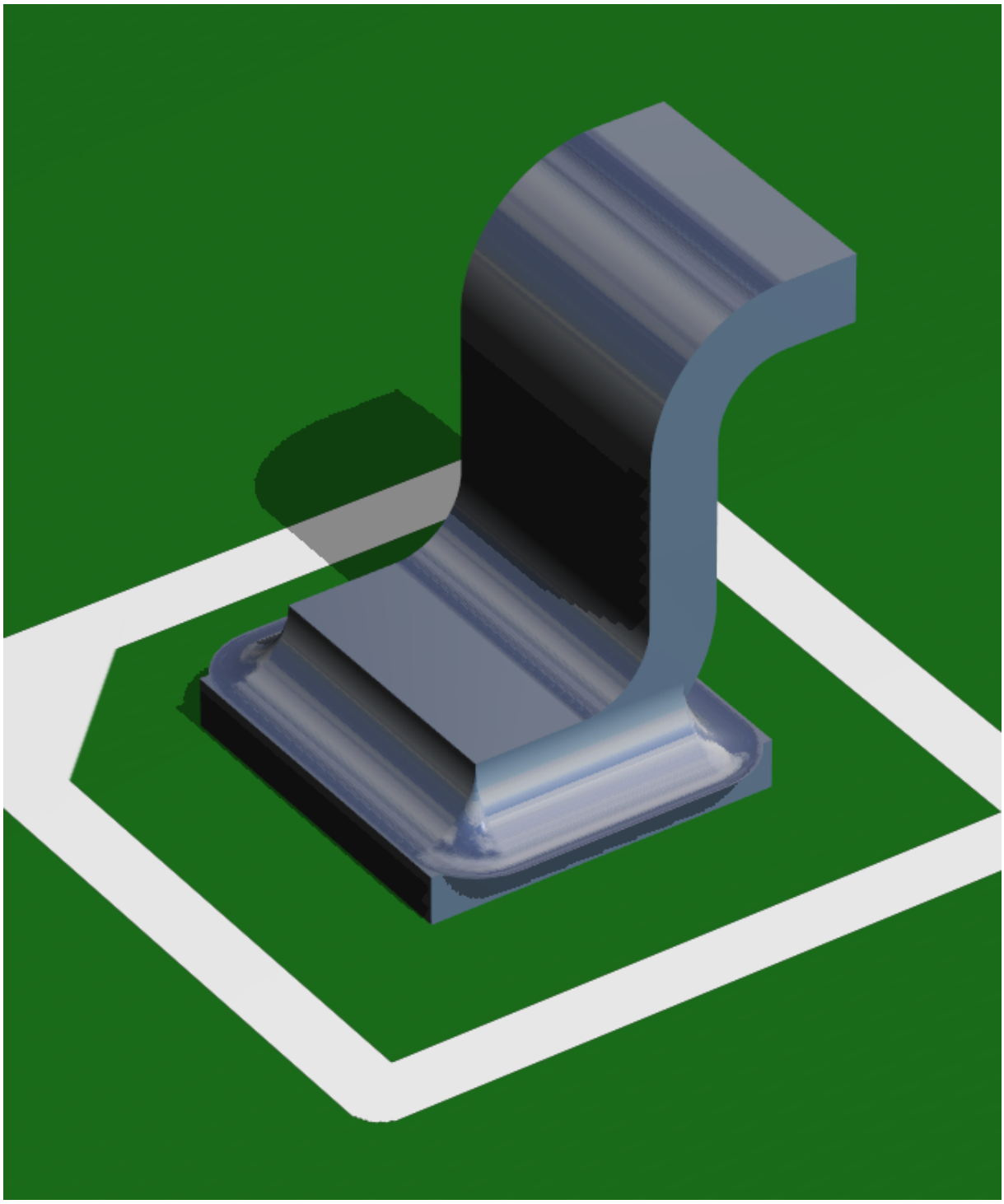
1.2.5.12.2.4 Automatic 3D Pins









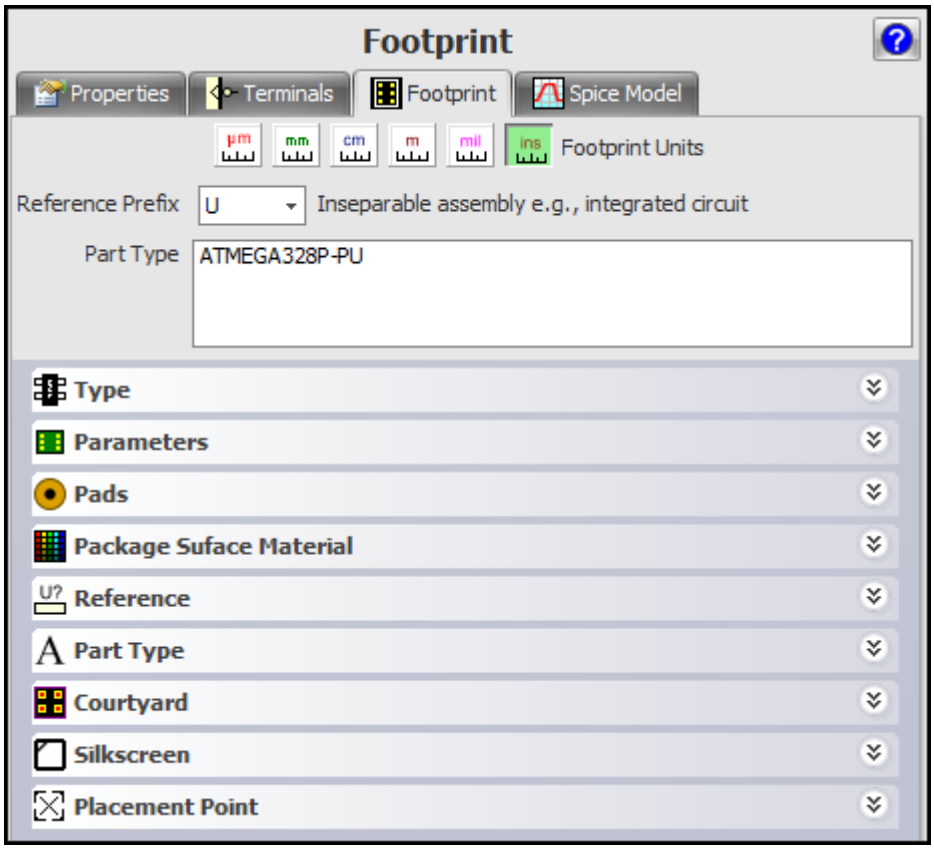


1.2.5.12.2.5 Gold Plating

You can optionally add gold plating to all pads in a part or to individual pads. Gold plating is normally applied to edge connectors.

When you generate Gerber manufacturing files, top and bottom gold Gerber files will also be generated. No Gerber files will be generated if there are no pads with gold plating.

1.2.5.12.3 Footprint Editor



Type

The type of the part.

Parameters

The types parameters editor.

Pads

Reference

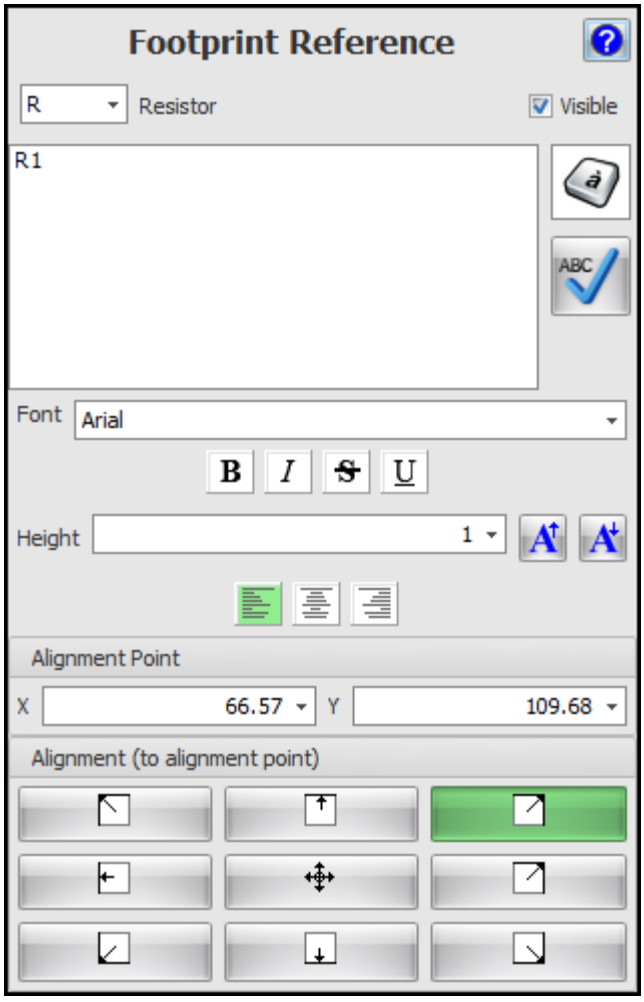
Part Type

Silkscreen

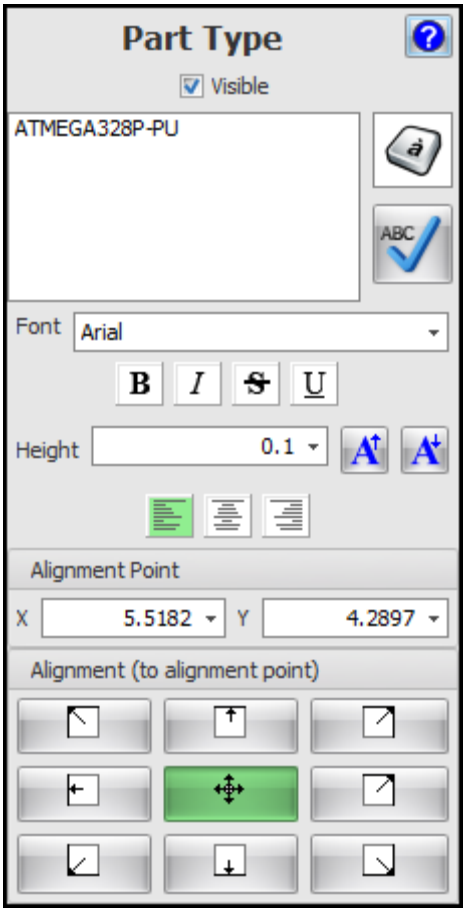
Placement

1. Footprint Editor
2. Footprint Reference Editor
3. Footprint Value Editor
4. Courtyard Editor
5. Silkscreen Editor
6. Placement Point Editor
7. Pad Editor

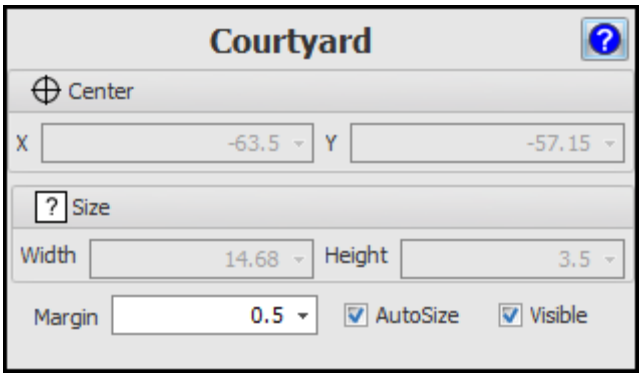
1.2.5.12.3.1 Footprint Reference Editor



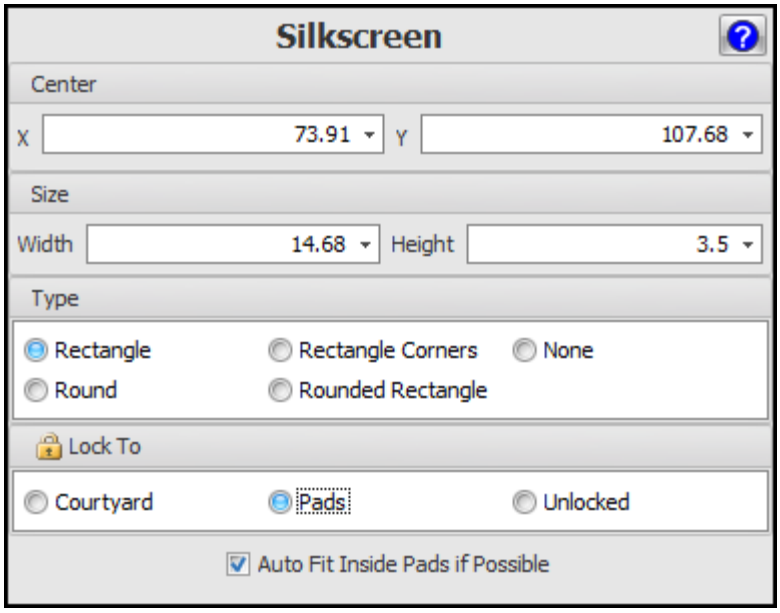
1.2.5.12.3.2 Footprint Value Editor



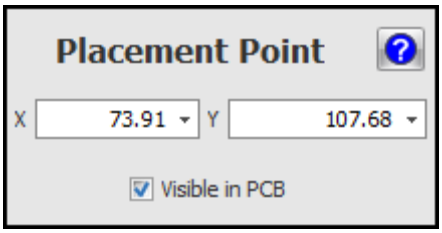
1.2.5.12.3.3 Courtyard Editor



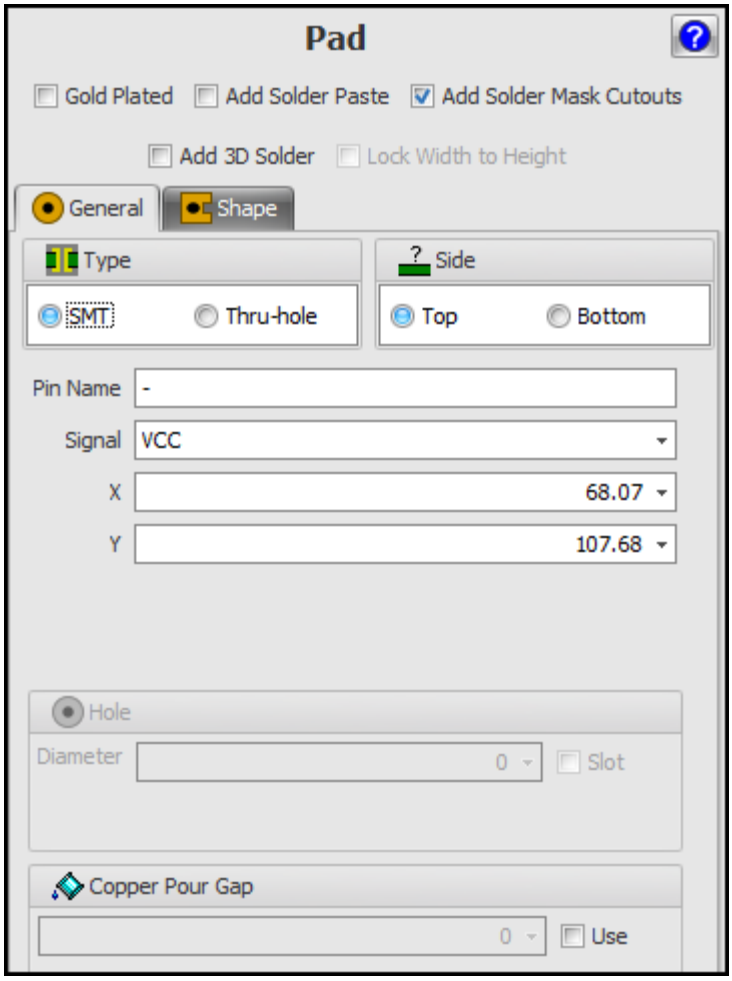
1.2.5.12.3.4 Silkscreen Editor

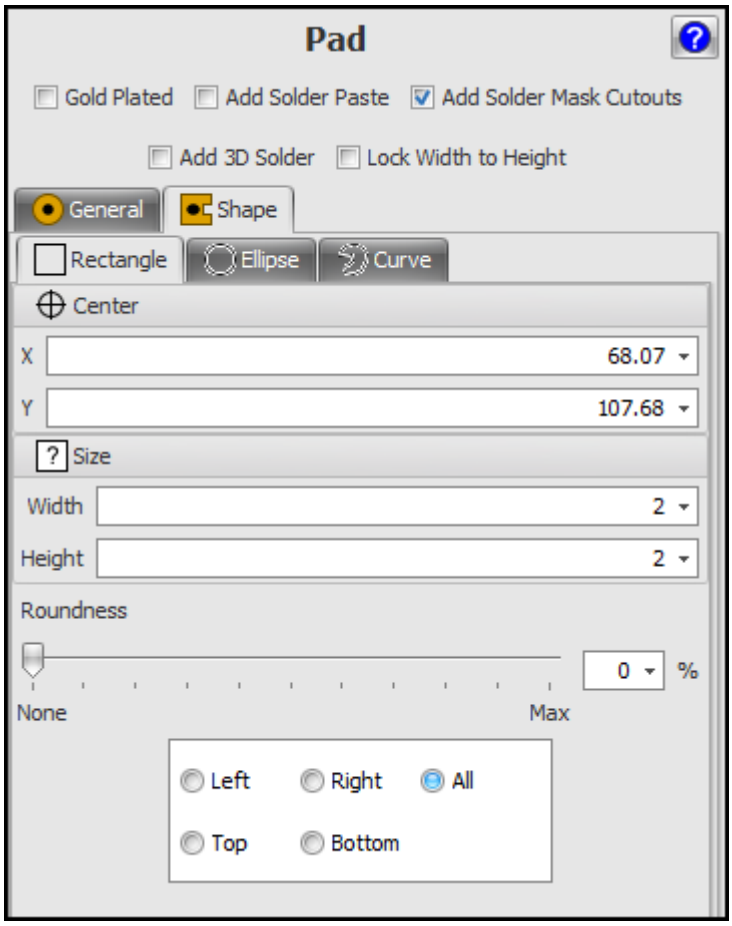


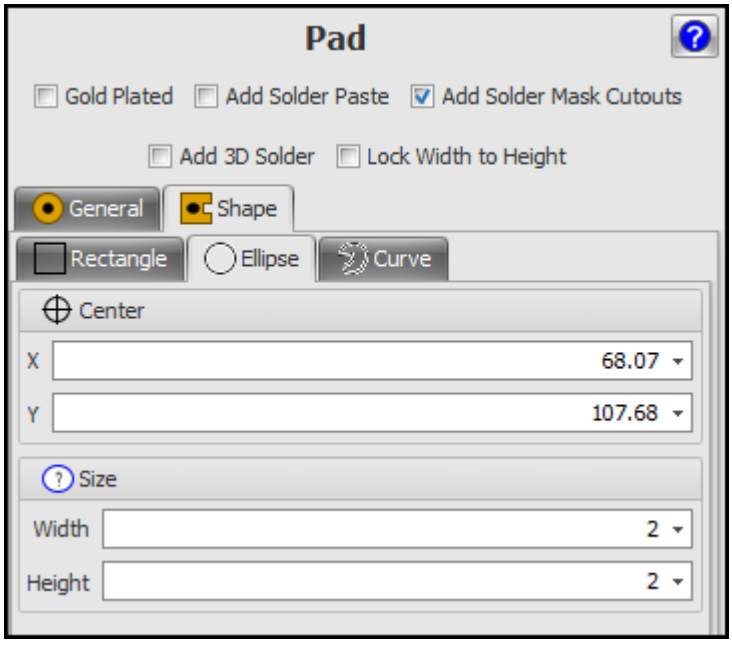
1.2.5.12.3.5 Placement Point Editor

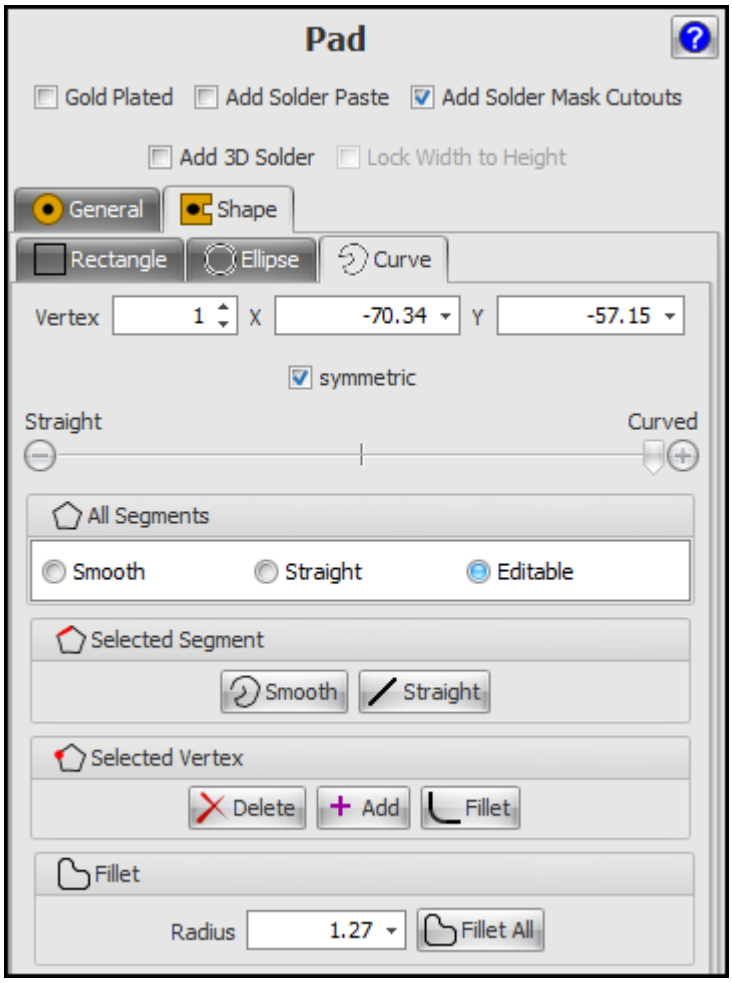


1.2.5.12.3.6 Pad Editor

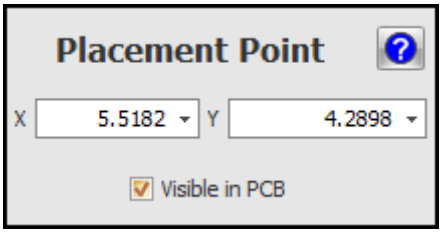








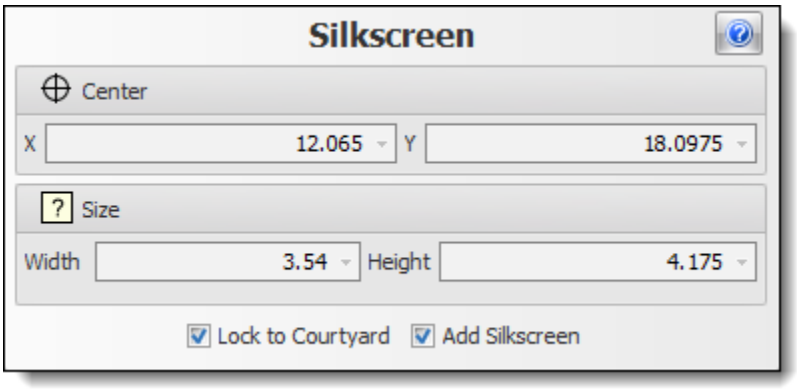
1.2.5.12.3.7 Placement Points



1.2.5.12.4 Silkscreens

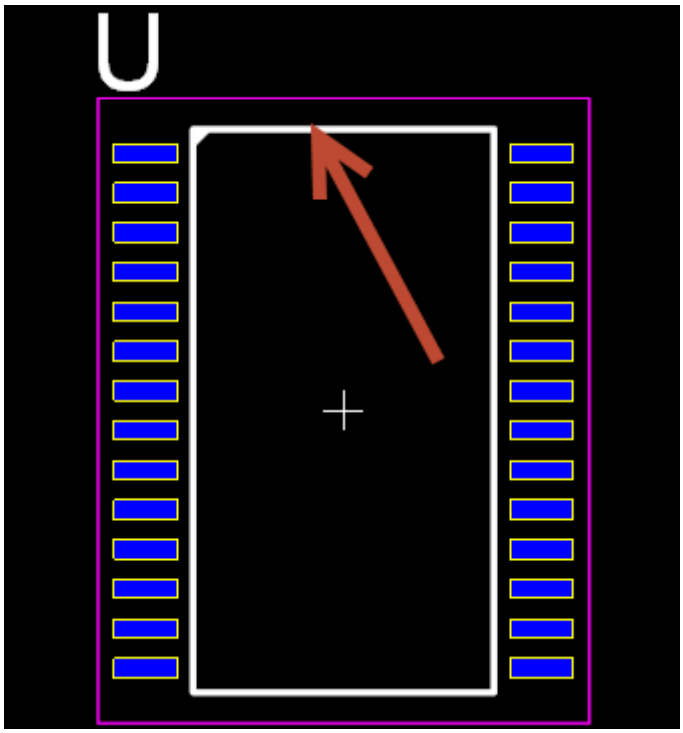
Silkscreens or artworks that are printed on the top and/or bottom of the PCB. The unknown electrical significance and are only for human consumption, mainly for use during manual assembly and for field service engineers. For cost sensitive PCBs the silkscreen will usually be omitted to say money.

Also a side-effect of silk screens when they are used to define part reference IDs is that the consume PCB area and can cause problems when you're trying to design a compact PCB.



1.2.5.12.4.1 Silkscreen Rectangles

Silkscreen rectangles are shapes drawn on the non-electrical silkscreen layer (usually white ink) that details for humans the shape of body of the part. They have no electrical significance and can be omitted to save costs. Pick and Place machines do not need them.

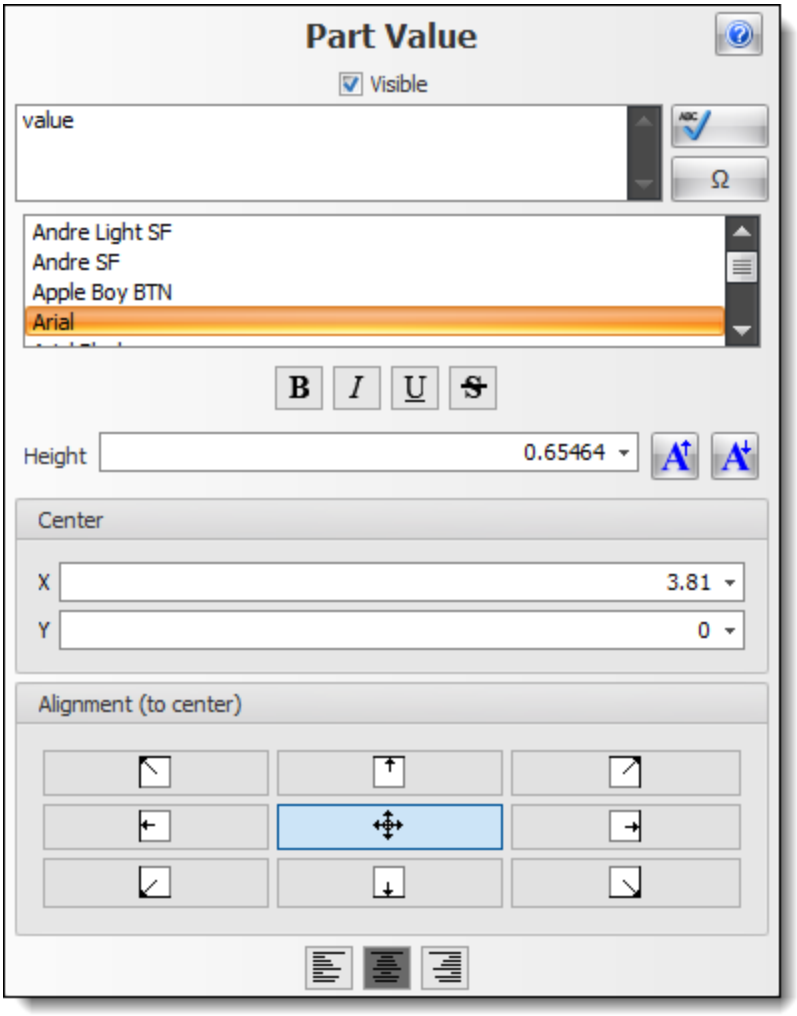


Silkscreen Rectangle (White)

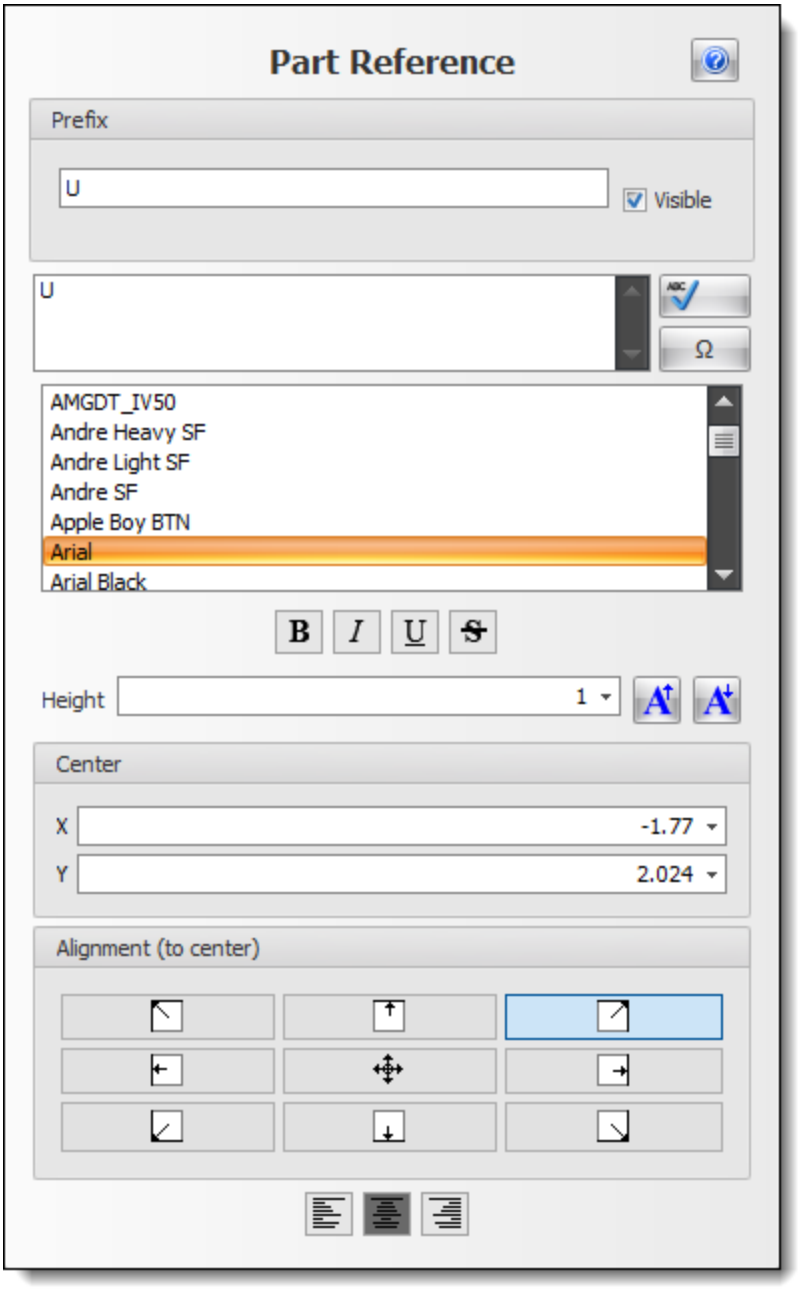
1.2.5.12.5 Footprint Part Values

Footprint values contain no electrical information and are usually not added to the final PCB.

They are often used by the part builder when it is automatically creating a 3-D part for the device. It will be used to assure the device type on the 3-D is a model.



1.2.5.12.6 Footprint Part Reference IDs



1.2.5.12.7 The Footprint Viewport

The footprint viewport show you a stylised view of the footprint or land pattern for your device. The footprint viewport is very similar to a PCB viewport but has characteristics that are relevant to PCB footprint design rally PCB viewport as characteristics relevant to laying out your parts and routing them together using electrical tracks.

The key electrical components of your footprint are clearly shown and differ quite significantly from an actual 3-D view.

Pads

Pads are shown as solid rectangles colored the same as a Top or bottom layer, or optionally coloured yellow. (You would have them colored yellow so you can clearly see what is a part in what is a normal copper area).

Pads are used as areas to glue the parts to the PCB. The glue being the electrical solder.

The placement point

Also other items are shown stylized, for instance the placement point which is used for pick and place. This is shown as a simple white on black cross. You move this cross to define the placement point to be used in pick and place.

Silkscreens and the part reference ID

You also see the silkscreen pattern. This consists of text that is the parts reference ID and other graphical objects usually used to define the placement of the part on the PCB.

Silkscreen patterns are for human consumption only and are not used electrically by the PCB. In fact, for cost conscious device,s you will not actually use a silkscreen and this will save you a lot of money. However, the silkscreen allows service personnel to locate parts on the device if they intend to repair the PCB.

The silkscreen is a real object that is actually placed on your final PCB like a placementpoint which you will never see on the PCB.

The courtyard rectangle

Another thing that you will see is the courtyard rectangle. This is a rectangular area that defines the area used by the part on your PCB and helps prevent overlapping of adjacent parts.

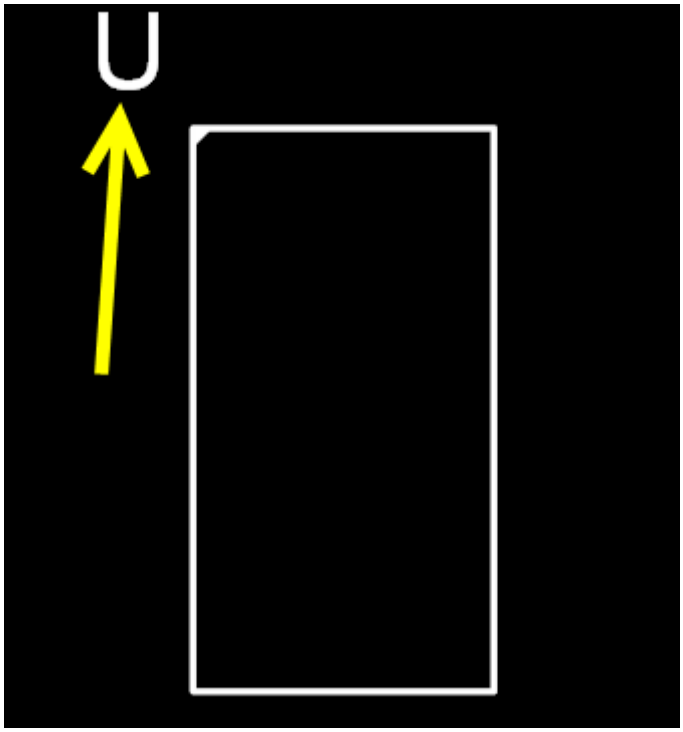
This view was in sharp contrast to the feed if you which shows a more photorealistic view of your PCB but did not show the abstract data. So you will see the footprint pads but you will not see the placement point for the courtyard rectangle.

Holes and cutouts

If you add any holes or cutouts to the footprint these are shown as black circles or the case of cutouts either circles ellipses, polygons, rectangles or curved black cutouts. The shape will have a colored line around it

1.2.5.12.8 The Footprint Reference

The footprint reference is a text marker on the silkscreen layer that identifies a specific part. They have no electrical significance and can be omitted to save costs. Pick and Place machines do not need them.



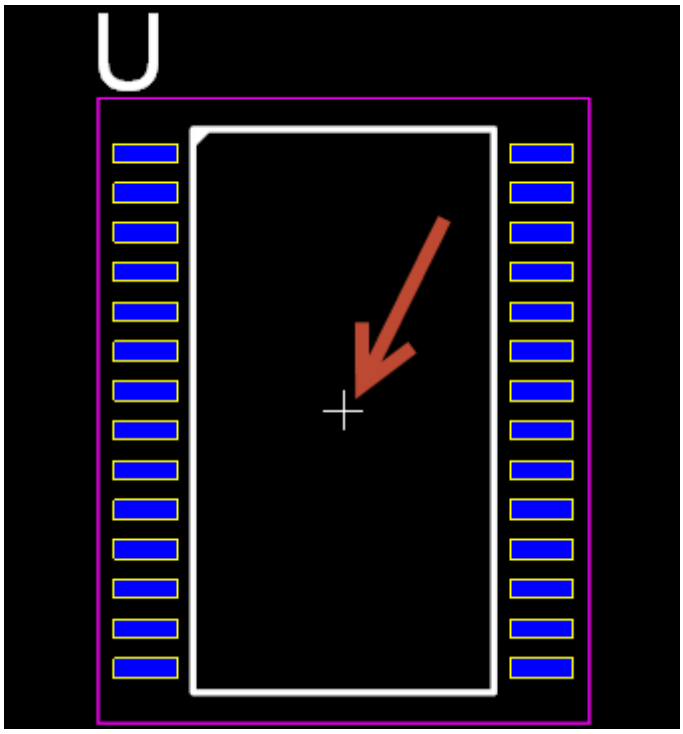
The Footprint Reference

1.2.5.12.8.1 Editing Footprint References

Select the footprint reference and change it's value in the properties panel or double click it to edit it's value.

1.2.5.12.9 The Footprint Placement Point

The footprint placement point defines the parts placement position for Pick and Place Pick and Place machines.



The Footprint Placement Point

1.2.5.12.9.1 Editing Footprint Placement Points

Select the footprint placement and change it's value in the properties panel or drag it in the viewport.

1.2.5.12.10 Adding 3D objects

You can add 3D objects to a part.

Use the 

button group in the **Add menu**.

1. Select either the top or bottom package layer as the current layer.
2. Add your graphics.
3. Set the height and base of the graphic using the properties panel.

1.2.5.12.11 Adding Graphics to Footprint

You can add graphics to any layer for a footprint.

If the graphics are on the top, bottom or inner layers (copper layers), then they will be in copper and do have electrical significance.

1.2.5.12.12 Customizing a Parametric Footprint

In AutoTRAX DEX, a parametric footprint is a part that is defined by a small number of numeric and textural parameters. From this small set of parameters, AutoTRAX DEX can generate whole families of parts with the schematic symbols, footprints (often called land patterns) and even the 3D part.

1.2.5.13 3D Packages

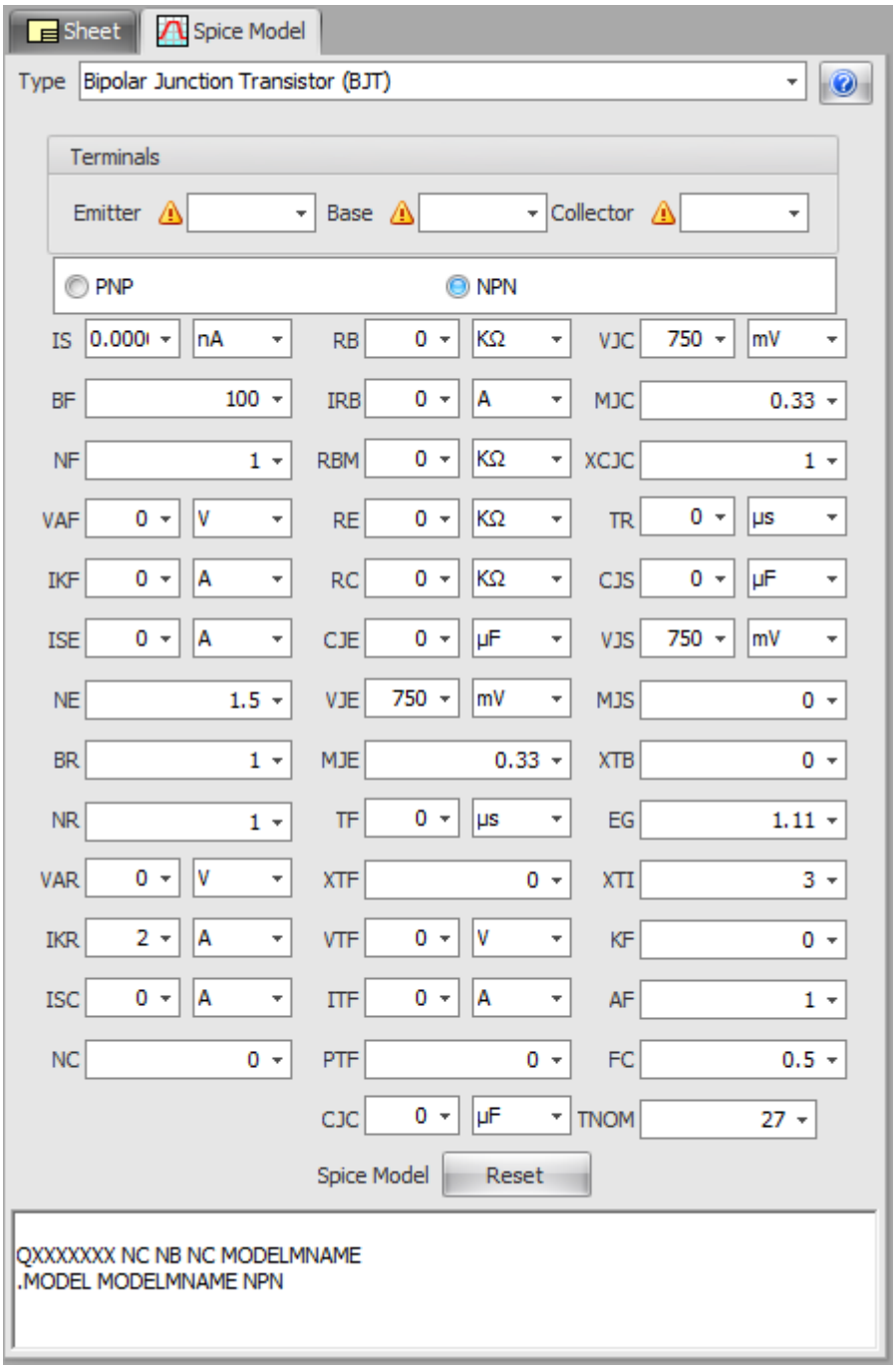
The Part Builder will automatically generate the 3D model for parametric parts.

1.2.5.14 Spice Models

You can assign SPICE simulation models to any part by selecting its schematic symbol.

1.2.5.14.1 Bipolar Transistors (BJT)

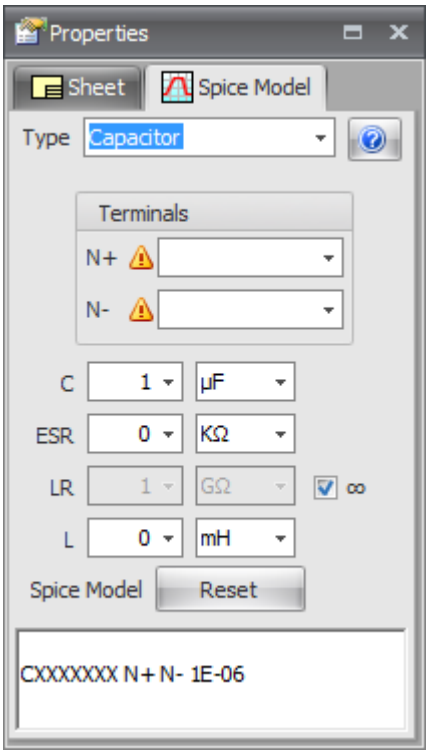
The Bipolar Transistor model dialog is shown below.



The Bipolar Transistor Model Dialog

1.2.5.14.2 Capacitors

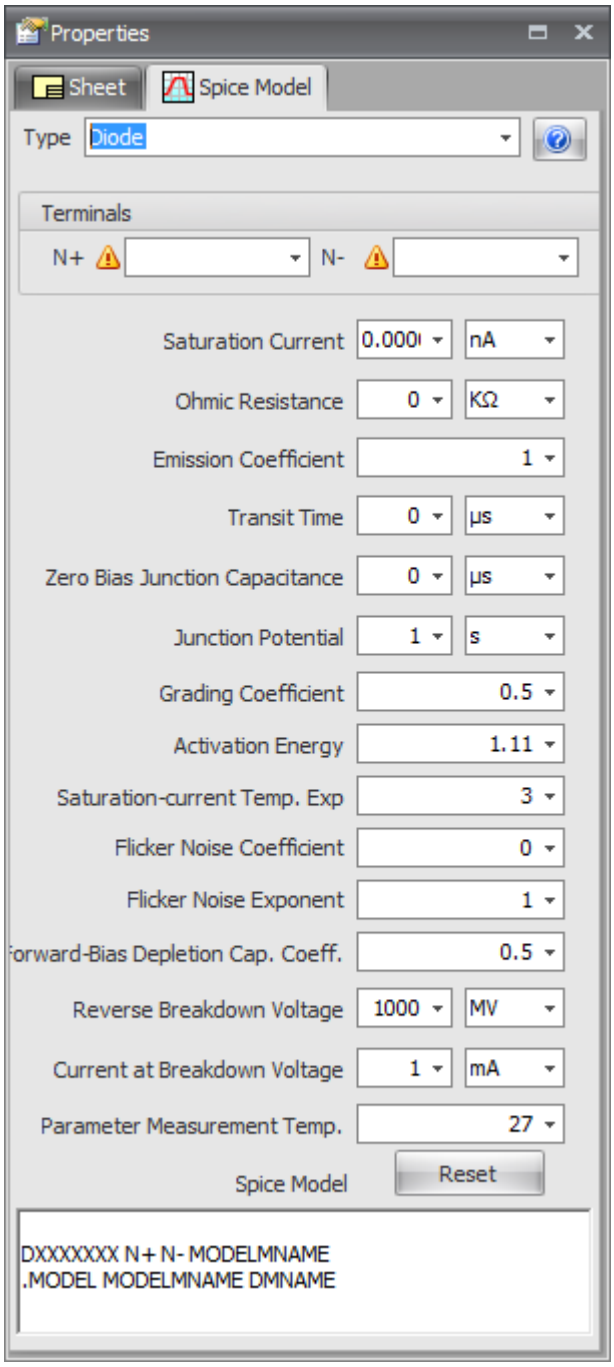
The capacitor model dialog is shown below.



The Capacitor Model Dialog

1.2.5.14.3 Diodes

The diode model dialog is shown below.



The Diode Model Dialog

1.2.5.14.4 Voltage and Current Sources

You can add the following types of voltage and current models.

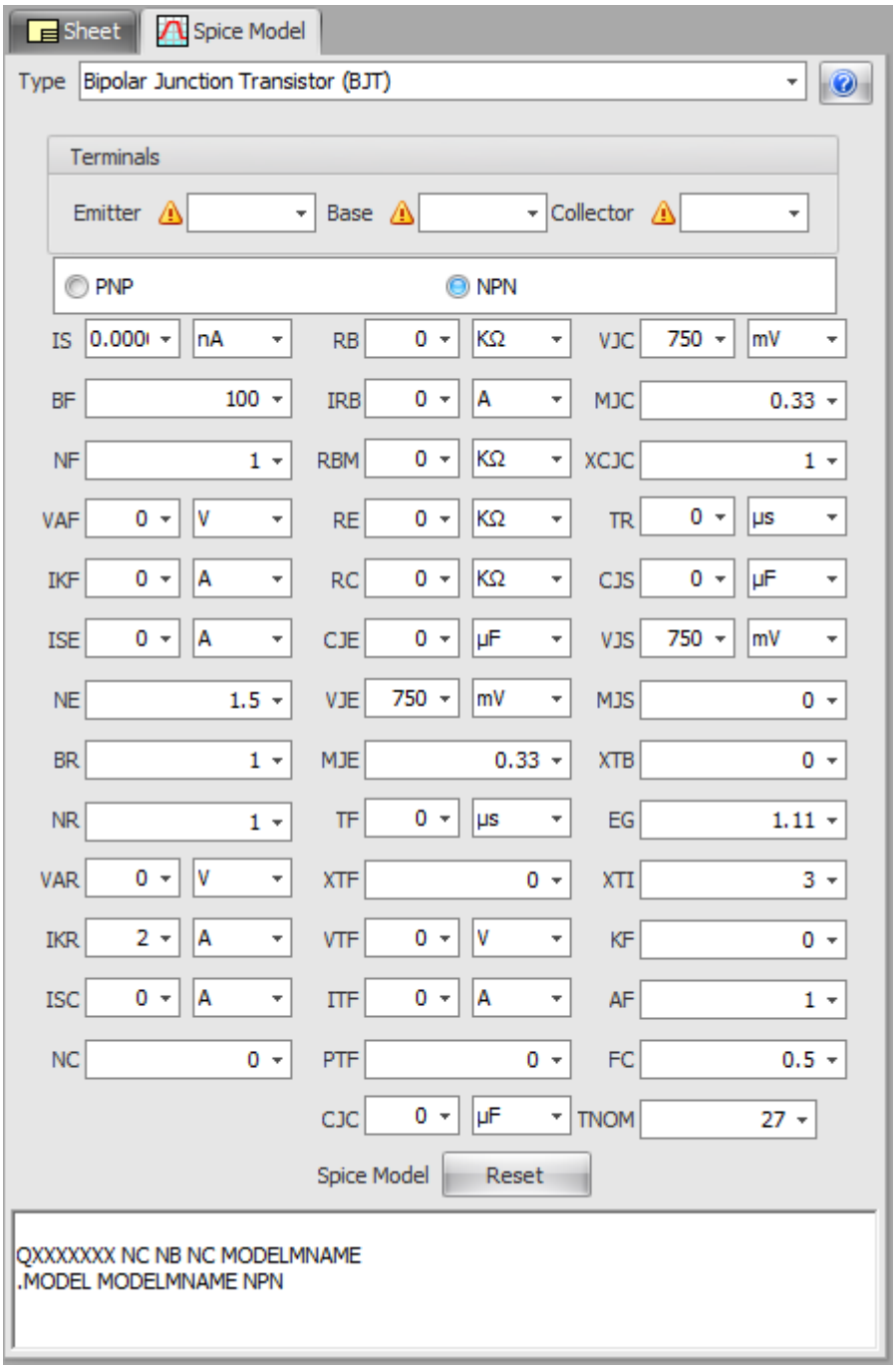
- Voltage Sources
- Current Sources
- Controlled Sources

1.2.5.14.4.1 Voltage Sources

You can add the following types of voltage source models.

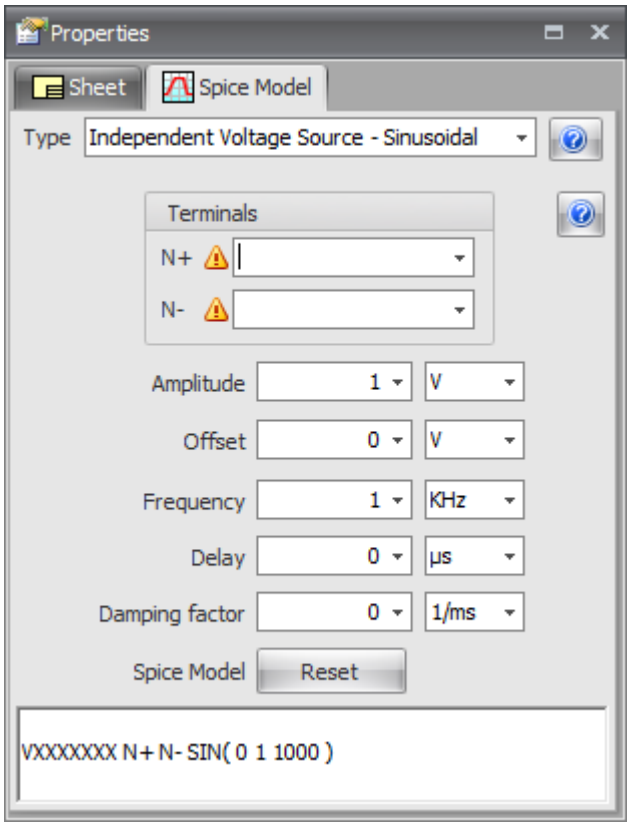
- Independent DC Voltage Sources
- Independent Sinusoidal Voltage Sources
- Independent Pulse Voltage Sources
- Independent Single Frequency FM Voltage Sources
- Independent Piece-Wise Linear Voltage Sources
- Independent Exponential Voltage Sources

The Independent DC Voltage Source model dialog is shown below.



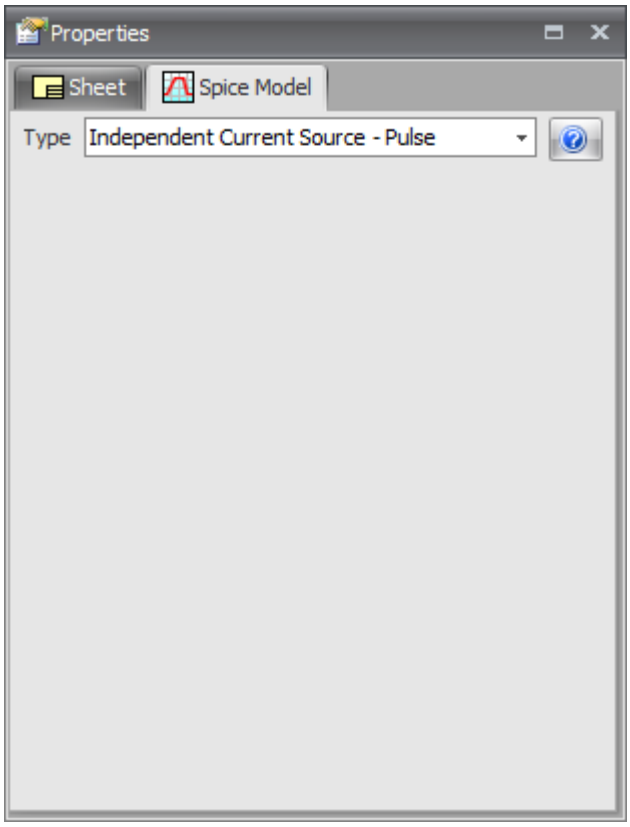
The Model Dialog

The Independent Sinusoidal Voltage Source model dialog is shown below.



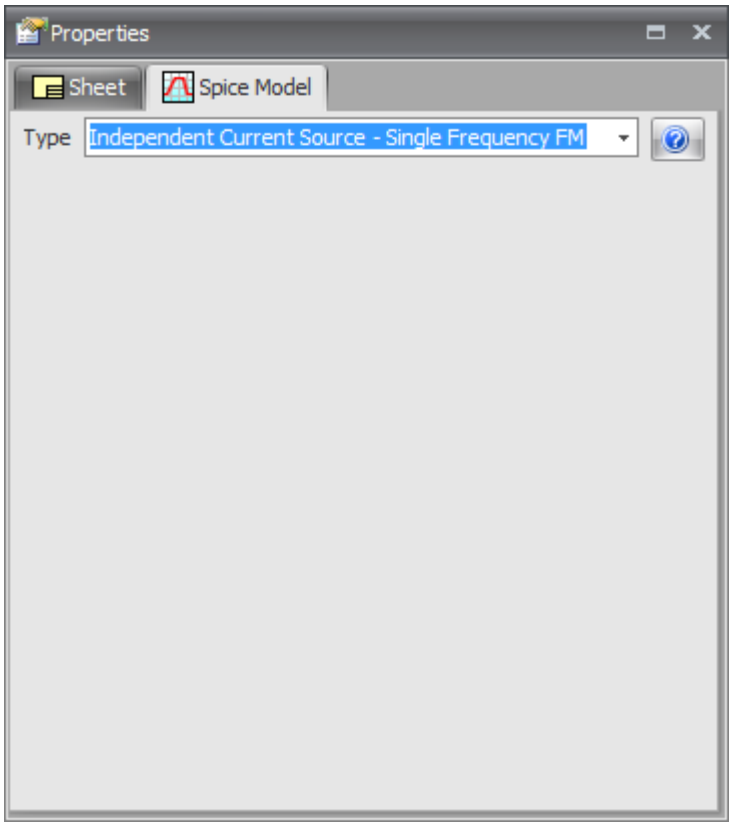
The Independent Sinusoidal Voltage Model Dialog

The Independent Pulse Voltage Source model dialog is shown below.



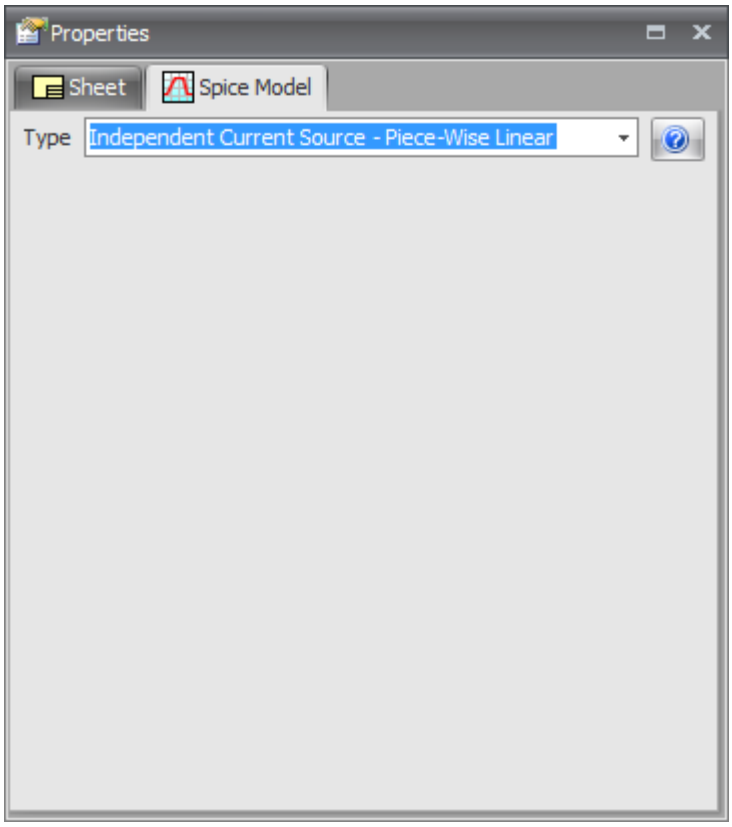
The Independent Pulse Voltage Sources Model Dialog

The Independent Single Frequency FM Voltage Source model dialog is shown below.



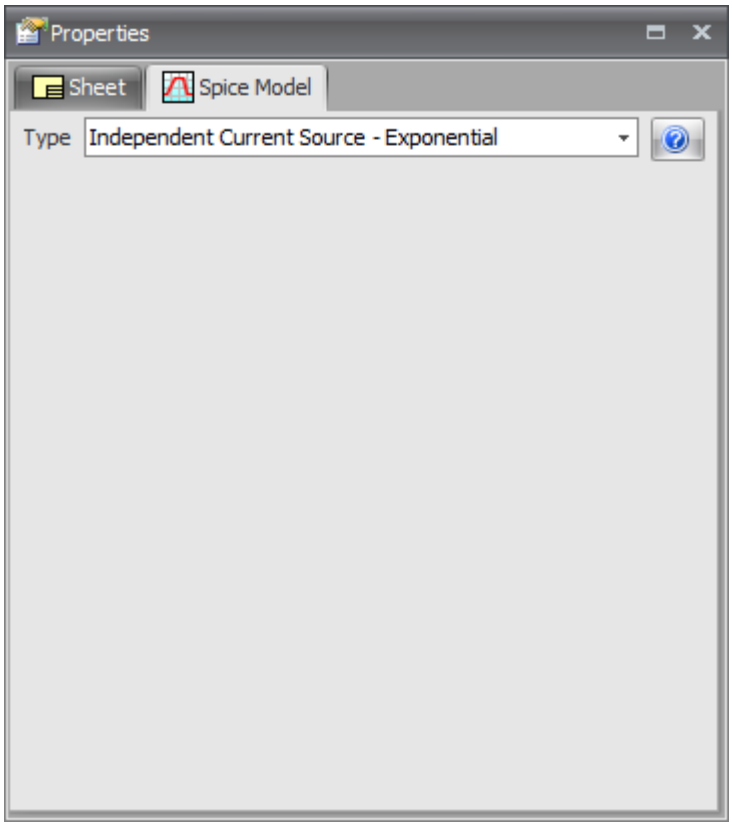
The Independent Single Frequency FM Voltage Model Dialog

The Independent Piece-Wise Linear Voltage Source model dialog is shown below.



The Independent Piece-Wise Linear Voltage Model Dialog

The Independent Exponential Voltage Source model dialog is shown below.



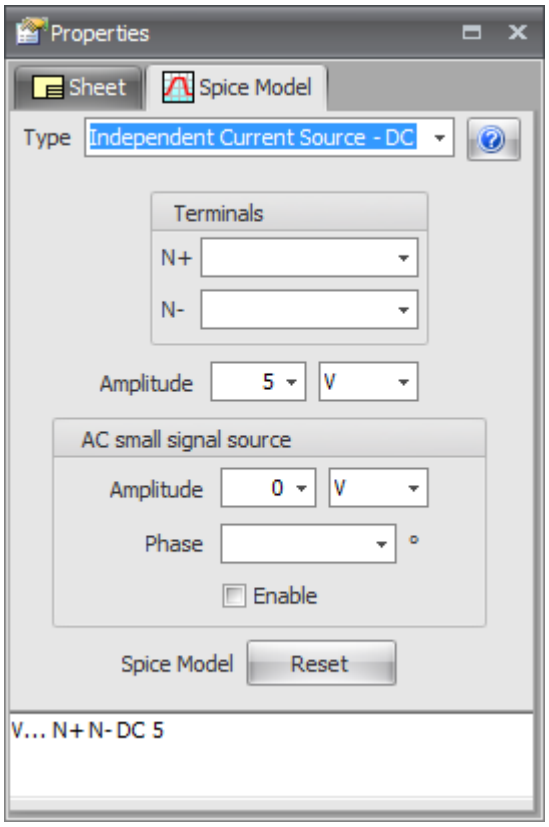
The Independent Exponential Voltage Model Dialog

1.2.5.14.4.2 Current Sources

You can add the following types of current source models.

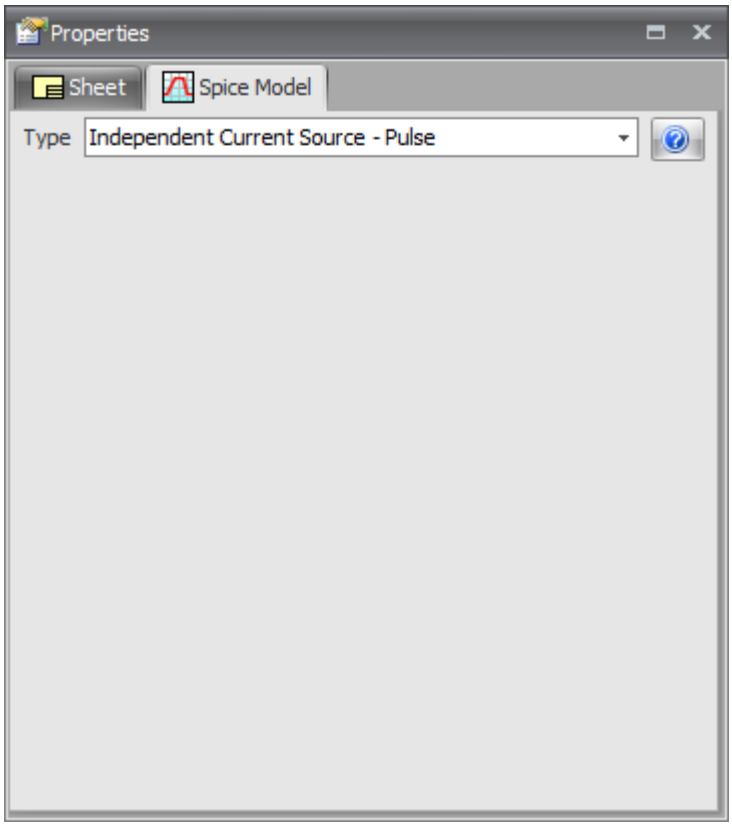
- Independent DC Current Sources
- Independent Pulse Current Sources
- Independent Sinusoidal Current Sources
- Independent Single Frequency FM Current Sources
- Independent Piece-Wise Linear Current Sources

The Independent DC Current Source model dialog is shown below.



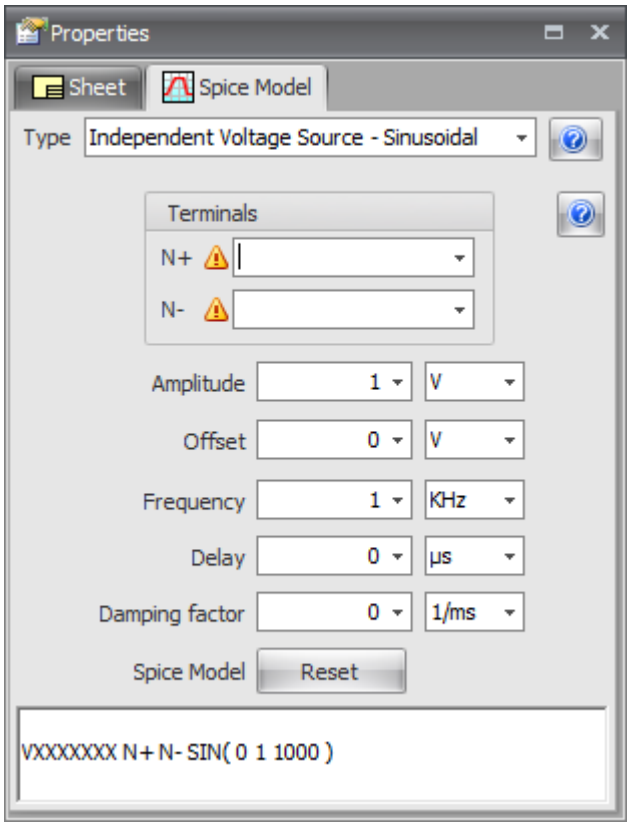
The Independent DC Current Model Dialog

The Independent Pulse Current Source model dialog is shown below.



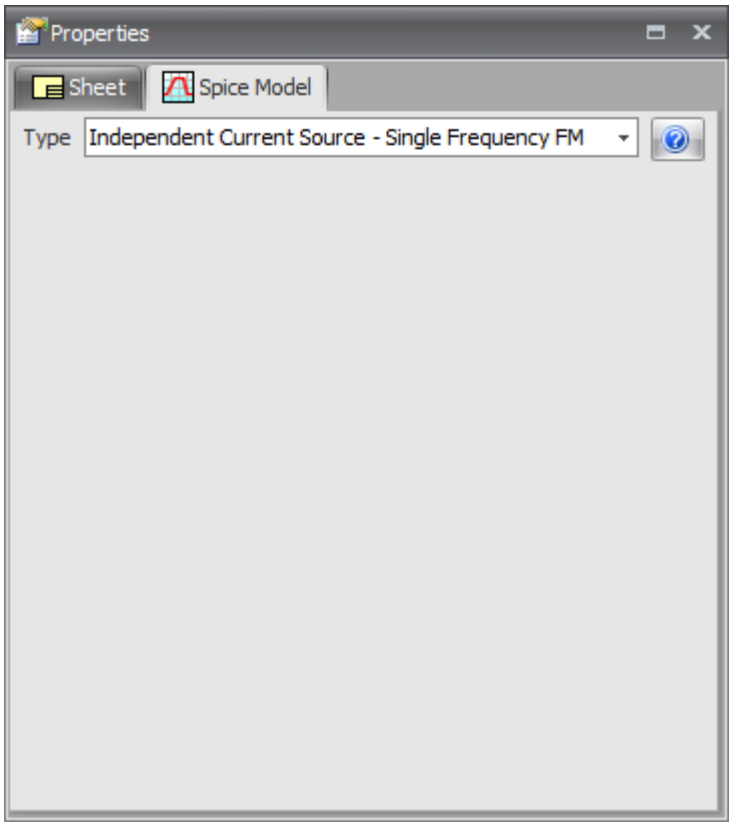
The Independent Pulse Current Model Dialog

The Independent Sinusoidal Current Source model dialog is shown below.



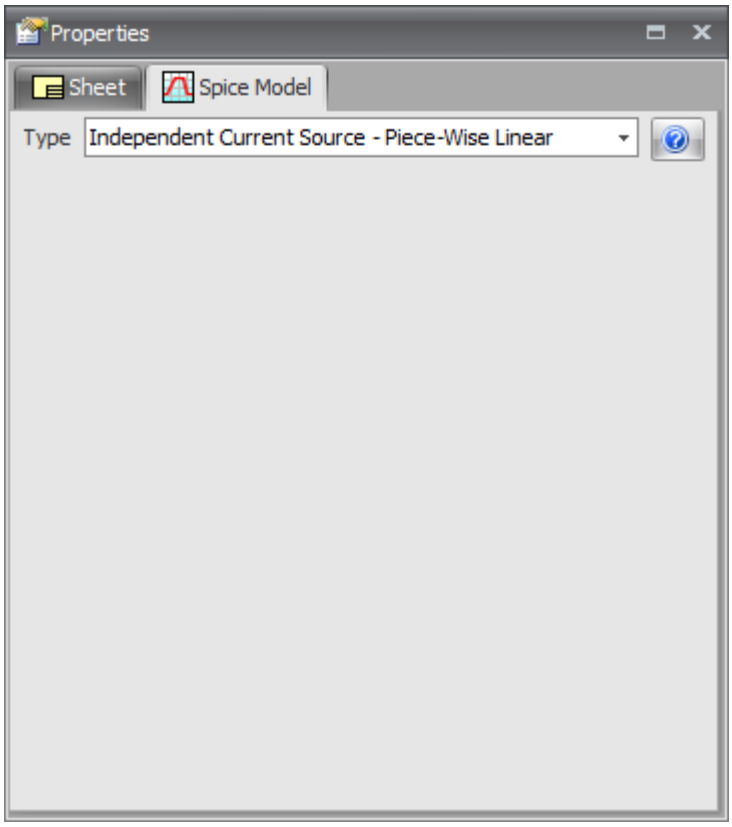
The Independent Sinusoidal Current Model Dialog

The Independent Single Frequency FM Current Source model dialog is shown below.



The Independent Single Frequency FM Current Model Dialog

The Independent Piece-Wise Linear Current Source model dialog is shown below.



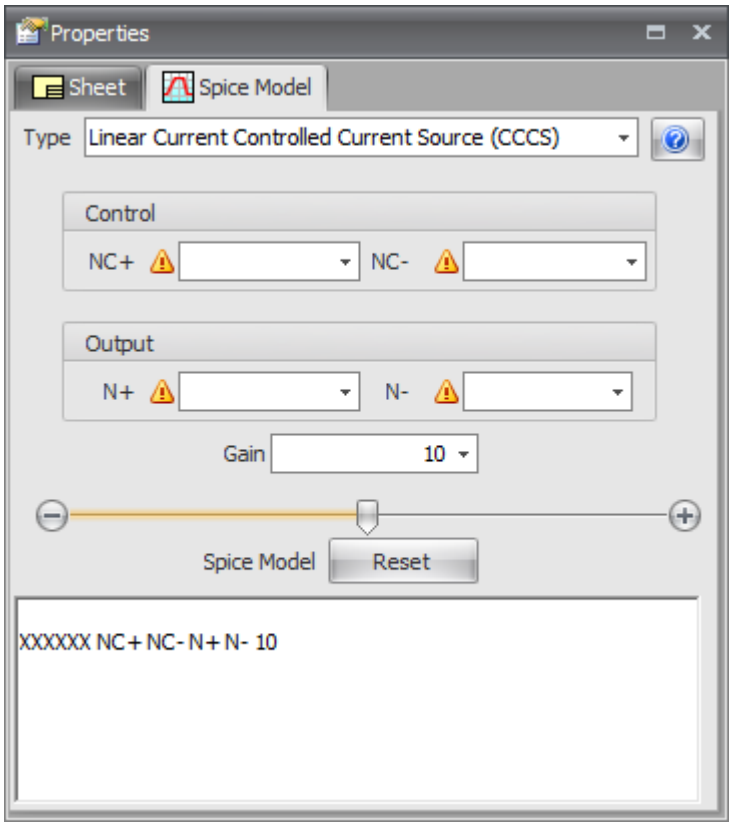
The Independent Piece-Wise Linear Current Model Dialog

1.2.5.14.4.3 Controlled Sources

You can add the following types of controlled source models.

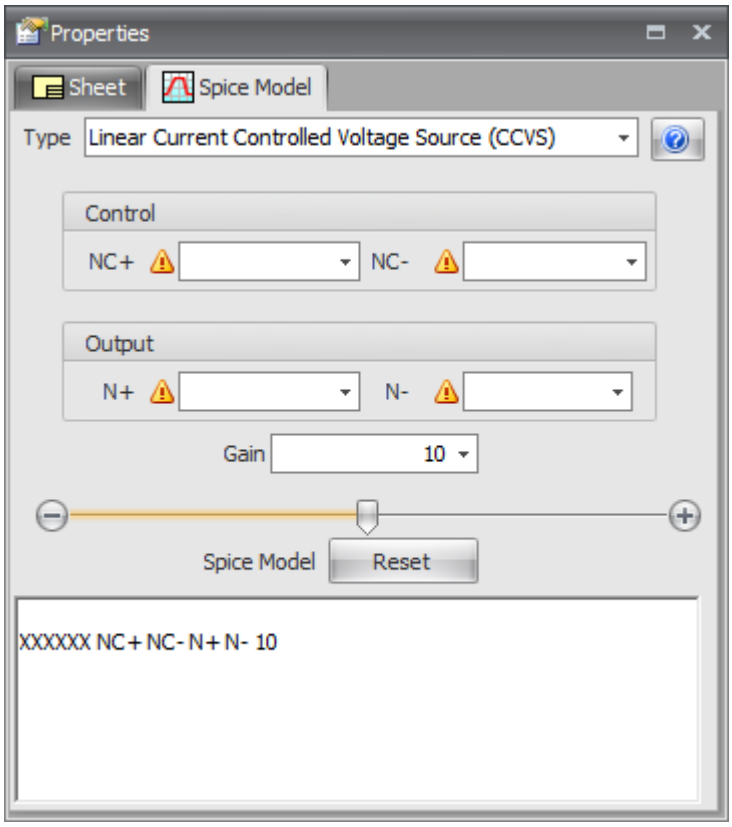
- Current Controlled Current Sources
- Current Controlled Voltage Sources
- Voltage Controlled Current Sources
- Voltage Controlled Voltage Sources

The Current Controlled Current Source model dialog is shown below.



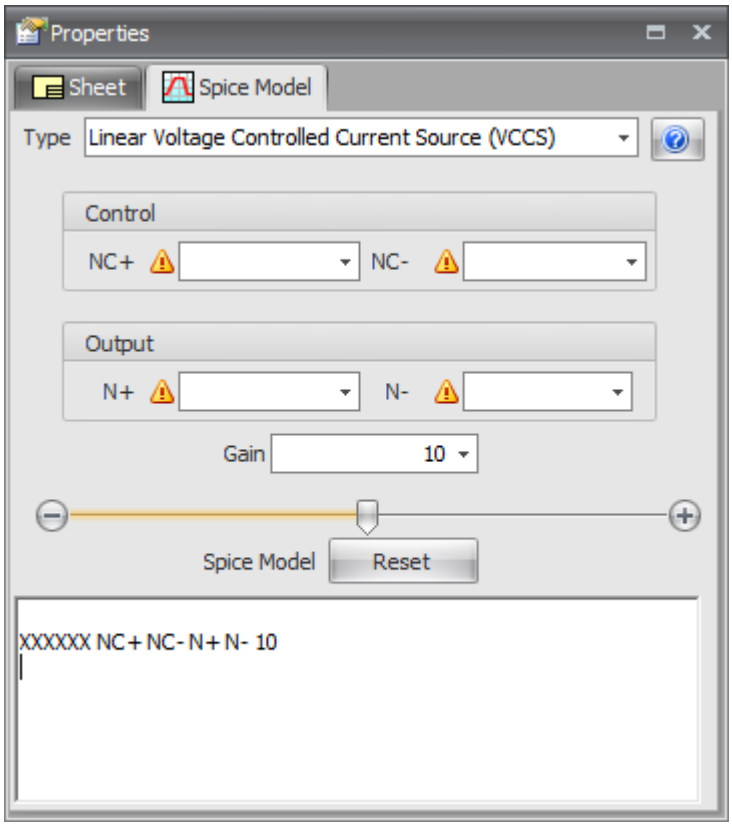
The Current Controlled Current Model Dialog

The Current Controlled Voltage Source model dialog is shown below.



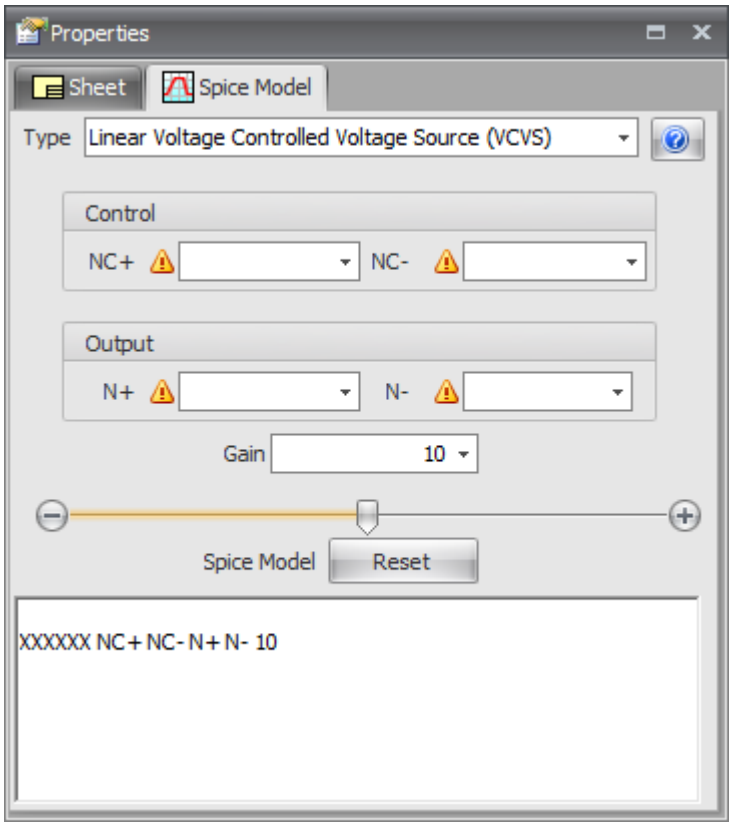
The Current Controlled Voltage Model Dialog

The Voltage Controlled Current Source model dialog is shown below.



The Voltage Controlled Current Model Dialog

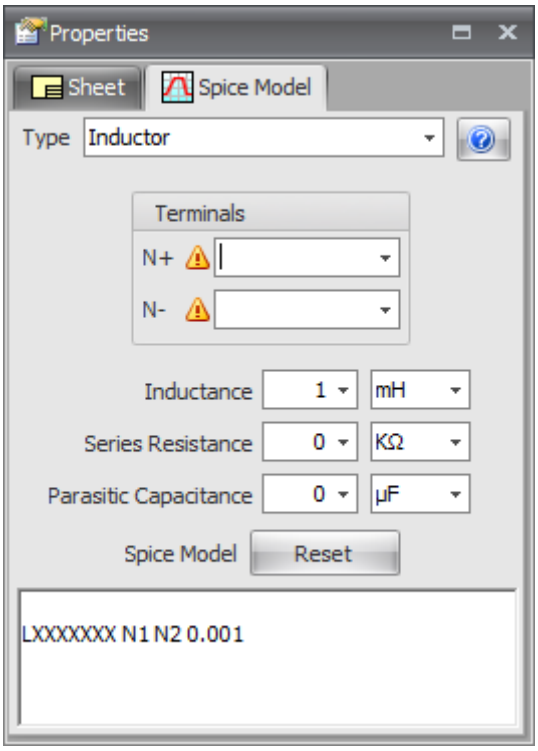
The Voltage Controlled Voltage Source model dialog is shown below.



The Voltage Controlled Voltage Model Dialog

1.2.5.14.5 Inductors

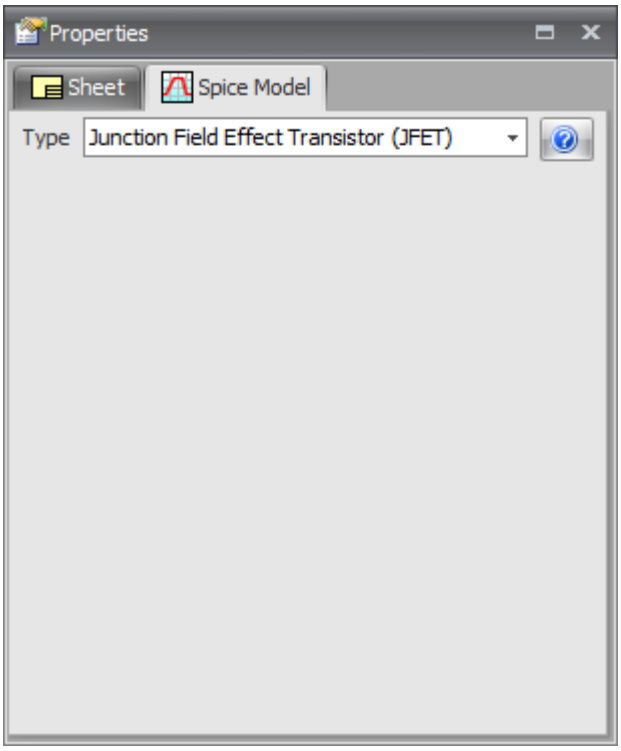
The Inductor model dialog is shown below.



The Inductor Model Dialog

1.2.5.14.6 Junction Field Effect Transistors (JFET)

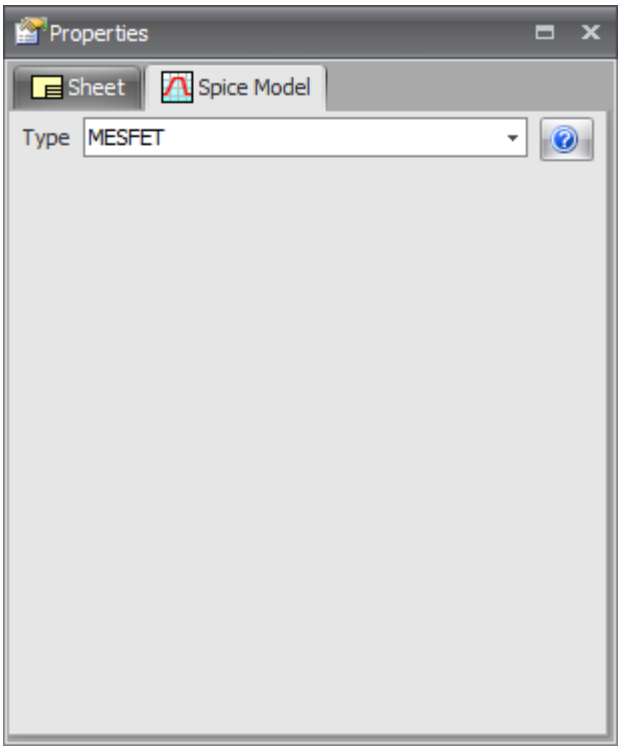
The JFET model dialog is shown below.



The JFET Model Dialog

1.2.5.14.7 MESFETs

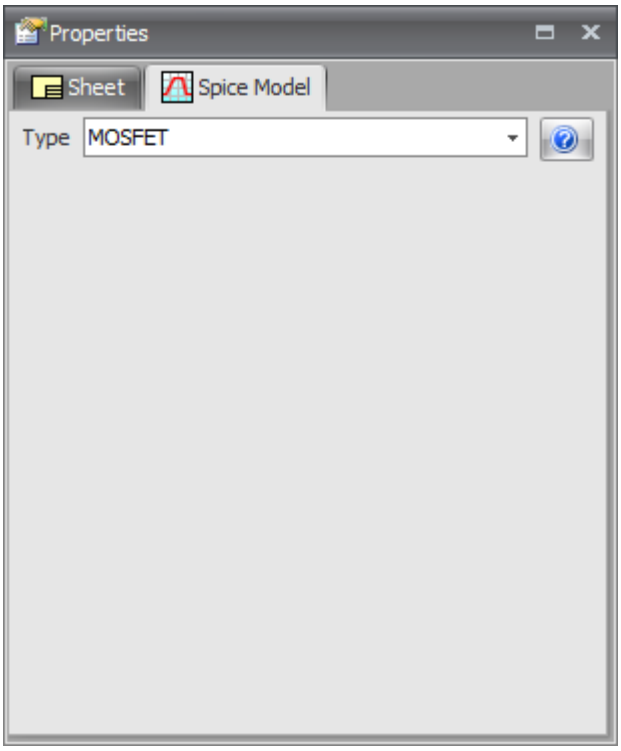
The MESFET model dialog is shown below.



The MESFET Model Dialog

1.2.5.14.8 MOSFETs

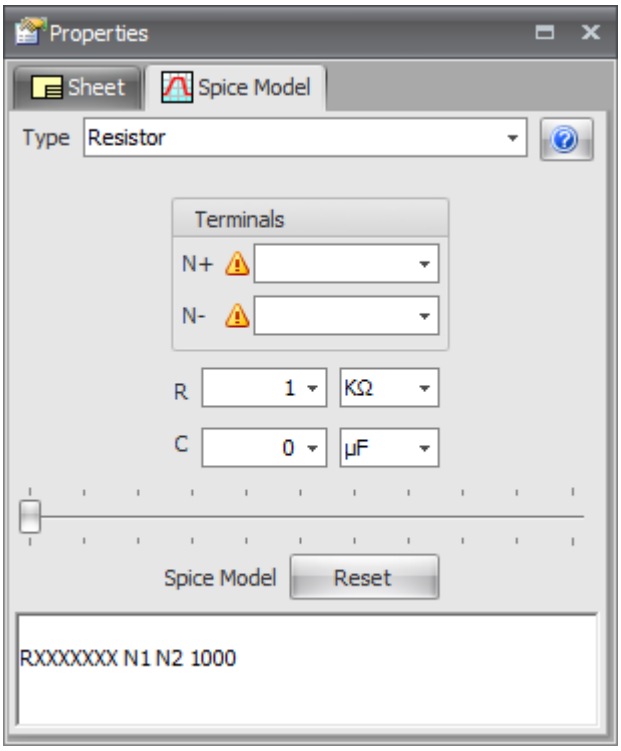
The MOSFET model dialog is shown below.



The MOSFET Model Dialog

1.2.5.14.9 Resistors

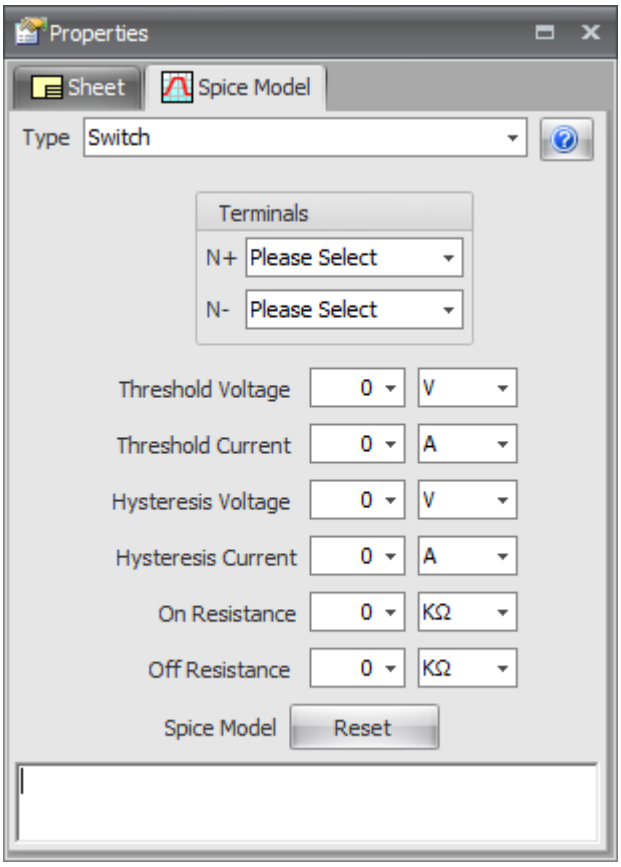
The Resistor model dialog is shown below.



The Resistor Model Dialog

1.2.5.14.10 Switches

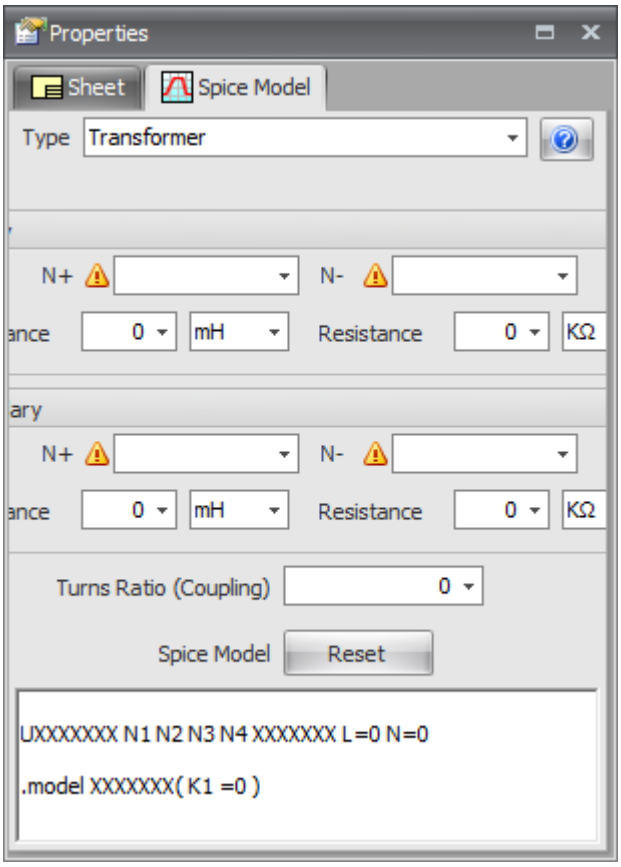
The Switch model dialog is shown below.



The Switch Model Dialog

1.2.5.14.11 Transformers

The Transformer model dialog is shown below.



The Transformer Model Dialog

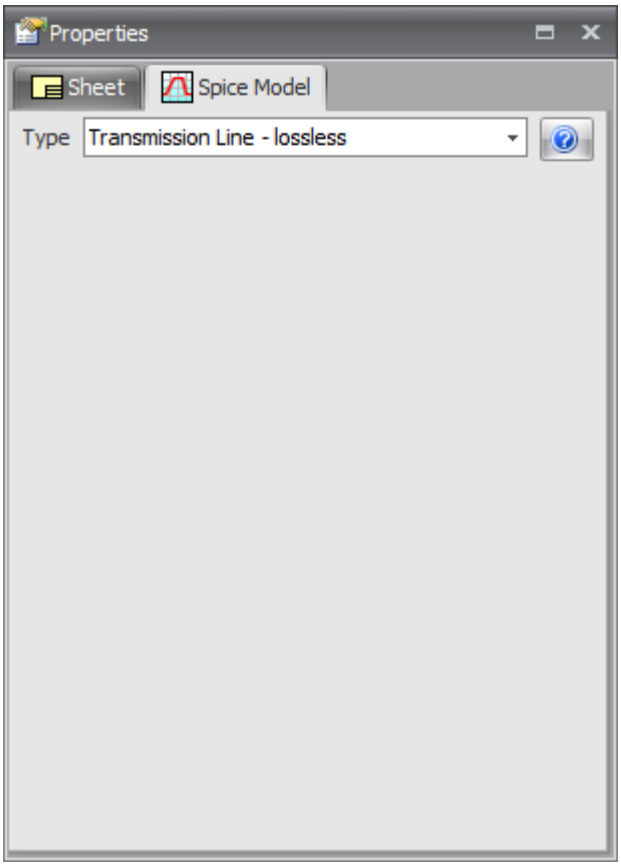
1.2.5.14.12 Transmission Lines

You can add the following types of transmission line models.

- Lossless Transmission Lines
- Lossy Transmission Lines (LTRA)
- Lossy Transmission Lines (URC)

1.2.5.14.12.1 Lossless Transmission Lines

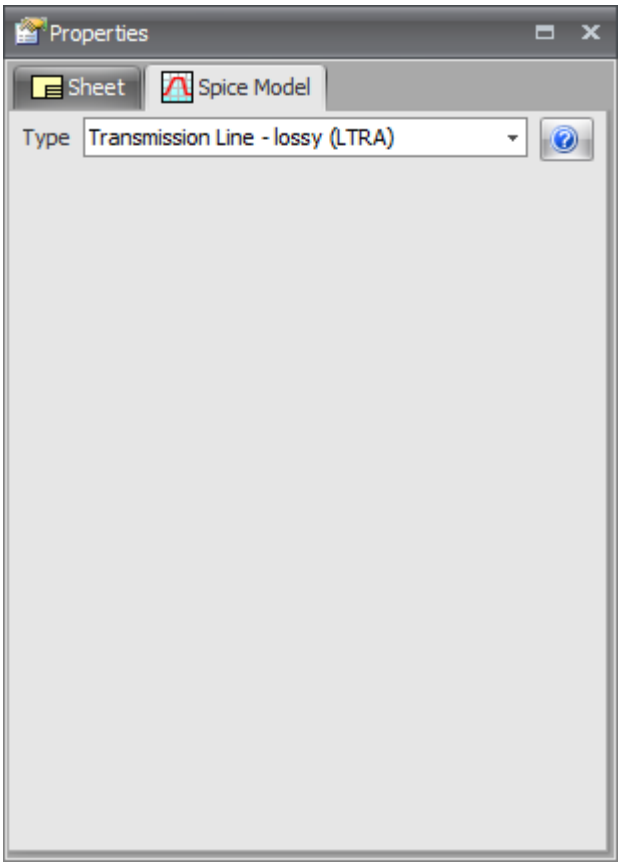
The Lossless Transmission Line model dialog is shown below.



The Lossless Transmission Line Model Dialog

1.2.5.14.12.2 Lossy Transmission Lines (LTRA)

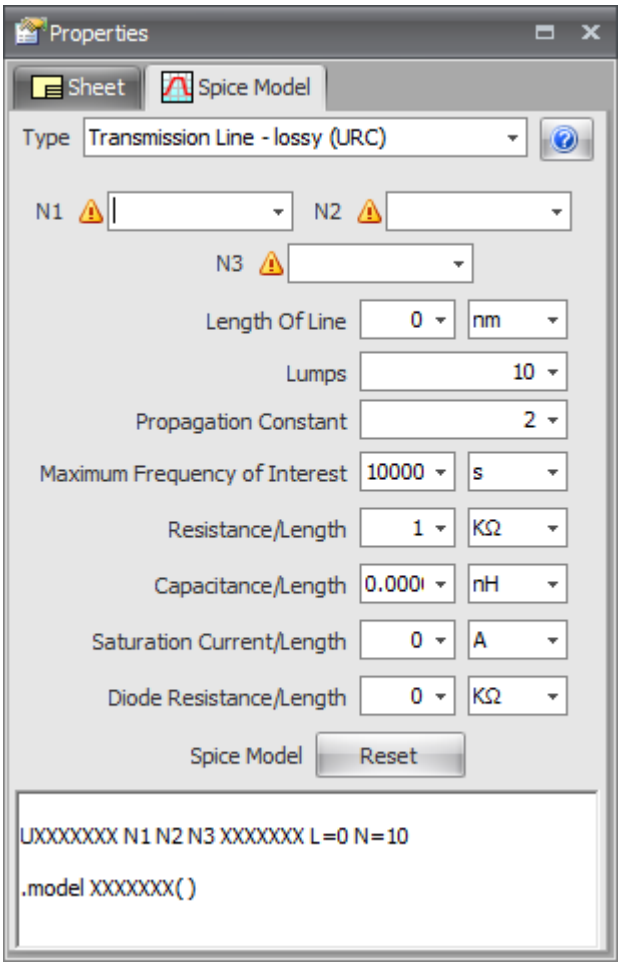
The Lossy Transmission Line model dialog is shown below.



The Lossy Transmission Line Model Dialog

1.2.5.14.12.3 Lossy Transmission Lines (URC)

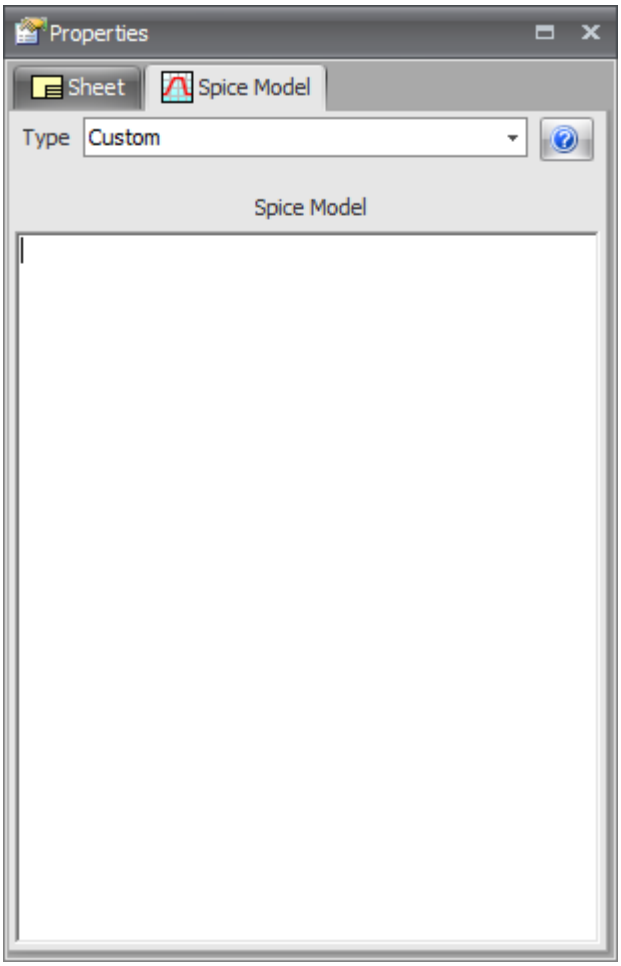
The Lossy Transmission Line model dialog is shown below.



The Lossy Transmission Lines Model Dialog

1.2.5.15 Custom

The Custom model dialog is shown below.



The Custom Model Dialog

1.2.5.16 Suppliers

The Suppliers dialog displays 3 expandable lists of suppliers: Semiconductors, Distributors, and Passive.

Click on the 

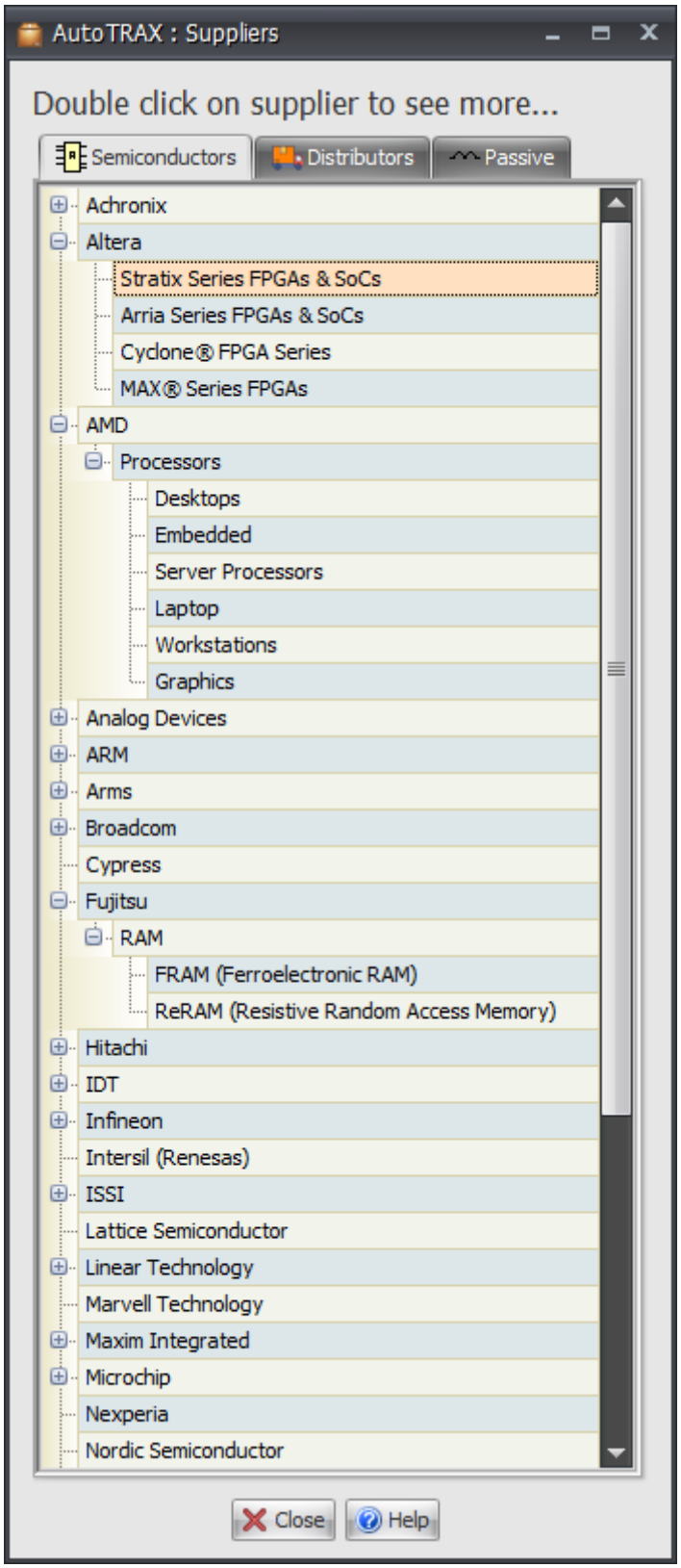
button in the **Home->Datasheets** menu to display the dialog.

Semiconductors

This is a list of manufacturers of semiconductors and devices you would use to make a PCB.

Expand the items by clicking on the ⊞ button.

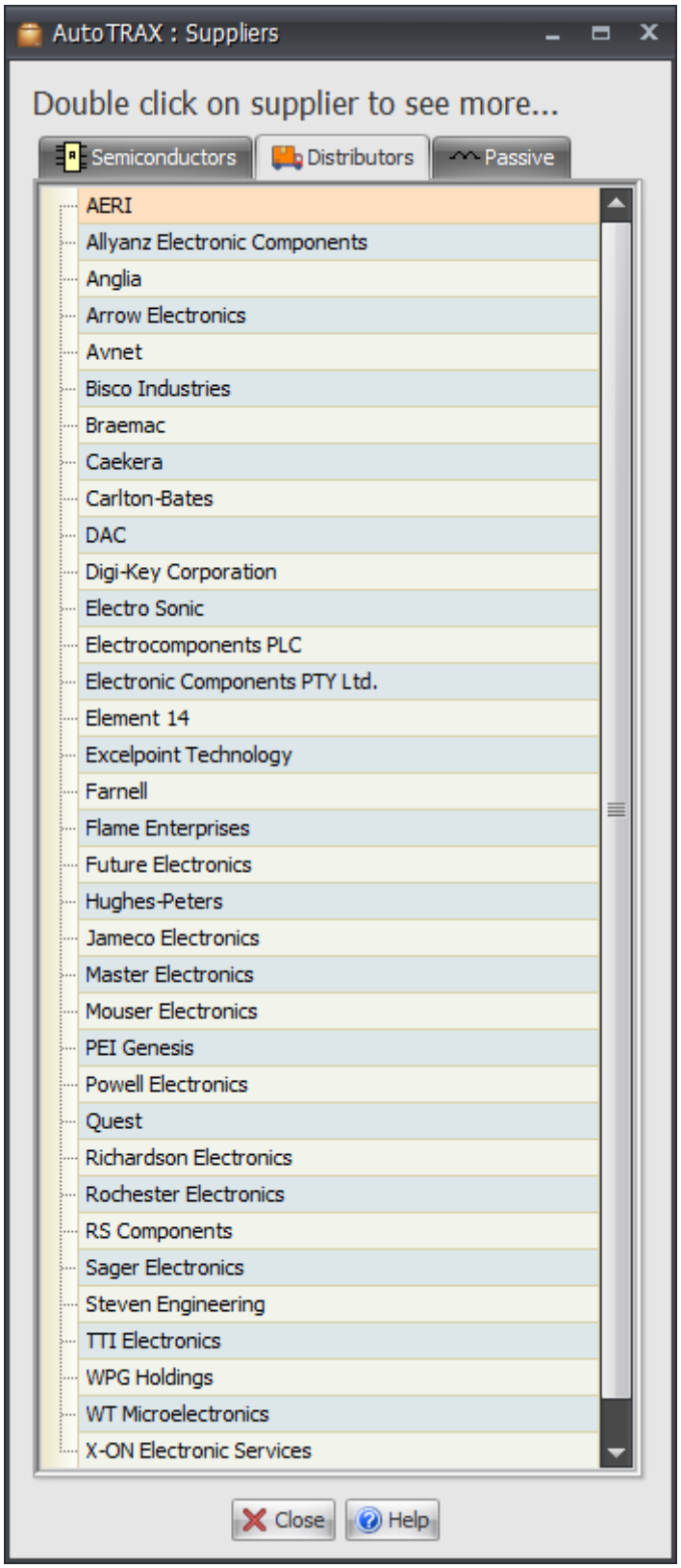
Double-click on a row to view their website in your web browser.



Distributors

This is a list of distributors of parts for your PCB.

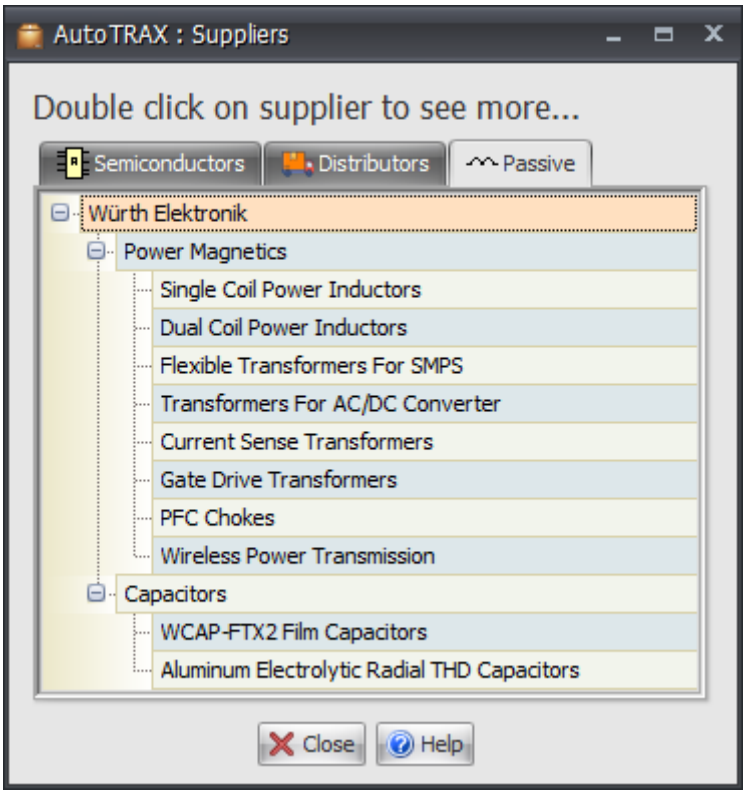
Double-click on a row to view their website in your web browser.



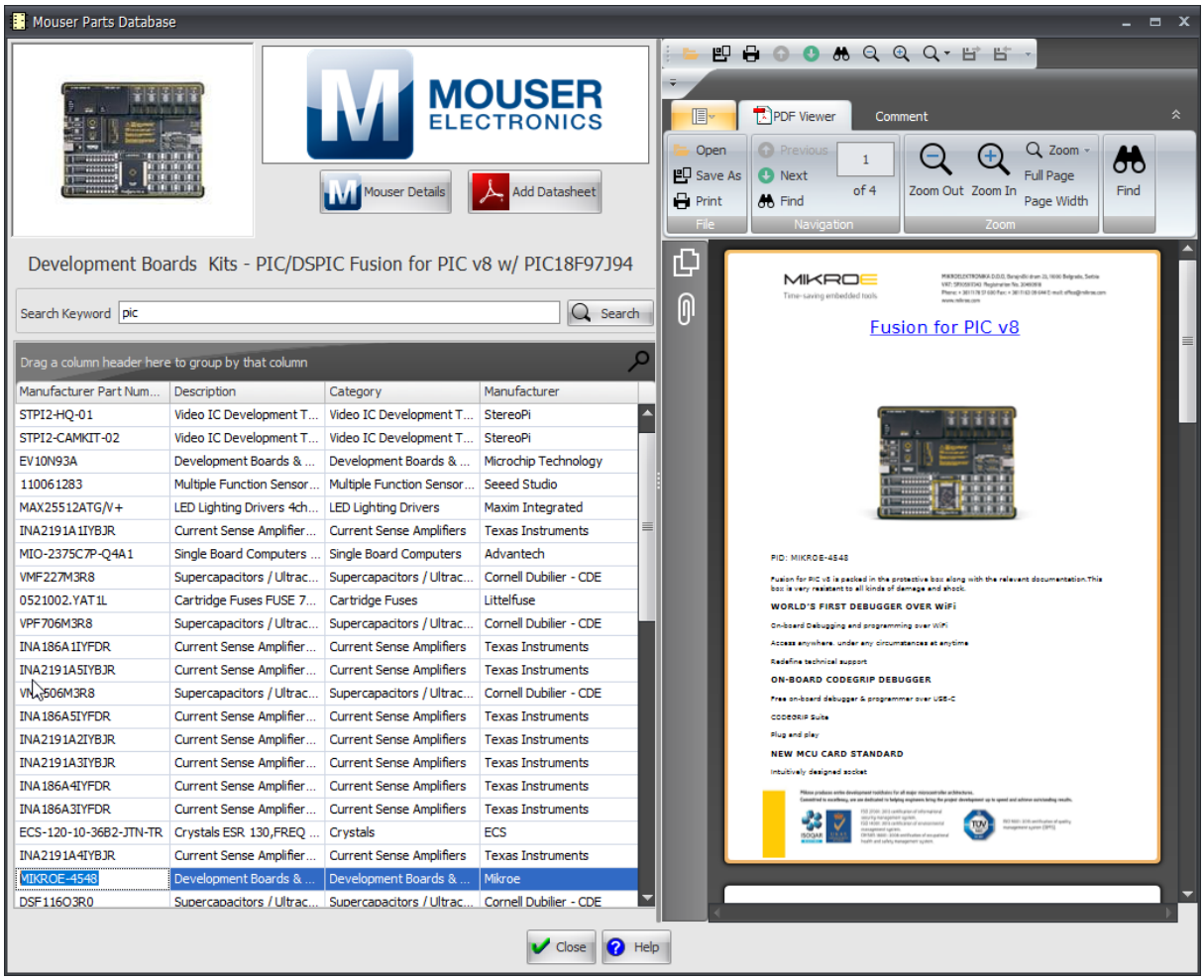
Passive

This is a list of manufacturers of passive parts for your PCB.

Double-click on a row to view their website in your web browser.



1.2.5.17 The Find Parts Dialog



1.2.5.18 Saving Your Part

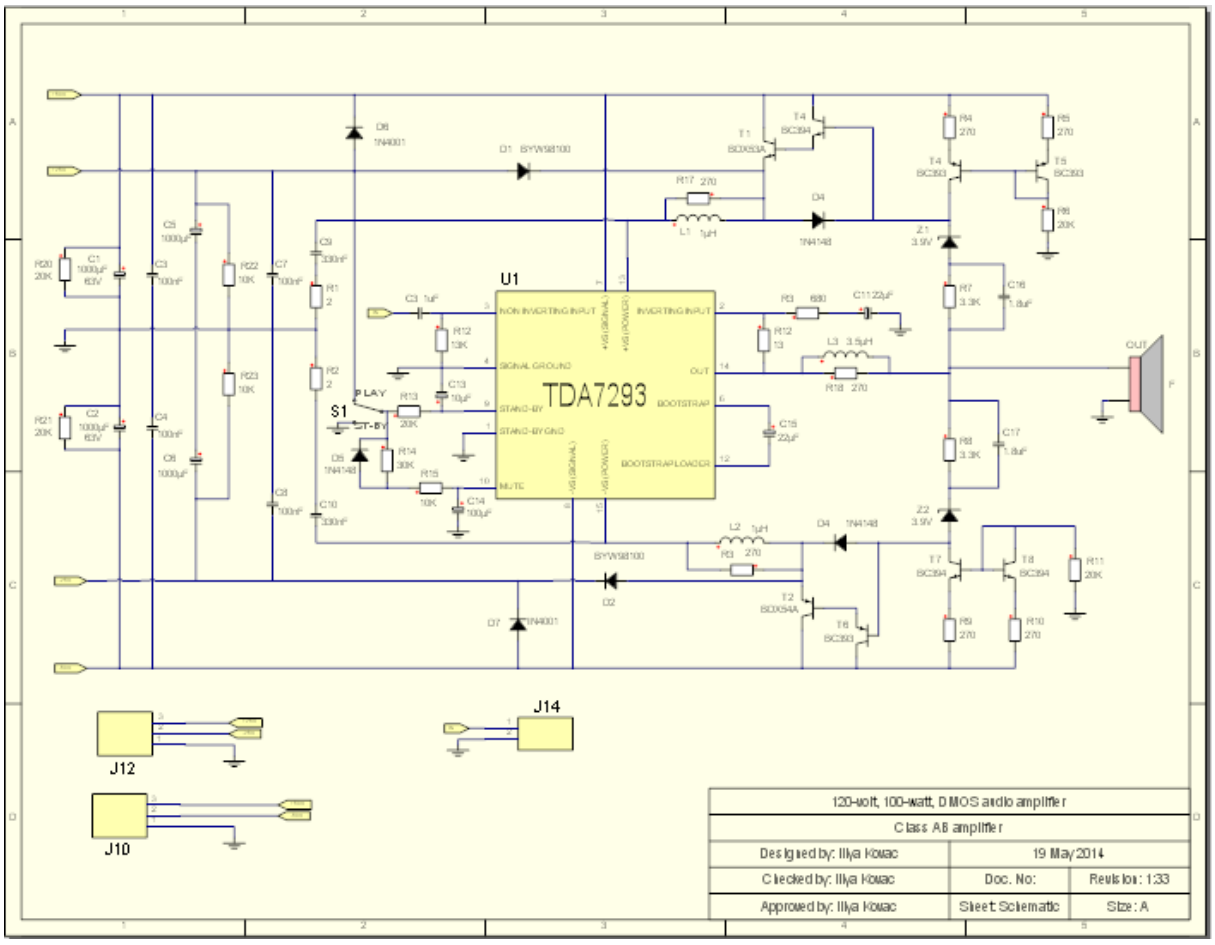
If editing a part you should save it to the parts library so you can drag and drop it onto schematics.

1.2.6 Projects

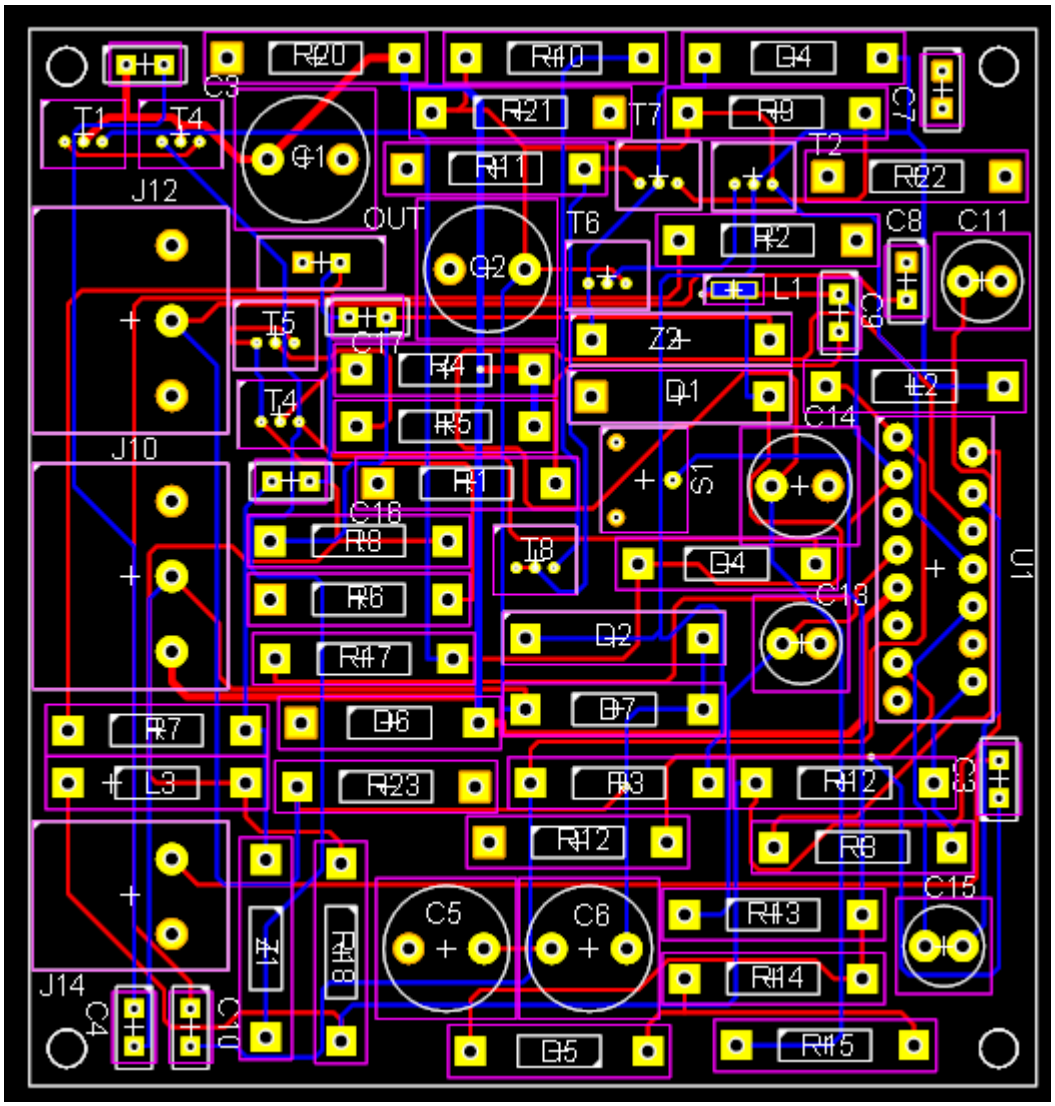
A project in AutoTRAX DEX is a self-contained file that contains all the details for your design. It does not rely on any external files (except font files). All schematics, text sheets and the PCB for your design is contained in this single file.

Your design project consists of one or more schematic diagrams and a single PCB.

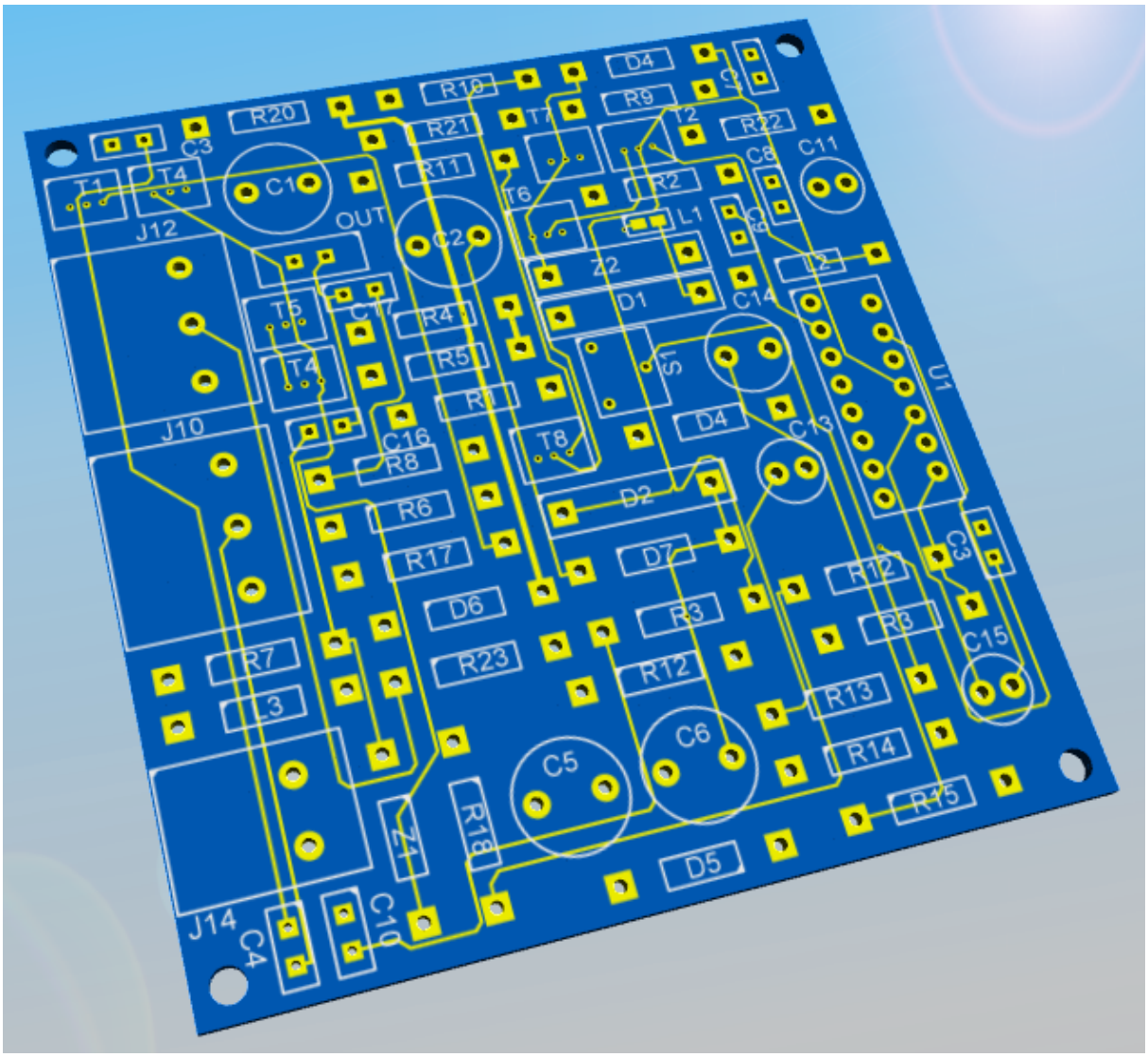
The schematics are the logical design for your project. The PCB is the physical manifestation of your design.



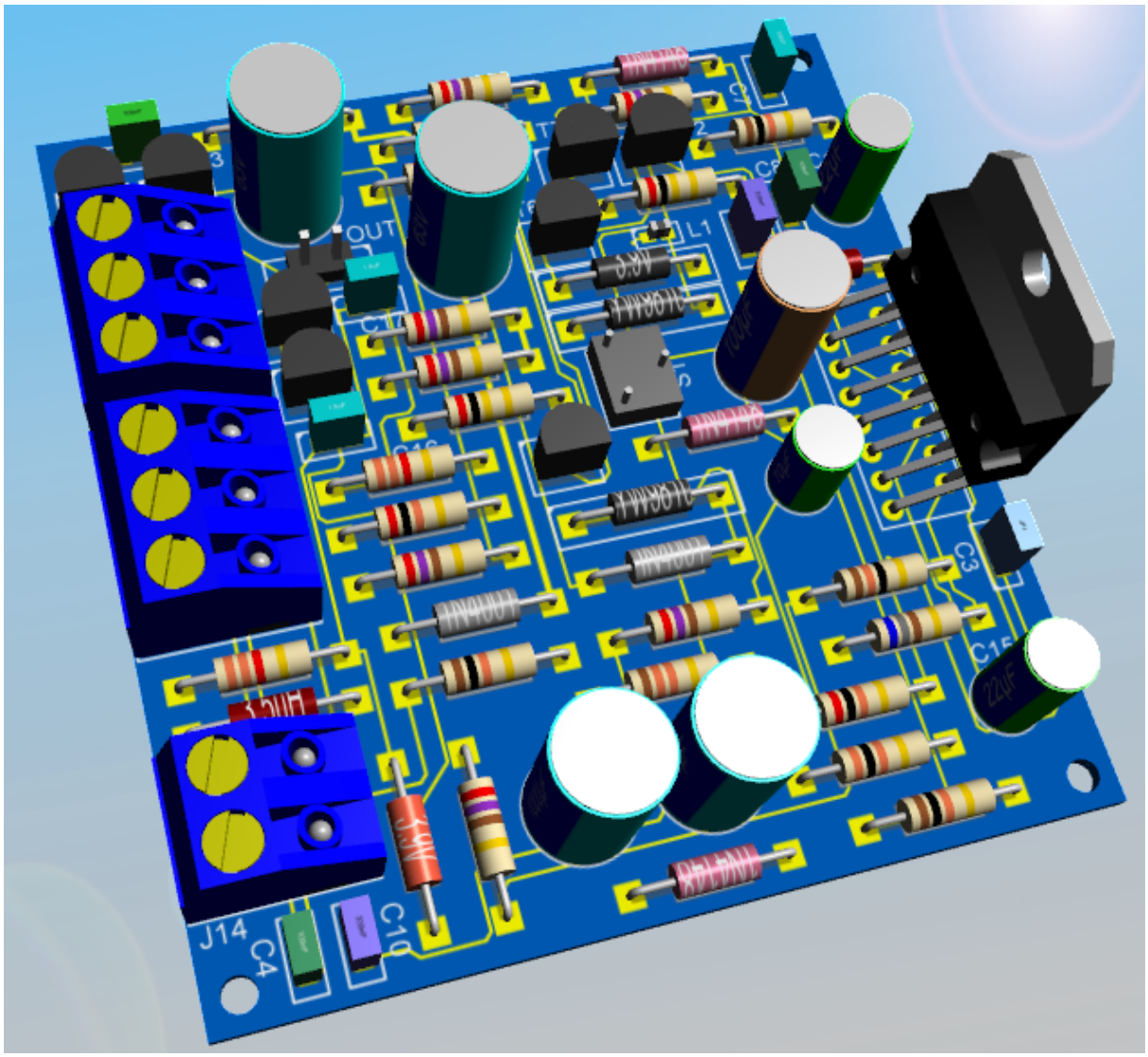
A Schematic



The PCB (2D Layout)



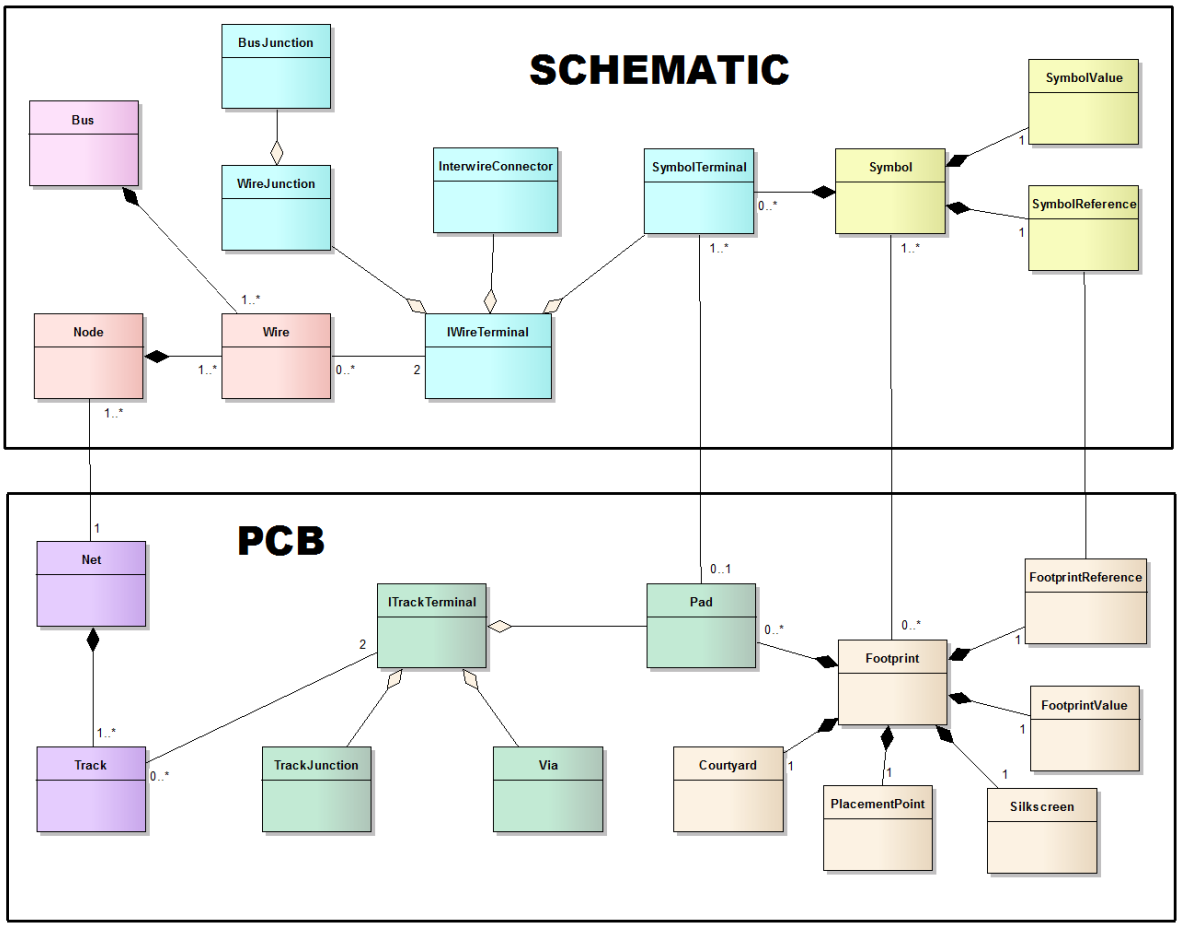
The 3D PCB without parts



The 3D PCB with parts

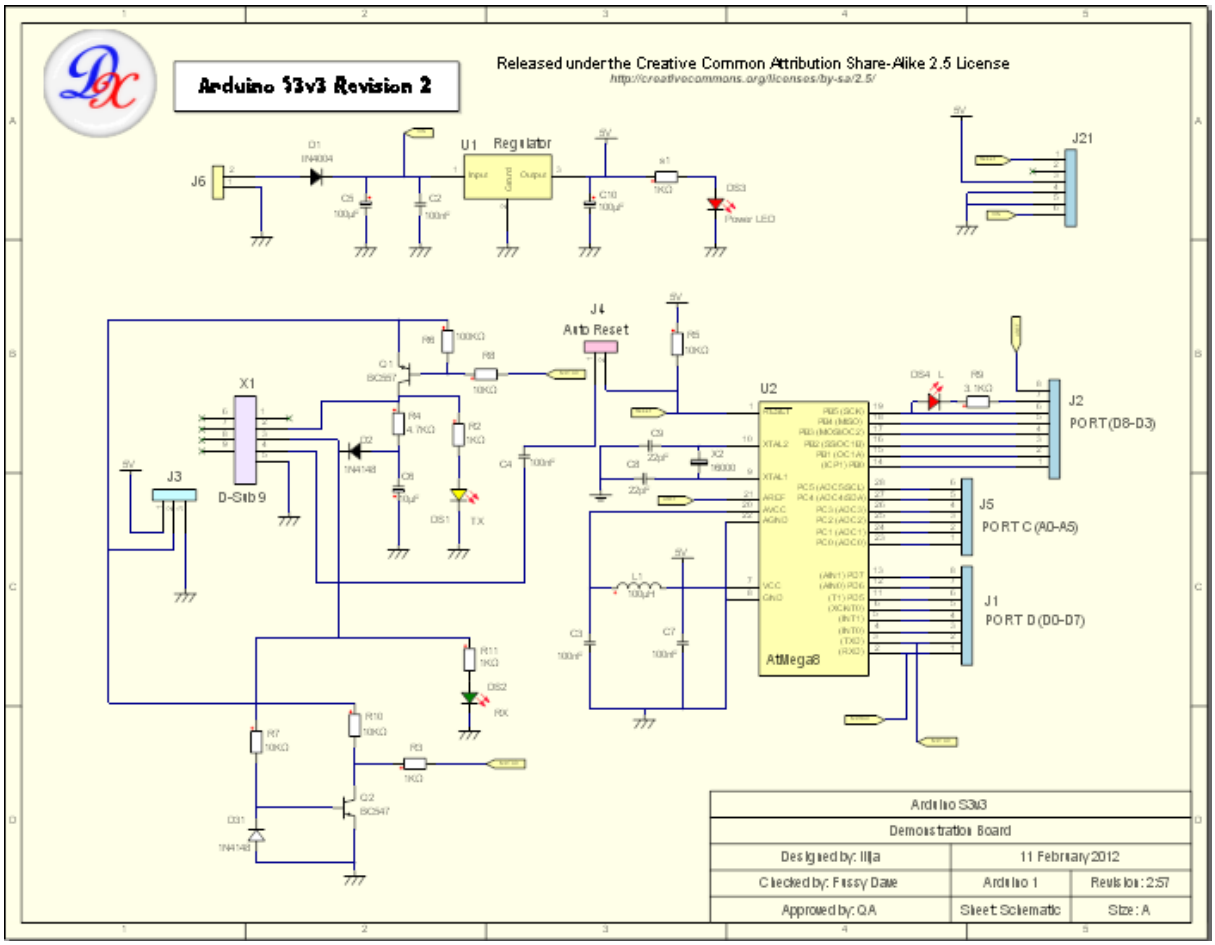
1.2.6.1 Project

The class diagram for the main classes in a AutoTRAX DEX project are shown below.



1.2.6.2 Schematics

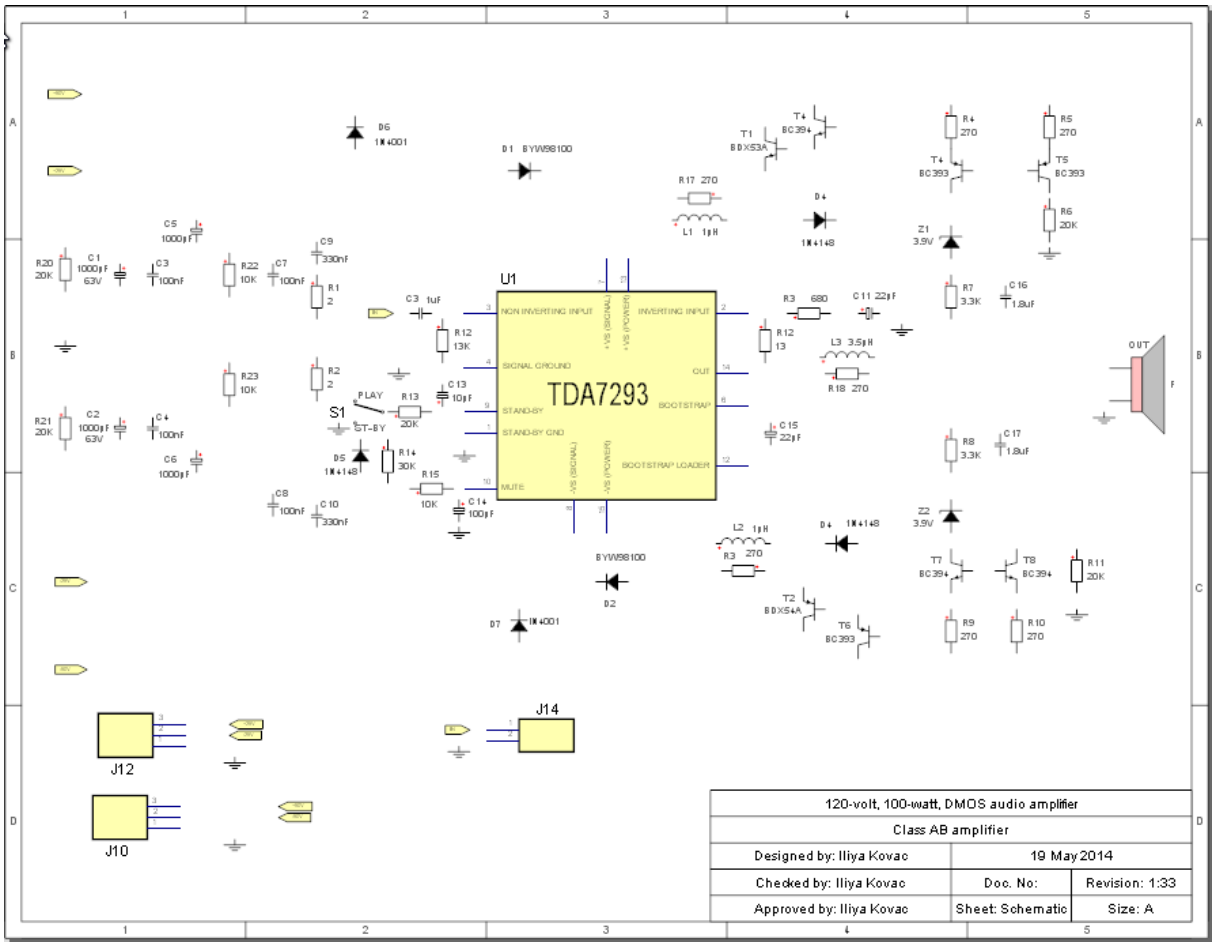
Schematics are Graphical Sheets that define the logical abstraction of the PCB design.



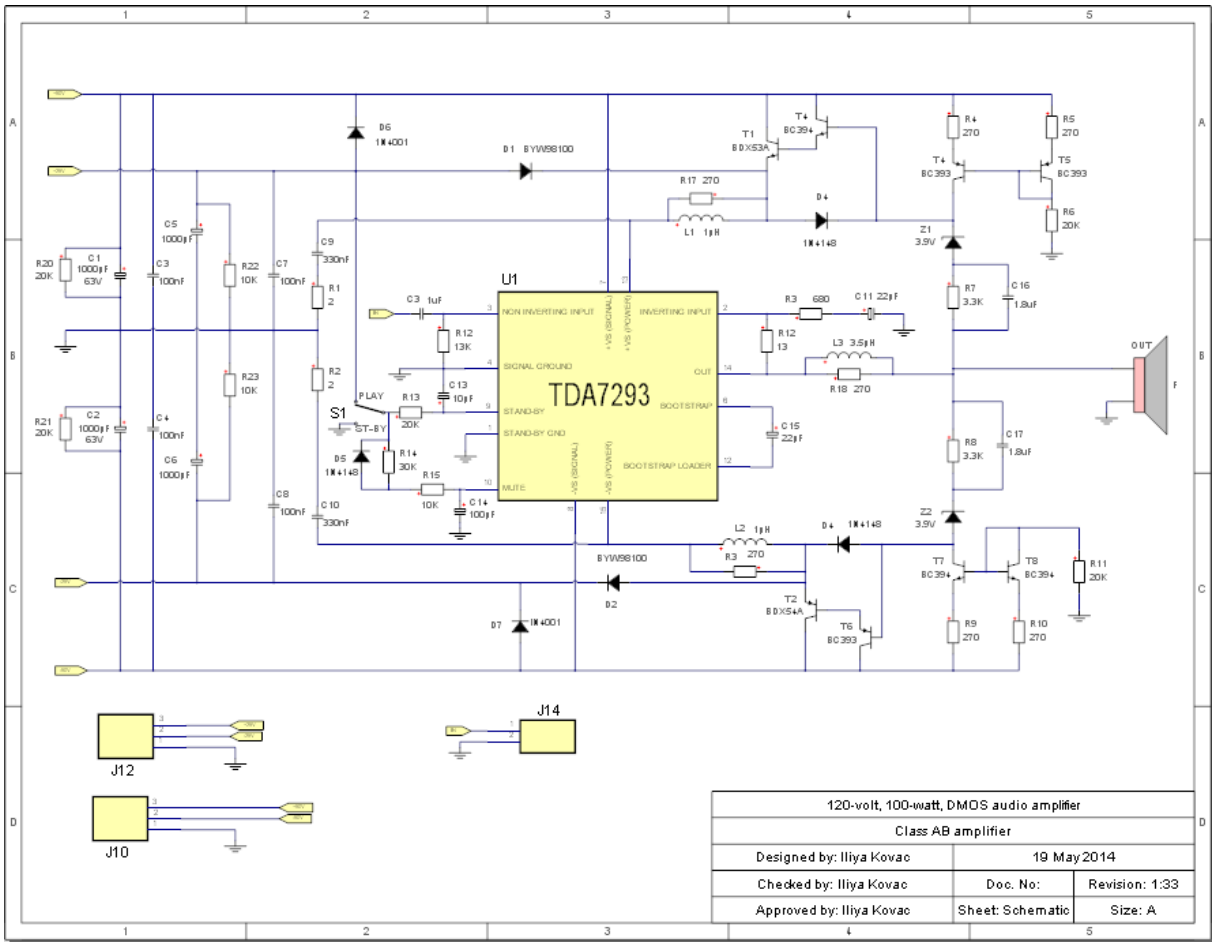
Typical Schematic

1.2.6.2.1 Schematics

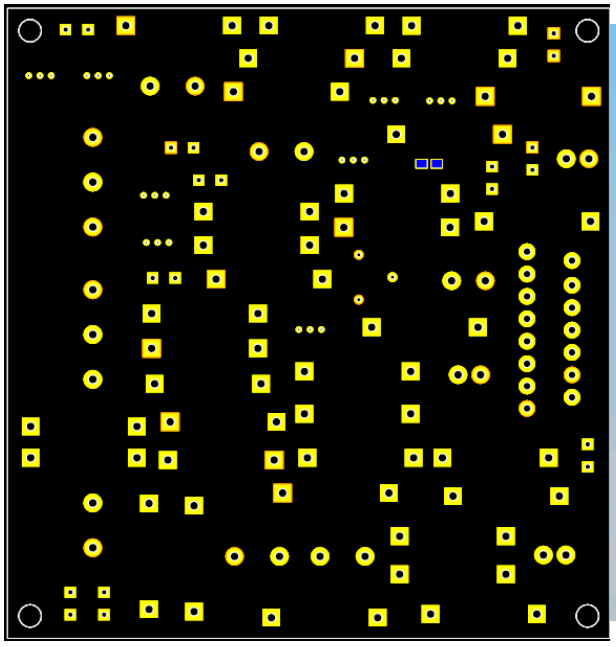
You place parts on a schematic by dragging them from the library panel or clicking on a part in the Parts menu.

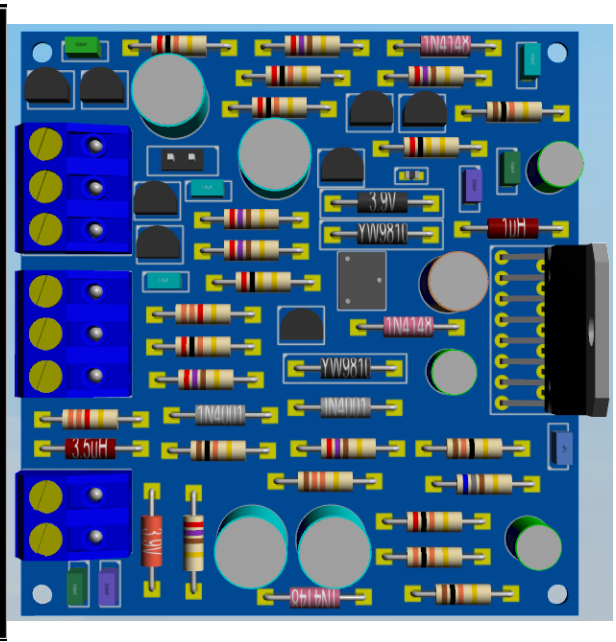


Parts placed on schematic

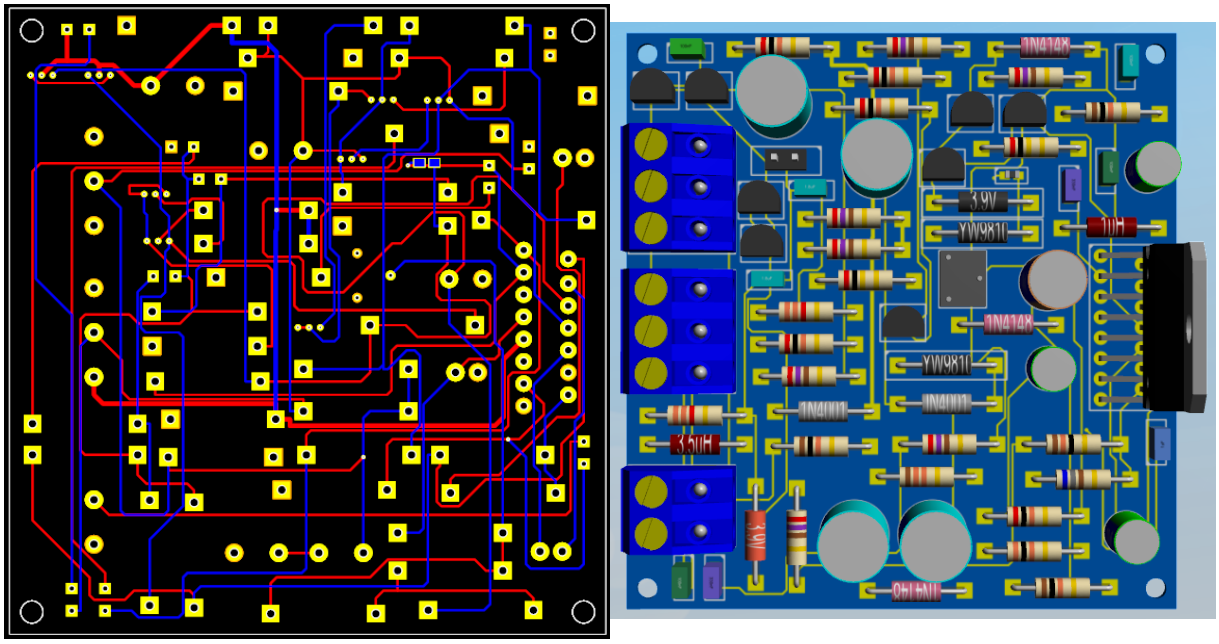


Parts wired together





PCB without schematic wiring



PCB after schematic wiring and PCB routing

1.2.6.2.2 Adding Parts

You can add parts to a schematic by:

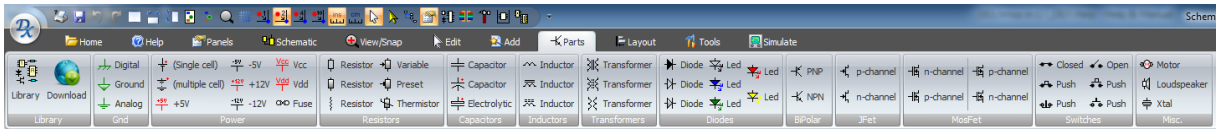
Using the Menu

Using the Library Panel

Adding Resistors

1.2.6.2.2.1 Using the Menu

You can add parts by clicking on any of the part buttons in the **Parts menu tab**.



Once clicked, move the mouse in the schematic viewport to drag the part.

Rotating the Part

Press the ***Space Bar*** to rotate the part 90° as you drag the part.

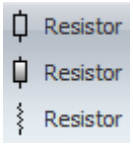
Menu Parts Library

The parts that are added from this menu are in the **XXXX**/___SystemDoNotRemove directory where **XXXX** is the location of the parts library.

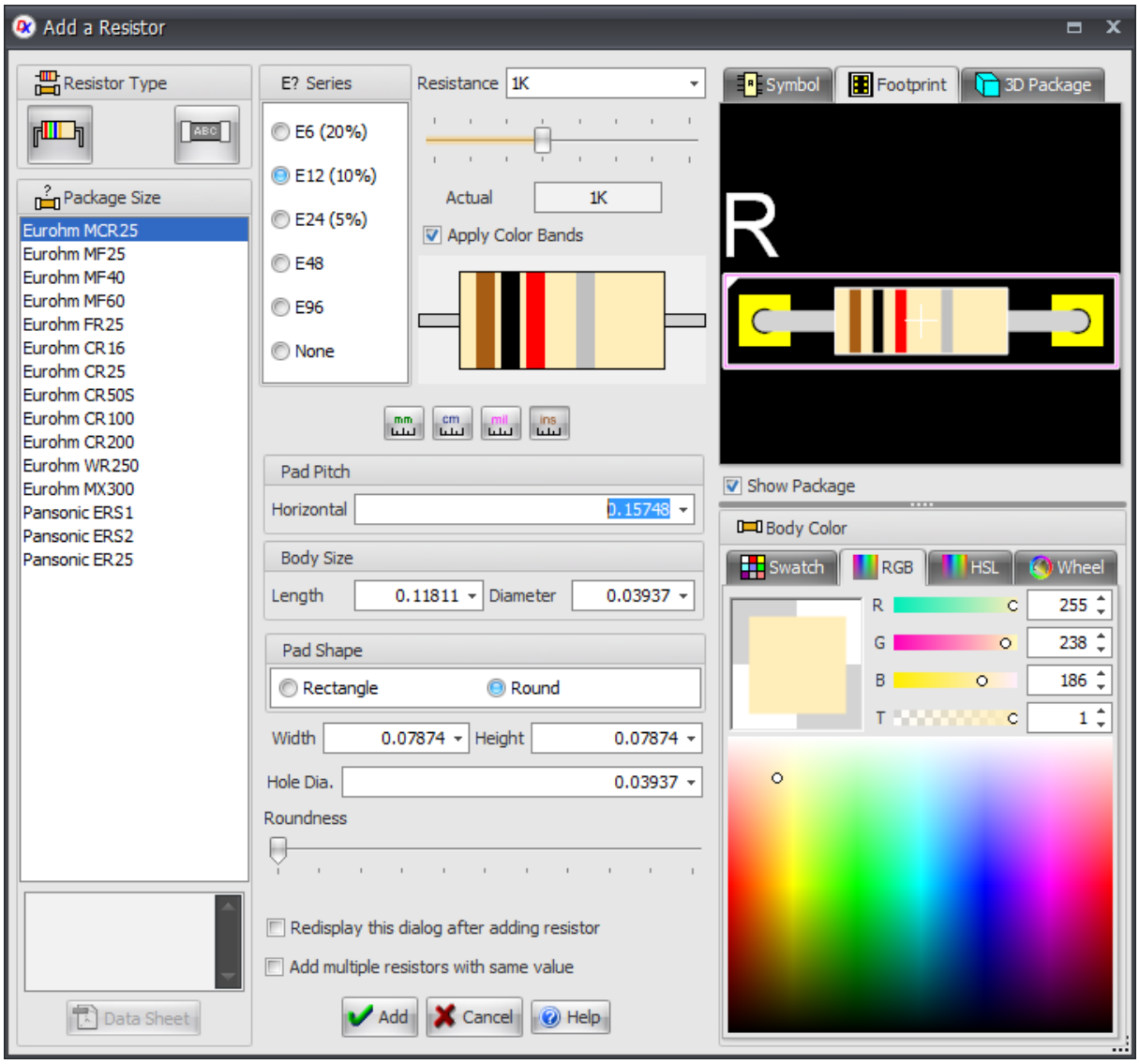
1.2.6.2.2.2 Using the Library Panel

You can use the library panel to drag parts onto your schematics.

1.2.6.2.2.3 Adding Resistors

To add a resistor click on of the 

buttons in the **Parts→Resistors** button group. The add resistor dialog shown below will appear.



Add Resistor Dialog

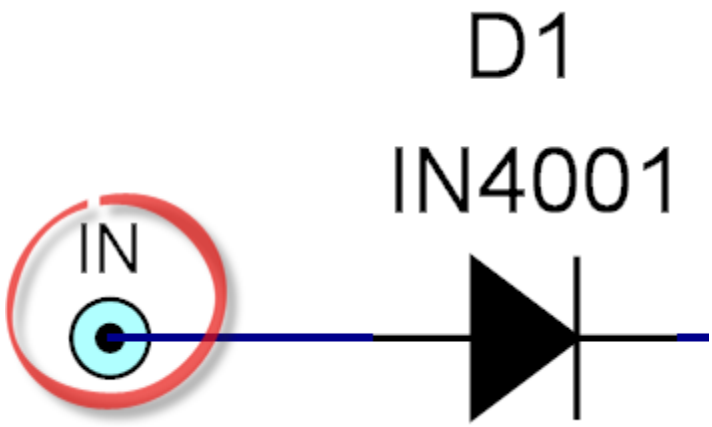
1.2.6.2.3 Hierarchical Design

In AutoTRAX DEX you can create an hierarchical top down design with your schematics if you wish.

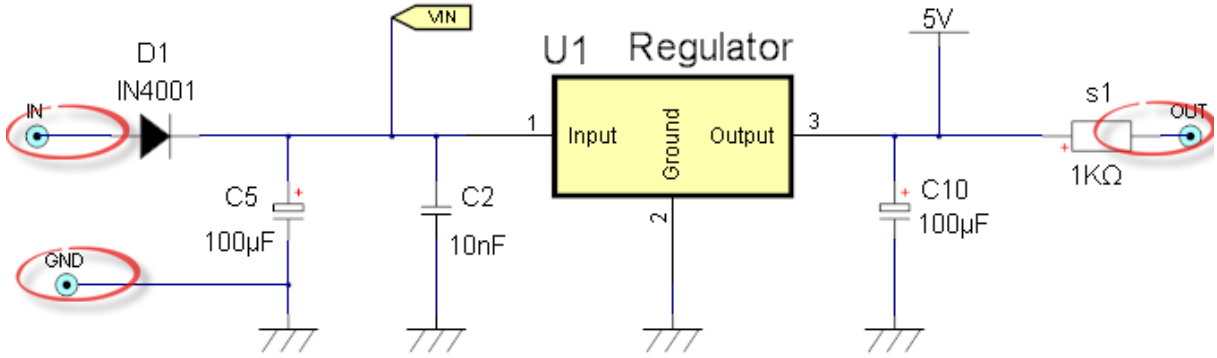
1.2.6.2.3.1 Sub-Systems

A sub-system is a circuit in a separate schematic that contains ports.

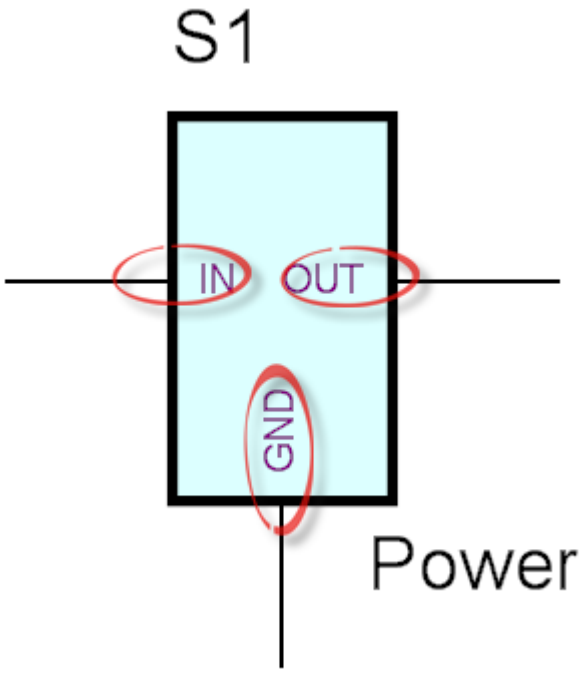
The sub-circuit can then be reference as sub-system symbols in other schematics.



Port

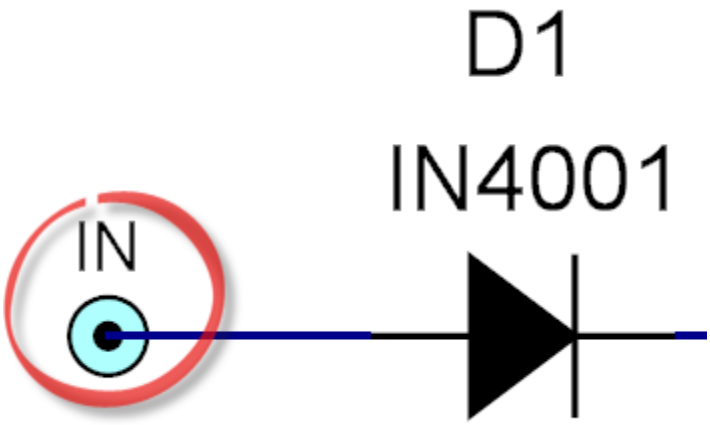


Sub-system with Ports

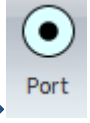


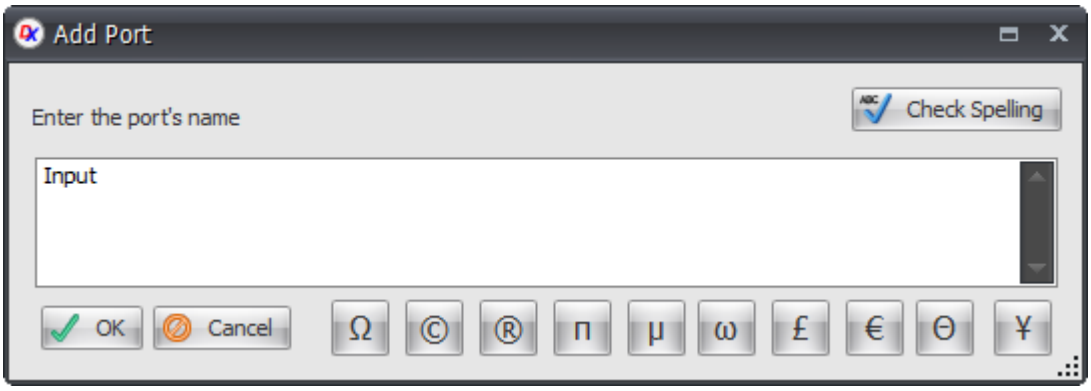
Sub-system reference

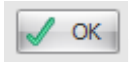
A port is a connection point in a sub-system that appears as a terminal on sub-system reference symbols.



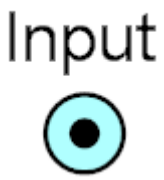
Port

To add a port to a sub-system click the **Add→Sub→**  button.



Enter the name for the port and click the 

button.



Drag the port and ***left-click*** to finally place it.

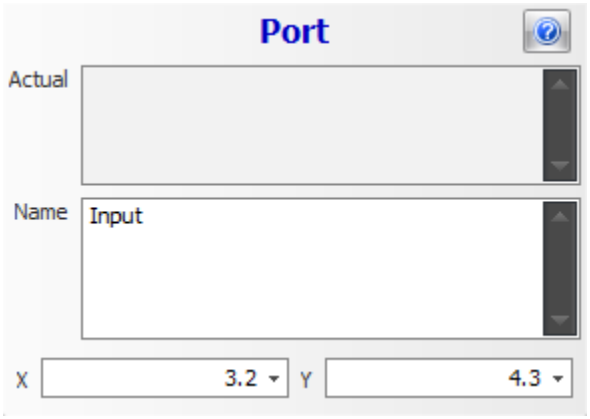
There are 2 ways to edit a port, dragging it or using the Properties panel.

Dragging

You can ***click and hold*** on a port to select it and then drag the mouse to move the coordinate. You will see the X and Y values of the coordinate change as you move it. Press the ***'s'*** key to turn snap on or off.

Properties Panel

If you select a port and the Properties panel is visible, the port properties dialog will be shown.



Actual

The actual for the port.

Name

The name for the port.

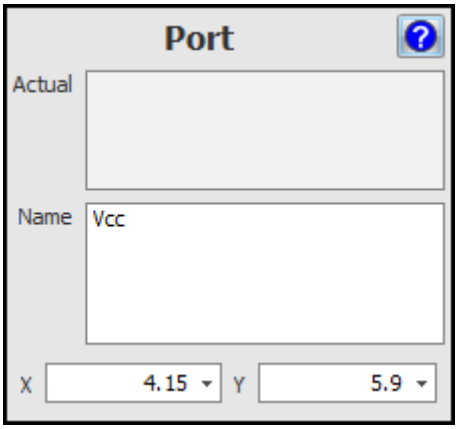
X

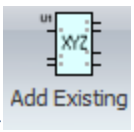
The X or horizontal position of the port.

X

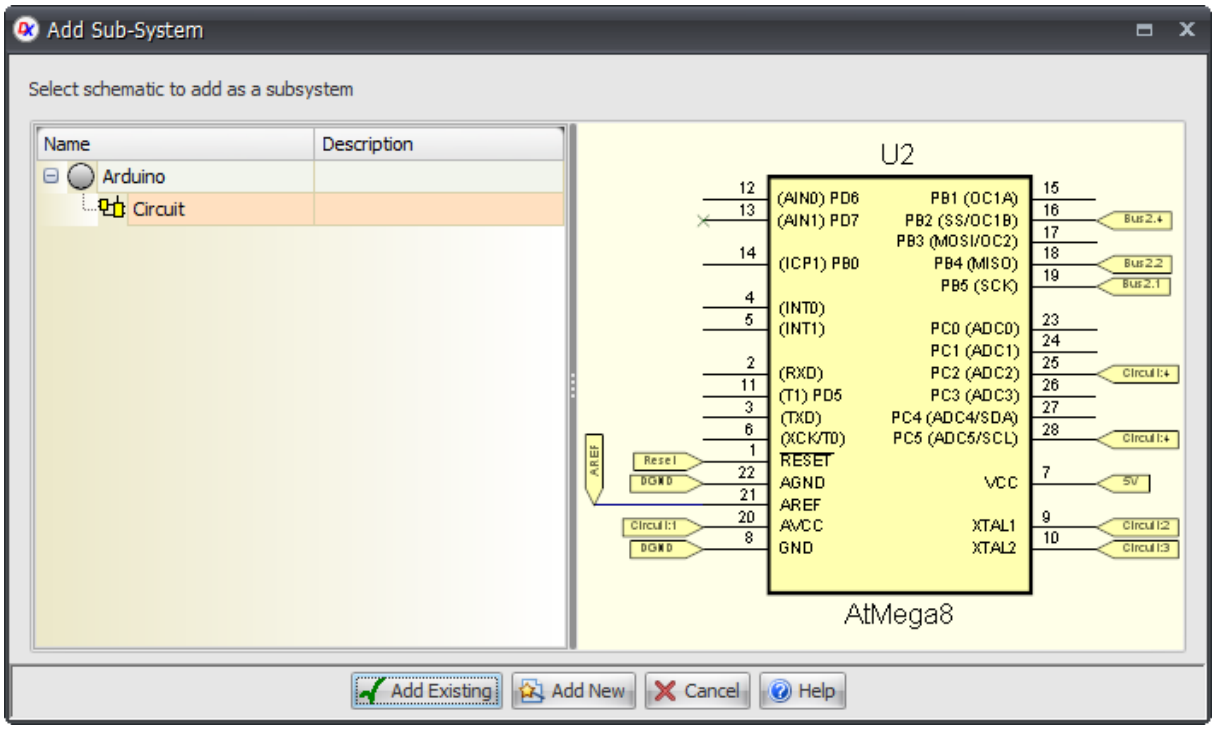
The Y or vertical position of the port

Pressing the  button displays this help page

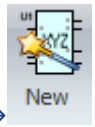


To add a reference to a sub-system click the Add→Sub→ 

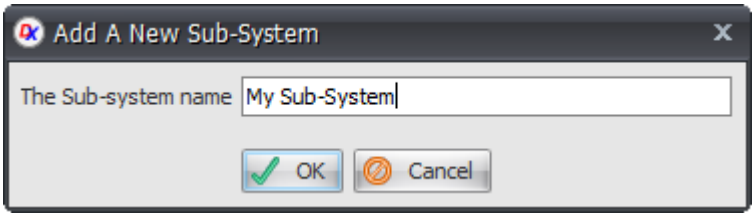
button. The add Sub-System dialog will appear.

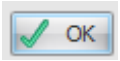


The Add Sub-System Dialog

To add a sub-system to your design click the Add→Sub→  button.

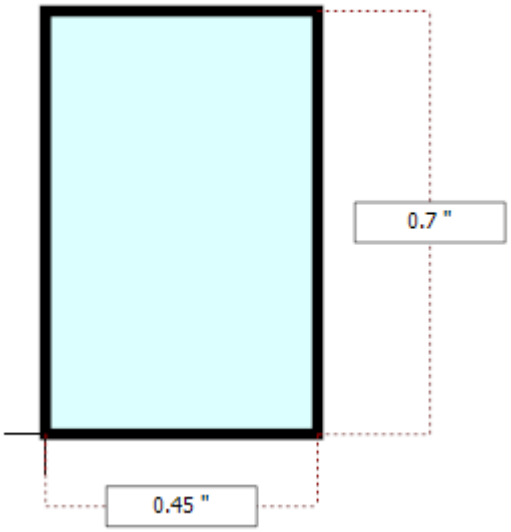
You will be prompted for the name of the sub-system.



Enter its name and click the  button.

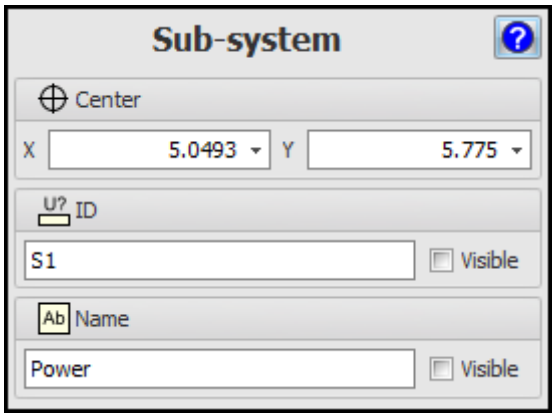
Move the mouse in the schematic viewport. As you mouse the mouse the point cursor shown below will appear. ***Left-click*** when the point cross is where you want the start corner of the sub-system or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to precisely place the start corner of the sub-system.

As you move the mouse you will see the sub-system re-size.

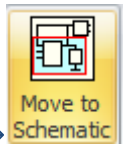


Click the left mouse button when it is the correct size press the Enter key followed by the Width and Height coordinates of the sub-system shape.

A sub-system is just like a Part symbol. See editing part symbols.



1.2.6.2.3.2 Refactor

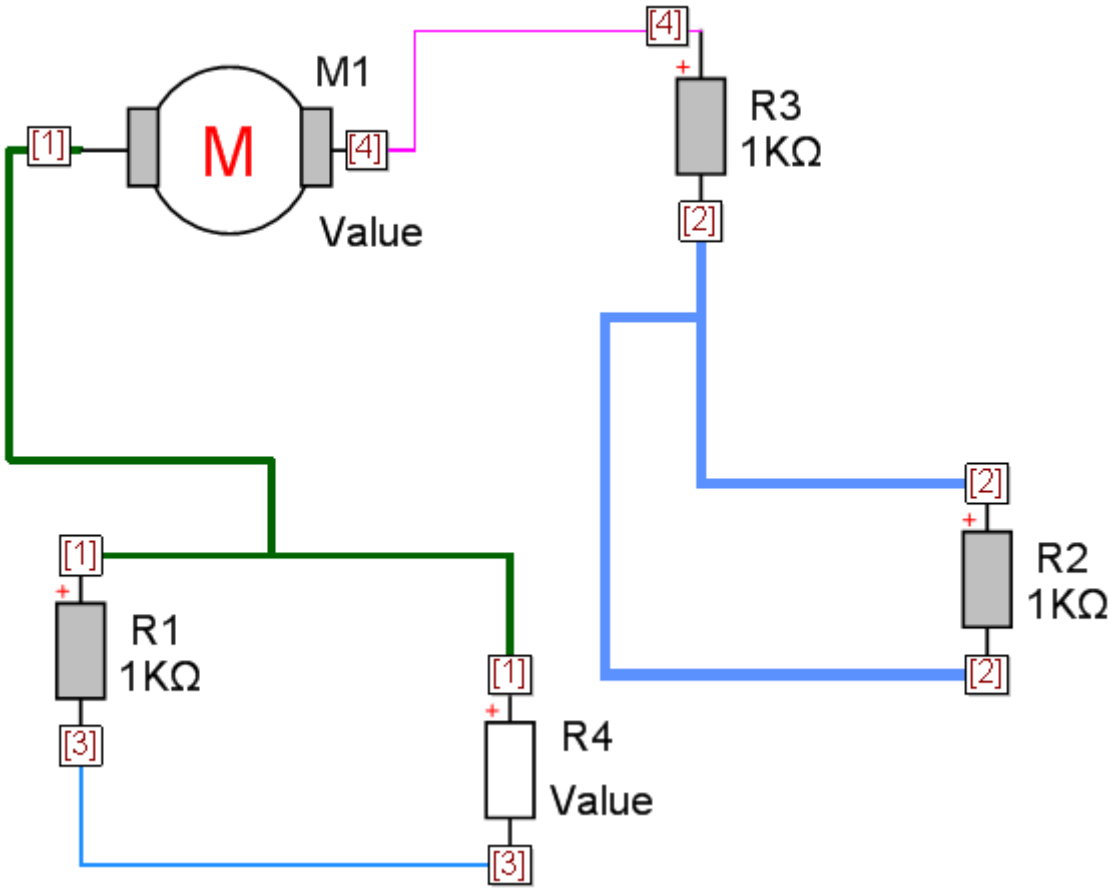
To refactor a part of a schematic design and place it in a separate schematic, select the parts to move and click the Edit→Refactor→ 

button.

A separate sub-system schematic will be placed and ports added.

1.2.6.2.4 Wiring Parts Together

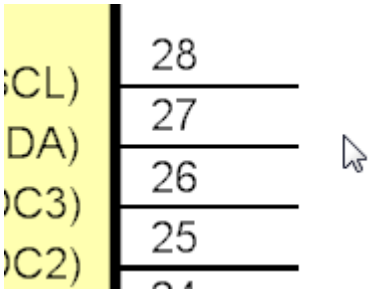
AutoTRAX DEX can wire part terminals together.
Each wire forms part of an electrical node.
Each node can have its own color and line width.
AutoTRAX DEX can optionally show node IDs where nodes join to part terminals.



1.2.6.2.4.1 Adding Wires

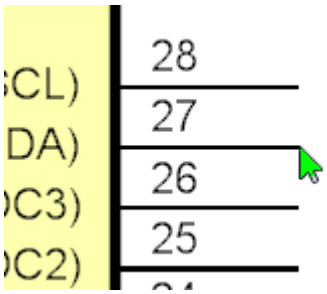
To add a wire:

As you mouse the mouse the standard mouse cursor will be shown.



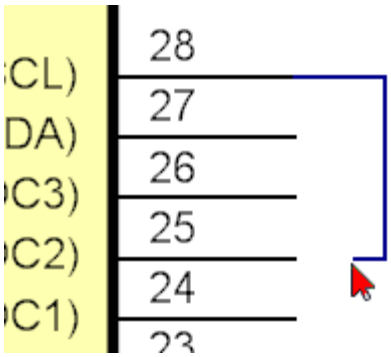
No Connect

Move the wire over a terminal. The cursor will change to green. Click the left mouse button to start the wire.



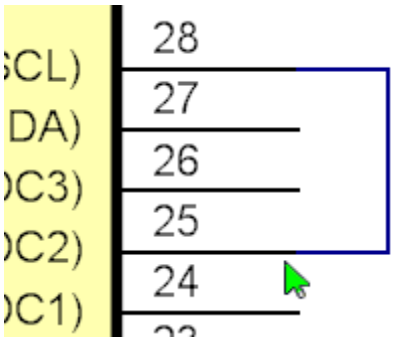
Start OK

Now drag the mouse. The wire will be drawn and a red arrow cursor will be shown.



End Not OK

Move the wire over a terminal. The cursor will change to green. Click the left mouse button to end the wire.



End OK

1.2.6.2.4.2 Editing Wires

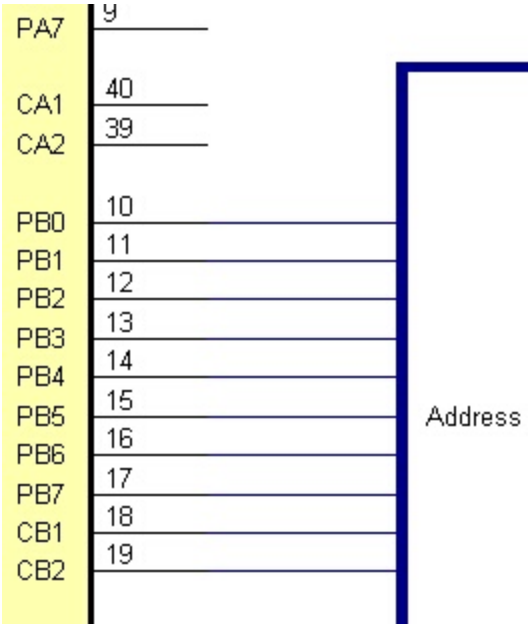
To edit a wire ***double-click*** on it.

To delete a wire, select it and the delete it.

1.2.6.2.4.3 Using Buses

A bus represents a collection of electrical wires on a schematic sheet. The collection of wires are related in functionality, e.g. an address bus contains a collection of bus wires, say A0 to A31 for a 32 bit wide address bus. You connect several wires to a bus and the bus 'carries' the wires around the schematic grouped into a virtual single cable conduit.

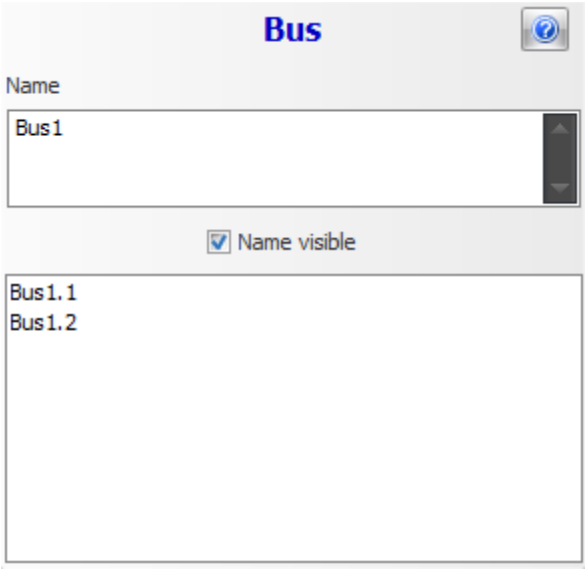
Below is bus named Address which carries connections from the device on the left.



To add a bus click the Add→Bus→ 

button.

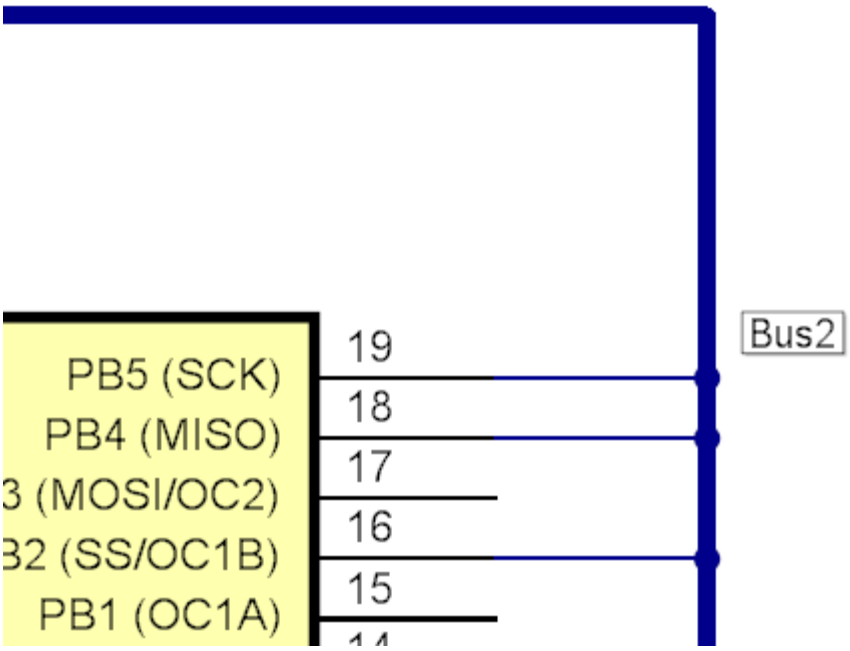
To edit a bus first select it.



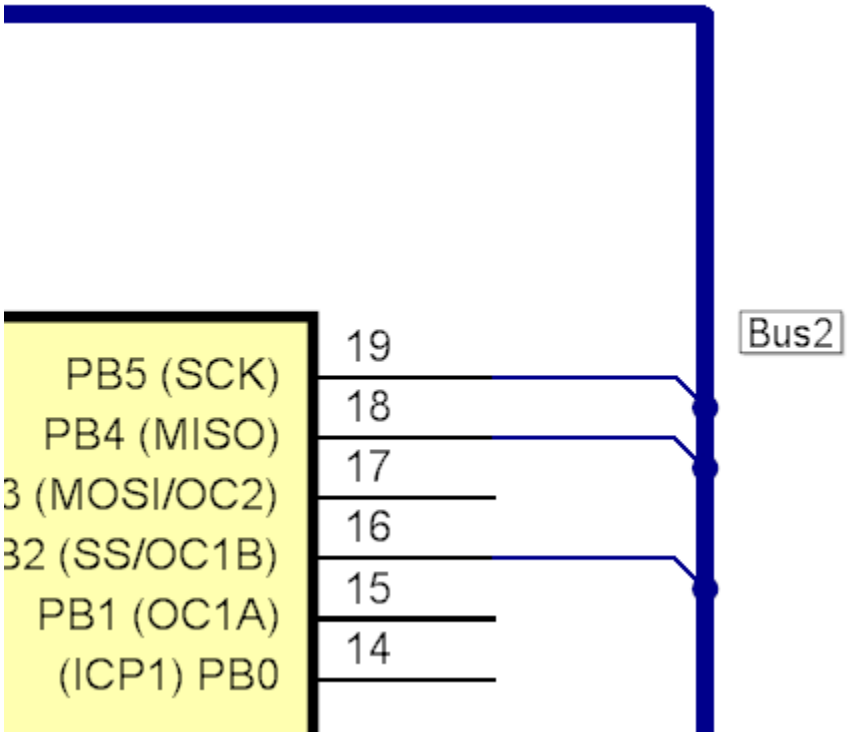
1.2.6.2.4.4 Beveling Wire to Buss Connections

To bevel/unbevel the wire to bus connectors click the Add→Bus→ 

button.



Unbeveled



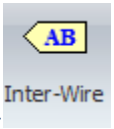
Beveled

1.2.6.2.4.5 Inter-Wire Connectors

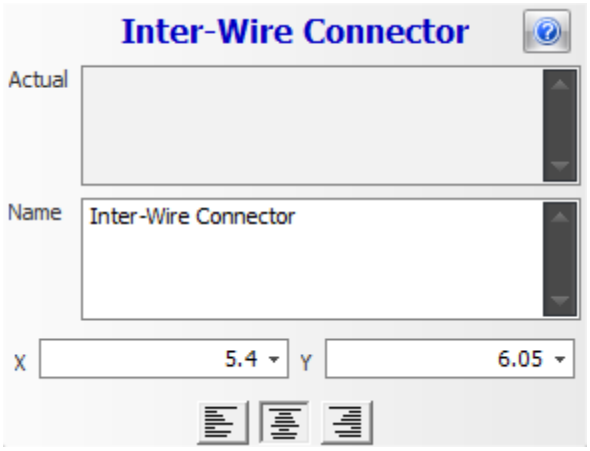
Also called off-page connectors.

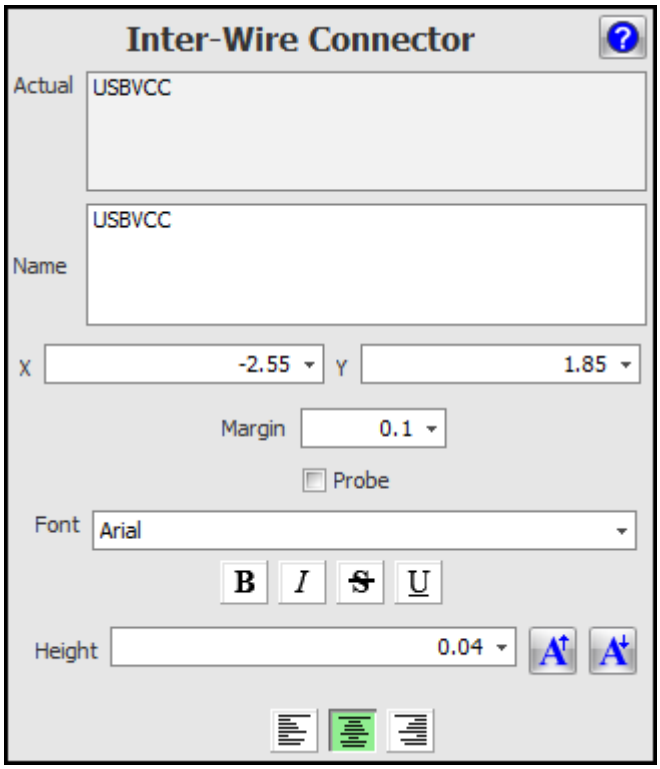
An off-page connector is a symbol terminal that a wire connected to it will connect to all wires connected to off-page connectors with the same node name. Like space world tunnels that science fiction enthusiasts love.

An inter-wire connector

To add a bus click the Add→Wire→  button.

To edit an inter-wire connector first select it.

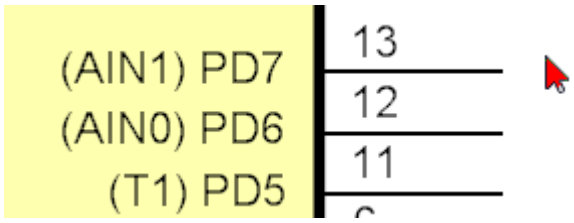




1.2.6.2.4.6 Marking a Terminal as No Connection Required

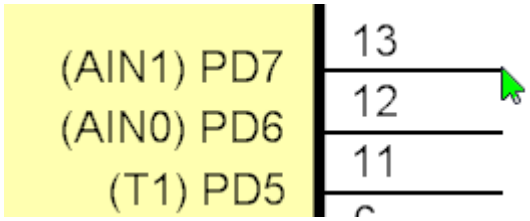
To mark a terminal as 'no connection required' click the Add→Wire→ button.

As you move the mouse in the schematic you will see a red cursor.



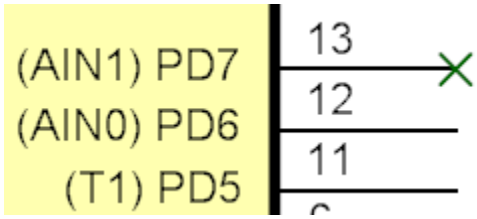
Red 'no terminal' cursor

If you move the mouse over a terminal it will turn green.



Green 'terminal' cursor

Left-click to mark a terminal as 'no connection required'. A green diagonal cross will be added to the terminal to signify that no connection is required for that terminal.



terminal marked as 'no connection required'

Deleting 'no connection required' Markers

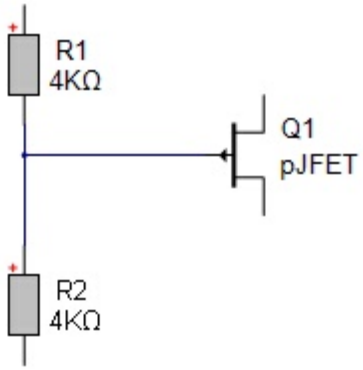
To remove the 'no connection required' marker, select it and the delete it.

1.2.6.2.4.7 Nodes and Nets

A node connects 2 or more electrical pins together or one or more schematics.

A node can consist of 1 or more wire segments. A wire connected a wire(using a wire junction) or a terminal to another wire via a wire(using a wire junction) or a terminal.

Below is a node consisting of 3 wire segments connecting R1, R2 and Q1.



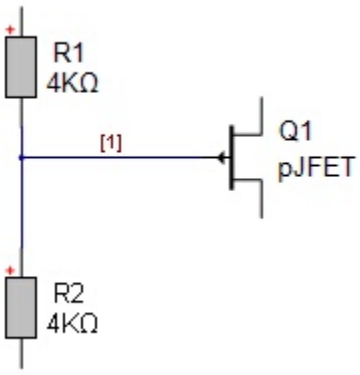
A terminal can be a

- A Symbol Terminal or
- An Inter-wire Connector or
- A Port.

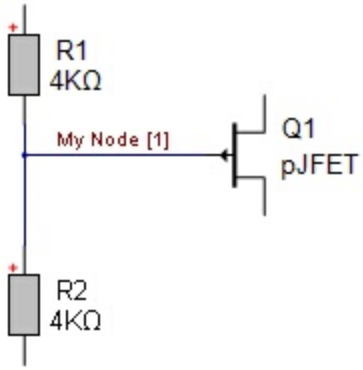
Symbols consist of zero or more symbol terminals and possibly some graphics to represent the symbol. Symbol terminals can be visible or invisible.

A net defines the physical representation of a node on a PCB and defines which part pins (pads on footprints) are connected together.

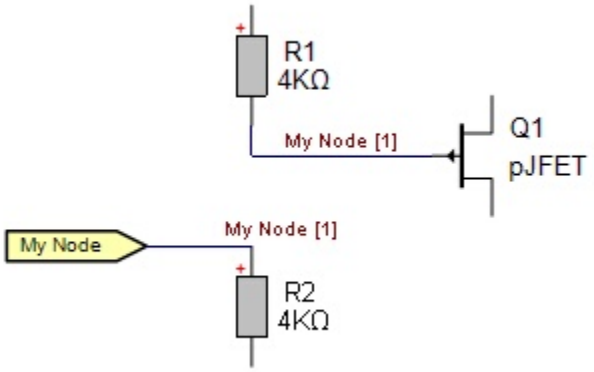
Each node has a node index and an optional node name. By default nodes do not have a node name. Below, the node is shown with the node index displayed. (Use ribbon menu View→Node Names).



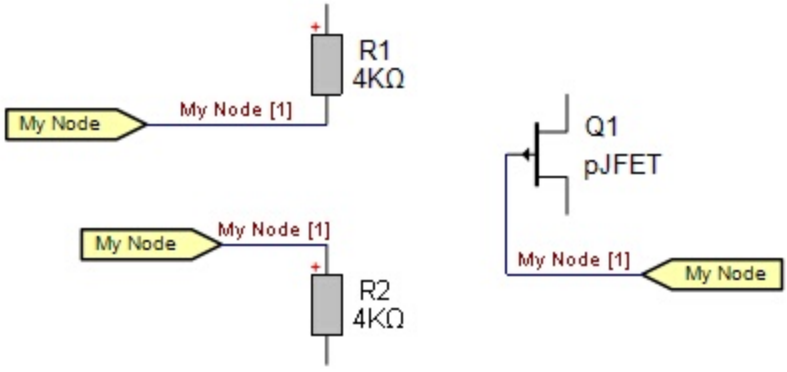
You can set the node name by selecting it by clicking on any of its wires and setting the nodes name using the properties panel. (***right-click*** and select Properties Panel). Below shows the node after the node's name has been set to 'My Node'.



Instead of connecting terminals together with wires you can connect terminals to an inter-wire connector and name the inter-wire connector node by selecting it and setting its name using the Properties Panel. Below the top terminal of R2 and been connected to an inter-wire connector whose name has been set to 'My Node'. This connects R1, R2 and Q1 together. You can place as many inter-wire connectors as you wish on the same schematic or different schematics and all wire connecting to them will be connected together into a single node. The PCB net will then connect all the package pins together.



You can take this to the extreme and connect all terminals to named inter-wire connectors as shown below. Again R1, R2 and Q1are connected together.

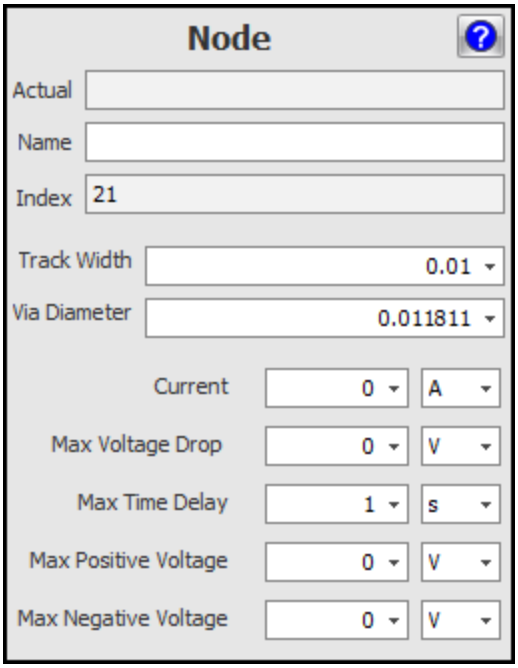


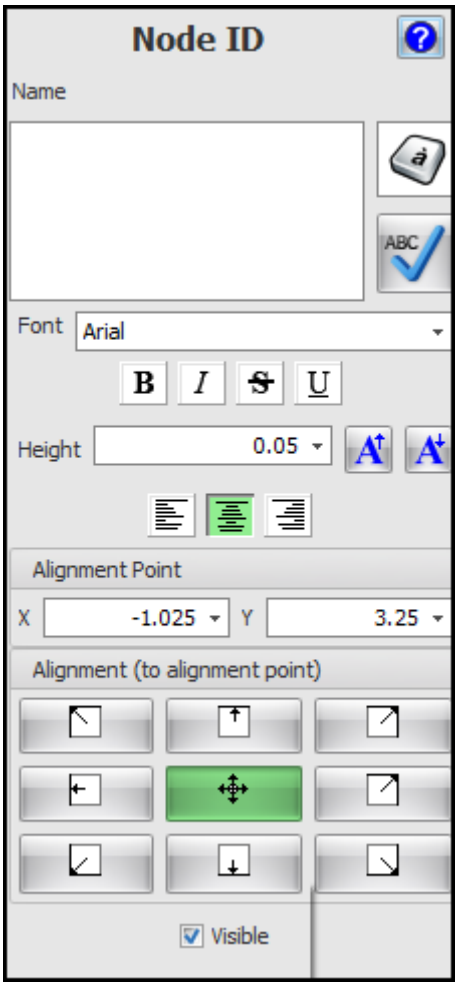
You use inter-wire connectors on the same sheet to reduce wire clutter. You would use them on different schematics to connect wires between sheets. Copy and paste inter-wire connectors works fine.

Ports can also be used to connect wires together on different sheets. Ports let you use a schematic as a sub-system on other sheets.

1.2.6.2.4.8 Nodes

Nodes are collections of wires on a schematic that connect 2 or more terminals together.





1.2.6.2.4.9 Wires

Wires are single connections between exactly two of the following:

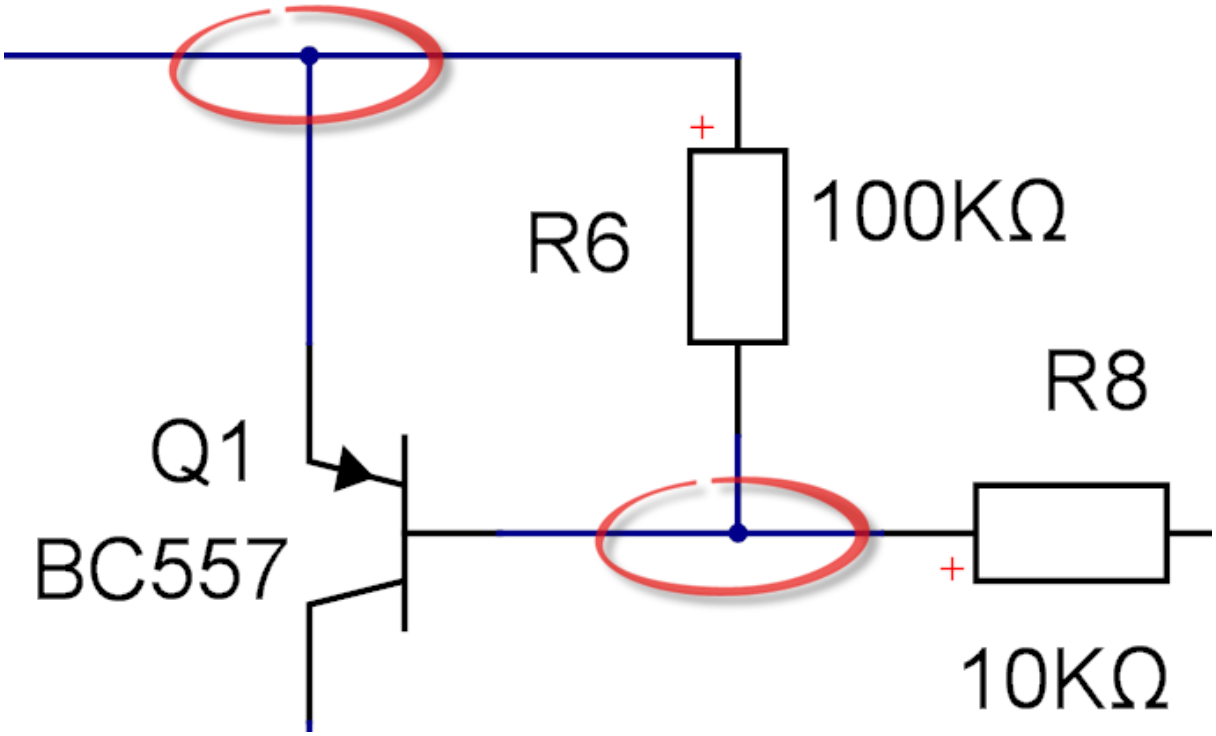
- symbol terminals
- wire junctions (where 3 or more wires meet)
- off-page connectors
- bus terminals. (terminal points on a bus)

Wire Junctions

All wire junctions are created/deleted automatically.

Created when starting a wire from another or ending a wire on another.

Deleted when only 2 wires connected to a wire junction. Junction trashed and wires added together.

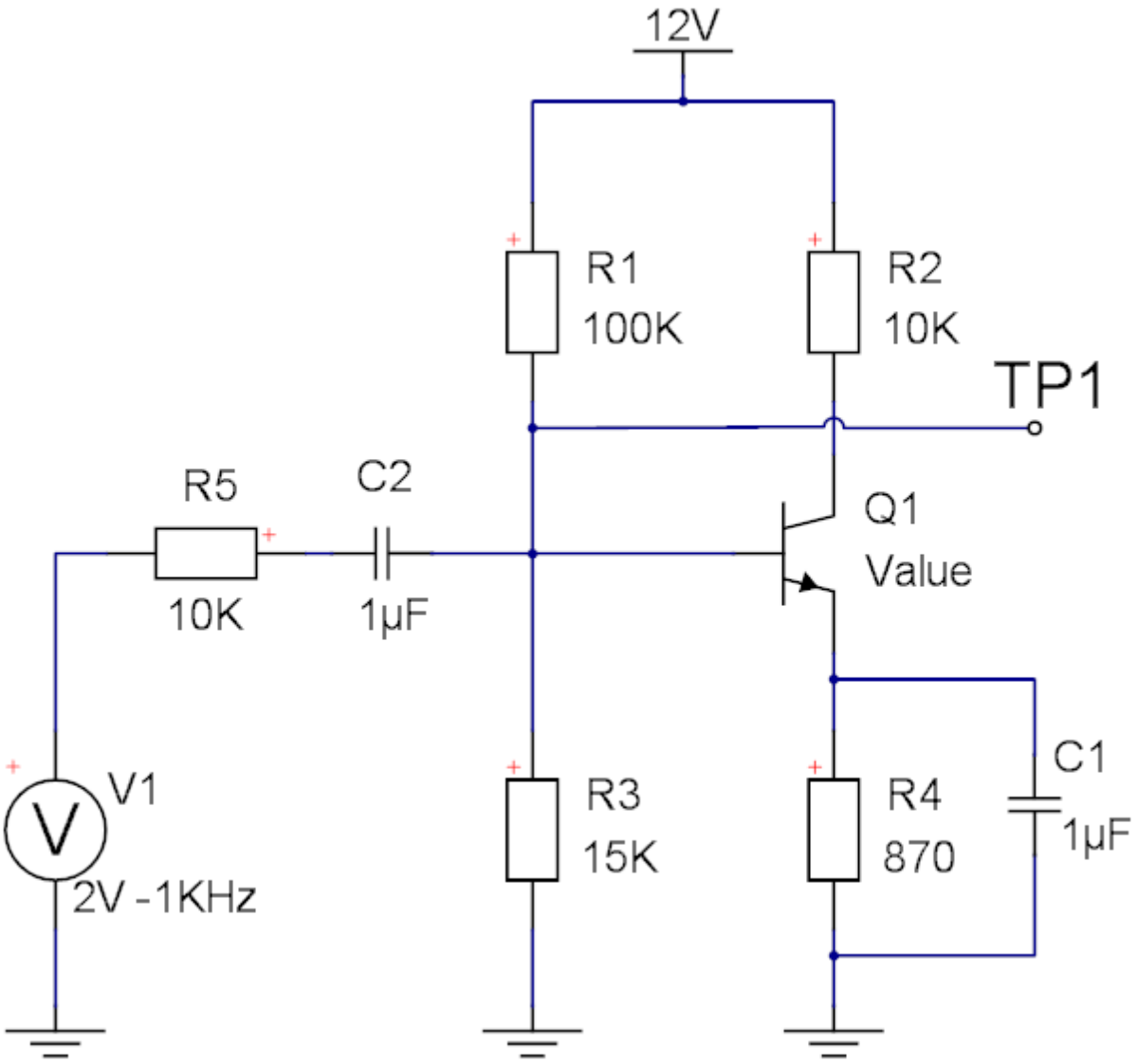


Wire Junctions

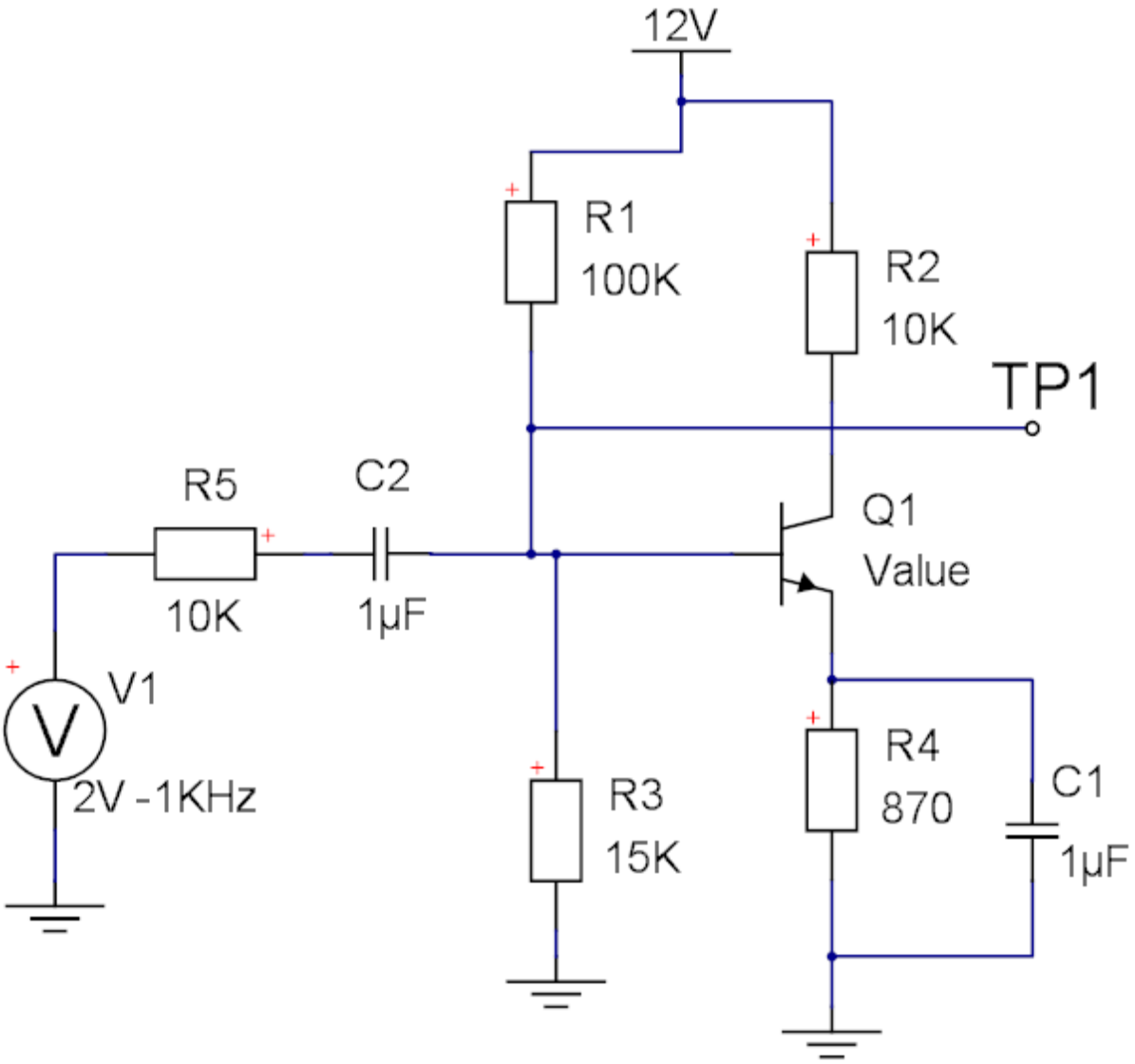
1.2.6.2.4.10 Style Guide

- Use Jumpers for wire crossings.
- Avoid wire crossings. You often can do this by moving and/or rotating parts. (TP1 can be moved)
- Align parts. e.g. V1, R1,R2 and C1, horizontally, R1 and R2 vertically.
- Use common point to make circuit function obvious and also pleasing to the eye.
- Center the schematic on the page.
- Show signal flow from left to right. Inputs on the left, outputs on the right.

However, as always with AutoTRAX DEX, the choice is yours.



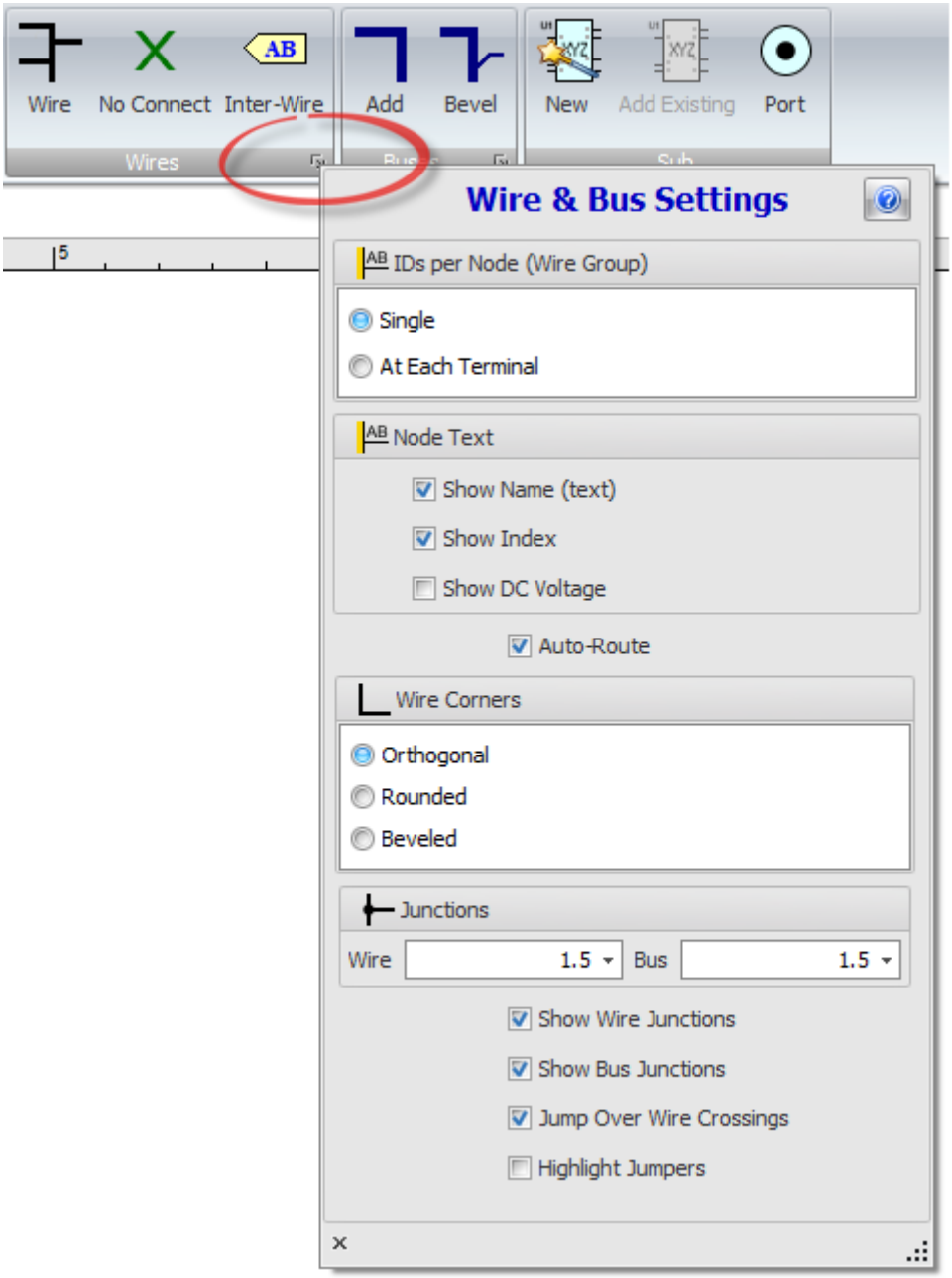
Good



Ugly and ambiguous (jumper)

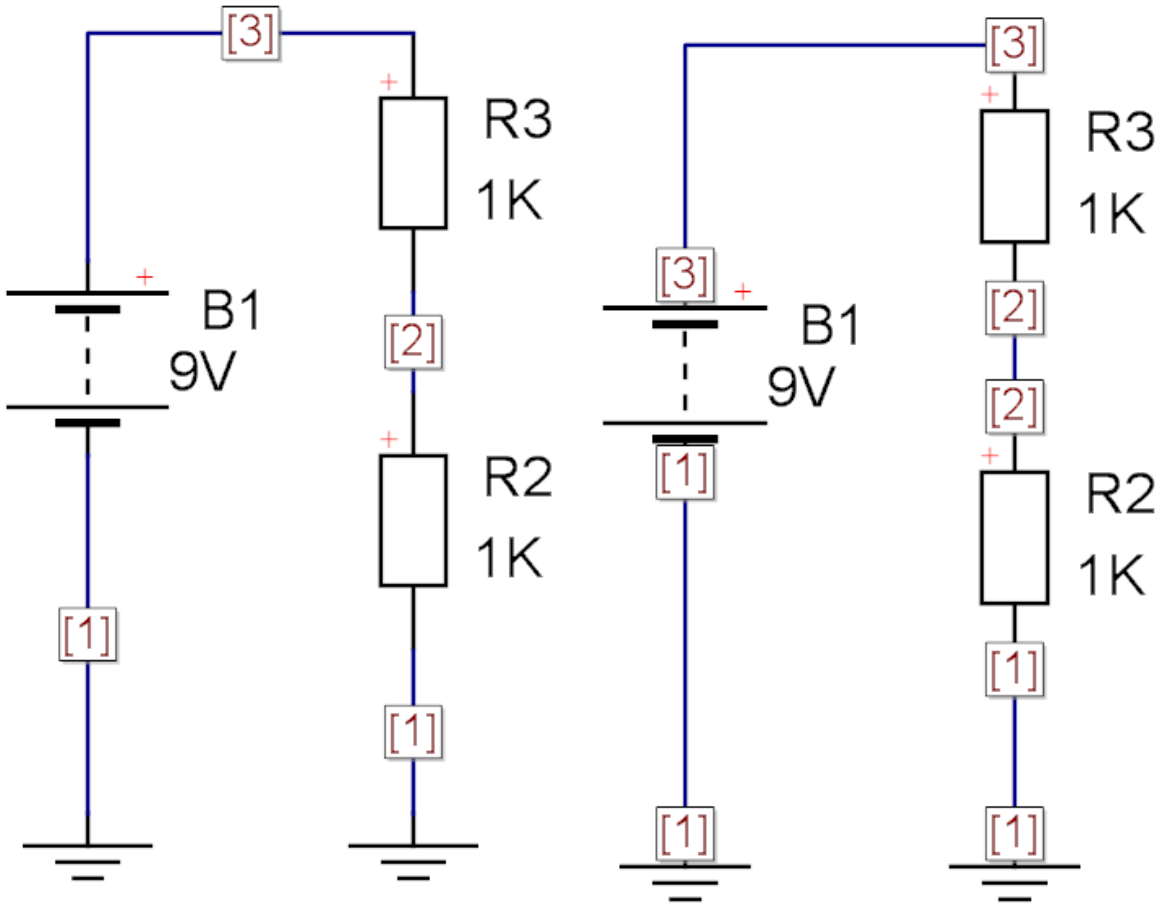
1.2.6.2.4.11 Wire Settings

To set the schematic wire setting click the small button at the bottom right of **Add→Wires** ribbon button group.



Click  to display this help topic.

IDs per Node (Wire Group)



Single **Check this to display only one node ID per node.**

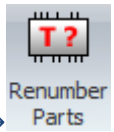
At Each Terminal **Check this to display a node ID at each terminal.**

Node Text

Show Name (text). Check to display the nodes textural name.

Show Index. Check to show the numerical node index.

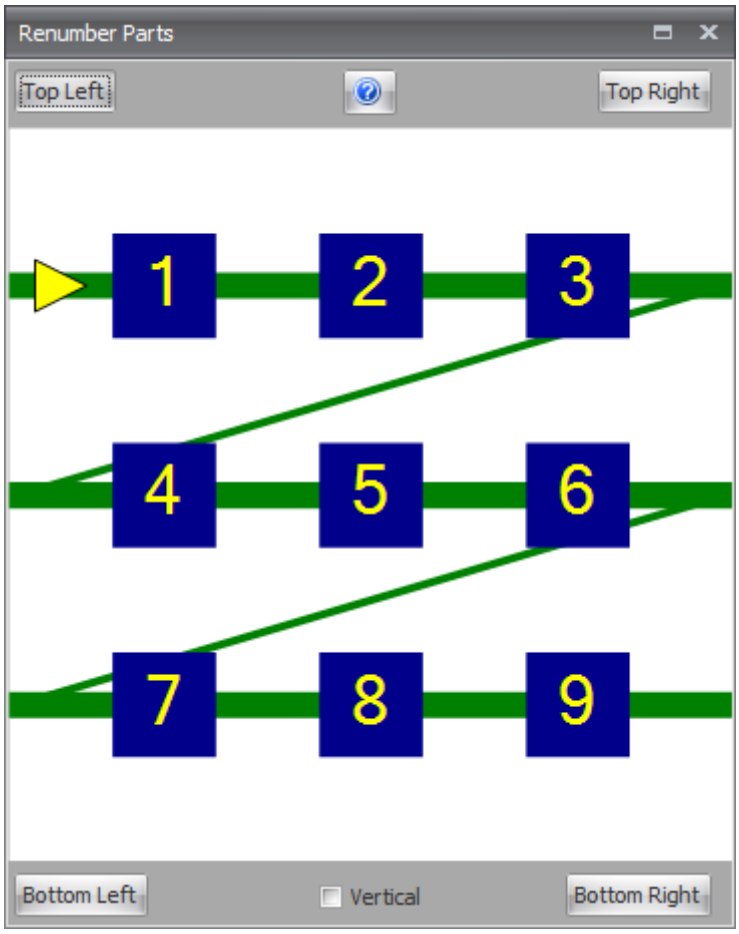
1.2.6.2.5 Renumbering Parts

To renumber parts click the Tools→Utilities→ 

button.

The renumber dockable panel shown below will appear.

This will renumber all part references (designators).



The Renumber Parts Panel

1.2.6.2.6 Reference Designators

In AutoTRAX DEX you can change the orientation, font and scaling of part reference designators in PCBs and schematics. In schematics you can also change the color.

1.2.6.2.7 Standard Reference Designators

A reference designator unambiguously identifies a component in an electrical schematic or on a printed circuit board. The reference designator usually consists of one or two letters followed by a number, e.g. R13, C1002. The number is sometimes followed by a letter, indicating that components are grouped or matched with each other, e.g. R17A, R17B. IEEE 315 contains a list of Class Designation Letters to use for electrical and electronic assemblies. For example, the letter R is a reference prefix for the resistors of an assembly, C for capacitors, K for relays.

Designator	Component Type
A	Separable assembly or sub-assembly (e.g. printed circuit assembly)
AT	Attenuator or isolator
BR	Attenuator or isolator
C	Capacitor
CN	Capacitor network
D	Diode (including Zeners, thyristors and LEDs)
DL	Delay line
DS	Display
F	Fuse
FB or FEB	Ferrite bead
FD	Fiducial
FL	Filter
G	Generator or oscillator
GN	General network
H	Hardware
HY	Circulator or directional coupler
J	Jack (least-movable connector of a connector pair) | Jack connector (connector may have "male" pin contacts and/or "female" socket contacts)

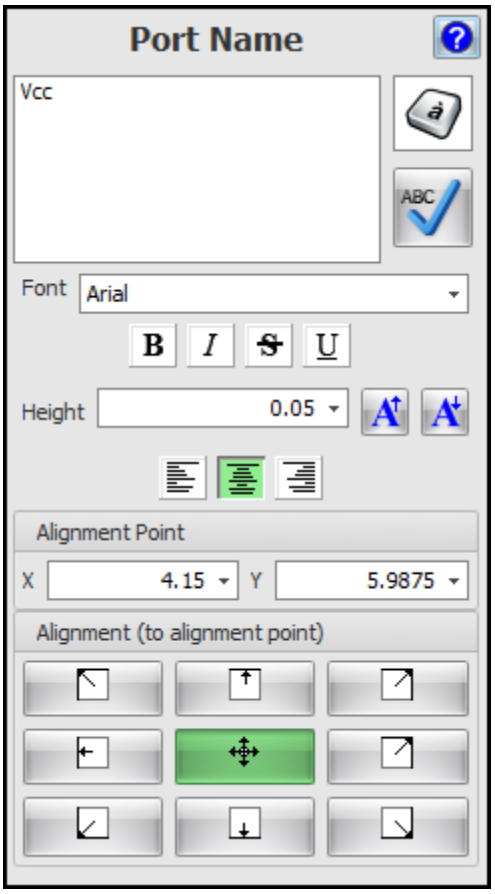
Designator	Component Type
JP	Link (Jumper)
K	Relay or contactor
L	Inductor or coil or ferrite bead
LS	Loudspeaker or buzzer
M	Motor
MK	Microphone
MP	Mechanical part (including screws and fasteners)
P	Plug (most-movable connector of a connector pair) | Plug connector (connector may have "male" pin contacts and/or "female" socket contacts)
PS	Power supply
Q	Transistor (all types)
R	Resistor
RN	Resistor network
RT	Thermistor
RV	Varistor
S	Switch (all types, including push-buttons)
T	Transformer
TC	Thermocouple

Designator	Component Type
TUN	Tuner
TP	Test point
U	Inseparable assembly (e.g., integrated circuit)
V	Vacuum tube
VR	Variable resistor (potentiometer or rheostat)
X	Socket connector for another item not P or J, paired with the letter symbol for that item (XV for vacuum tube socket, XF for fuse holder, XA for printed circuit assembly connector, XU for integrated circuit connector, XDS for light socket, etc.)
Y	Crystal or oscillator
Z	Zener diode

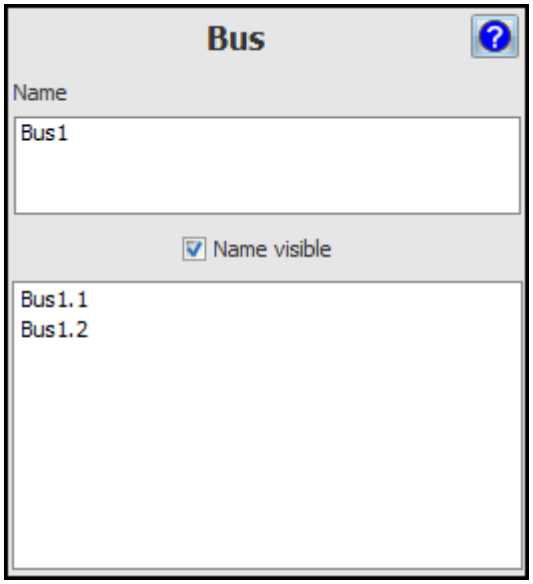
1.2.6.2.8 Saving Designs for Re-use

You can save all or part of a design to the design library panel and drag these sub-circuits onto schematics in other projects. This video shows you how.

1.2.6.3 Port Name Editor

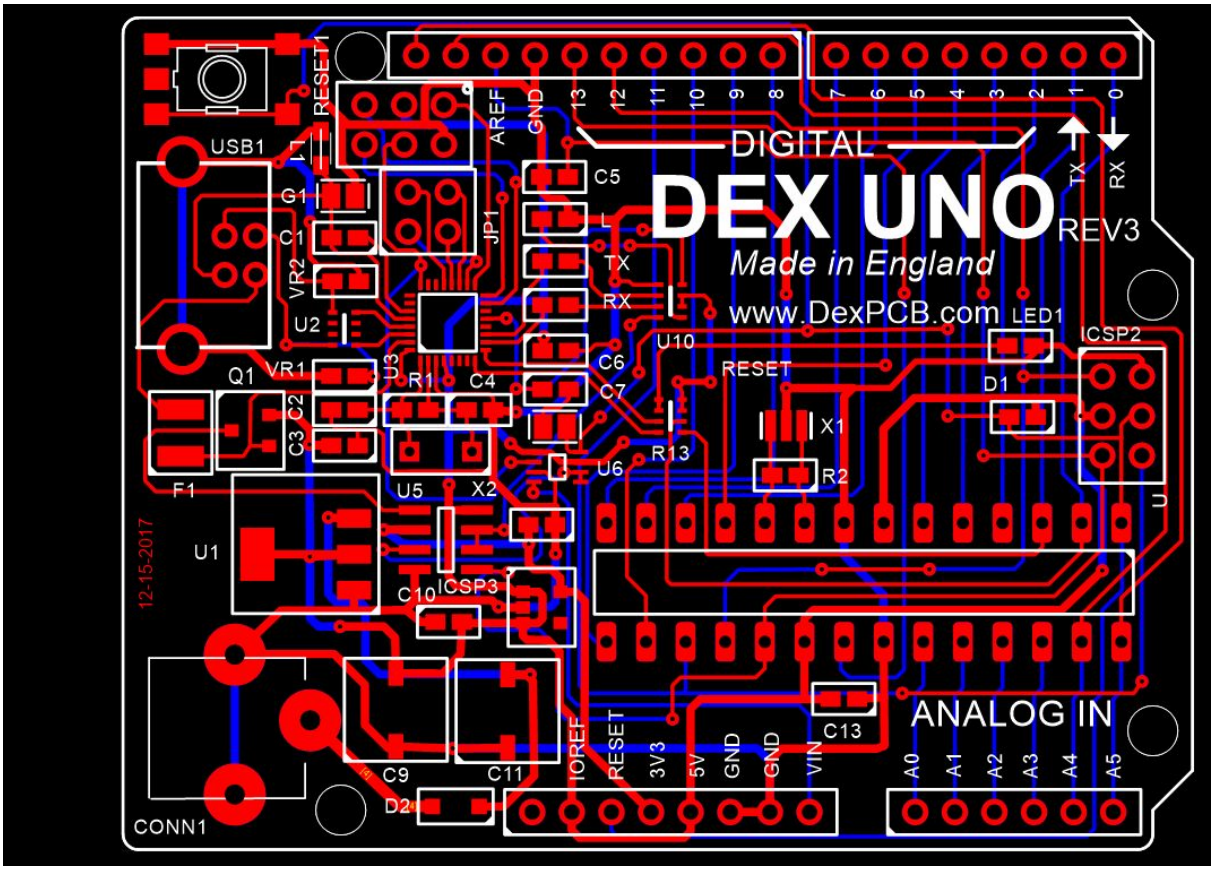


1.2.6.4 Bus Editor

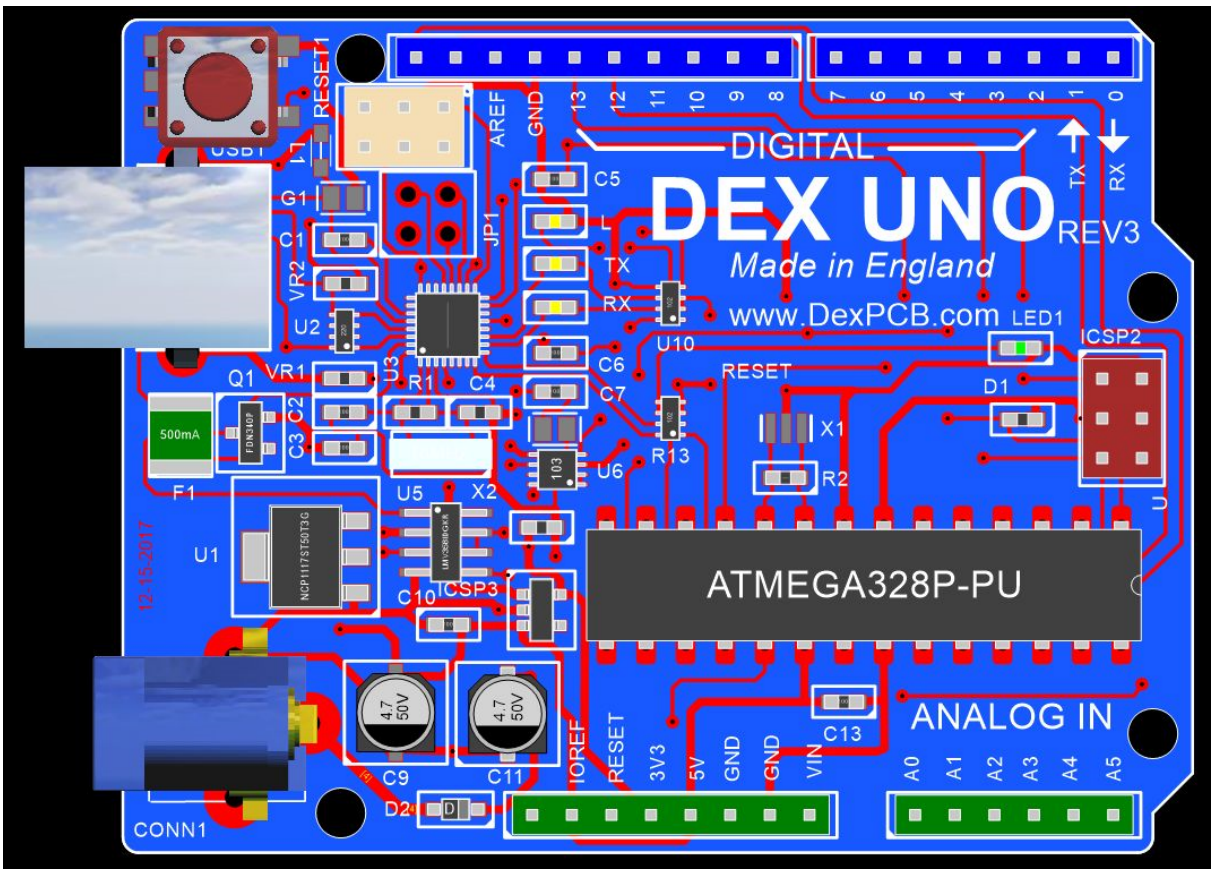


1.2.6.5 The PCB

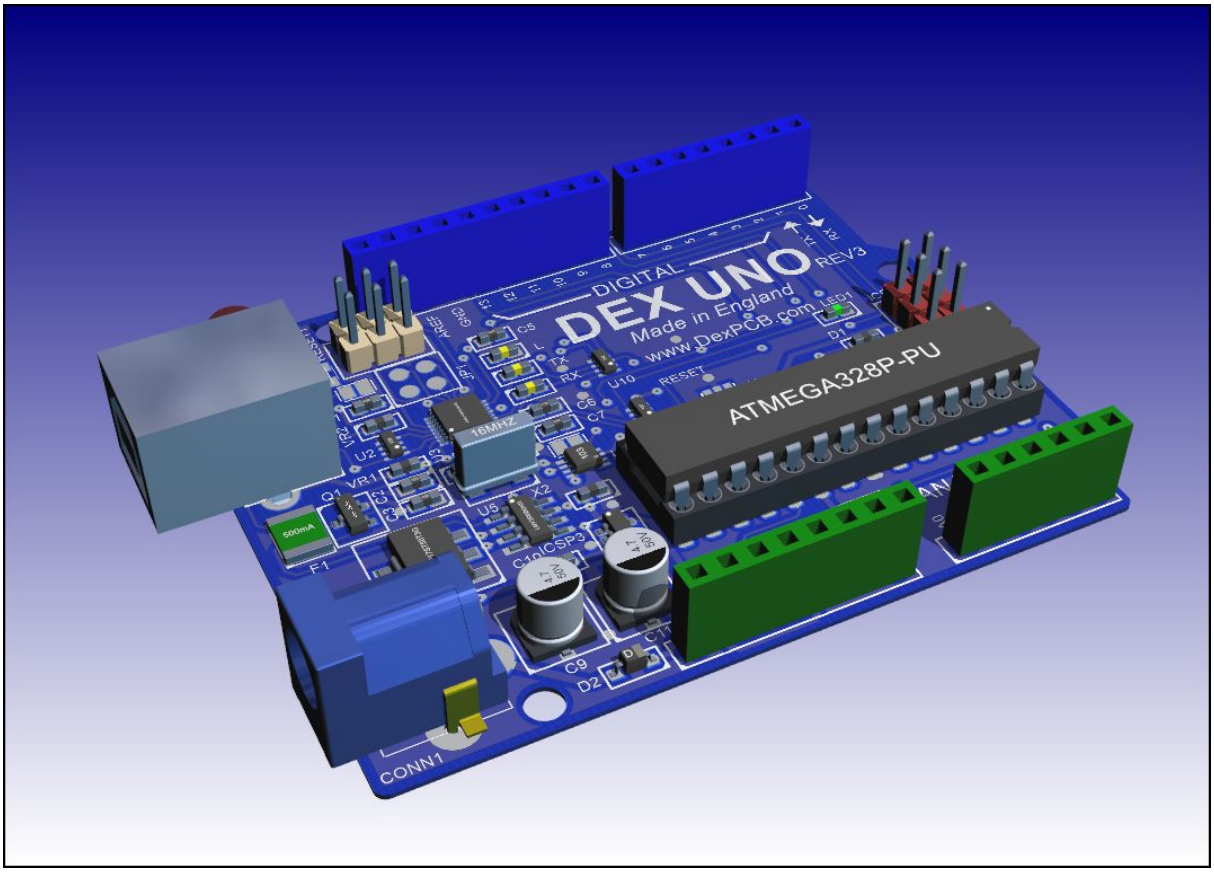
The PCB is the physical manifestation of your design and represents your ultimate goal: To make a PCB.



A typical PCB



The PCB showing the PCB filled and physical parts on the top of the PCB



The PCB viewed in 3-D

It consists of a PCB border, parts and electrical connections.

A PCB border

The border can have any number of cutouts and round hole. Cutouts are usually created with a CNC cutter and are expensive. Round holes are simply drill out.

Parts

You place the footprints for your physical parts inside the border and usually not covering holes and cutouts. However, sometimes you may place a part over a cutout/hole so design reasons.

Parts can be placed on the top or the bottom of the PCB.

The parts are connected together either by electrical copper tracks or via internal layers of solid copper called signal/power planes.

Copper Tracks (electrical connections)

Tracks are parts of electrical PCB nets.

A Net is a collection of pads that need to be electrically connected together. AutoTRAX DEX calculate a optimal unrouted non-cyclic* graph of unrouted track.

Nets map 1:1 with schematic nodes.

You manually route the unrouted tracks. That's the fun part. You can also use an auto-router.

**non-cyclic - does not form loops*

1.2.6.6 Creating Projects

AutoTRAX DEX integrates all the data for a design into a single project file.

A project file is a self contained XML file (with optional compression), that contains all the details of your design. It does not rely on any external files.

The project file contains:

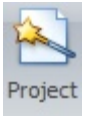
1. Zero or more schematics.

2. Zero of more text documents.

3. The PCB for the project. -- A project can only contain one PCB.

The schematics and text documents can be arranged in a hierarchical fashion or left in a flat arrangement.

The schematics are firmly attached to the PCB and AutoTRAX DEX's internal data structure ensures that the schematics and the PCB are always in sync. Any changes to a schematic are instantly reflected in the PCB. Likewise, any changes in the PCB are immediately reflected in the schematics.

To create a new project select the New Project button 

in the **Home** tab of the ribbon menu.

1.2.6.7 Setting The Default Project

You can set the default project template by first creating a project, setting it up exactly as you want all new projects to be and then click the 

in the File Menu. Then, every time you create a new project, it will start off as a mirror image of the default project you saved.

The file is named Default.project and is in the library '___SystemDoNotRemove' directory .

You can always reset the default project to the factory default using the File Settings.

1.2.6.8 Schematic-PCB Cross Selection

You can optionally select PCB footprints when you select schematic symbols and vice versa. You can also do this for schematic wires and nets/tracks in the PCB. This video lets you quickly see how the parts relate to each other in the schematics and the PCB.

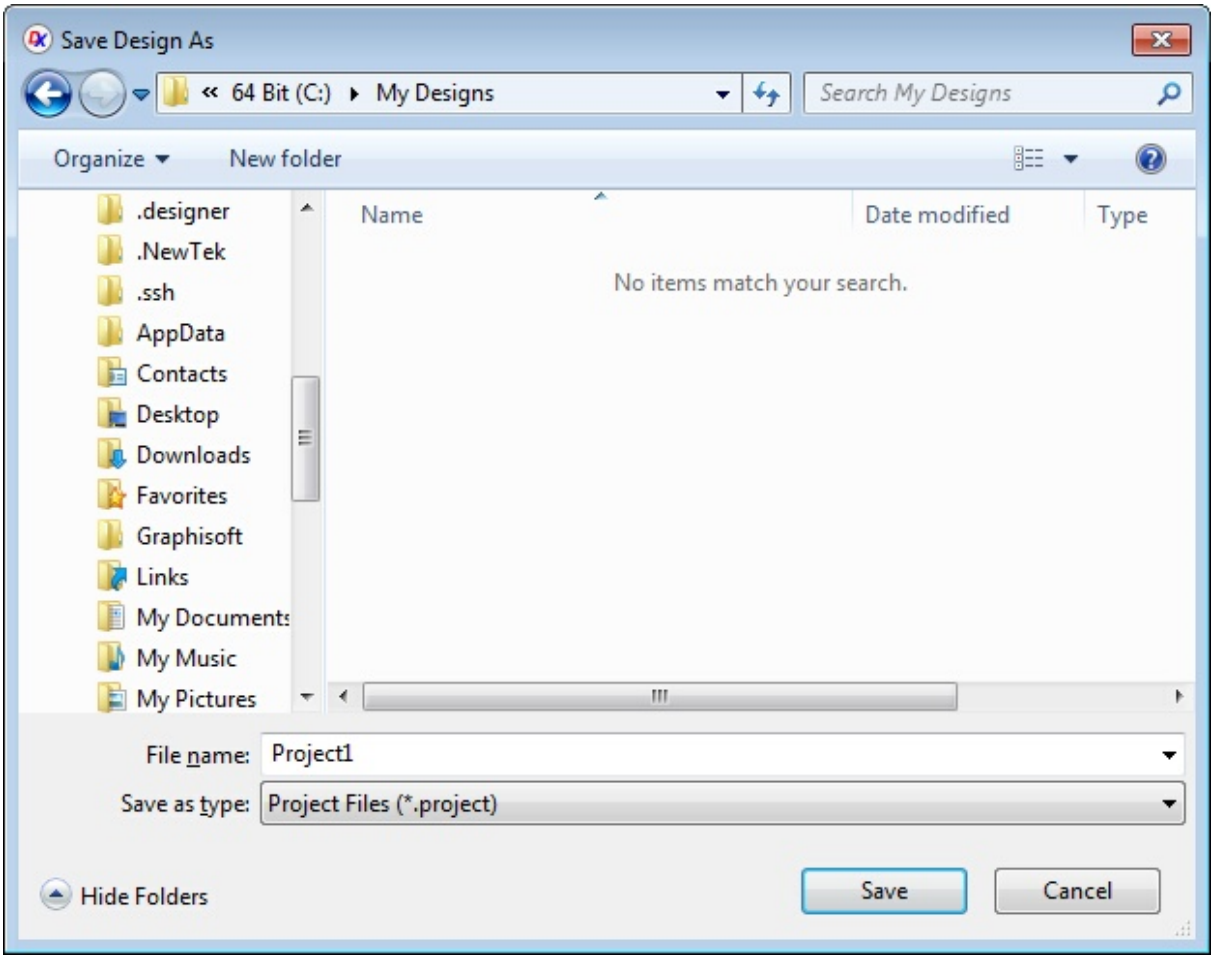
1.2.6.9 Saving Projects

To save you work click the Save button 

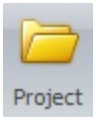
in the **Home tab** of the ribbon menu. If you have never saved your work, you will be prompted for the file name to save to as show below.

To save your work to a different file click the Save As button 

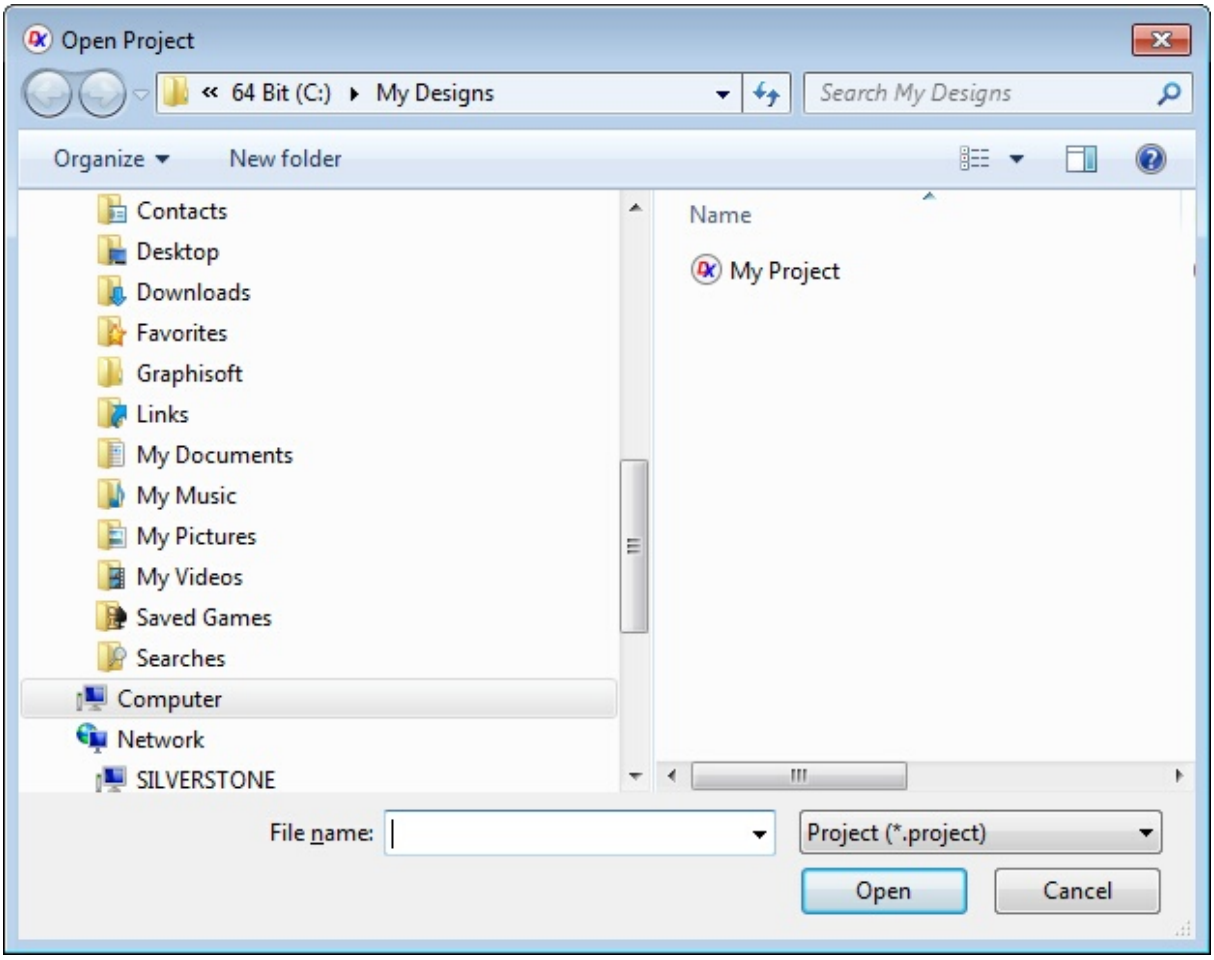
in the **Home tab** of the ribbon menu. You will be prompted for the file name to which to save.



1.2.6.10 Opening Projects

To open an existing project click on the **Open button** 

in the **Home tab** of the ribbon menu.



1.2.6.11 The PCB

A printed circuit board, or PCB, is used to mechanically support and electrically connect electronic components using conductive pathways, tracks or signal traces etched from copper sheets laminated onto a non-conductive substrate. When the board has only copper tracks and features, and no circuit elements such as capacitors, resistors or active devices have been manufactured into the actual substrate of the board, it is more correctly referred to as printed wiring board (PWB) or etched wiring board. Use of the term PWB or printed wiring board although more accurate and distinct from what would be known as a true printed circuit board, has generally fallen by the wayside for many people as the distinction between circuit and wiring has become blurred. Today printed wiring (circuit) boards are used in virtually all but the simplest commercially produced electronic devices, and allow fully automated assembly processes that were not possible or practical in earlier era tag type circuit assembly processes.

A printed circuit board (PCB) mechanically supports and electrically connects electronic components or electrical components using conductive tracks, pads and

other features etched from one or more sheet layers of copper laminated onto and/or between sheet layers of a non-conductive substrate. Components are generally soldered onto the PCB to both electrically connect and mechanically fasten them to it.

Printed circuit boards are used in all but the simplest electronic products. They are also used in some electrical products, such as passive switch boxes.

PCBs can be single-sided (one copper layer), double-sided (two copper layers on both sides of one substrate layer), or multi-layer (outer and inner layers of copper, alternating with layers of substrate). Multi-layer PCBs allow for much higher component density, because circuit traces on the inner layers would otherwise take up surface space between components. The rise in popularity of multilayer PCBs with more than two, and especially with more than four, copper planes was concurrent with the adoption of surface mount technology. However, multilayer PCBs make repair, analysis, and field modification of circuits much more difficult and usually impractical.

A PCB populated with electronic components is called a printed circuit assembly (PCA), printed circuit board assembly or PCB Assembly (PCBA). In informal use the term "PCB" is used both for bare and assembled boards, the context clarifying the meaning.

Viewing the PCB

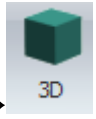
Viewing the PCB in 3D

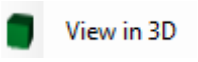
Semi-Transparent PCBs

PCB Internals

The PCB Border

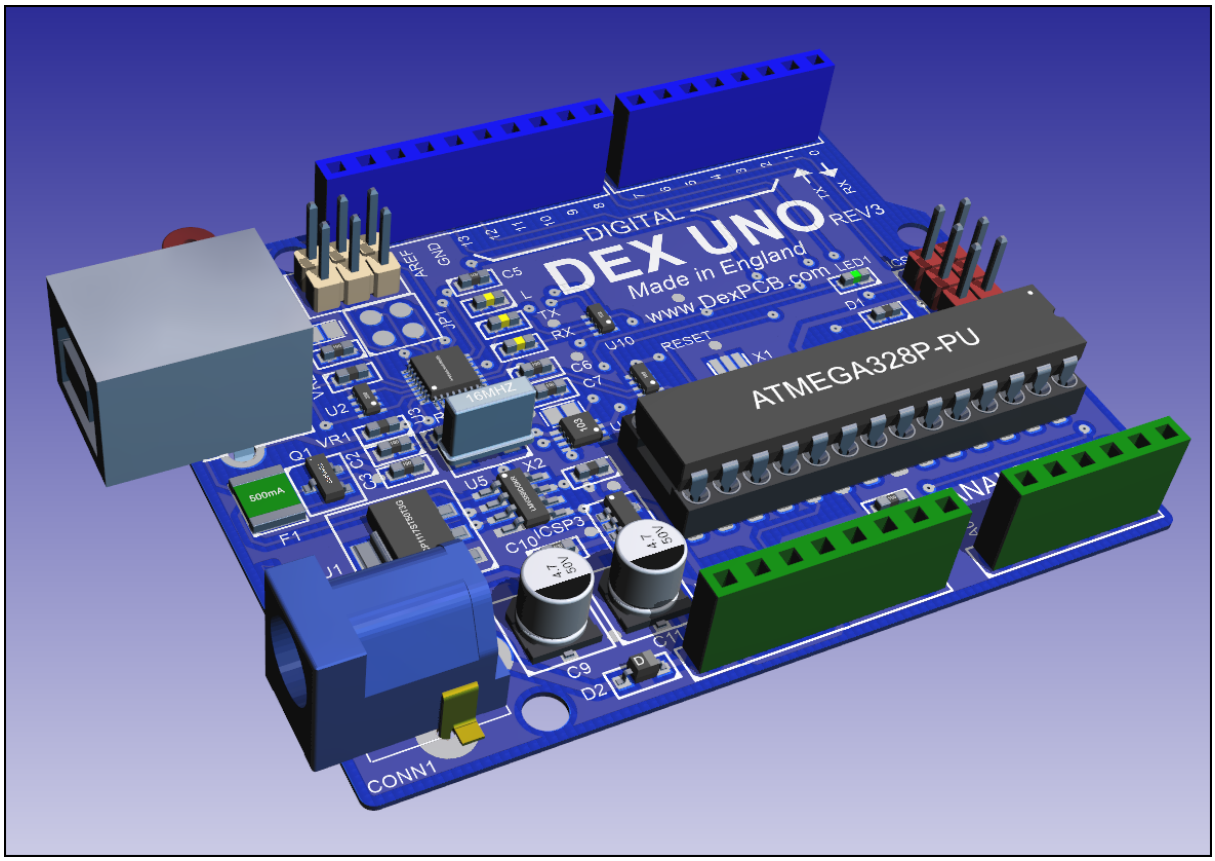
1.2.6.11.1 Viewing the PCB in 3D

To view the PCB in 3D click the Panels→Window→ 

button. Alternatively ***right-click*** in a PCB viewport and select 

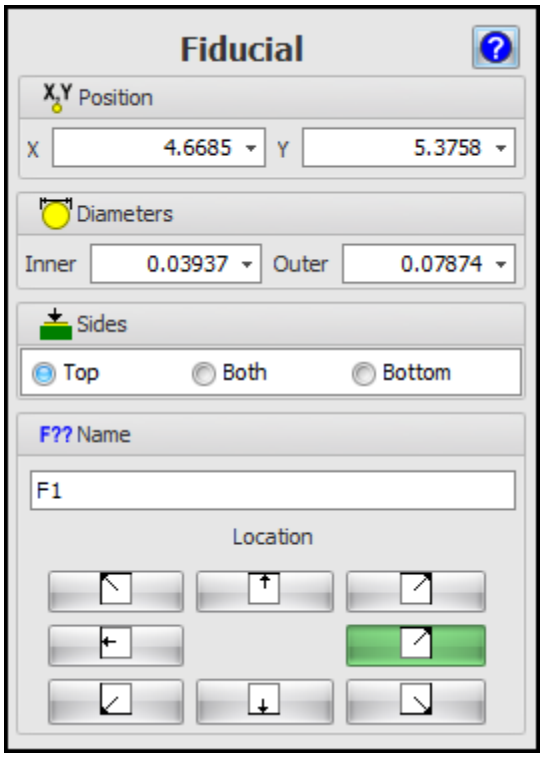
.

This will display a 3D view of the PCB.

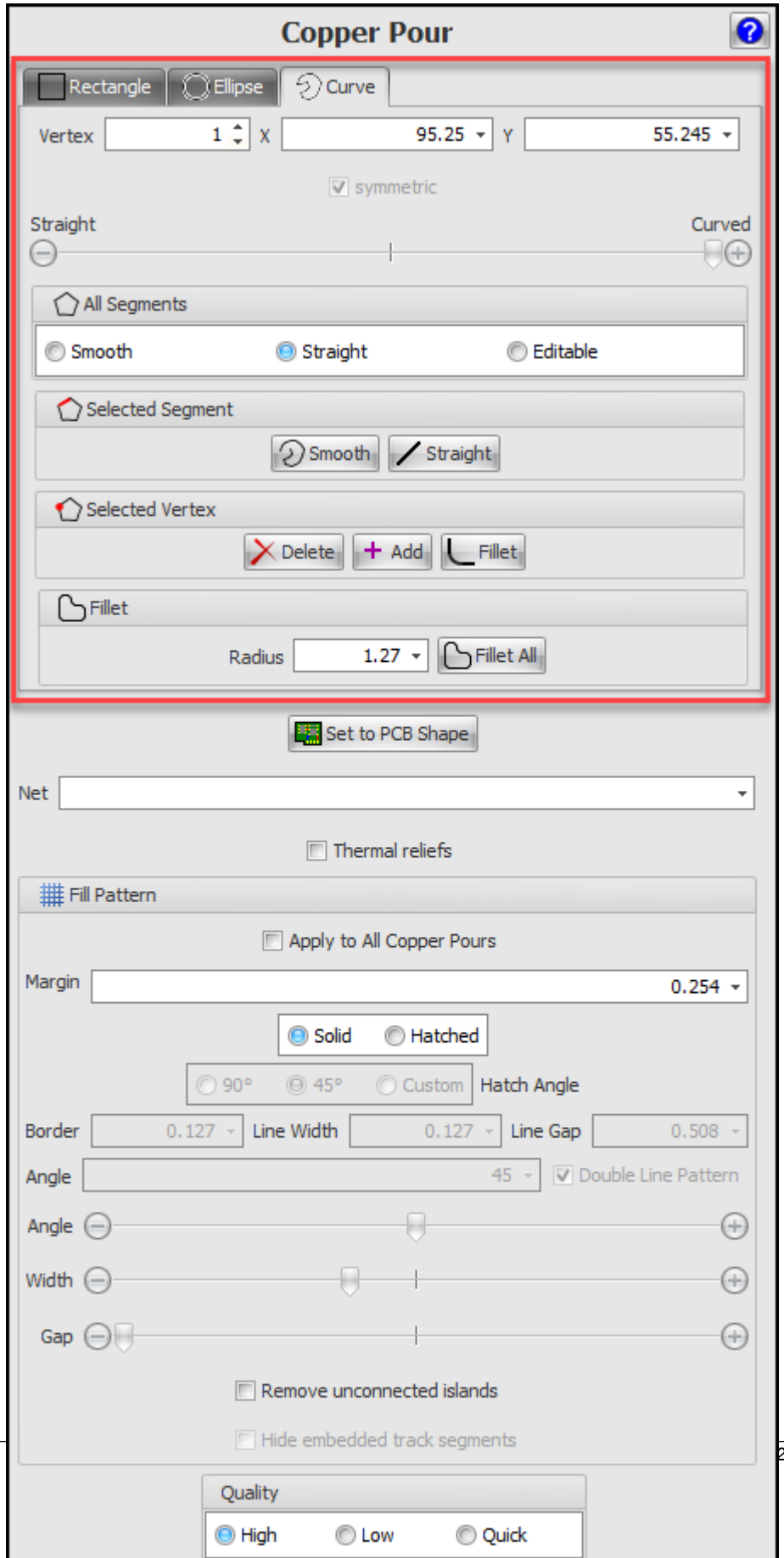


Typical 3D view of a PCB

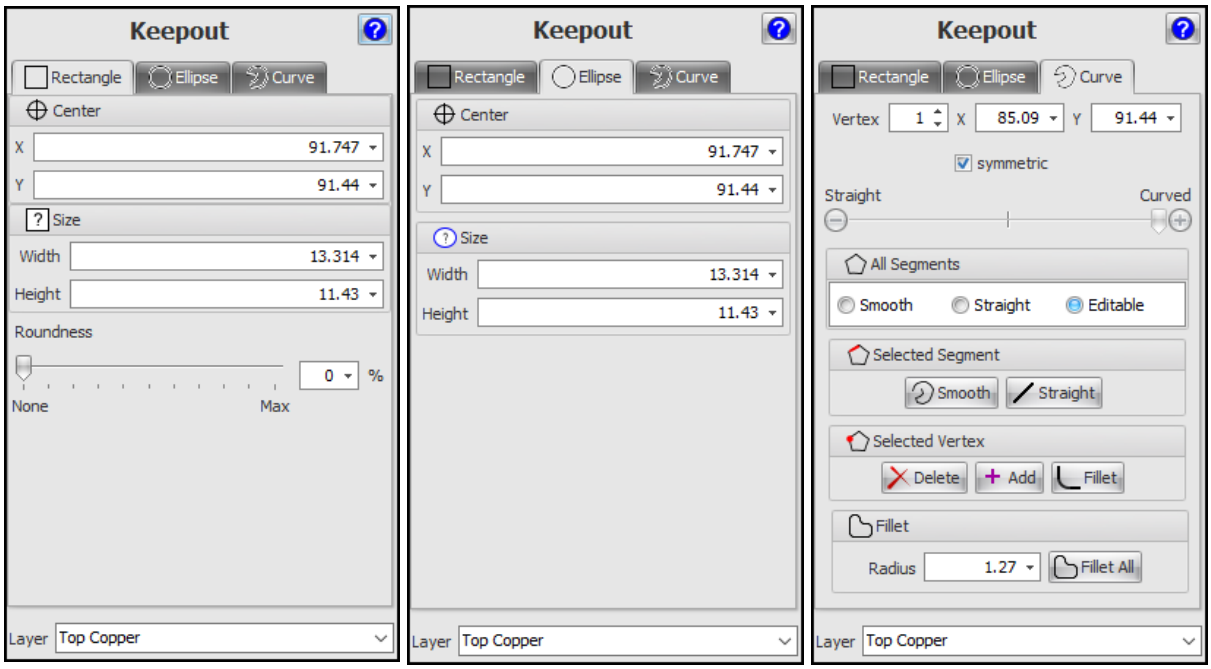
1.2.6.11.2 Fiducial Editor



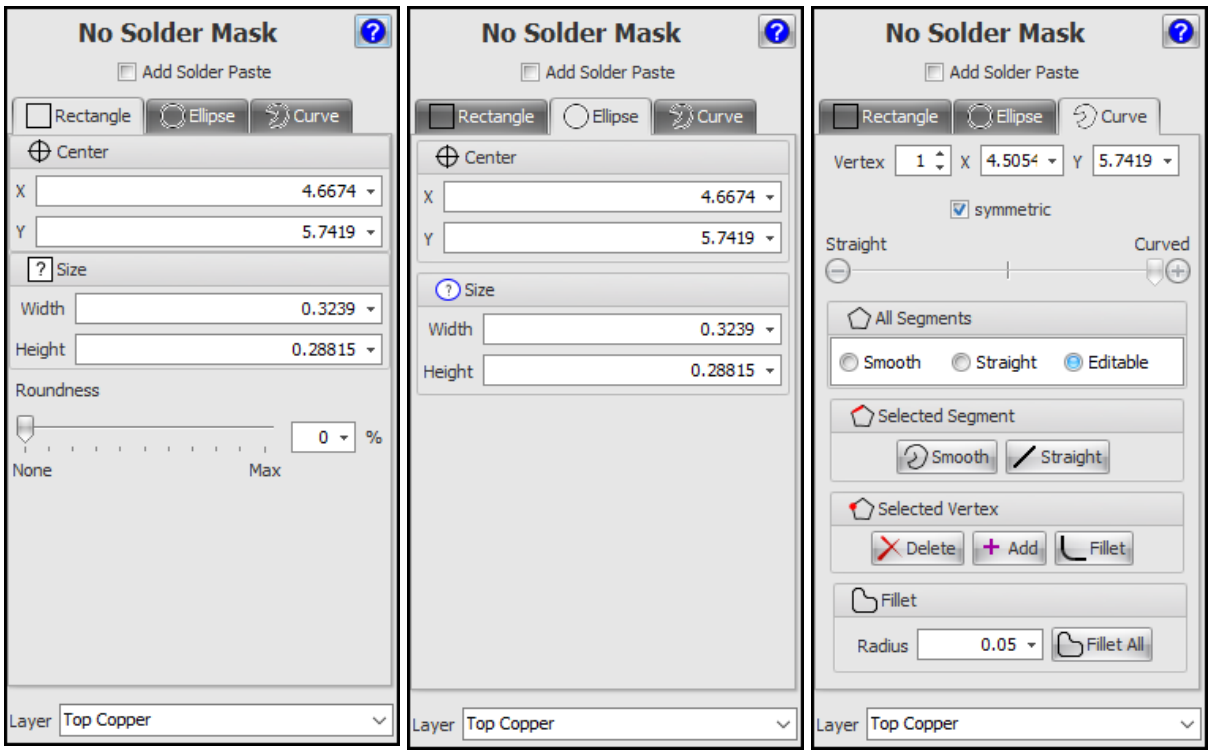
1.2.6.11.3 Copper Pour Editor



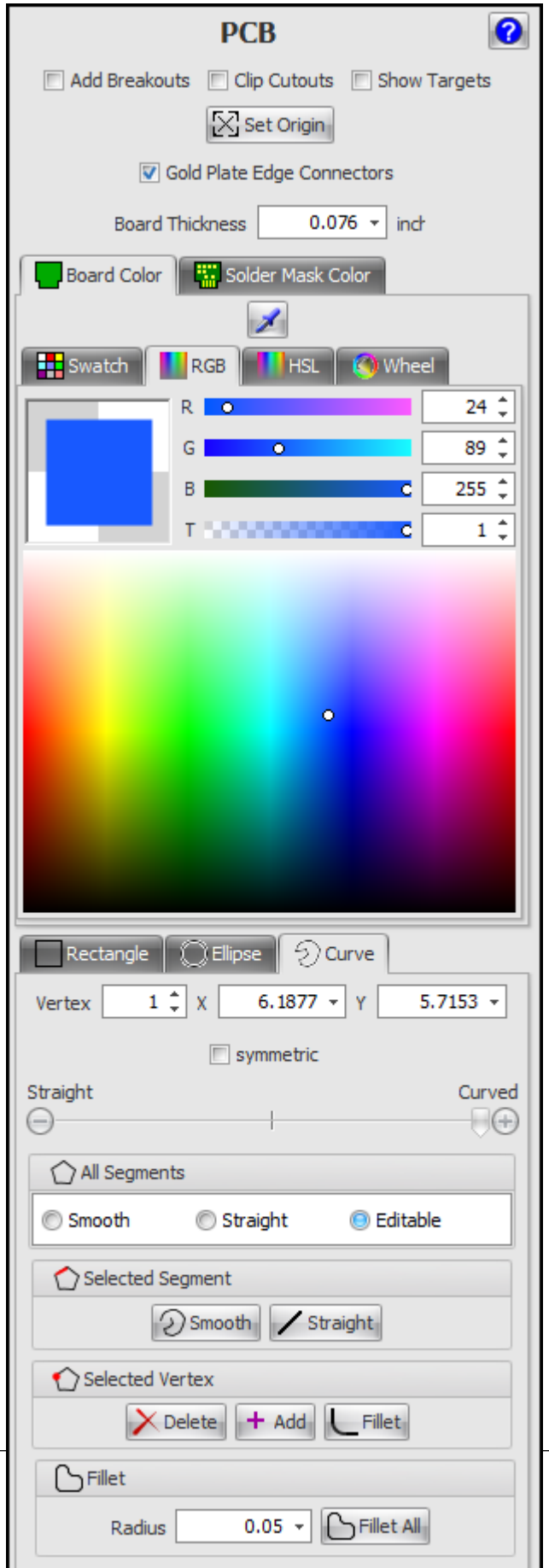
1.2.6.11.4 Keep-Out Editor



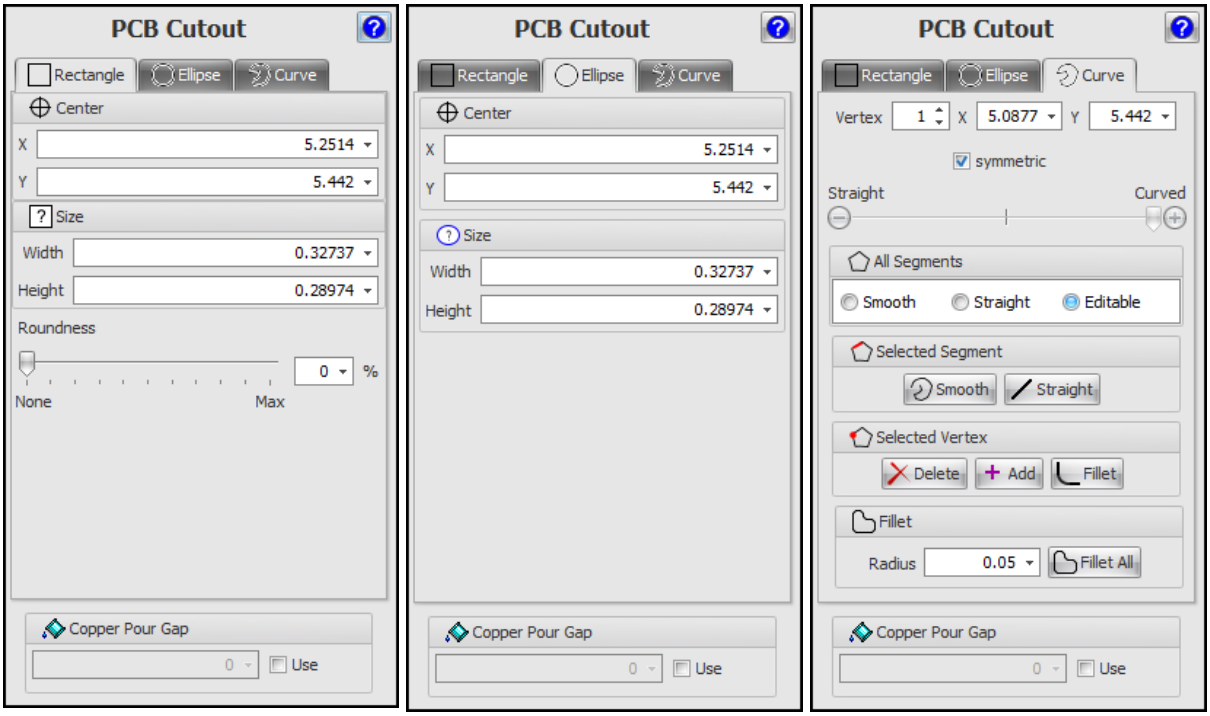
1.2.6.11.5 No-Mask Editor



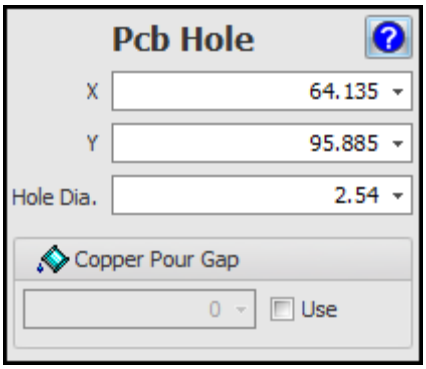
1.2.6.11.6 PCB Border Editor



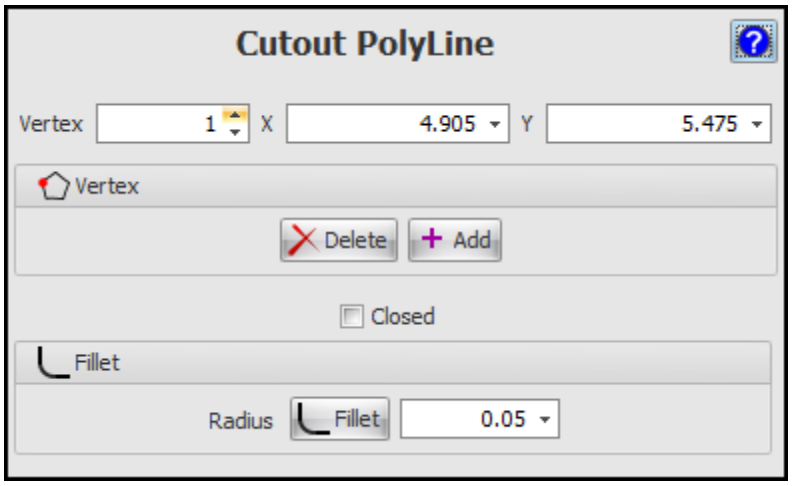
1.2.6.11.7 PCB Cutout Shape Editor



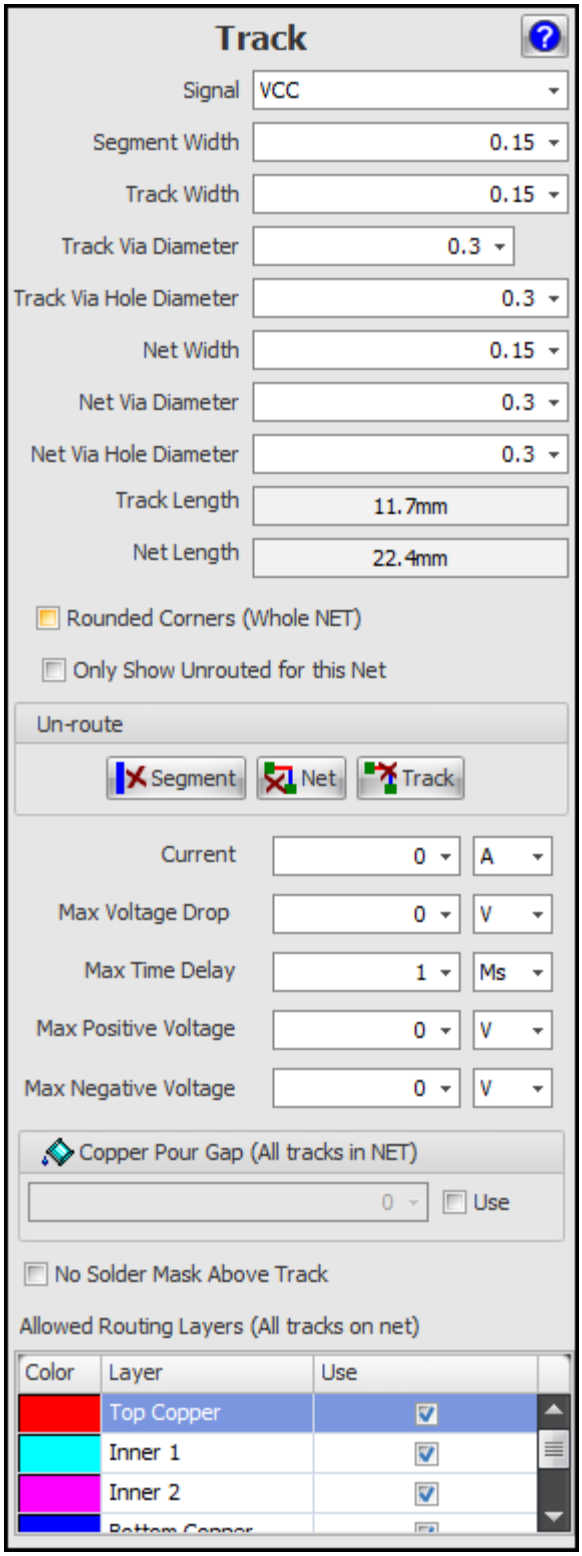
1.2.6.11.8 PCB Hole Editor



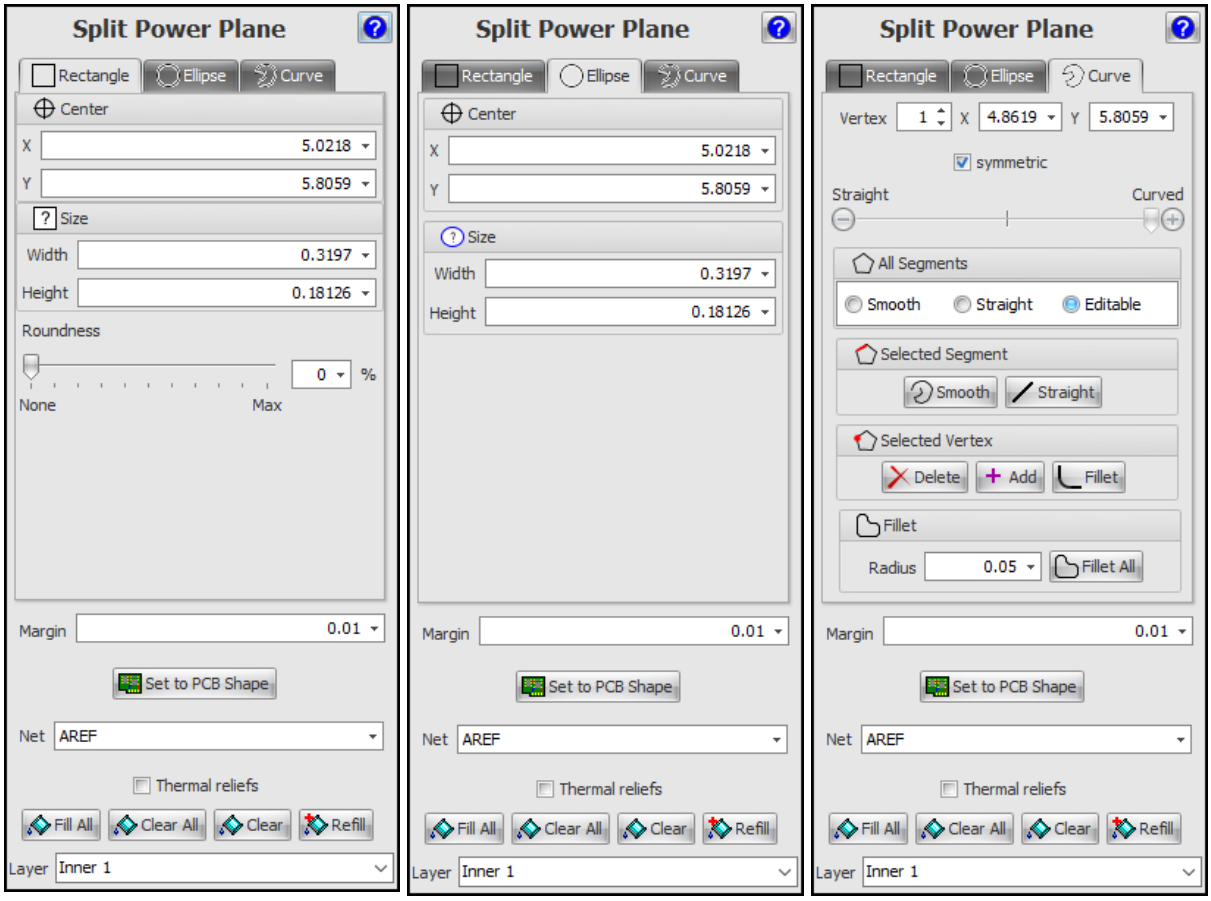
1.2.6.11.9 Pcb Polyline Cutout Editor



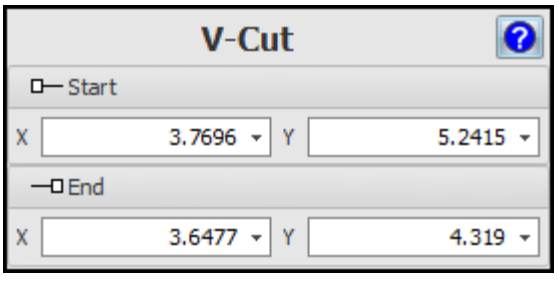
1.2.6.11.10 Track Editor



1.2.6.11.11 Split Power Plane Editor

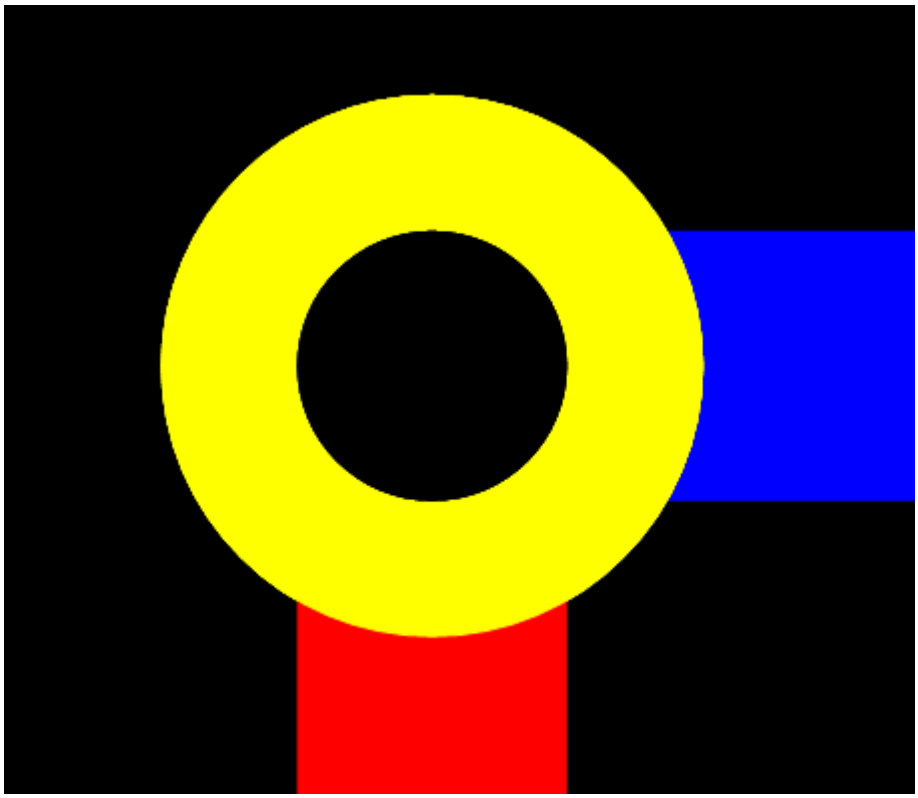


1.2.6.11.12 V-Cut Editor

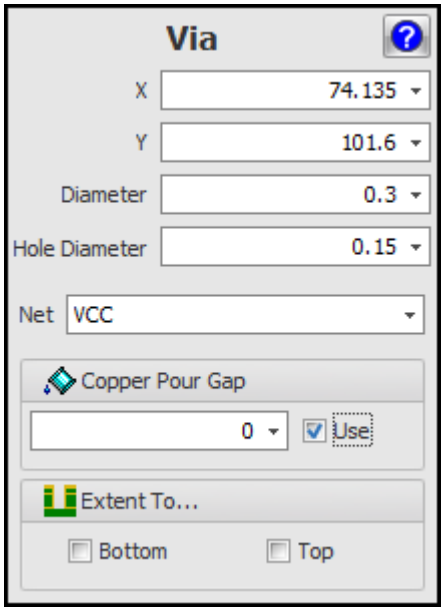


1.2.6.11.13 Via Editor

For more details of vias see this...



A typical Via



The Via Editor

Setting the Position and Size of the Via

The center X coordinate of the via.

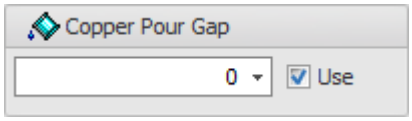
The center Y coordinate of the via.

The diameter of the via.

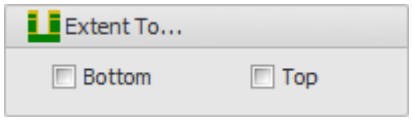
The diameter of the via's hole.

Select the name of a copper pour's net to connect to.

Copper Pour Gap



Extending Buried Vias to the Top and/or the Bottom of the PCB



1.2.6.11.14 Design for manufacturability

In the PCB design process, DFM leads to a set of design guidelines that attempt to ensure manufacturability. By doing so, probable production problems may be addressed during the design stage.

Ideally, DFM guidelines take into account the processes and capabilities of the manufacturing industry. Therefore, DFM is constantly evolving.

As manufacturing companies evolve and automate more and more stages of the processes, these processes tend to become cheaper. DFM is usually used to reduce these costs. For example, if a process may be done automatically by machines (i.e. SMT component placement and soldering), such process is likely to be cheaper than doing so by hand.

1.2.6.11.15 PCB Standards

The IPC

IPC, the Association Connecting Electronics Industries, is a trade association whose aim is to standardize the assembly and production requirements of electronic equipment and assemblies. It was founded in 1957 as the Institute for Printed Circuits. Its name was later changed to the Institute for Interconnecting and Packaging Electronic Circuits to highlight the expansion from bare boards to packaging and electronic assemblies. In 1999, the organization formally changed its name to IPC with the accompanying tagline, Association Connecting Electronics Industries.

IPC is accredited by the American National Standards Institute (ANSI) as a standards developing organization[2] and is known globally for its standards. It publishes the most widely used acceptability standards in the electronics industry.

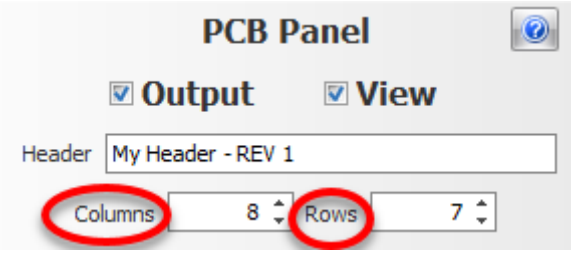
IPC is headquartered in Bannockburn, Illinois, United States and maintains additional offices in Washington, D.C.; Taos, New Mexico; Arlington County, Virginia, in the United States; Stockholm, Sweden; Brussels, Belgium; Moscow, Russia; Bangalore, India; and Shanghai, Shenzhen and Beijing, China.

Find out more...

1.2.6.11.16 Panelised PCBs

When you submit your design to your PCB manufacturer, in addition to just sending a single PCB design and allowing the manufacturer to create the PCB as they wish, you can also ask the manufacturer to create a PCB panel.

Number of PCB's Per Panel



Having a higher number of PCB's per panel will reduce manufacturing costs and is essential for automated assembly using pick and place machines if the PCB is small.

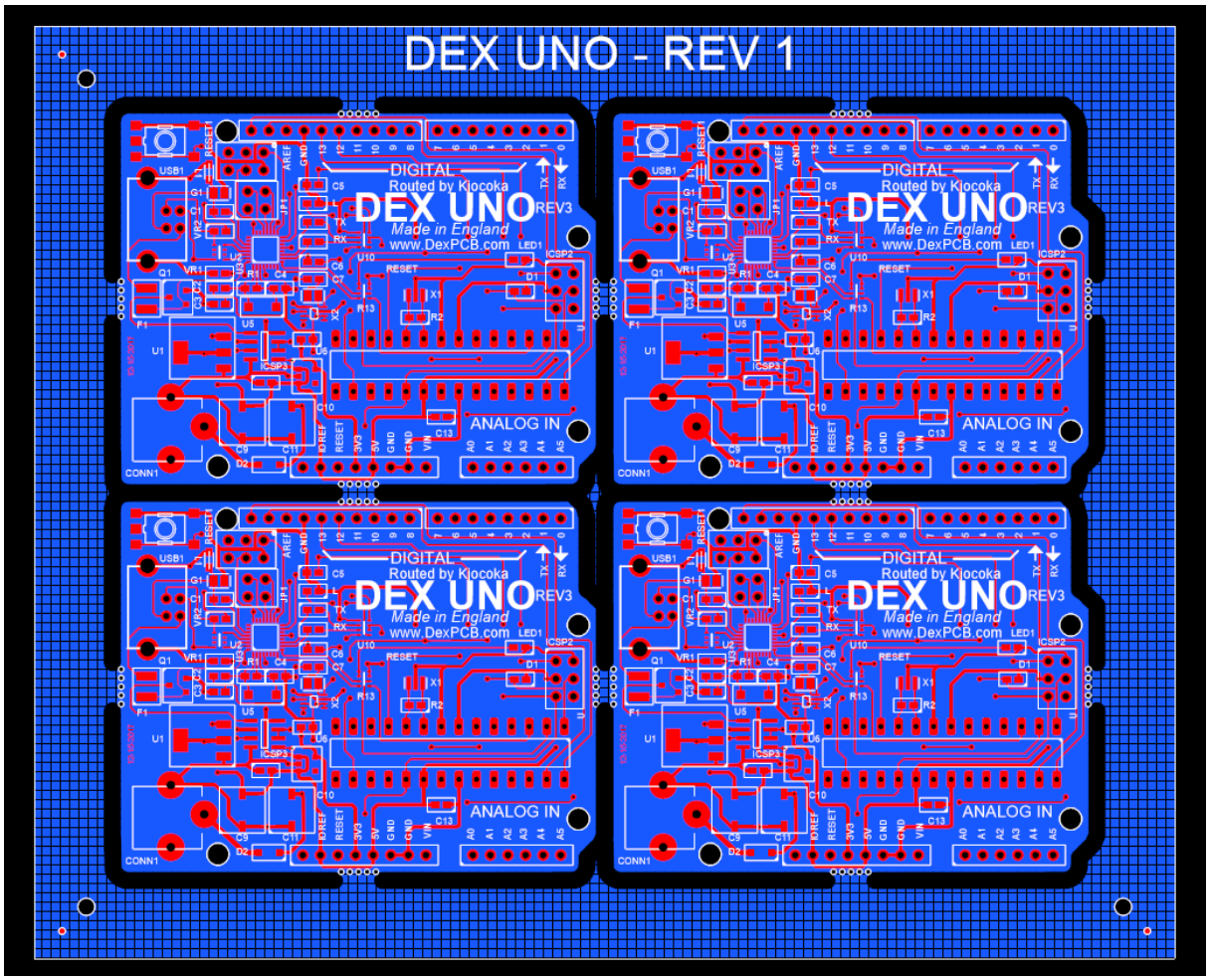
For an order with a small number of units, single-up panels are usually OK.

But when orders get larger, over 25 pieces, you are best to consider making the panels larger in order to improve the assembly process.

If the individual PCB's themselves are large, then a single-up panel may be necessary, regardless of the number of units ordered. For example if the individual PCB is over say 200mm then it is probable that it would need to remain a single-up panel.

However, if the PCB is under that, and the quantity is sufficient, then putting more PCB's into a panel would be preferred.

2 x 2 PCB Panel (4 PCBs)

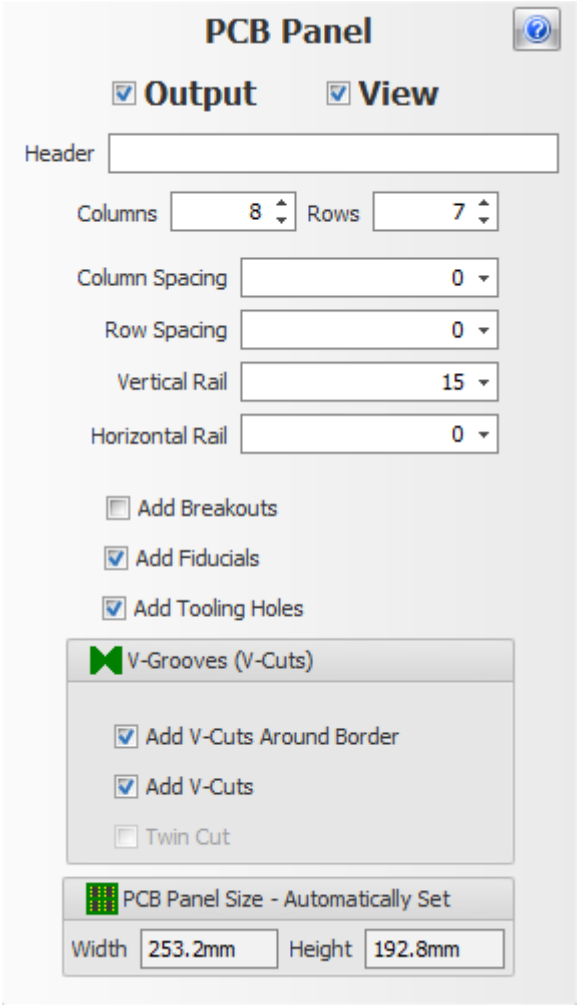


1.2.6.11.16.1 Creating a PCB Panel

To create a PCB panel, in the properties panel while viewing the PCB with nothing selected you will see the PCB Panel section.

If you expand it, you will see the controls below.

The PCB Panel Control in the Properties Panel



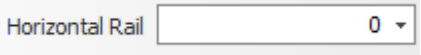
With the PCB Panel control you can:

Set the number of columns and rows of the PCB in the panel.

Set the spacing between the columns and rows.

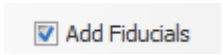
Optionally add a pair of vertical rails on the left and right of the panel.

Optionally add a pair of horizontal rails on the top and bottom of the panel.

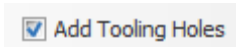


Add Breakouts - useful for non-rectangular PCBs.

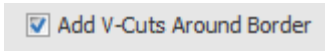
Add 3 fiducials to allow setup for a pick and place machine.



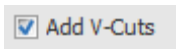
Add 4 Tooling holes.



Add V-Grooves around the border of all the PCBs.

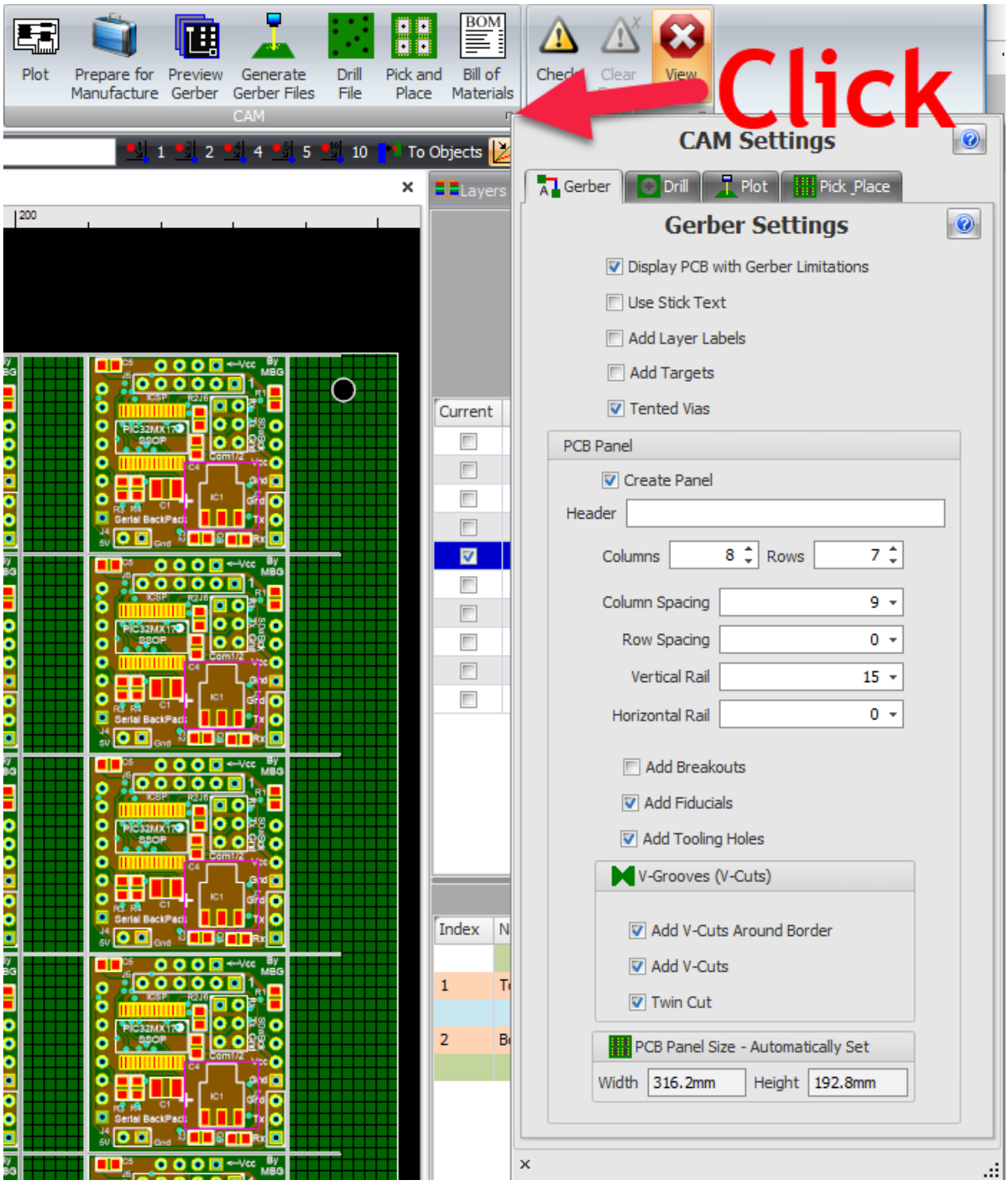


Add V-Grooves between the rows and columns of PCBs.



Add twin cuts if you have non-zero column/row spacings.

You can also edit most of the parameters in the CAM settings pop up.



1.2.6.11.16.2 PCB Breakaway Panels

To produce a cost-effective layout through optimum PCB material usage, you are recommended to consult with your PCB manufacturer to determine the optimum panel size. The PCB should be design to utilize the manufacturers suggested usable area. Smaller PCBs may be ganged of red in the uniform apparel format to simplify fixture and reduce excessive angling during assembly.

The main reason for having your boards delivered in an array is to make automated assembly faster and less expensive. Running an array of boards through a pick-and-place machine is far more efficient than sending them through one at a time. Arrays are also desirable because they allow the addition of tooling rails, tooling holes, and fiducials, all of which help your assembler.

Tooling Rails are the 'holding frame' for the array. They provide stability and make it easier to handle the arrays throughout the assembly process. Tooling holes and fiducials are usually added to the tooling rails.

Tooling Holes are non-plated holes added to the rails so the array can be pinned down to prevent unwanted shifting during assembly. They are typically 0.125" in diameter but can be drilled to your required specification.

Fiducials are copper markers on the rails, which aid automated pick-and-place assembly equipment by providing a uniform reference point. The copper fiducial is typically 0.04" in diameter and will not be covered with mask to make it easier for assembly equipment to see.

A PCB panel may include several into circuit boards arranged in a matrix or simply one printer circuit board requiring additional material retained for efficient assembly. The large PCB are several smaller PCBs are retained in the panel separated after all assembly is completed. Separating the individual PCBs from the panel can be done using several methods which include the – will scarring, and see routing and slotted routes with breakaway tabs.

What type of array to use?

There are 3 types of arrays:

1. Scored
2. Tab Routed
3. A mixture of both.

So how do you decide which one is right for your design?

Scoring is the preferred choice for two reasons.

1. Scoring has the advantage of a more consistently smooth board edge,
2. and it wastes less material, which can mean cost savings especially for larger quantities or high layer count printed circuit boards where every square inch counts.

You should consider tab routing when your design has an irregular shape or if you need space between your boards to allow for overhanging components.

A mix of scoring and tab routing can be used when some board sides are straight, which can be scored, while irregular sides must be tab routed.

There are two ways to create PCB Breakaway Panels.

- Mouse bites (a.k.a. breakaway This is a test of the font height tab on routing)
- V – score (a.k.a. V–Grove)

V–Grooves

V–Groves are cuts, usually on both sides of a PCB and generally in a straight line. They are made with a cutter with a cut angle of 30° or 45°

You need tooling rails along the two longest sides and there should be no space between the boards. The score will be made along the shared board outline. Scoring is performed only parallel to the x- and y-axes, not diagonally. Because scoring runs all the way across the array in a straight line, the board outline should be straight for scored arrays.

The score depth is approximately ⅓ of the total material thickness. Score lines will be made on the top and bottom sides of the PCB. This will leave a remaining web of material equal to approximately ⅓ of the total thickness of the board. If your board is1mm thick, the remaining material will be ~0..33mm.

Standard PCB material is basically fiberglass. Even though your board is scored, it will still be sturdy so be careful when separating the boards. You should expect some amount of flexing before the score line will give. Some fiberglass strands are common along a scored edge. A quick skim with a sander will take care of any rough areas.

Jump Scoring

It is possible to stop a score from continuing all the way across an array; this is referred to as jump scoring. However, this is not a recommended array solution due to the limitations and expense of this process. We strongly encourage the use of tab routing rather than jump scoring.

Scoring cannot be stopped precisely enough to end a score line like a CNC route. To make a complete score, the scoring blade must travel past the end point and come to a stop. You need a minimum of a 7 mm gap between the stop score location and anything that is not intended to be scored.

Tab Routed Arrays

If your design has an irregular shape, then tab routing may be required. AutoTRAX DEX will add a gap, equal to the router cut width, between the boards to allow the router bit to pass between them. Small tabs of material will remain to hold the

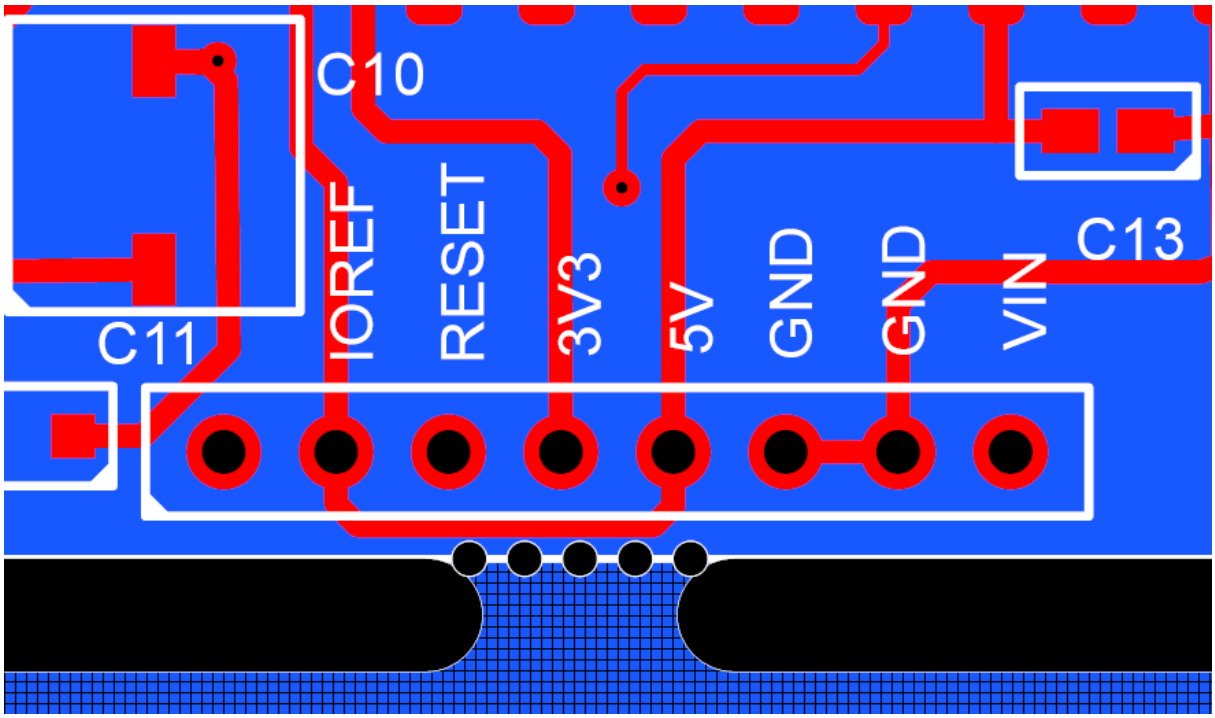
boards in place. To make separation easier, you can add small non-plated holes to the tabs called 'mouse bites to perforate the tab. If you desire a smooth edge on your PCBs, these areas will need to be sanded after the boards are removed from the array.

Because tab routed arrays are inherently less sturdy than scored arrays,You should add tooling rails to all 4 sides of the array. This will give your array of PCBs more support during handling and assembly processes.

Mouse Bites

Each break-out tab has it's own set of 5 automatically created mouse-bite holes.

Break-apart tab with mouse-bites



Breakout

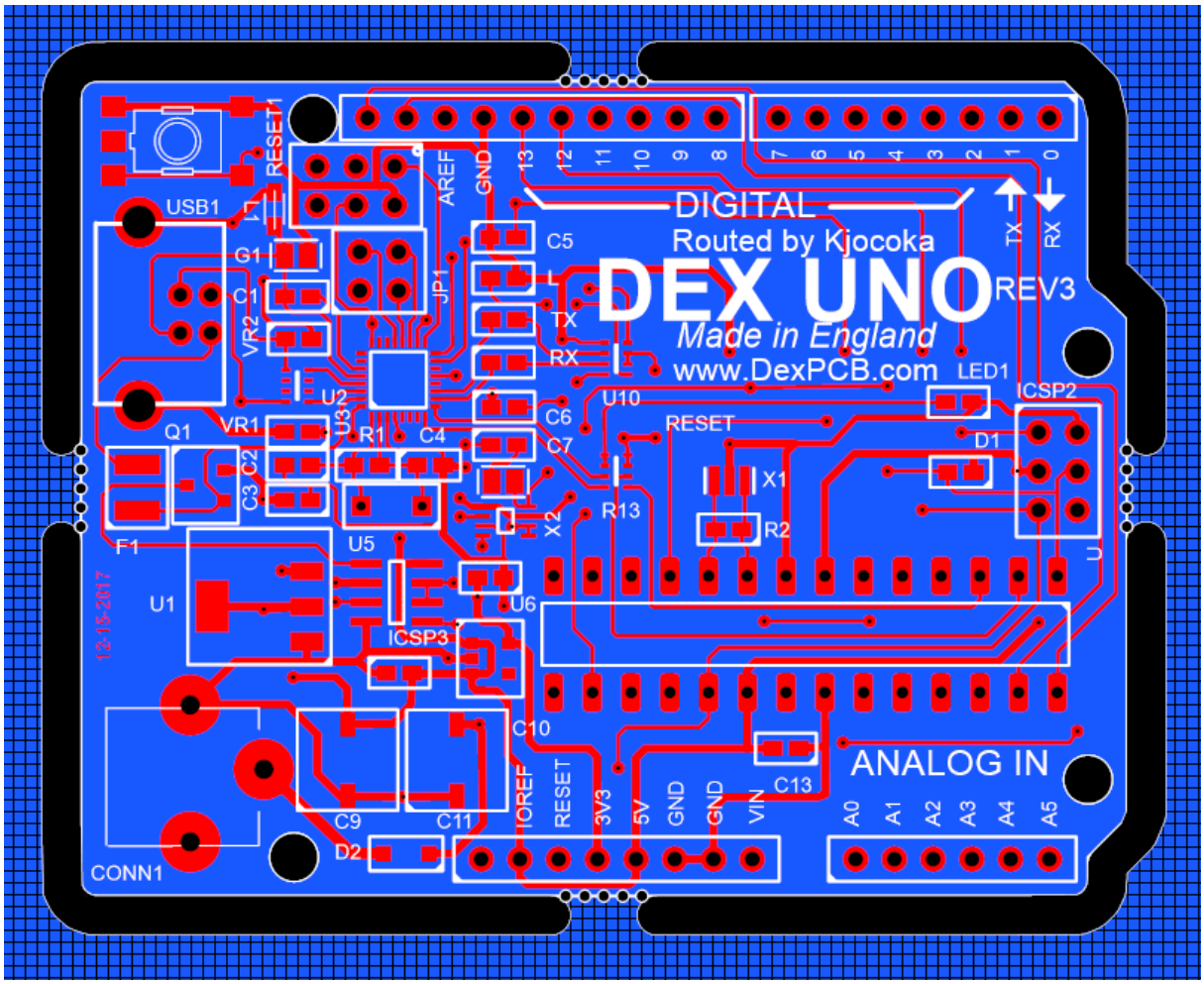
The spacing between breakaway tabs can range from 60 mm to 90 mm, but the IPC recommend 75mm or 3" . Try not to exceed 100 mm between tabs and try to evenly space them apart.

1.2.6.11.16.3 Tab Routed Arrays

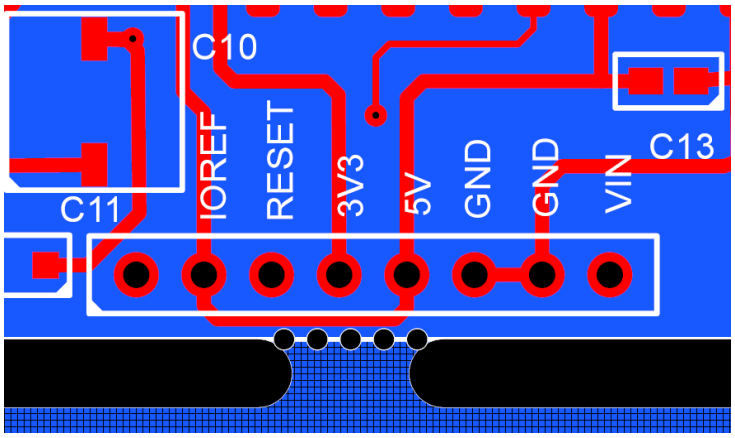
If your design has an irregular shape, then tab routing may be required. AutoTRAX DEX will add a gap, equal to the router cut width, between the boards to allow the router bit to pass between them. Small tabs of material will remain to hold the

boards in place. To make separation easier, you can add small non-plated holes to the tabs called 'mouse bites to perforate the tab. If you desire a smooth edge on your PCBs, these areas will need to be sanded after the boards are removed from the array.

Because tab routed arrays are inherently less sturdy than scored arrays,You should add tooling rails to all 4 sides of the array. This will give your array of PCBs more support during handling and assembly processes.



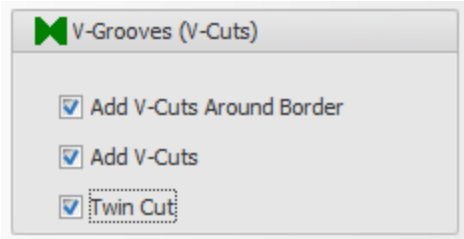
Single PCB with routed break-apart tabs on 4 sides



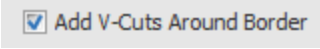
Magnified View of break-apart tab with mouse-bites

1.2.6.11.16.4 Adding V-Grooves

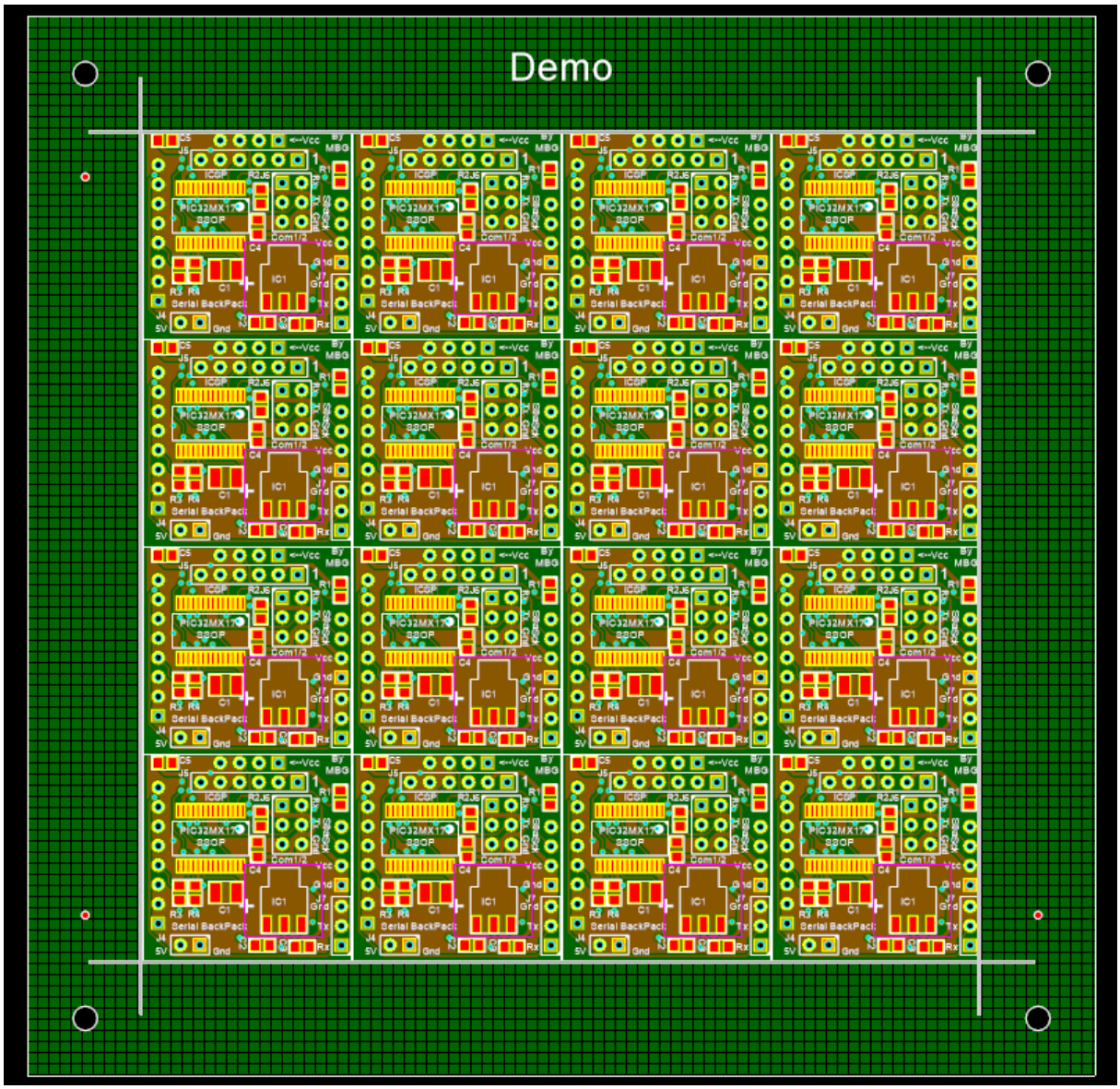
You can add V-Grooves to your panel. These are used to quickly snap apart the panel to separate out the individual PCBs.



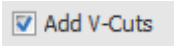
Add V-Grooves around the border of all the PCBs.



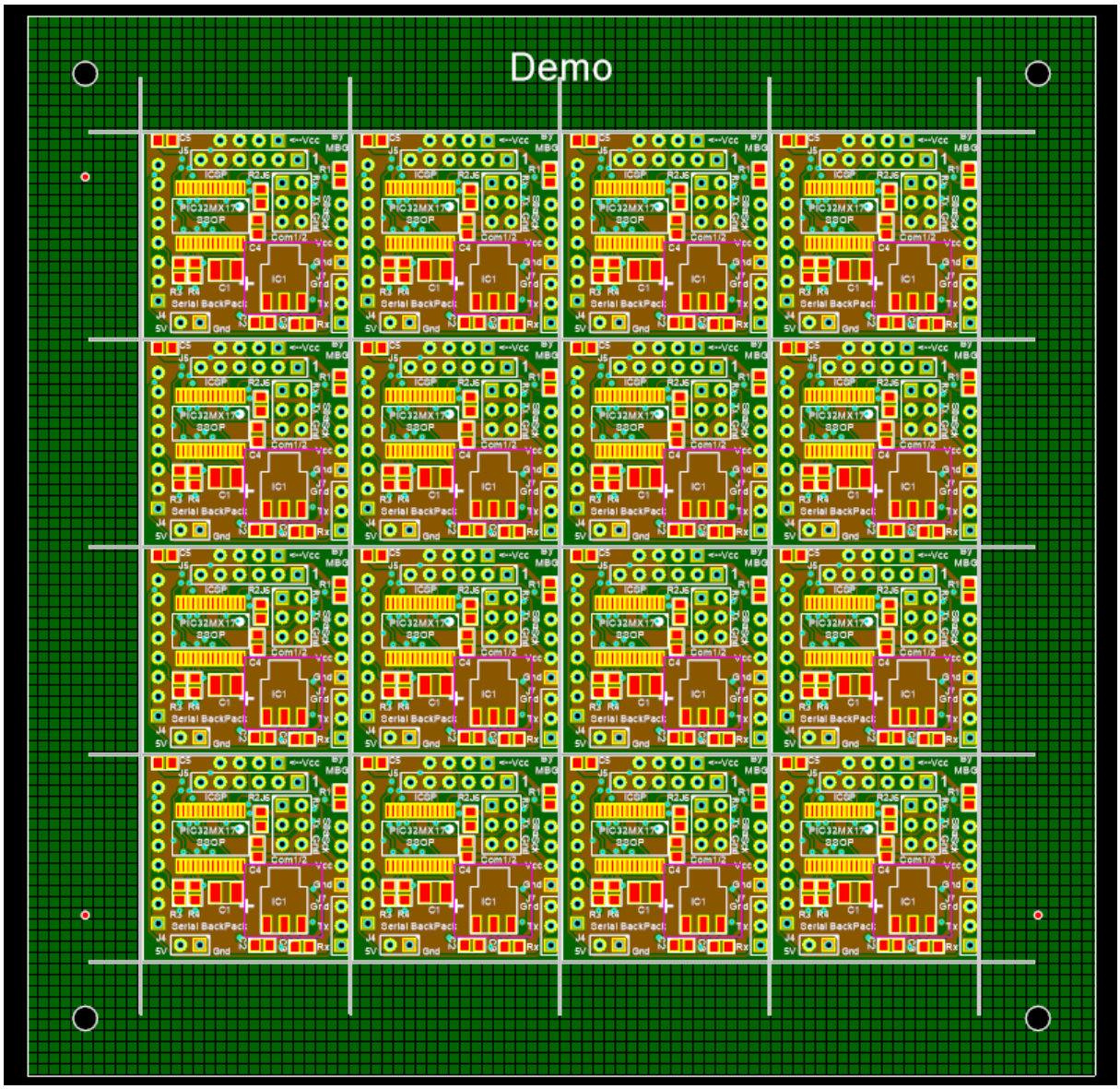
V-Grooves on all 4 edges of the PCB array



Add V-Grooves between the rows and columns of PCBs.

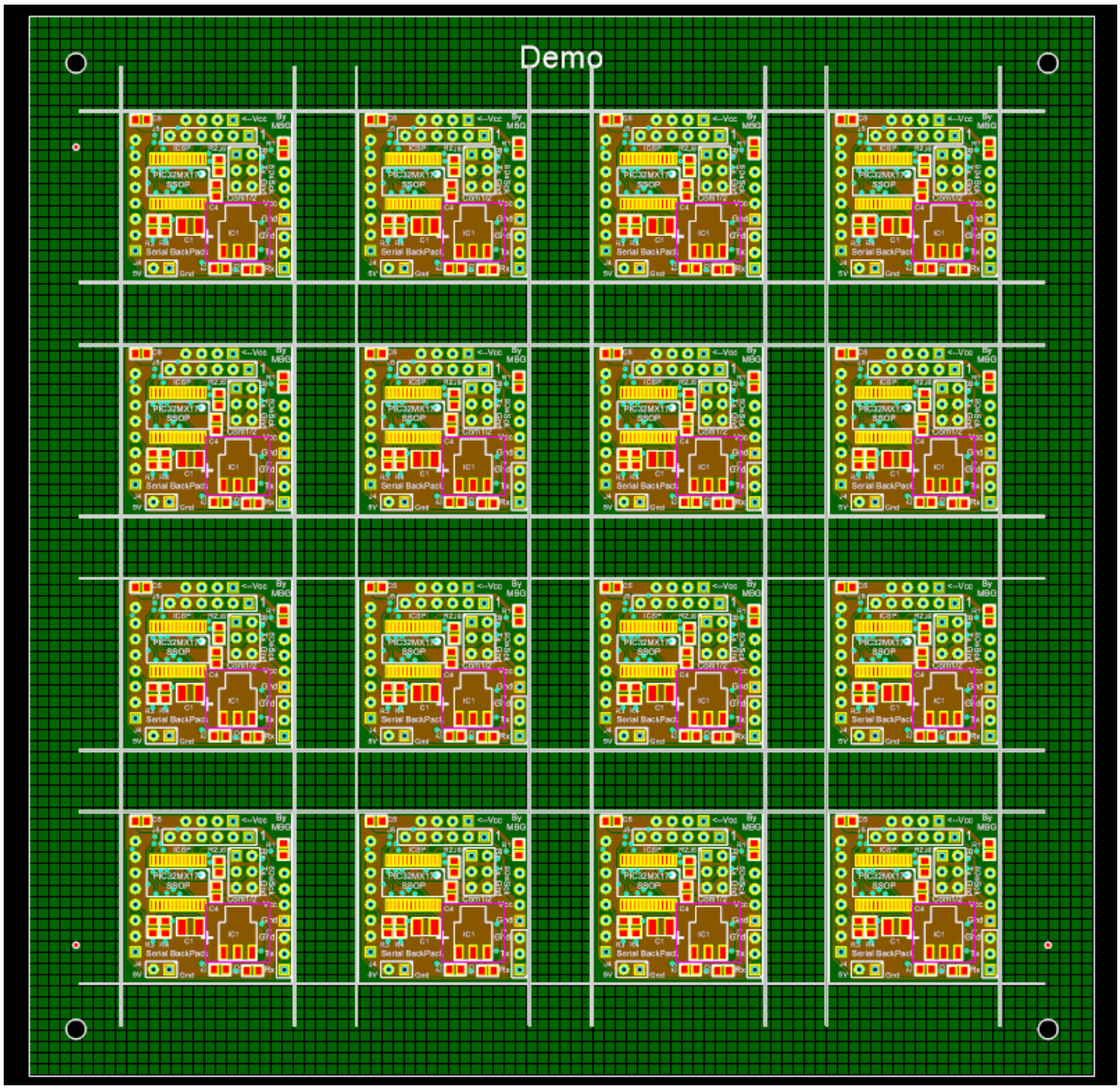


V-Grooves on all 4 edges of the PCB array and between PCBs

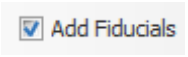


Add twin cuts if you have non-zero column/row spacings.

Twin cuts between PCB rows and columns



1.2.6.11.16.5 Adding Fiducials

To add 3 fiducial marks to your panel check the 

checkbox in the Panels control.

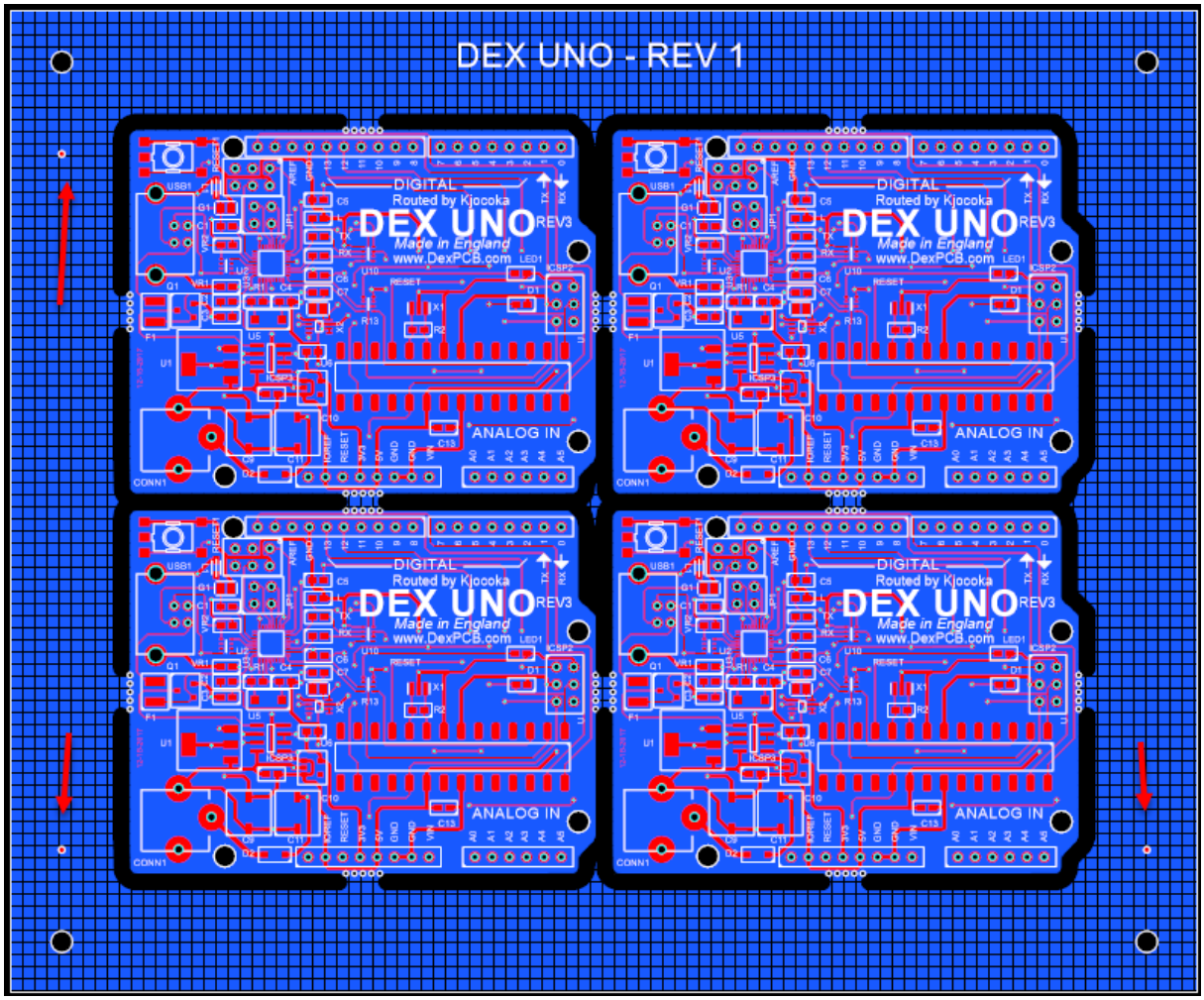
If checked, 3 fiducial marks in copper will be added to 3 corners of your panel as shown below.

Panel fiducials are called **global fiducials** by the IPC.

You can also added local

Pick and Place Origin

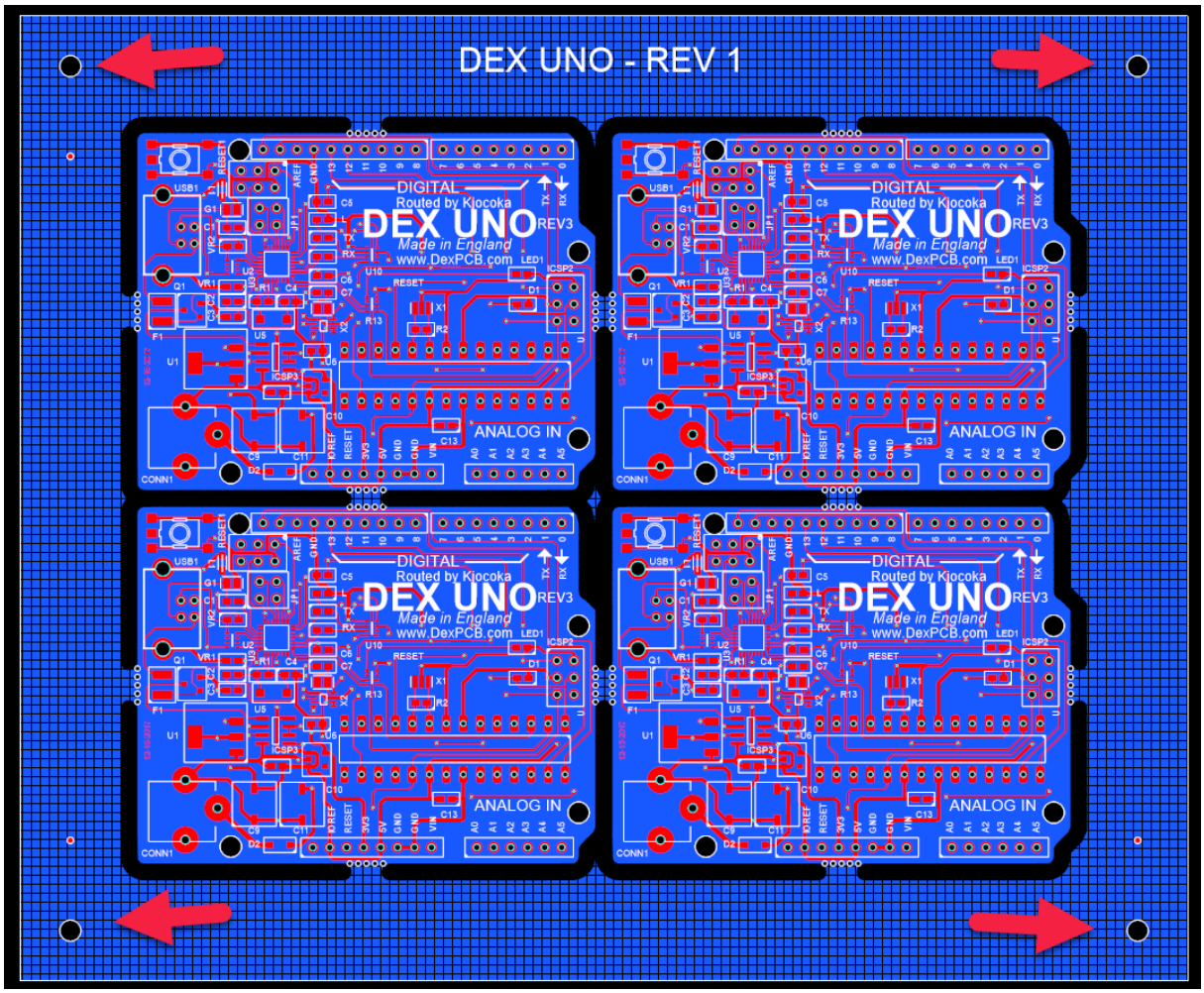
If you add automatic fiducial to your panel then the origin for the Pick and Place file coordinates is set to the lower left fiducial else it is set to the bottom left corner of the panel.



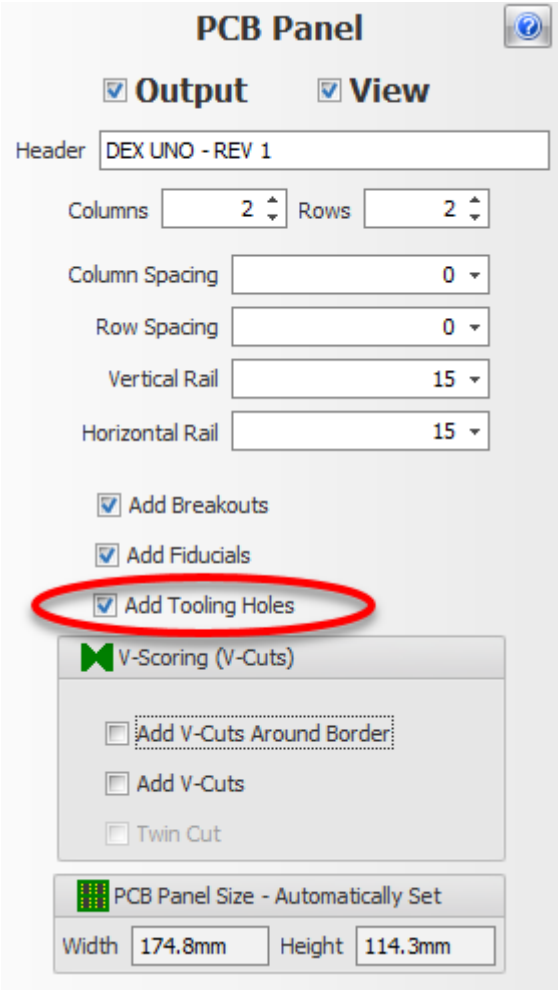
1.2.6.11.16.6 Adding Tooling Holes

Tooling holes, while not as relevant as they used to be, are still valuable to have and do not usually cost any extra to add. You can put 4 of them in the corners and they are 3.175mm in diameter (which is 1/8" which was a standard pin support size that most of the industry standardized on in the 80's)

You can add 4 tooling holes 1/8" in diameter to the 4 corners of the panel



Check Add Tooling Holes to automatically add 4 tooling holes.



1.2.6.11.16.7 Assembly Rails

When using pick and place machines to populate your PCBs you should add assembly rails to the edges of your panel.

The purpose of a frame around the panel is to make sure that if there are any parts that hang over the edges, they have clearance from machinery.

Rails should typically have a width of half an inch or 12.7mm.

In order to save money, you may want just 2 rails instead of 4. In other words you will sometimes want rails on 2 sides instead of rails on all 4 sides.

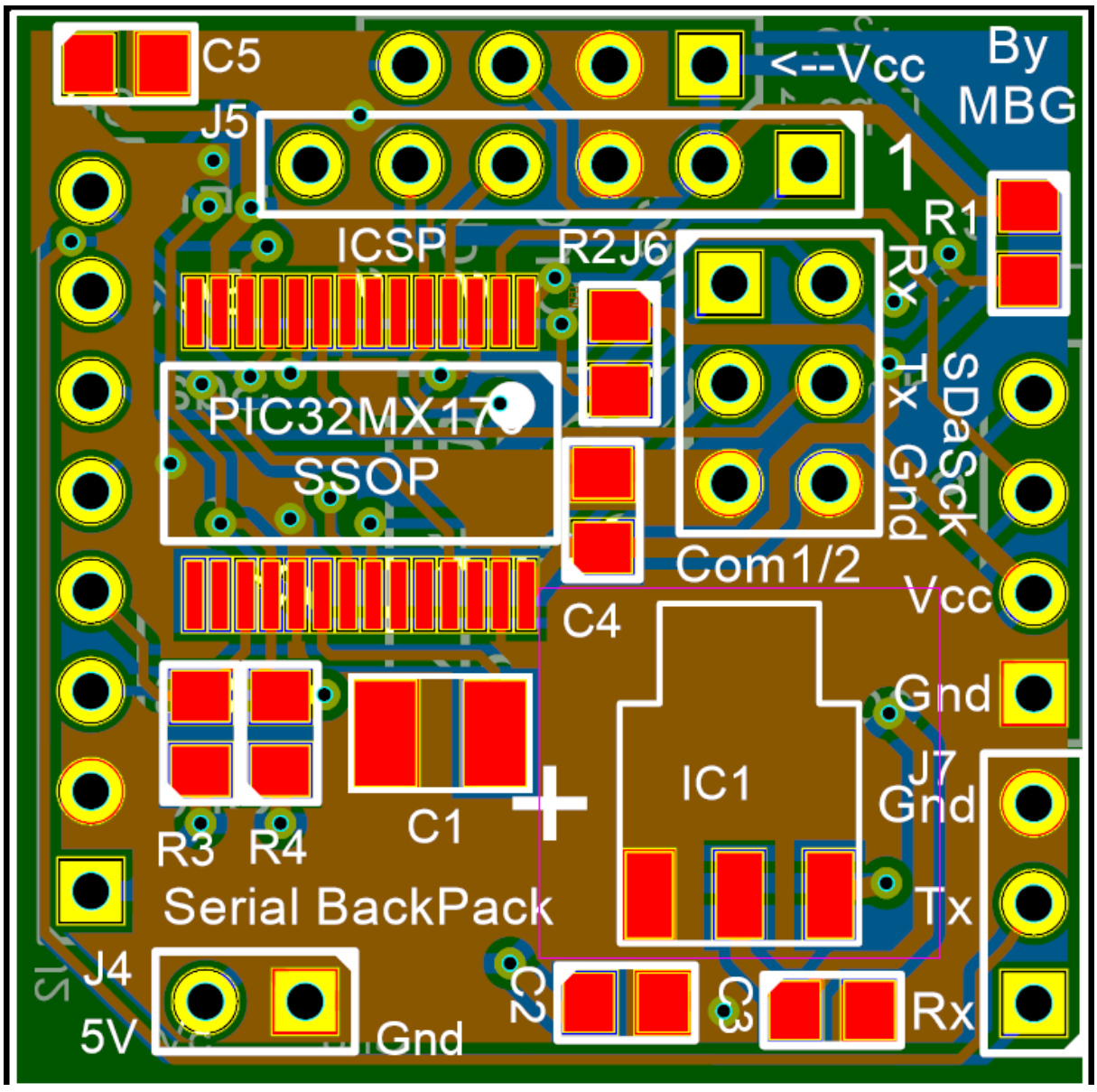
If you are using only 2 rails then they should be on the long edge of the panel, not the short edge.

You can add optional assembly rails on:

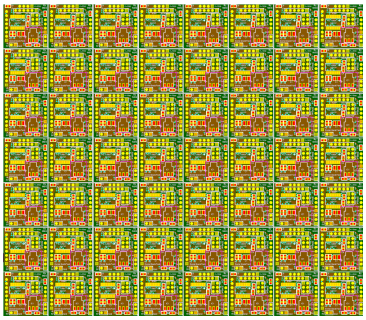
1. Both the left and right side of the array of PCBs and/or
2. Both the top and bottom side of the PCB array.

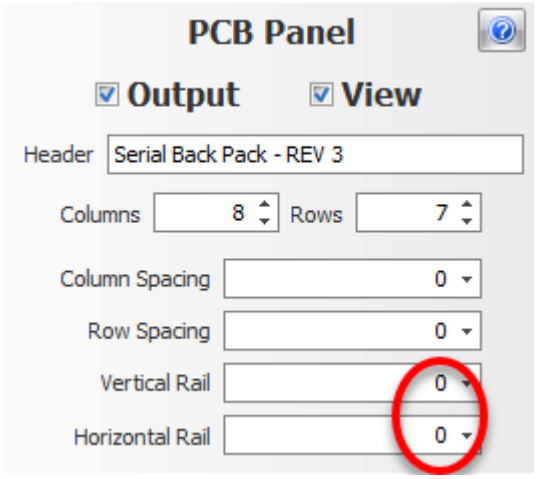
Below are some examples of using assembly rails.

Single Design Before Panelization

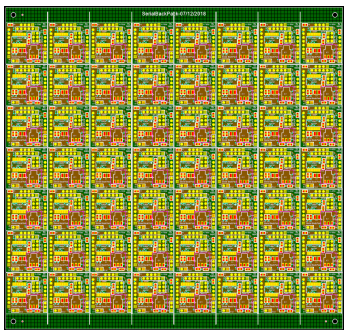


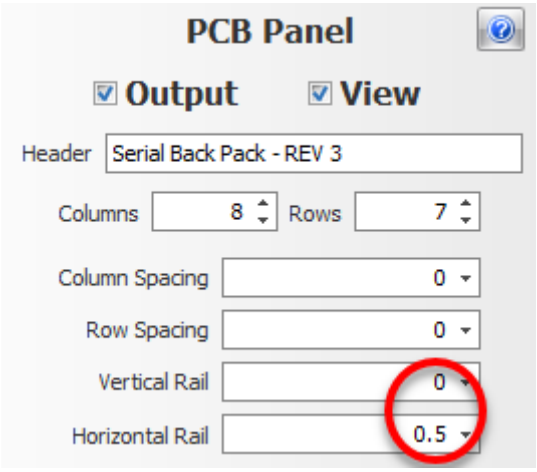
Panel for a PCB arrayed 56 times (8x7) With No Assembly Rails



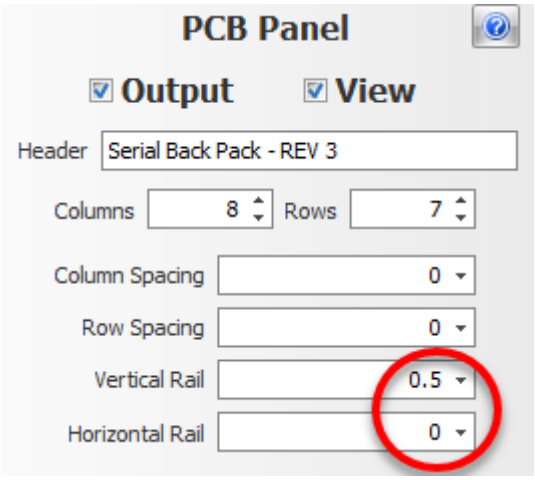


Top and Bottom Assembly Rails

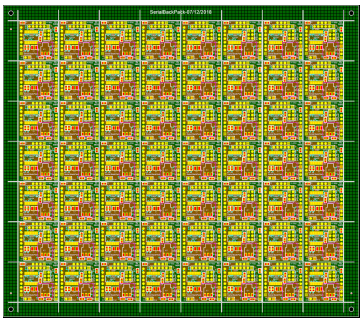
3. 

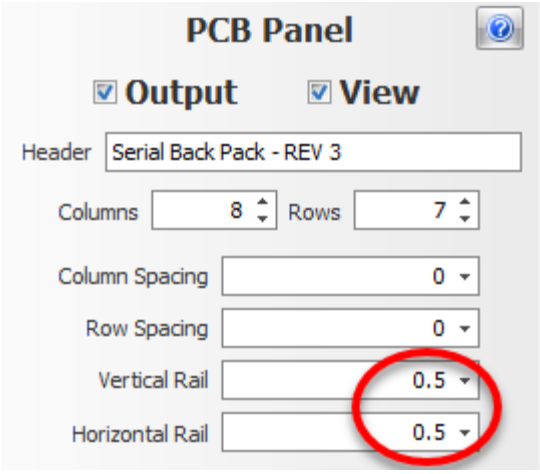


Left and Right Assembly Rails



Assembly Rails on all 4 Sides

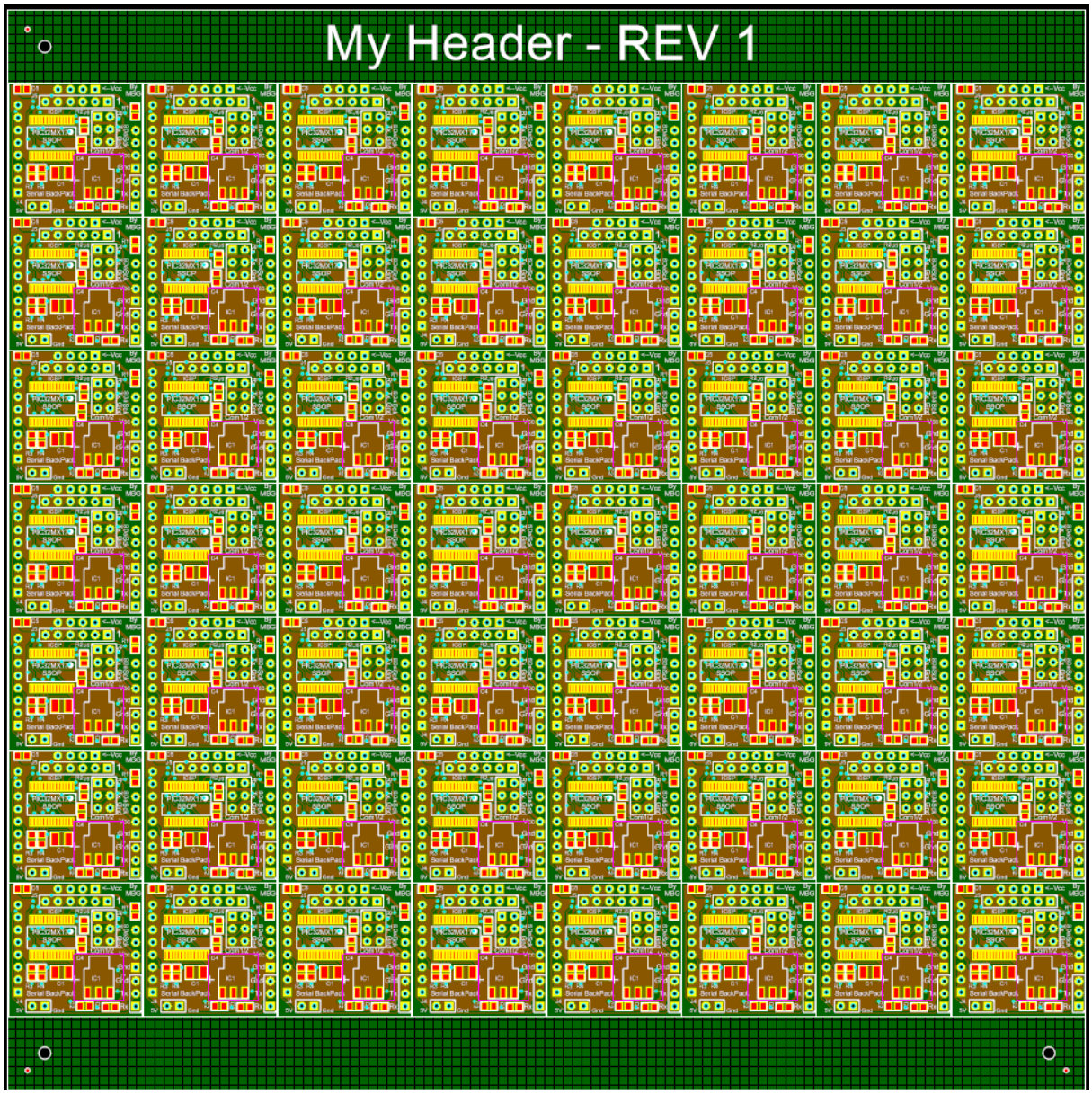




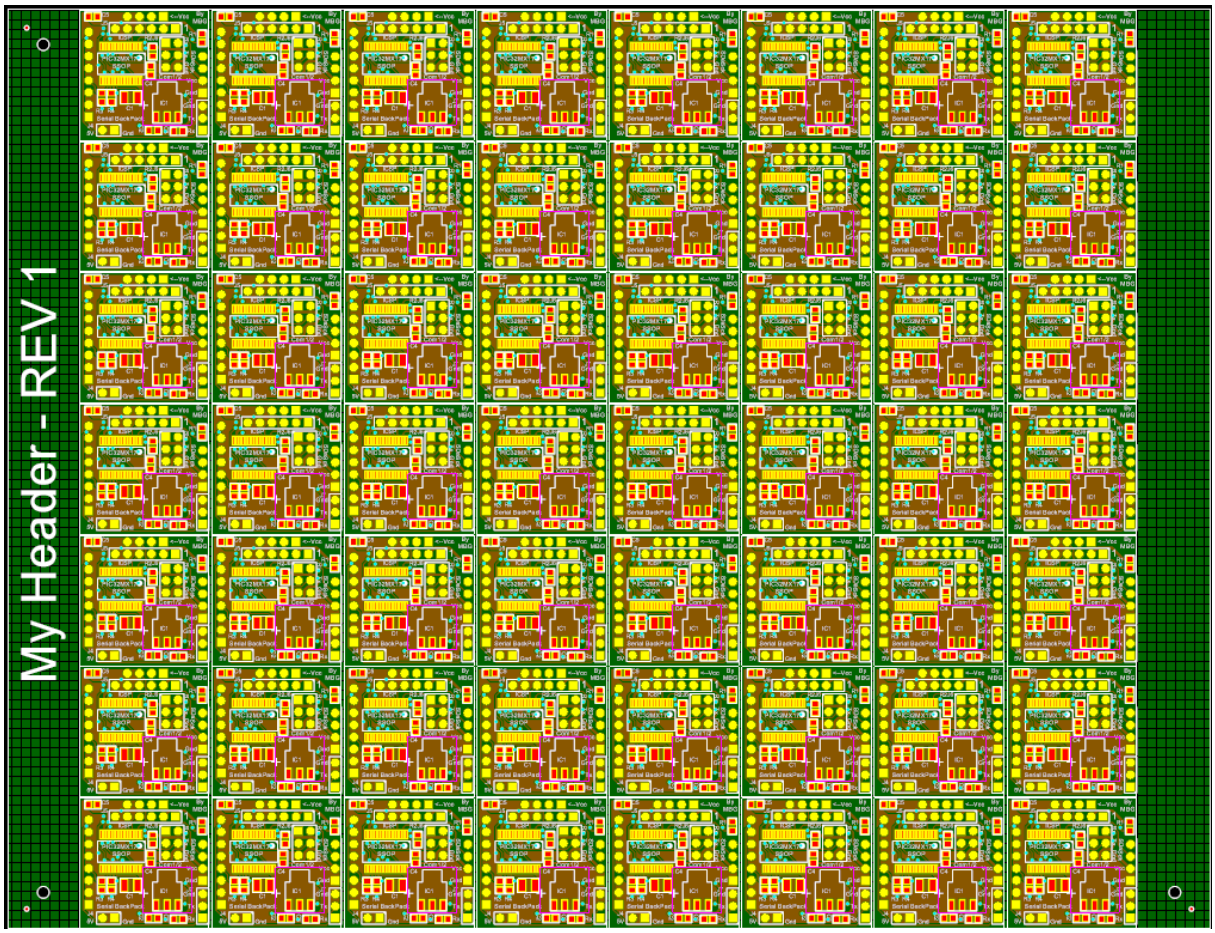
1.2.6.11.16.8 The Panel Header

You can optionally add a text header to your panel. The header will only appear if you have added a set of vertical or horizontal rails. The header will appear on the top rail if the rail exists else it is added to the left rail.

Header on top rail



Header on the left rail



1.2.6.11.16.9 The Panel's Pick and Place File

Pick and Place Origin

If you add automatic fiducial to your panel then the origin for the Pick and Place file coordinates is set to the lower left fiducial else it is set to the bottom left corner of the panel.

Creating the Pick and Place File

The pick and place file is added to your zip file if you use the Prepare for Manufacturing option.

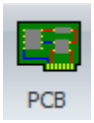
Part of a typical pick and place file

Note: the duplication of part references for each part on each arrayed PCB

	A	B	C	D	E	F	G	H	I
1	Designator	Footprint	Center-X(Mil)	Center-Y(Mil)	Ref-X(Mil)	Ref-Y(Mil)	Layer	Rotation	Comment
2	AD1		2645.3	-41.3	2645.3	-41.3	Top	90	
3	AD1		5445.3	-41.3	5445.3	-41.3	Top	90	
4	AD1		2645.3	2158.7	2645.3	2158.7	Top	90	
5	AD1		5445.3	2158.7	5445.3	2158.7	Top	90	
6	C1	100	955.3	1408.5	955.3	1408.5	Top	180	
7	C1	100	3755.3	1408.5	3755.3	1408.5	Top	180	
8	C1	100	955.3	3608.5	955.3	3608.5	Top	180	
9	C1	100	3755.3	3608.5	3755.3	3608.5	Top	180	
10	C10	100	1217.5	438.5	1217.5	438.5	Top	0	
11	C10	100	4017.5	438.5	4017.5	438.5	Top	0	
12	C10	100	1217.5	2638.5	1217.5	2638.5	Top	0	
13	C10	100	4017.5	2638.5	4017.5	2638.5	Top	0	
14	C11	4.7 50V	1366.4	213.2	1366.4	213.2	Top	90	
15	C11	4.7 50V	4166.4	213.2	4166.4	213.2	Top	90	
16	C11	4.7 50V	1366.4	2413.2	1366.4	2413.2	Top	90	

1.2.6.11.17 Viewing the PCB

To view the PCB for a design:

- Click on the 

button in the **Schematics/Panels** button group or
- the **Panels→Windows** button group or ***right-click*** in a schematic view and click on the View PCB button or
- In the Project Panel, ***double-click*** on the PCB in the project tree.

1.2.6.11.18 Semi-Transparent PCBs

You can make your PCB semi-transparent and even separate the layers so you can see the internal structure of your PCB.

1.2.6.11.19 PCB Internals

You can make your PCB semi-transparent and even separate the layers so you can see the internal structure of your PCB.

1.2.6.11.20 The PCB Border

The PCB border defines the physical board on which your parts are placed and is the support for the copper tracks that connect the electrical terminals of the parts.

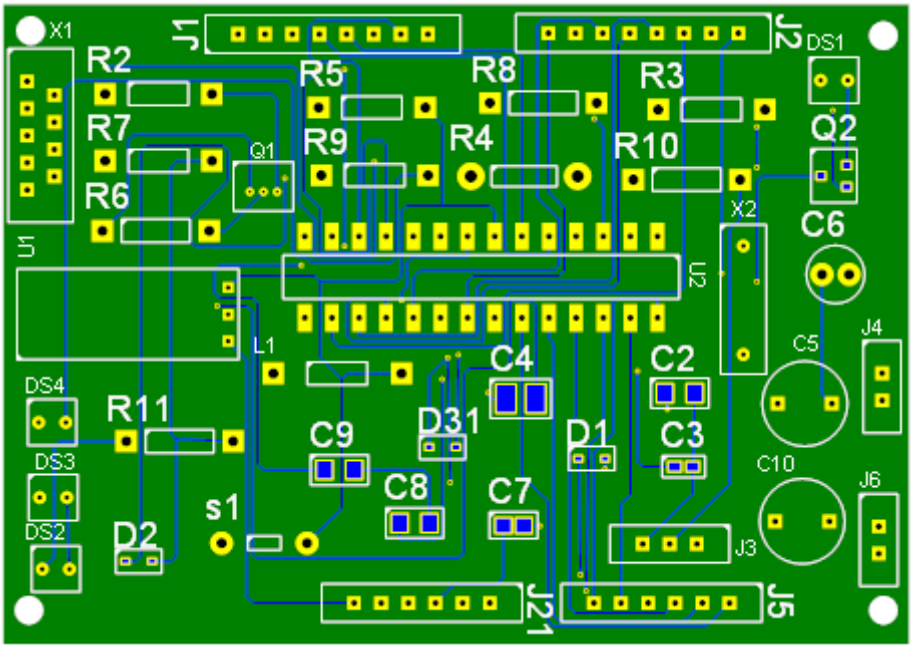
Creating a Rectangular PCB

Creating a Circular/Elliptical PCB

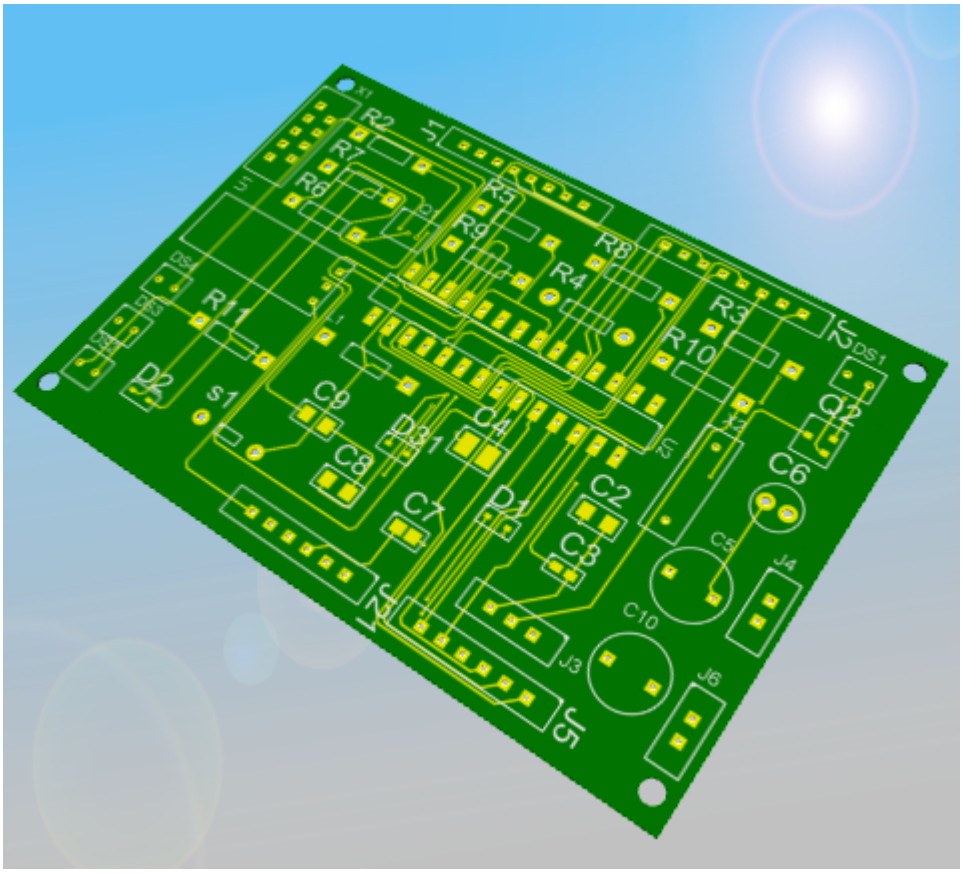
Adding Holes to a PCB

Adding Cutouts to a PCB

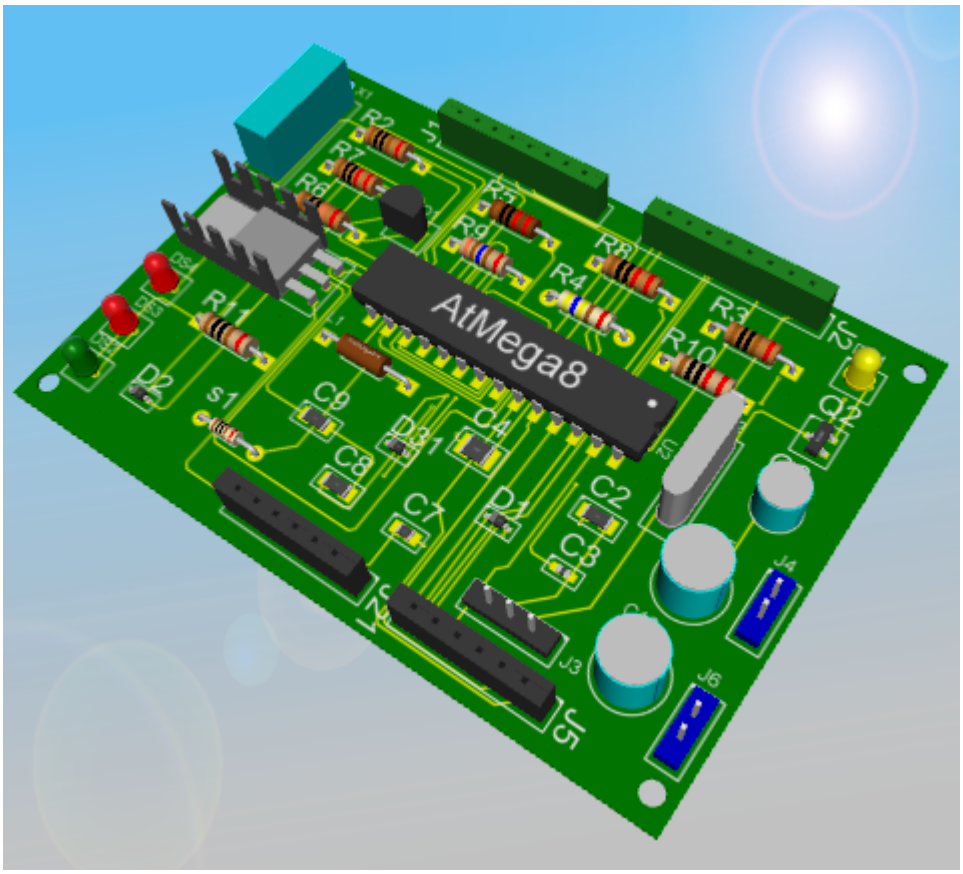
Basic rectangular PCB



PCB with pads for components and tracks



3D view of PCB



3D View of PCB with parts mounted

1.2.6.11.20.1 Creating a Rectangular PCB

A rectangular PCB is a PCB defined by a rectangle only. All corners are 90°.

To set the PCB border click on one of the 3 buttons in the **PCB→PCB** button group

To create a simple rectangular PCB click the  button.

Now as you move the mouse inside the PCB viewport you will see the PCB first corner cursor shown below.

PCB first corner cursor

To set the first corner, click the left mouse button or type in the actual X and Y coordinate (press the ***Enter*** key to start numeric input from the keyboard).

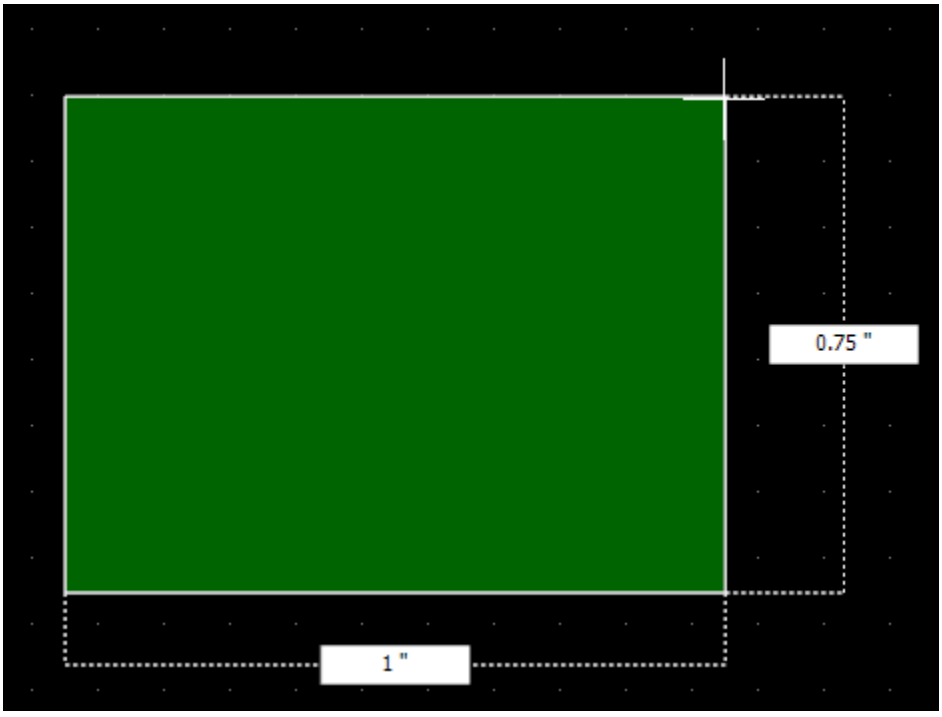
Now as you drag the mouse the PCB will change shape as shown below. As you drag the mouse the width and height of the board will be instantly updated.

Making the Rectangle Square. If you hold down the ***CTRL*** key then the x dimension (width) and y dimension (height) are the same.

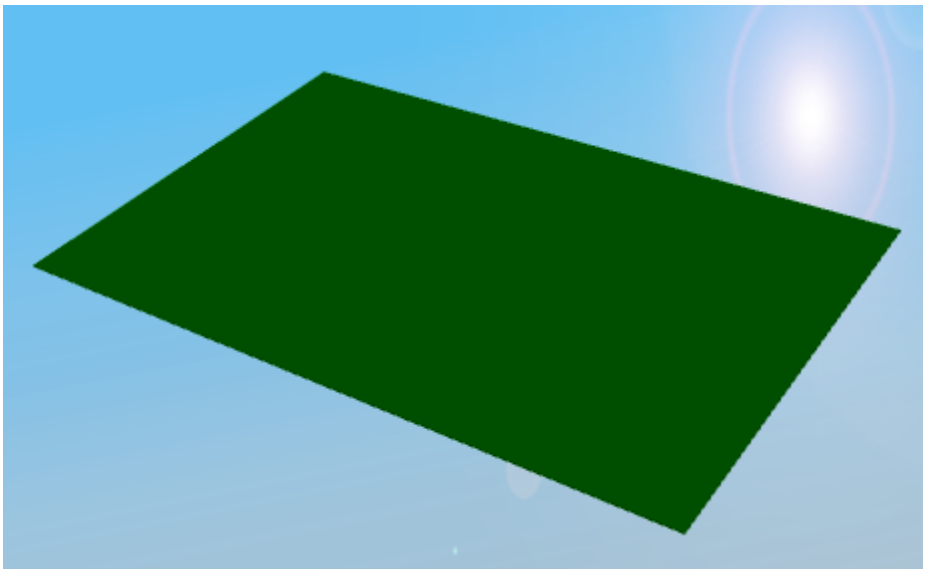
Centering the Rectangle. If you hold down the Shift key then the rectangular PCB you are adding is centered at the start point.

Creating a centered Rectangle. You can use both the ***CTRL*** key and the Shift key at the same time.

Left-click and hold to complete the PCB border when it is the size you want or enter the width and height using the keyboard (press the ***Enter*** key to start numeric input from the keyboard).



PCB shape defined as you drag the mouse



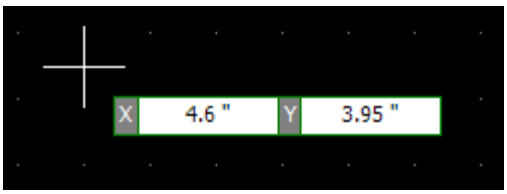
3D view of a rectangular PCB

1.2.6.11.20.2 Creating a Circular/Elliptical PCB

To create a circular or elliptical PCB click the button in the **PCB→PCB** button group 

To create a simple rectangular PCB click the button.

Now as you move the mouse inside the PCB viewport you will see the PCB first corner cursor shown below.



To set the first corner, ***click*** or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value of the starting point, then ***Enter***.

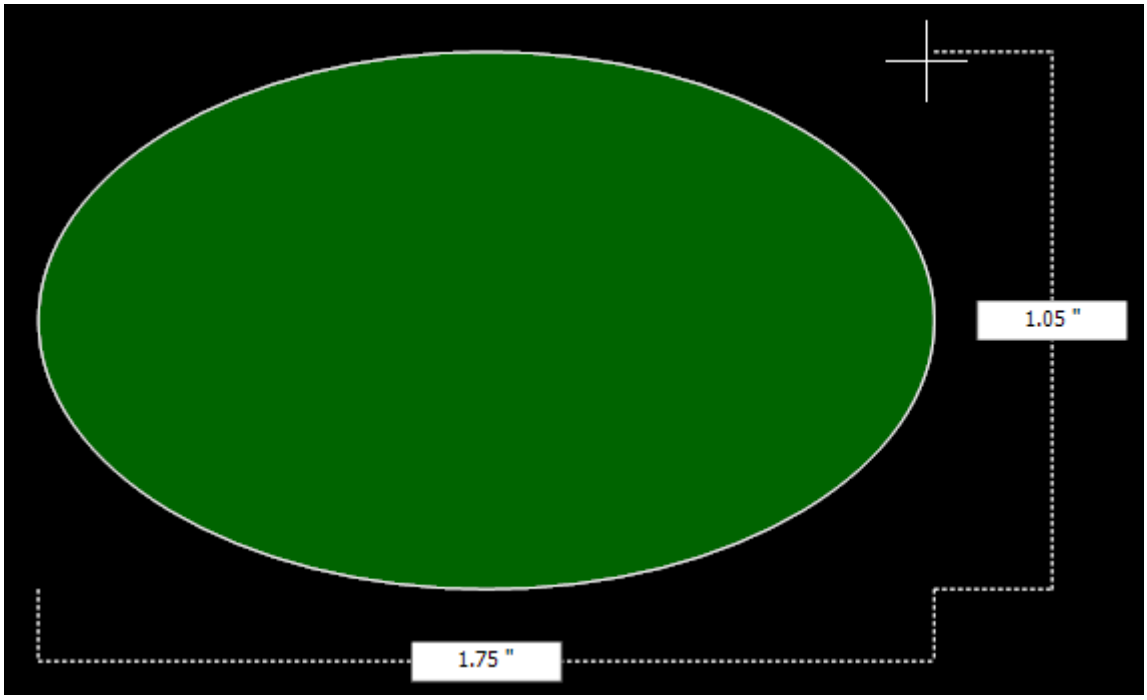
Now as you drag the mouse the PCB will change shape as shown below. As you drag the mouse the width and height of the board will be instantly updated.

Making the Ellipse a Circle. If you hold down the ***CTRL*** key then the x dimension (diameter) and y dimension (diameter) are the same.

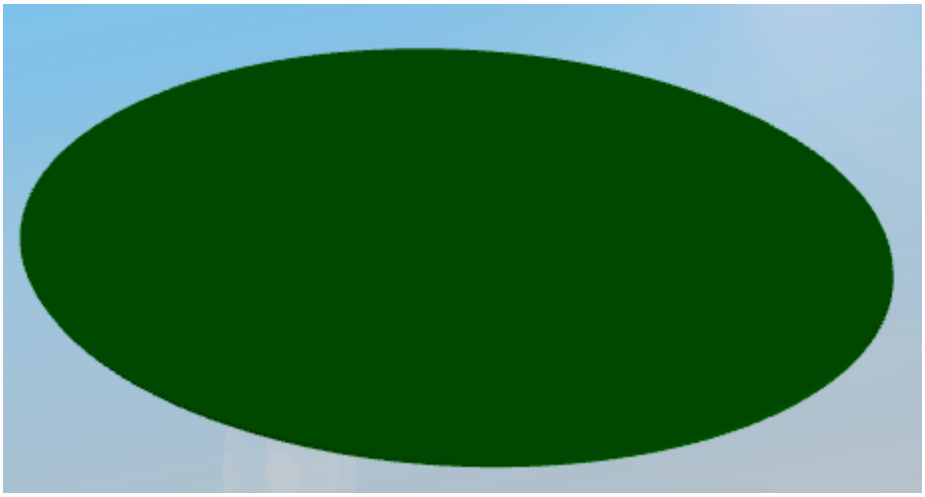
Centering the Circle/Ellipse. If you hold down the ***Shift*** key then the circular or elliptical PCB you are adding is centered at the start point.

Creating a centered Circle. You can use both the ***CTRL*** key and the ***Shift*** key at the same time.

Left-click to complete the PCB border when it is the size you want or enter the width and height using the keyboard (press the ***Enter*** key to start numeric input from the keyboard).



PCB shape defined as you drag the mouse



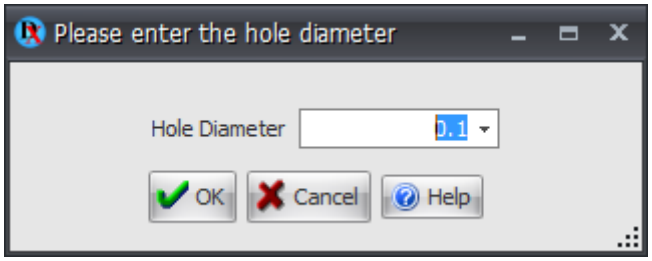
3D view of an elliptical PCB

1.2.6.11.20.3 Creating a Polygonal PCB

To create a polygonal PCB border click the **Add→PCB→** button.

1.2.6.11.20.4 Adding Holes to a PCB

To add a hole to a PCB click on the Drill Hole button in the **Add→Cutouts** button group. You will then be prompted for the diameter of the hole. Enter the hole diameter and click the OK button to add it. The value you enter is remembered by AutoTRAX DEX and used the next time you add a hole.



As you move the cursor in the PCB viewport the hole will move. You will notice that the PCB will now be shown as a solid color, this is so you can see a true representation of the hole. The X and Y coordinate of the center of the hole will be displayed as shown below.

You can either:

- Move the mouse so the hole is positioned where you want it and ***left-click hold*** and drag, then release to create the hole
- ***Left-click*** to start the hole then hit ***Enter,*** enter the X value, Enter, then the Y value, and finally enter to place the hole.

NOTE: PCB holes are different to circular cutouts as they will be output as commands to the drill file. Circular cutouts are not output as drill commands.

1.2.6.11.20.5 Adding Cutouts to a PCB

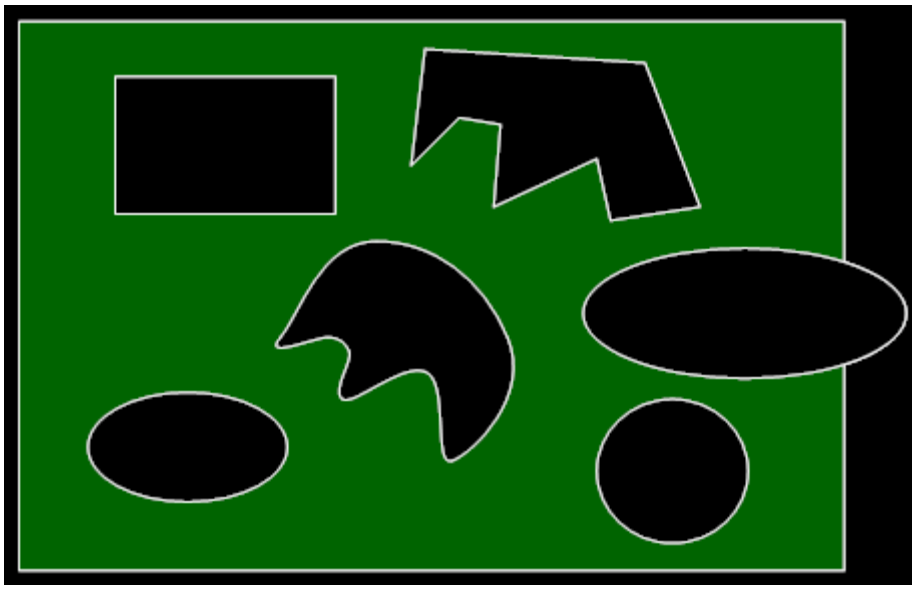
A cutout is a section of PCB that has been removed.

You can add 3 different types of PCB cutouts, rectangular, polygonal or circular/elliptical as shown below. Cutouts can extend to outside the basic PCB shape as shown in the elliptical cutout on the right of the PCB.

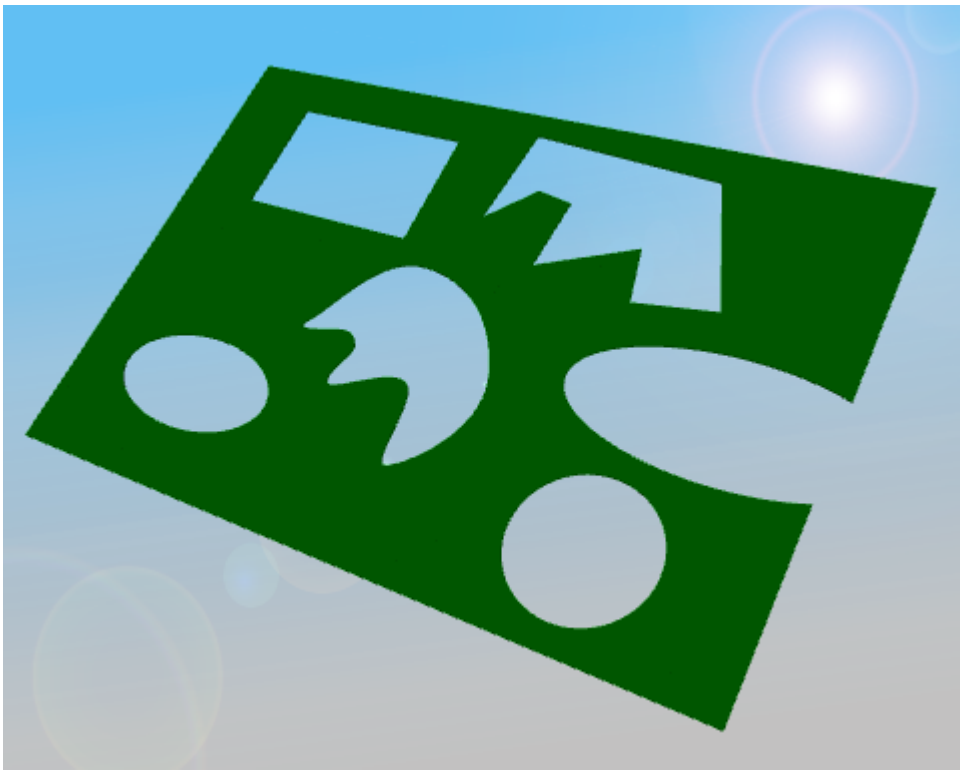
Adding a Rectangular PCB Cutout

Adding a Circular/Elliptical PCB Cutout

Adding a Polygonal PCB Cutout



Cutouts in a PCB



3D view of cutouts

1.2.6.11.20.6 Adding a Rectangular PCB Cutout

To create a rectangular PCB cutout click the Add→Cutout→ button.

1.2.6.11.20.7 Adding a Circular/Elliptical PCB Cutout

To create a rectangular PCB cutout click the Add→Cutout→ button.

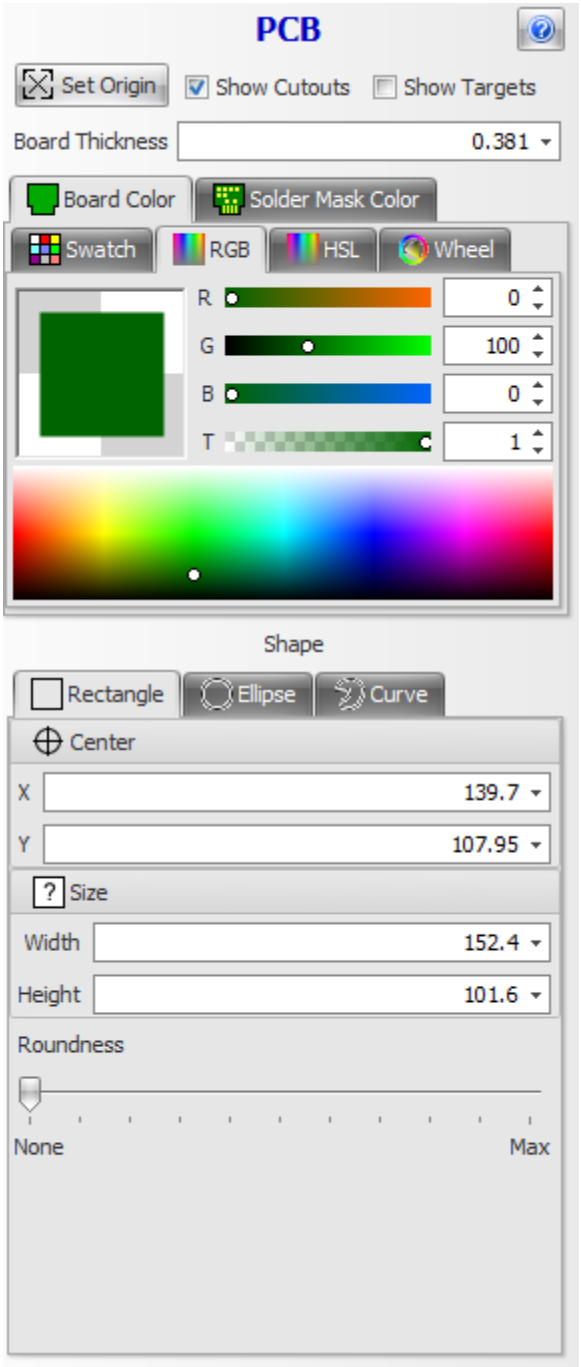
1.2.6.11.20.8 Adding a Polygonal PCB Cutout

To create a rectangular PCB cutout click the Add→Cutout→ button.

1.2.6.11.20.9 Editing the PCB Border

To edit a PCB, ***double-click*** on one of its edges.

You can also use its properties dialog by first selecting it.



The PCB properties dialog

1.2.6.11.21 PCB Layers

The number of layers you can have in AutoTRAX DEX is limited only by the amount of memory in your PC. AutoTRAX DEX can cope with hundreds of layers; far more than your PCB manufacturer can make.

To set the current layer and the layer visibilities use the Layers panel.

PCB-Layers

Editing the PCB Layers

1.2.6.11.21.1 PCB-Layers

Printed circuits and footprints both contain the following layers.

Document

the document layer contains graphics that you can add to document your design. There are no restrictions on color and style etc.

Net

the net layer displays unrouted track segments.

If a footprint is placed outside the PCB area and the footprint net does not connect to any footprints inside the PCB area then the net that connect to the footprint will not appear. This is to avoid clutter. However when you move the footprint that is outside the PCB area the nets will be displayed.

Top Package

the top package layer contains graphics that define the 3-D content on the top of the PCB/footprint.

Top Silkscreen

the top silk screen layer defines the graphics that will be used to print a graphics design on the top of the PCB.

Top Copper

The top copper is used to define the copper tracks, but shapes, copper filled regions etc. that will be on the underside of the PCB.

Bottom Copper

The bottom copper is used to define the copper tracks, but shapes, copper filled regions etc. that will be on the underside of the PCB.

Bottom Silkscreen

The top silk screen layer defines the graphics that will be used to print a graphics design on the top of the PCB.

Bottom Package

The bottom package layer contains graphics that define the 3-D content on the underside of the PCB/footprint.

Background

The background can be used for graphics that you want to be displayed underneath all the other layers. It is useful for adding objects such as images that you want to use as a trace pattern.

Inner Layers

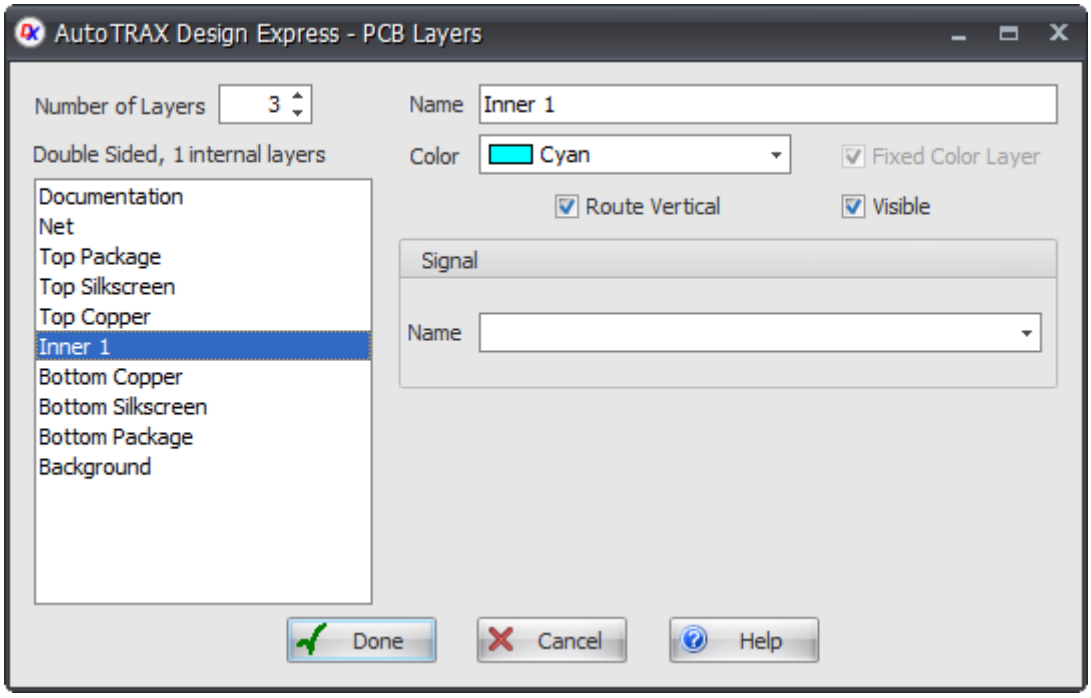
PCBs can also have a number of inner layers.

1.2.6.11.21.2 Editing the PCB Layers

You can set and edit the layers using the 

button at the top left of the layers panel.

The layer editor dialog box shown below will appear.



Set the number of layers using the 

control.

To edit the properties of an individual layer first select it using the left layers list box.

Each layer can have it's own name. Se it using the

control.

All layers except the top and bottom package layers. the documentation layer, and the background layer have a fixed color. You can set the fixed color for these layers using the 

layer color control.

Check/uncheck 

to set the preferred auto-routing direction of the layer. This only applies to electrical layers.

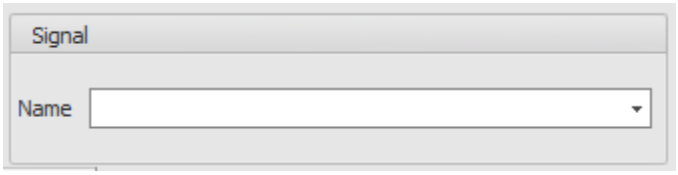
Check/uncheck 

to hide/show a layer.

Signals

Layers can act as signal planes. Enter the name of the signal for the layer. If a layer has a signal name then it will be filled with copper with holes for pads and vias that do not connect to the layer.

Tracks with the same net name as the layer will connect to that layers via any pads/vias that pass through to that layer. These pads/vias will connect directly to the signal layer.

Enter the signal name in the 

control.

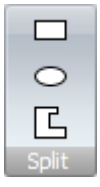
Clear the signal name (set it to nothing) to make the layer act as a normal layer.

1.2.6.11.21.3 Solder Mask Differences

This video shows you solder mask differences in PCBs

1.2.6.11.21.4 Split Power Planes

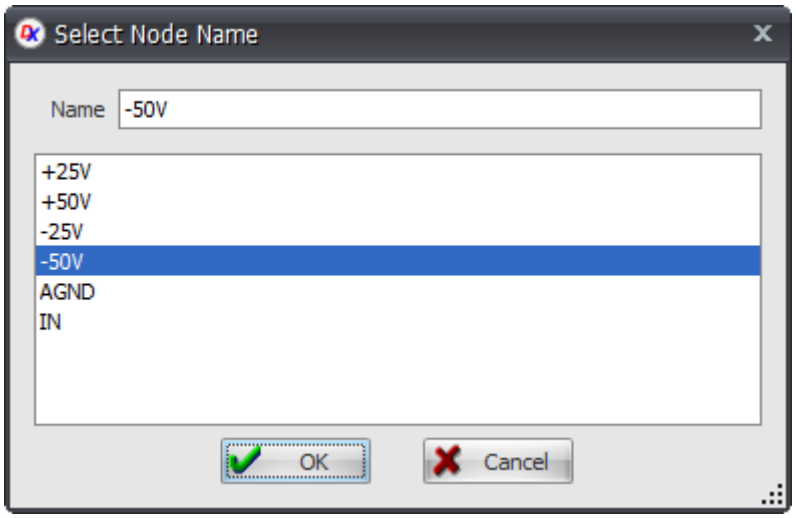
Split power planes work with the Electra router. Spread power planes will not work with the internal router.

To add a split power plane click on the PCB→Split Buttons 

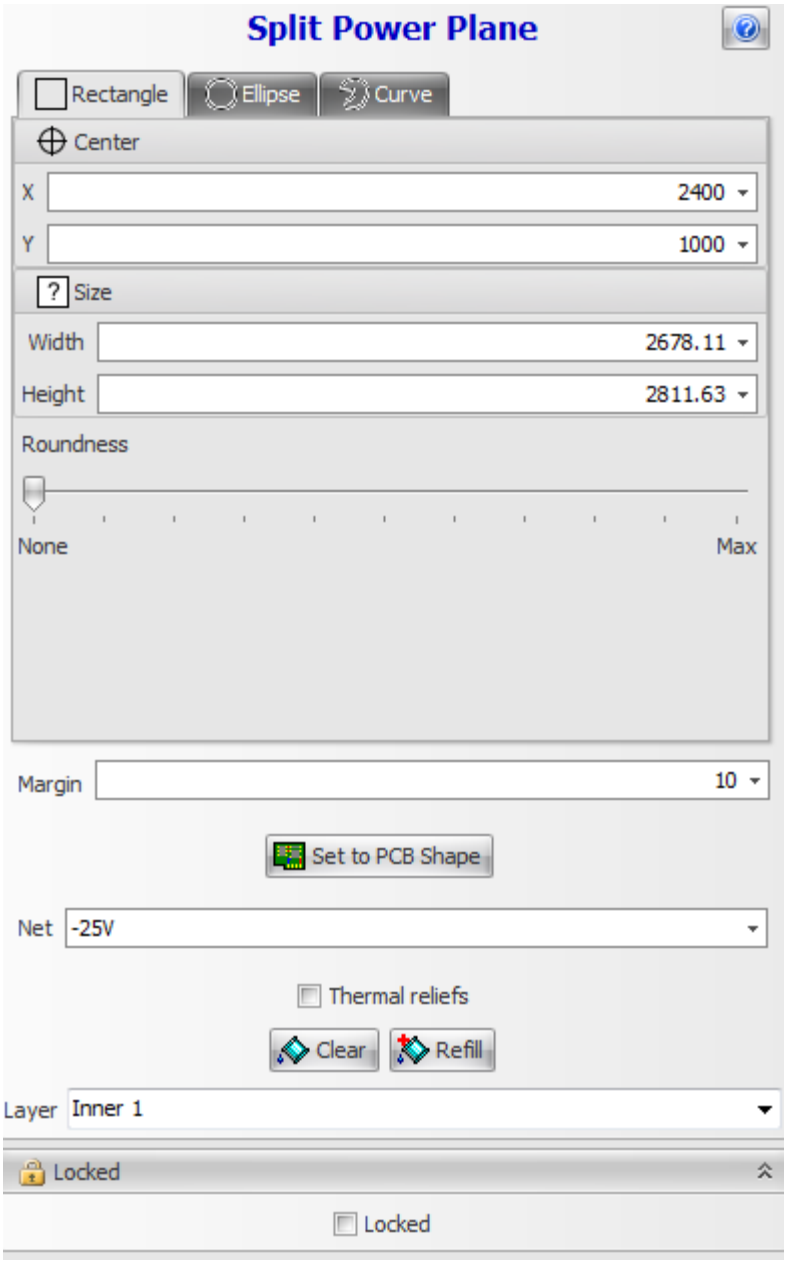
A split power plane can only be added to an internal layer.

The Electra router will also be enabled.

Once add you will be prompted for a signal/node name.



1.2.6.11.21.5 Editing Split Power Regions



1.2.6.11.22 Pads

There are two different classes of pads, surface mount technology **SMT** pads and thru-plated hole technology **TPH** pads

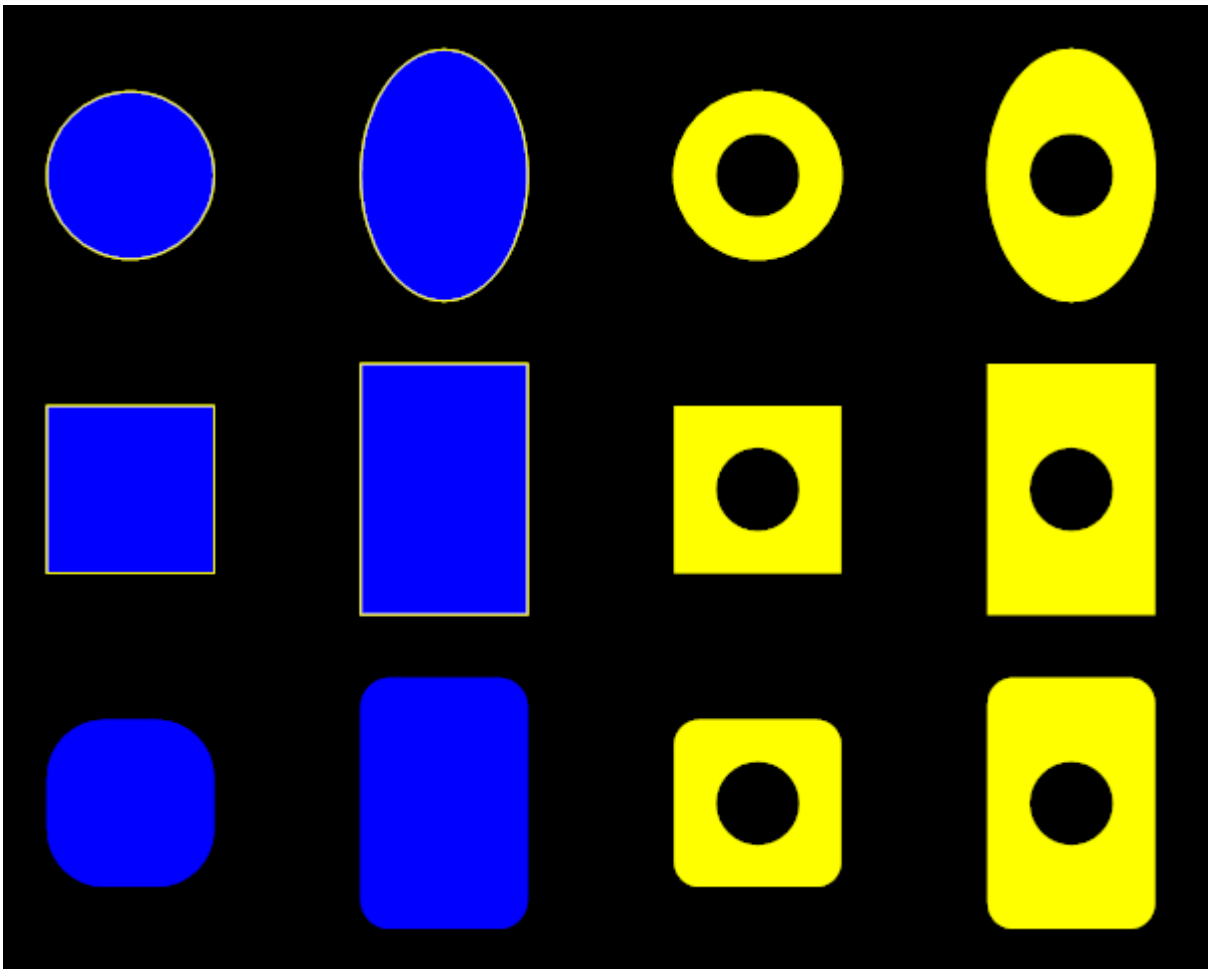
SMT Pads

SMT pads are cover areas on only one side of the PCB. Electronic parts adhere to the pads using the solder on the pads.

TPH Pads

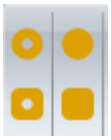
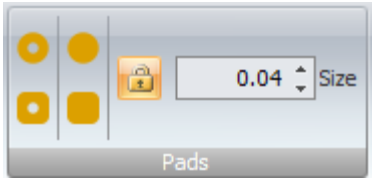
TPH pads have copper areas on both the top and the bottom sides of the PCB. In addition a hole is drill through the circuit board to connect the top pad area to the pad area on the bottom. The electronic pin for the part is inserted through the hole and often the sides of the hole are plated with copper. However, if you are making a single sided board then the copper will most likely be placed on the underneath side of the PCB unless the PCB is being made with SMT parts, in which case you will need to have the copper pads on the side of the board where you mount the SMT parts, in other words the top side.

You can add the following types of pads.



Pads

1.2.6.11.22.1 Adding Pads

To add a pad to a PCB or footprint click the one of the 4  buttons in the **Add→Pads** in the ribbon button group. 

Adding TPH pads

Click the  button to add a round/elliptical TPH pad.

Click the  button to add a rectangular TPH pad.

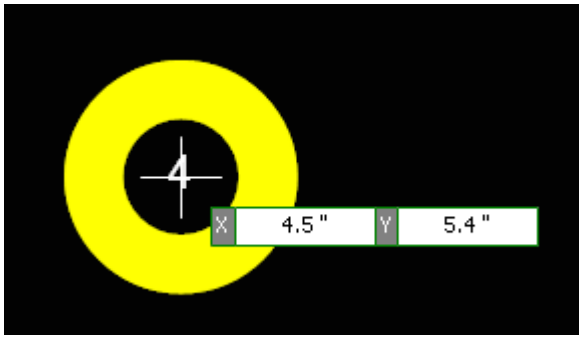
Adding SMT pads

Click the  button to add a round/elliptical SMT pad.

Click the  button to add a round/elliptical SMT pad.

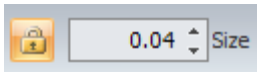
Placing the Pad

After you have clicked on one of the 4 buttons you will see the pad with a point input cursor. Move the pad to the position you wish and ***left-click*** or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to exactly place the center of the pad.

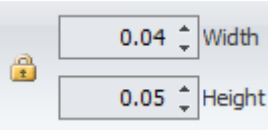


Pad with point input cursor

Settings



sets the size of square/circular pads



sets the size of rectangular/elliptical pads

Adding Pads in PCBs

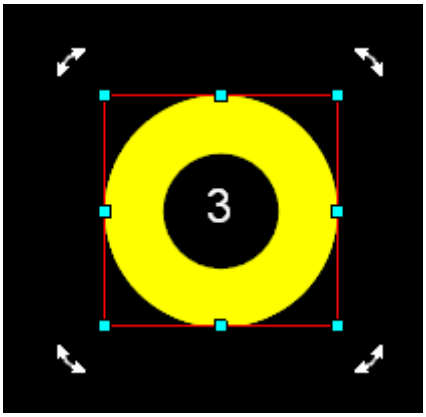
If you add a pad in a PCB, a schematic symbol for the pad is automatically created and added to your first schematic. You can then use this to make connections.

1.2.6.11.22.2 Editing Pads

To edit a pad you can select it and then drag it to its new position. You can also rotate and scale it.

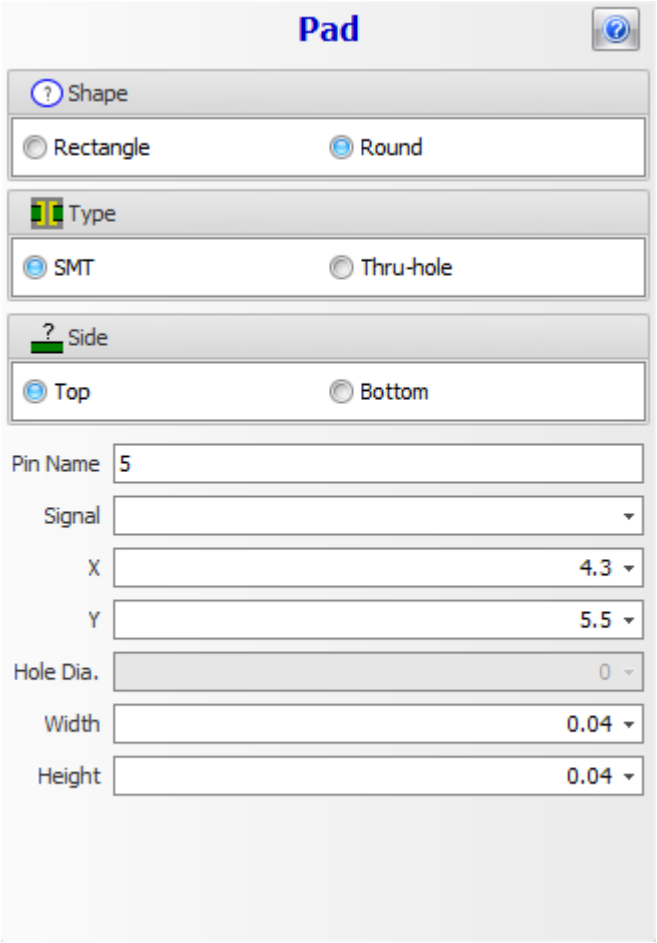
Drag any of the 4  rotate manipulator points to rotate it.

Drag any of the 6 ▫ manipulator points to scale it.



Selected Pad

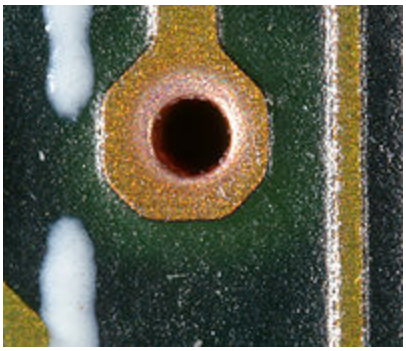
You can also use the Pads Properties dialog in the properties panel.



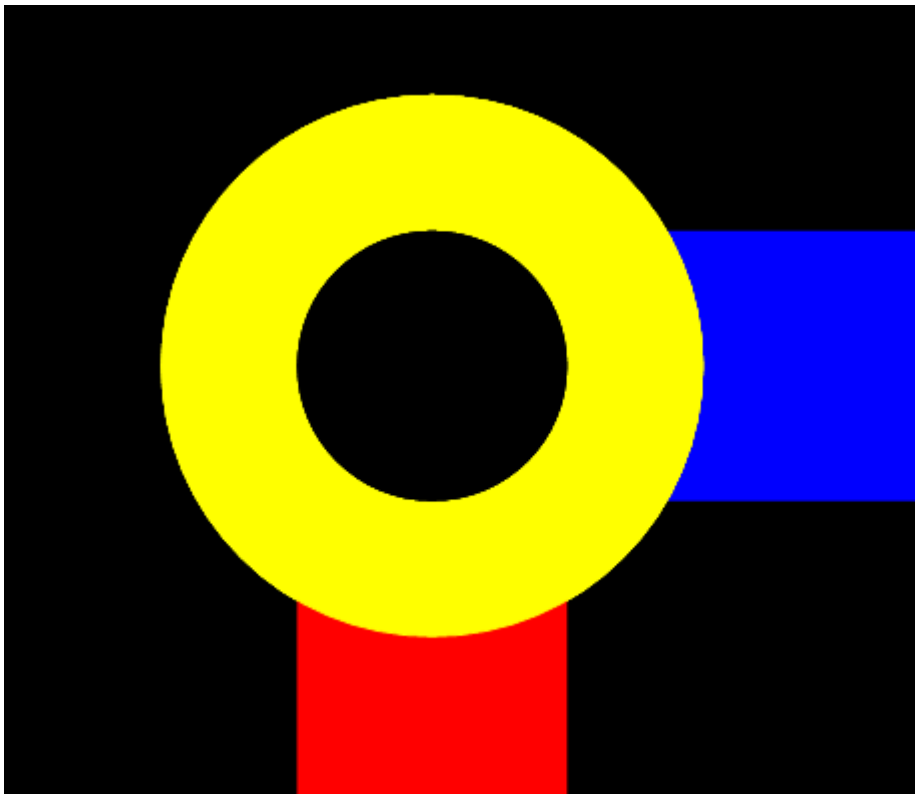
Pad Properties Panel

1.2.6.11.23 Vias

A via is a hole that is used to form an electrical connection between two signal layers of a PCB. Vias are like round pads, which are drilled and usually through-plated when the board is fabricated.



A through-hole via



A typical Via

There are 2 types of vias in AutoTRAX DEX.

Automatic Vias

The first type of via is one that is automatically created by a track in order to switch layers.

Manually Created Vias

The second type of via is one created by you to provide an electrical connection between layers.

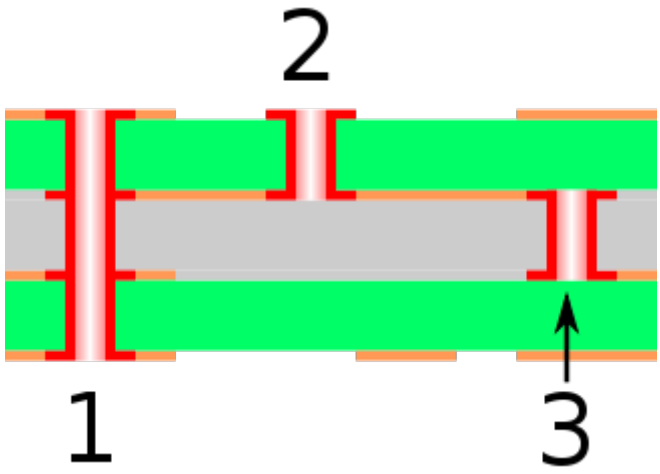
A via consists of two pads, in corresponding positions on different layers of the board, that are electrically connected by a hole through the board. The hole is made conductive by electroplating, or is lined with a tube or a rivet. High-density multi-layer PCBs may have micro vias: blind vias are exposed only on one side of the board, while buried vias connect internal layers without being exposed on either surface. Thermal vias carry heat away from power devices and are typically used in arrays of about a dozen.

A via consists of:

- Barrel — a conductive tube filling the drilled hole
- Pad — connects each end of the barrel to the component, plane or trace

- Antipad — a clearance hole between barrel and no-connect metal layer

A via may be at the edge of the board so that it is cut in half when the board is separated; this is known as a castellated hole and is used for a variety of reasons, including allowing one PCB to be soldered to another in a stack.

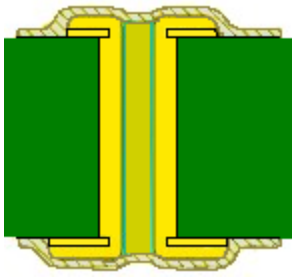


Different types of vias

(1) Through hole.
(2) Blind via.
(3) Buried via.
The grey and green layers are non-conducting, while the thin orange layers and vias are conductive

Tented Vias

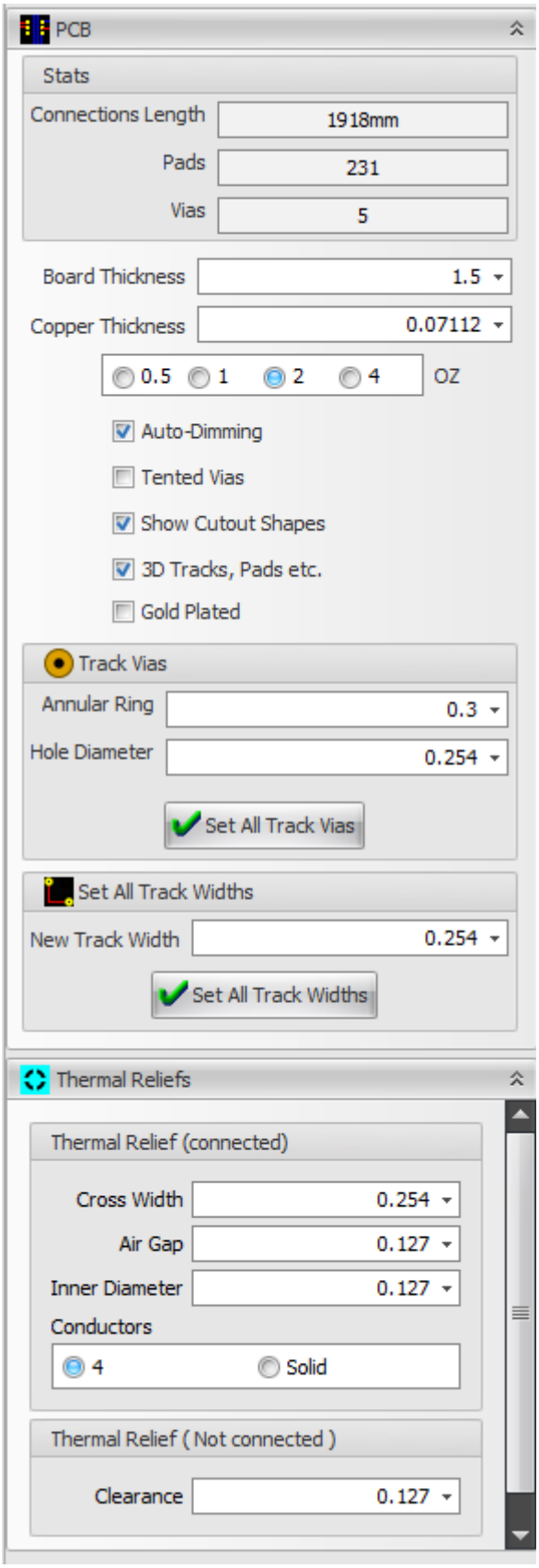
A tented via is a via with dry film solder mask which completely covers both the via pad and its plated-through hole. In some cases this may also be referred to as a Capped Via or Cap plating, although the phrase tented via is much more common.



A Tented Via

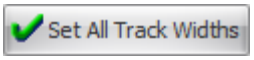
1.2.6.11.24 PCB Sheet Editor

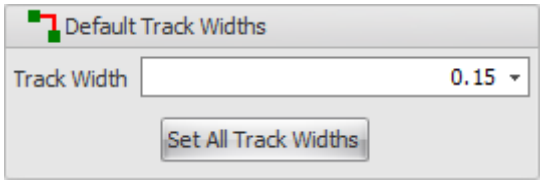
The addition properties for the PCB are settable in the setting sections below. The sheet editor is displayed in the Properties Panel when nothing is selected.



1.2.6.11.25 Setting All Track Widths

To set the width of all tracks on a PCB enter the width into the Set All Tracks Widths section of the PCB Sheet Properties Editor.

Enter the new track width and click 

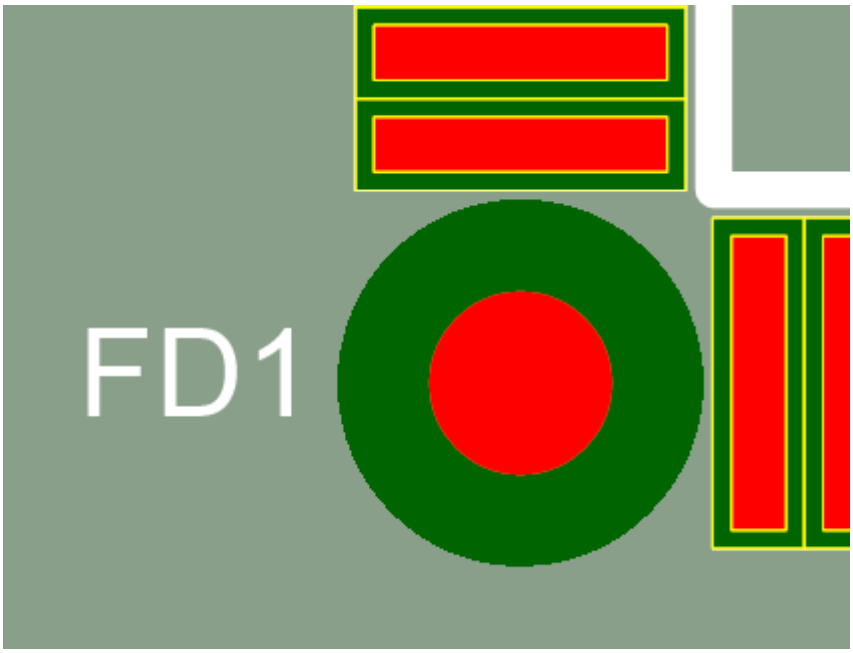


1.2.6.11.26 Fiducials

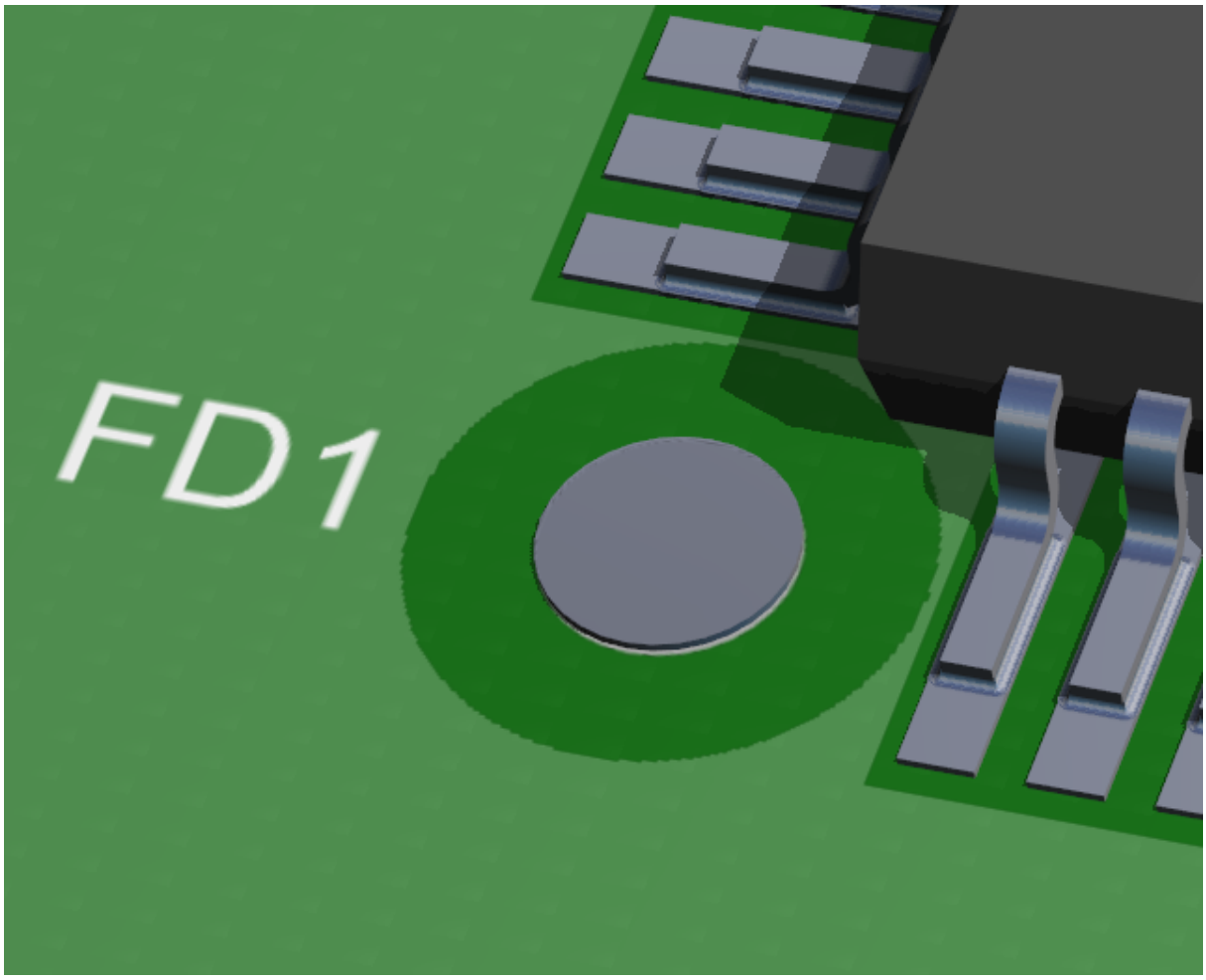
A fiducial marker or fiducial is an object placed in the field of view of an imaging system which appears in the image produced, for use as a point of reference or a measure. It may be either something placed into or on the imaging subject, or a mark or set of marks in the reticule of an optical instrument.

A fiducial mark is a printed board artwork feature that is created in the same process as the conductive pattern and that of is a measurable point for component mounting with respect to a land pattern or land patterns.

A PCB fiducial is a circular area of copper on either the top of the bottom side of the PCB that is used for Alignment using optical recognition systems. The fiducial mark and this PCB copper tracks must be etched in the same step to ensure alignment.



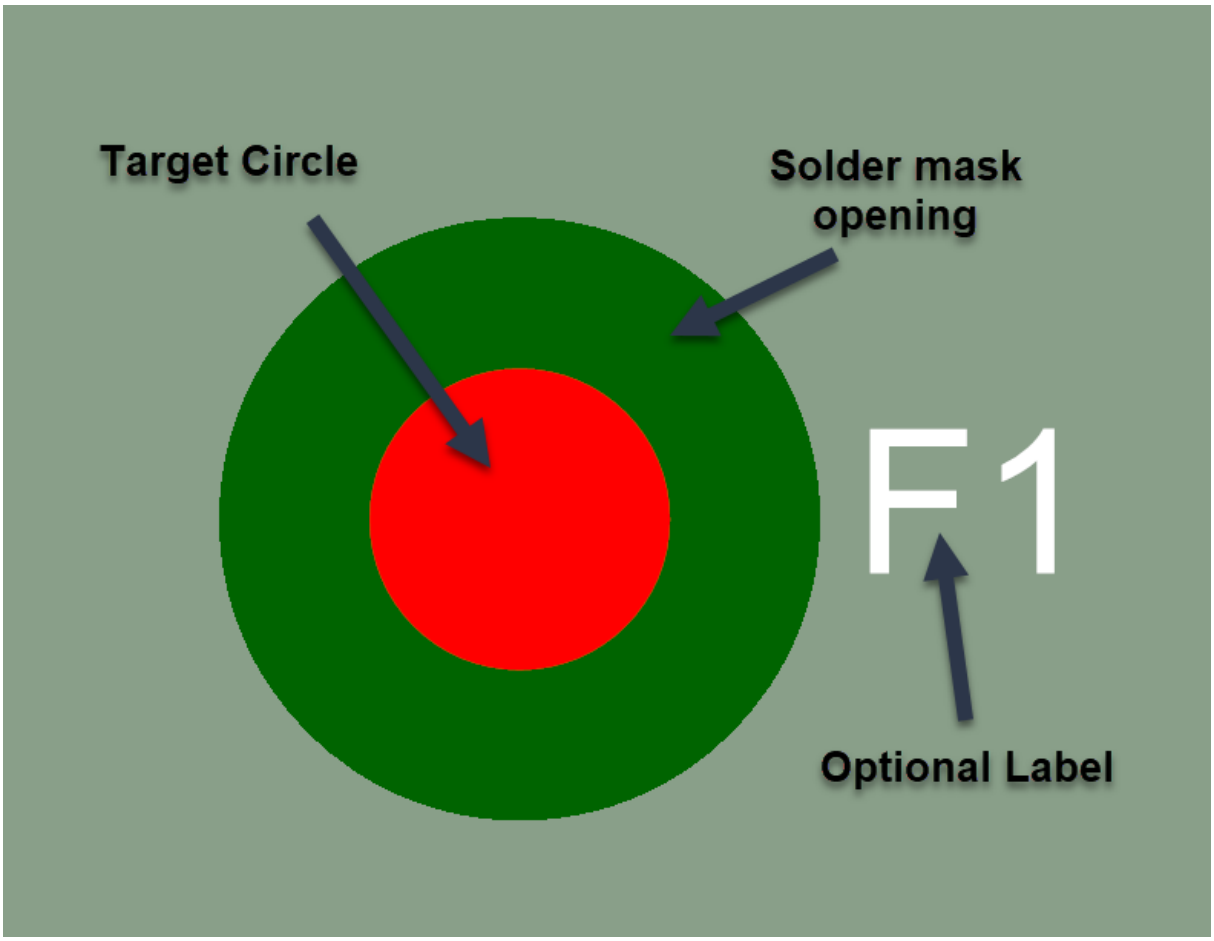
A PCB fiducial in the 2D View



A PCB fiducial in the 3D View

A fiducial marks is made up of 3 parts.

1. A solid circle of copper on the top or bottom copper layer. This is the alignment target.
2. An opening in the solder mask so the circular alignment target is clearly visible.
3. An option text label to the side of the opening in the solder mask



A Fiducial Mark

The fiducial marks provide a common set of data points for all steps in the PCB assembly process. This allows all pieces of equipment used for assembly to accurately locate the parts based on the circuit pattern.

In printed circuit board (PCB) manufacturing, fiducial marks, also known as circuit pattern recognition marks, allow SMT placement equipment to accurately locate and place parts on boards. These devices locate the circuit pattern by providing common measurable points. They are usually made by leaving a circular area of the board bare from solder-mask coating. Inside this area is a circle exposing the copper plating beneath. This center metallic disc can be solder-coated, gold-plated or otherwise treated, although bare copper is most common if not a current-carrying contact. Alternatively, it is possible to use clear solder-mask lacquer to cover the fiducials. In order to minimize rounding errors it was good practice to place fiducials in the same grid (or some multiple of it) that was used to place the parts, however, this isn't always possible on high-density boards nor is it a requirement any more with modern high-precision machines.

Most placement machines are fed boards for assembly by a rail conveyor, with the board being clamped down in the assembly area of the machine. Each board will

clamp slightly differently than the others, and the variance—which will generally be only tenths of a millimeter—is sufficient to ruin a board without proper calibration. Consequently, a typical PCB will have multiple fiducials to allow placement robots to precisely determine the board's orientation. By measuring the location of the fiducials relative to the board plan stored in the machine's memory, the machine can reliably compute the degree to which parts must be moved relative to the plan, called offset, to ensure accurate placement.

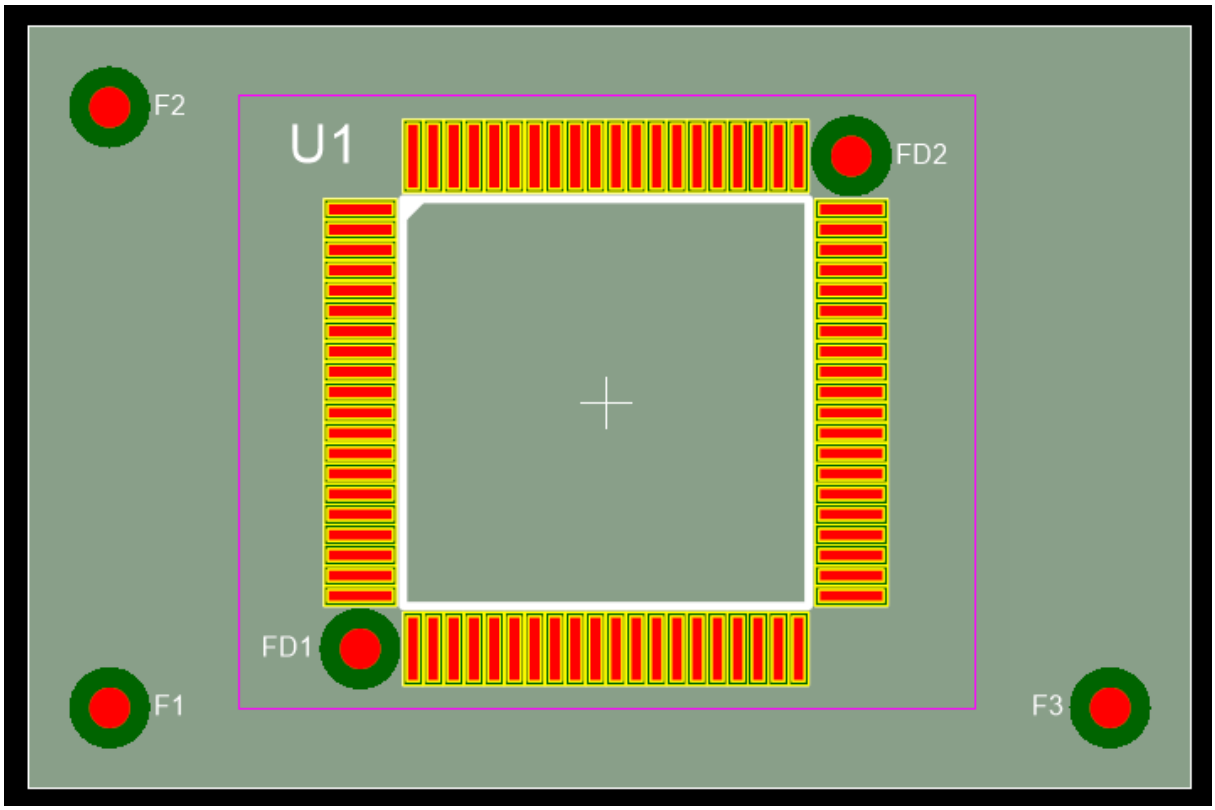
Using three fiducials enables the machine to determine PCB offset in both the X and Y axes, as well as to determine if the board has rotated during clamping, allowing the machine to rotate parts to be placed to match. Such fiducials are also called global fiducials. Global fiducials are also used in conjunction with stencil printing. Without them the printer would not print the solder paste in exact alignment with the pads. Parts requiring a very high degree of placement precision, such as ball grid array packages, may have additional local fiducials near the package placement area of the board to further fine-tune the targeting. Local fiducial, however, cannot be used in the stencil printing process.

Conversely, low end, low-precision boards may only have two fiducials, or use fiducials applied as part of the screen printing process applied to most circuit boards. Some very low-end boards may use the plated mounting screw holes as ersatz fiducials, although this yields very low accuracy.

For prototyping and small batch production runs, the use of a fiducial camera can greatly improve the process of board fabrication. By automatically locating fiducial markers, the camera automates board alignment. This helps with front to back and multilayer applications, eliminating the need for set pins.

There are 3 types of fiducial marks. These are panel fiducials, global fiducials and local fiducials.

Panel fiducials are used on board panels where 2 are more PCBs are manufactured together.

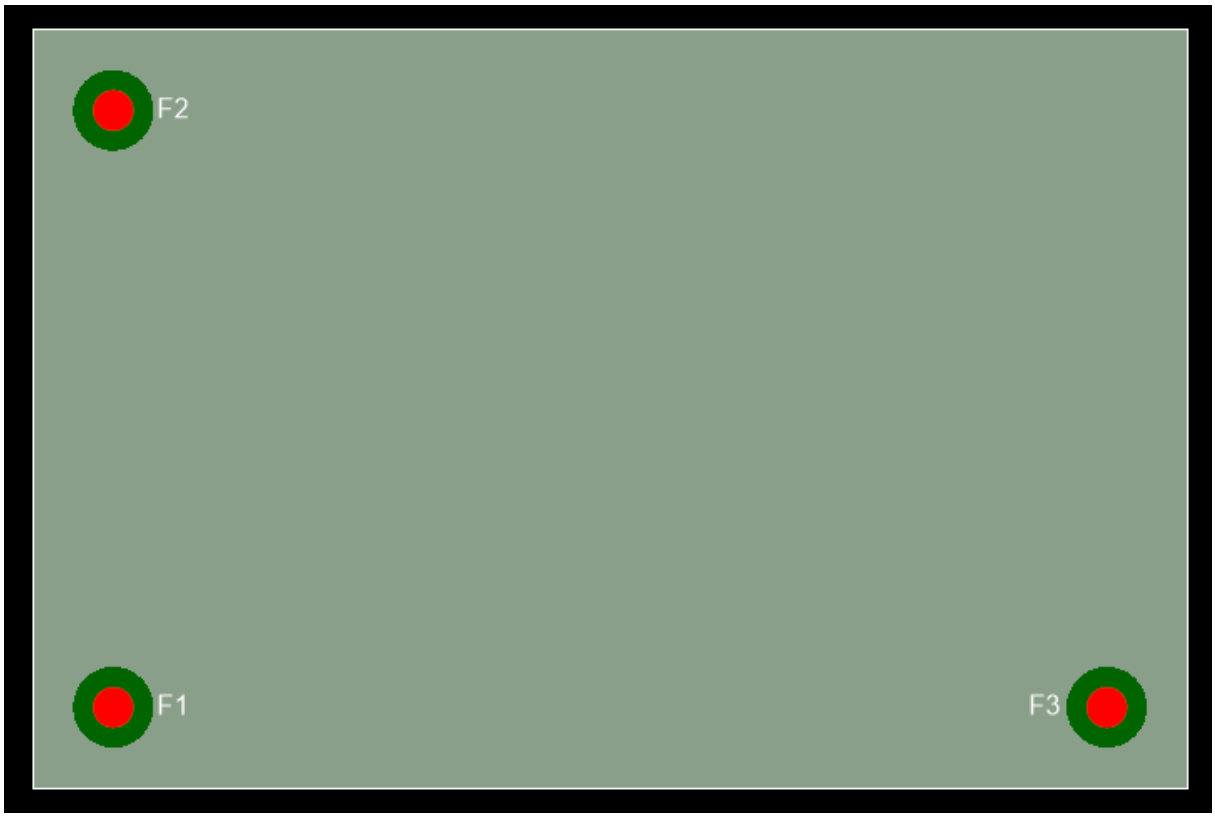


PCB with global and local fiducials.

Global Fiducials

Global fiducials are used to locate the position of all circuit features on an individual PCB. When multiple PCBs are created on the same PCB panel then these global fiducials are referred to as panel fiducials.

A minimum of three fiducial panel/global marks are required for correction of offsets (X, Y positions), rotational offsets (angle position). These should be located orthogonally as far as possible on the PCB panel.



Very small PCB with 3 global fiducials F1, F2 and F3

It is good practice to locate global fiducials in a three point grade are shown below. The first fiducial is located at the lower left usually at the Norton – not location. The second and third fiducials are located vertically or horizontally aligned to the first fiducial. These fiducials should be located on both the top and bottom layers of your PCB that contains surface mount as well as through all components since even so all assemblies systems are beginning to utilize visual alignment systems.

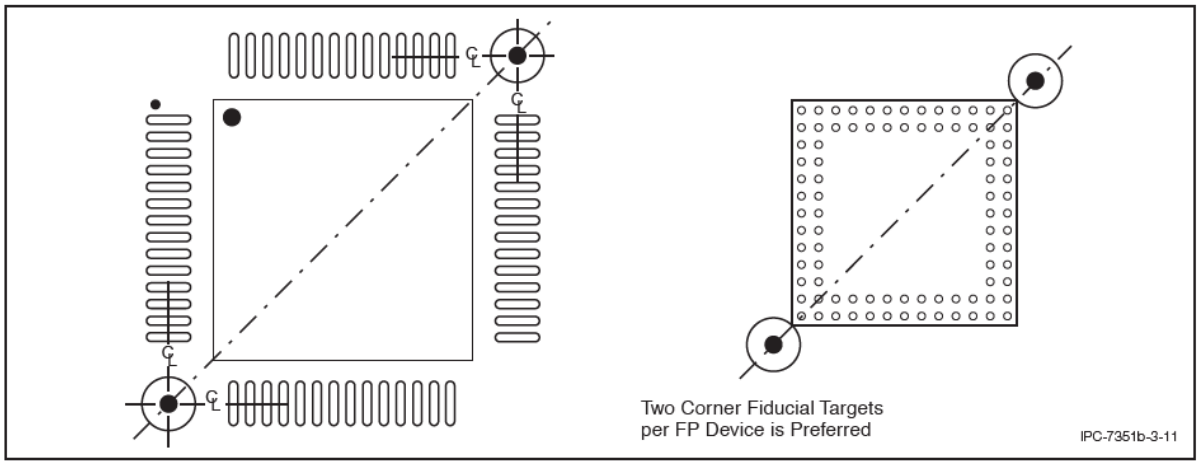
Local Fiducials

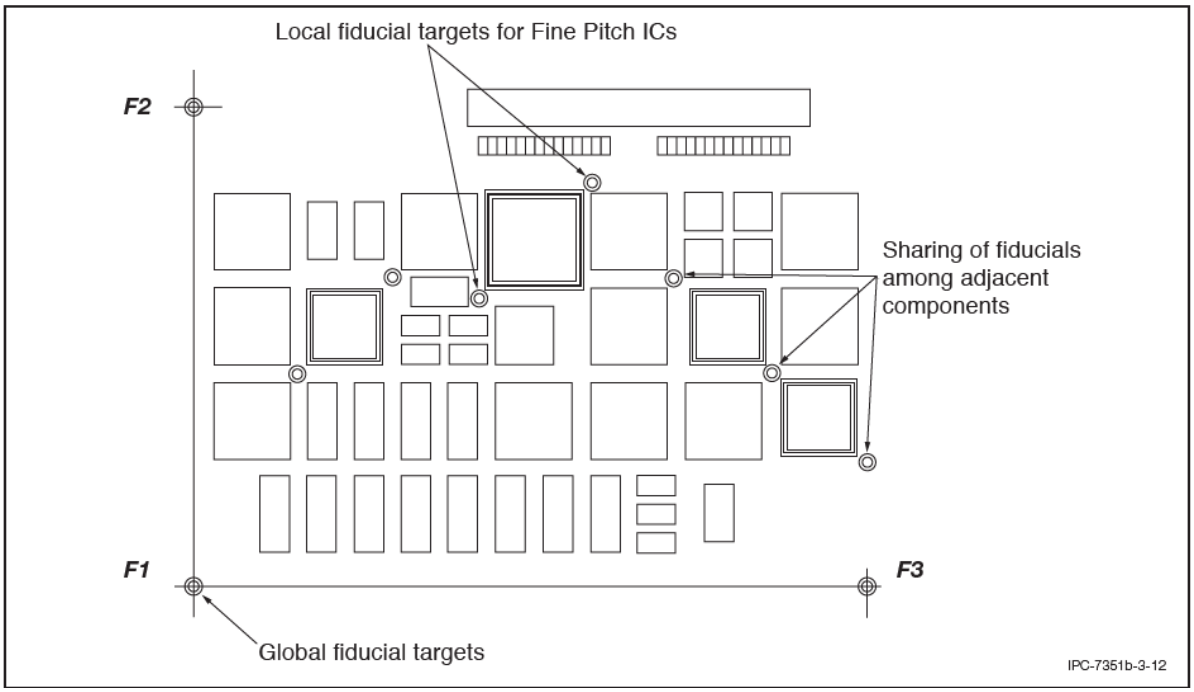
Local fiducial marks are used to locate the position of an individual component requiring more precise placement.

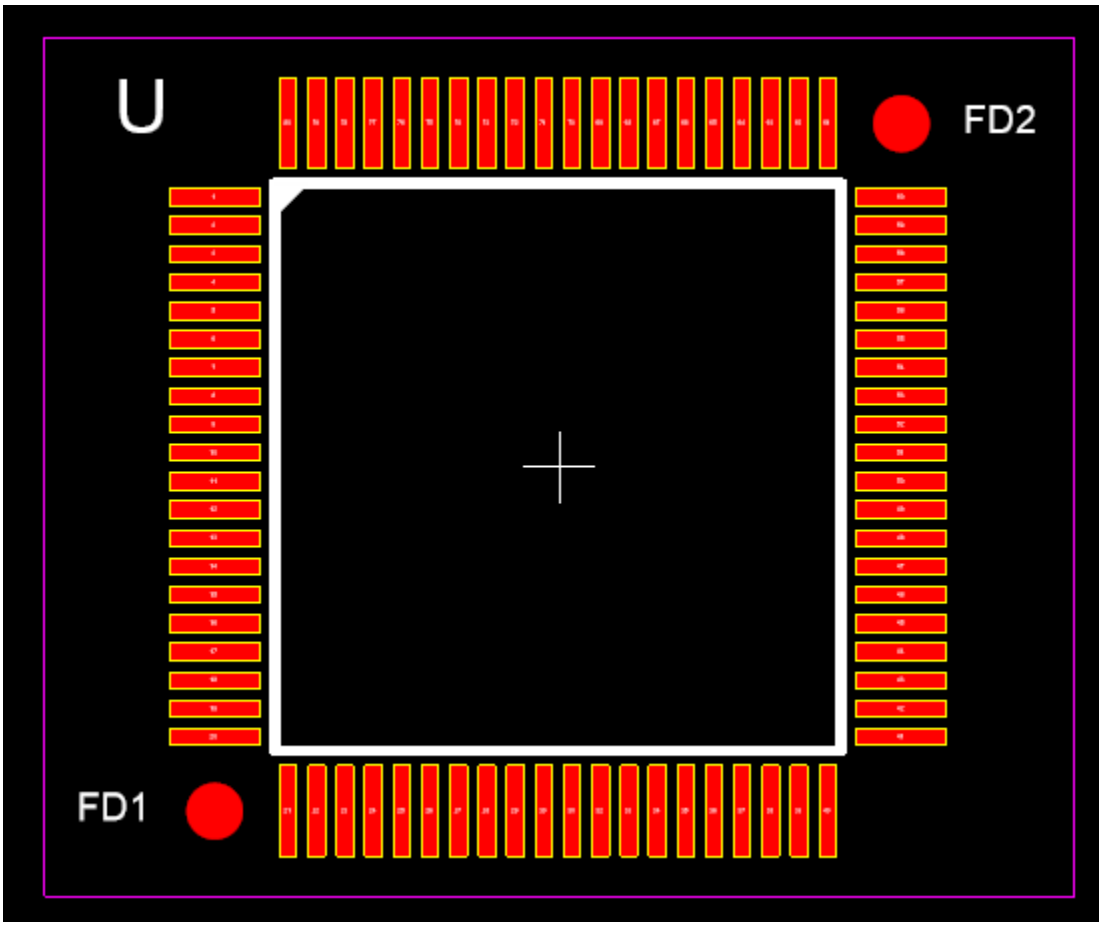
For these components, 2 local fiducial marks are required for correction of translational offsets (X and Y position) and rotational offsets (θ position). This can be to marks located directly opposite each other within the outside perimeter of the land pattern.

All fine pitch components should have 2 local fiducial marks designed into the footprint to ensure that enough fiducials are available every time a component is placed, removed and/ or replaced on the PCB. Fiducial must have a solder mask opening large enough to keep the optical target absolutely free of solder mask. If solder mask should get into the optical target area them some vision alignment systems may fail due to insufficient contrast in the target.

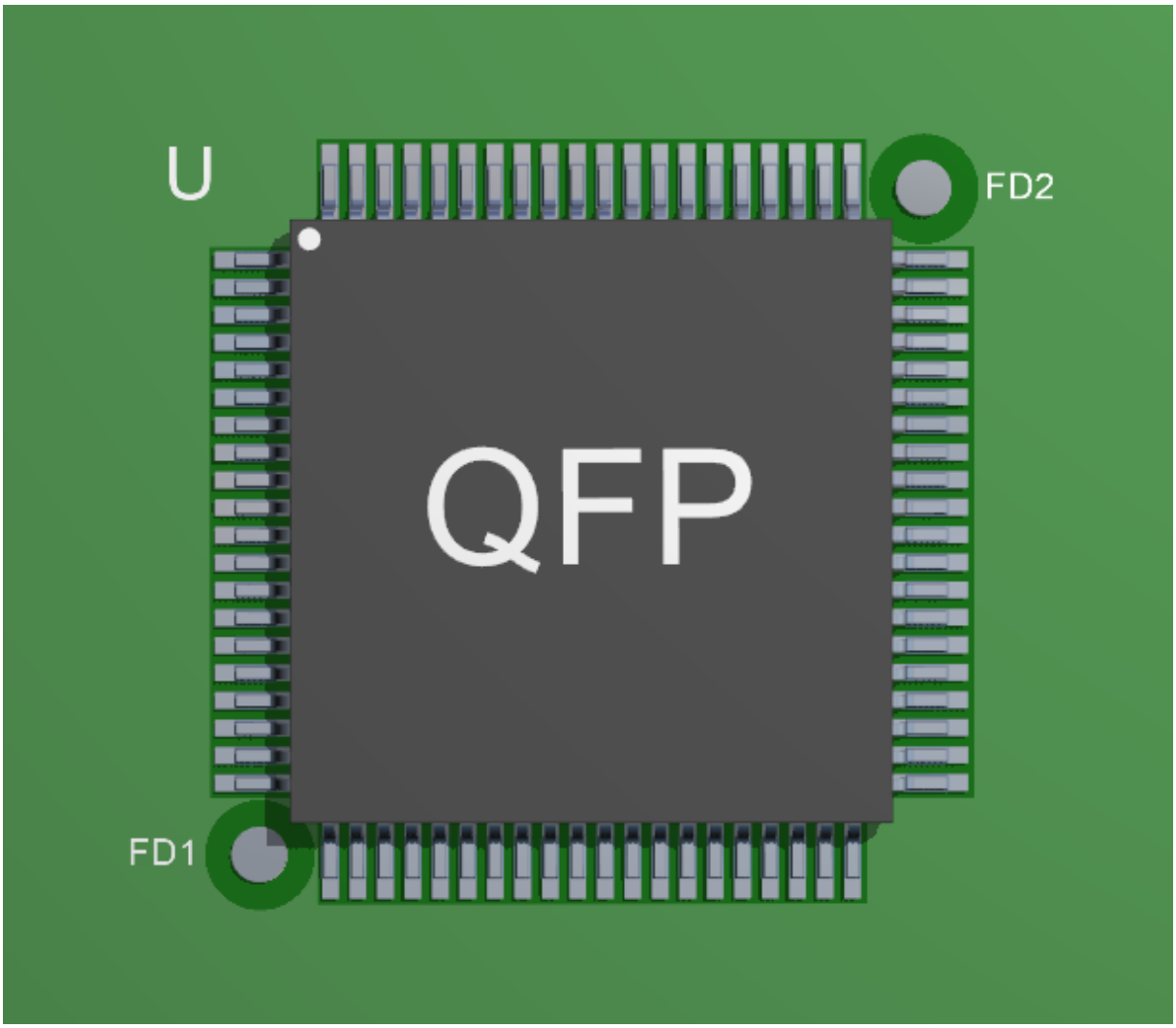
If space is limited, you may be able to share fiducial from adjacent component within the local constraint. Also, in such a space limited situation and the PCB is deemed by your manufacturer to be sufficiently small, component fiducials can be omitted and global fiducials used in their place.







QFP with 2 corner local fiducials (FD1 and FD2)

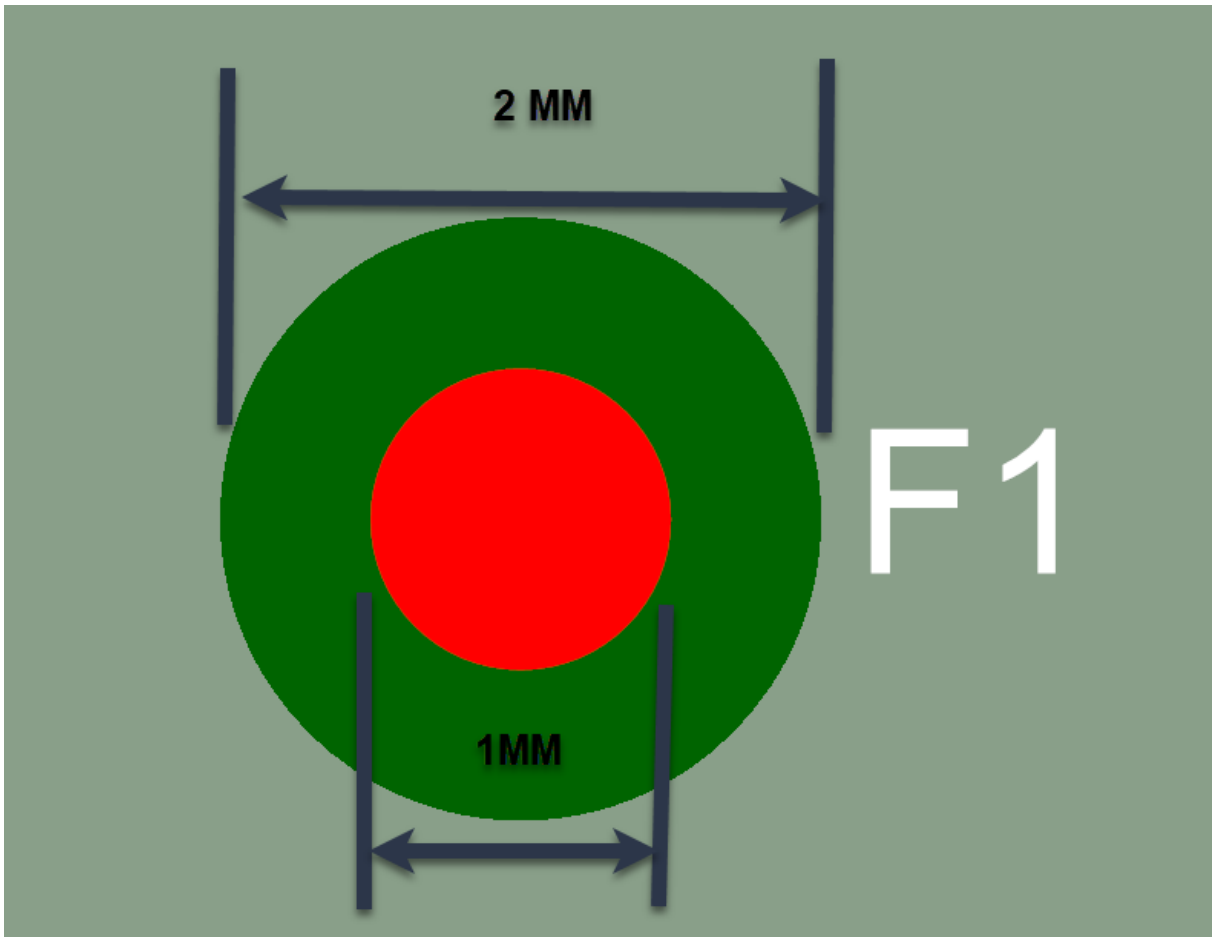


QFP with 2 corner local fiducials (FD1 and FD2)

The size and shape of fiducials

The best fiducial shape is a solid full-circle and that is what AutoTRAX DEX does. The preferred diameter of the fiducial mark is 1 mm as per the IPC – 7351B specification. You are recommended to keep the size is 1 mm although you can change it if you wish.

Fiducial marks should not vary in size on the same printer circuit board by more than 25 µm. A clear area devoid of any other circuit features are markings should exist around the fiducial mark. The minimum size of the clear area is recommended to be twice the radius of the central mark.



Recommended Fiducial Dimensions - IPC 7351B

Zonal fiducials

To facilitate the accurate placement of multiple SMD components that are not near component specific local fiducials are global fiducials you can place additional solo fiducial targets within an area of the PCB to compensate for PCB dimensional stability.

Material

the fiducial mark is copper which may or may not be tend. If you have a solder mask on the PCB it must not cover the fiducial mark of the clearance area. Note: that excessive oxidation of a fiducial marks copper may degrade the readability.

Edge clearance

the age of the fiducial should be no closer to the edge of the PCB than the sum of 4.75 mm and the minimum fiducial clearance required. If less then you may need to add a PCB board only fixture.

Contrast

the best performance is achieved when a consistent high contrast is present between the fiducial mark and the PCB base material.

The background of all fiducial marks must be the same.

1.2.6.11.26.1 Adding Fiducials

Unlike practically all other EDA programs, AutoTRAX DEX is internally fiducials as first class objects and you do not need to and make them using a variety of different graphical objects. Your life is made a lot easier and fiducials are far easier to add to your designs.

It is possible to automatically add fiducials to a PCB.

It is also possible to automatically add fiducials to footprints using the parametric part builder.

In both cases it simply involves checking a tick box.

1.2.6.11.26.2 Adding A Fiducial

You can manually add a fiducial to a Project PCB or to a Footprint land pattern.

To add a fiducial.

1. Click the Fiducial button 

in the Add / PCB menu.
2. Move the mouse cursor in the PCB/land pattern viewport and click the left mouse button to finish.

You can automatically add global footprints to a PCB using the PCB board is property panel.

Select the PCB border by clicking on it.

In the PCB border properties panel click the add fiducials checkbox to automatically add three global fiducials.

The first one at the lower left of the PCB.

Another one above the first fiducial for.

Another one horizontally to the right of the first fiducial.

The origin of the PCB will be set to the center point of the first fiducial.

There are two different ways of adding global fiducials to your PCB.

Global fiducials are used globally by all parts of the PCB and not by any specific part. Specific parts use what are called local fiducials. This naming convention is from the IPC specification.

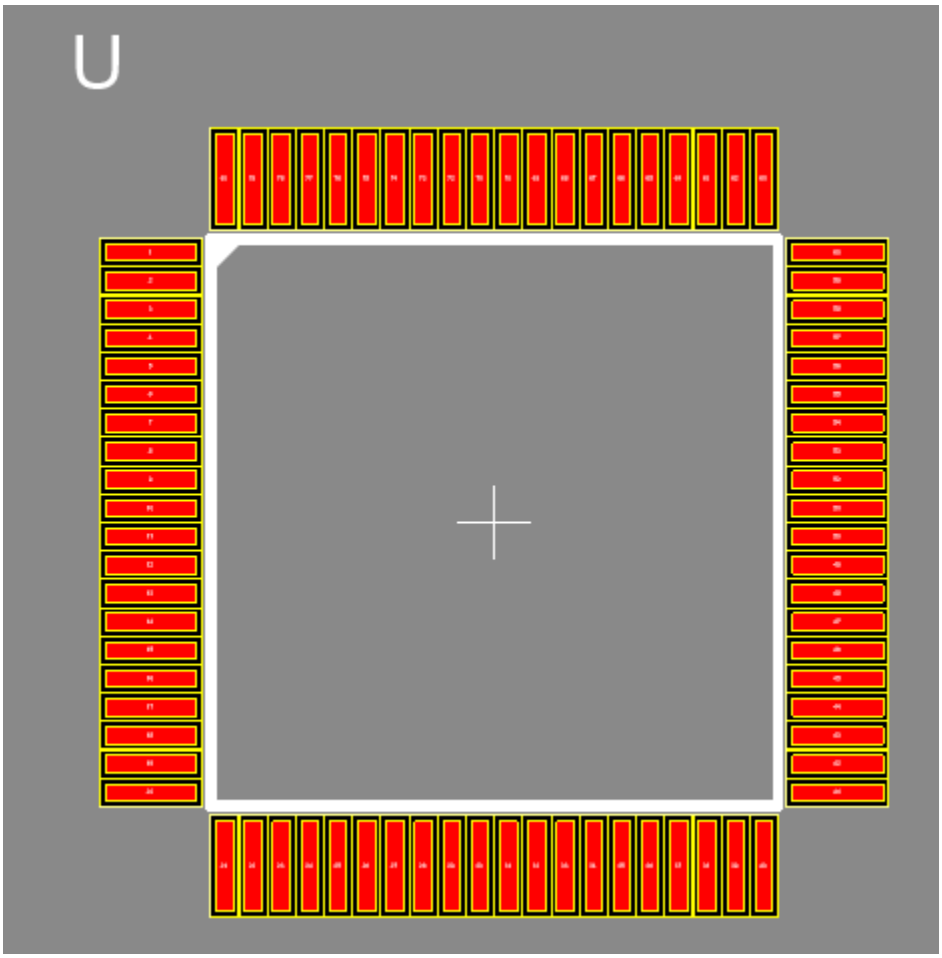
1. The first way is to use the automatic features that are built into the PCB border properties panel. This is the easiest way to add fiducials.
2. The second way is to manually place the fiducial is by first clicking on the fiducial button in the add ribbon menu then moving the mouse cursor to the location that you want the fiducial to be and then clicking the mouse button.

1.2.6.11.26.3 Adding Fiducials to a Footprint

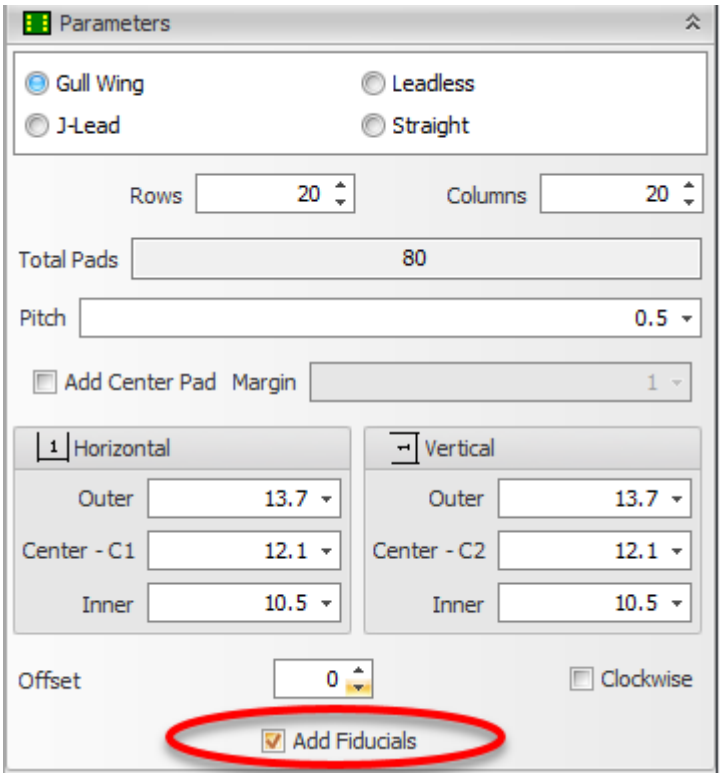
You can quickly add fiducials to a footprint by either using the Part Builder or by actually manually placing the fiducials.

Fiducials added to a footprint of called local fiducials as opposed to the global fiducials that are added to a PCB. Generally you will use these local fiducials only for locating the individual footprint and it would not be used for located other footprints.

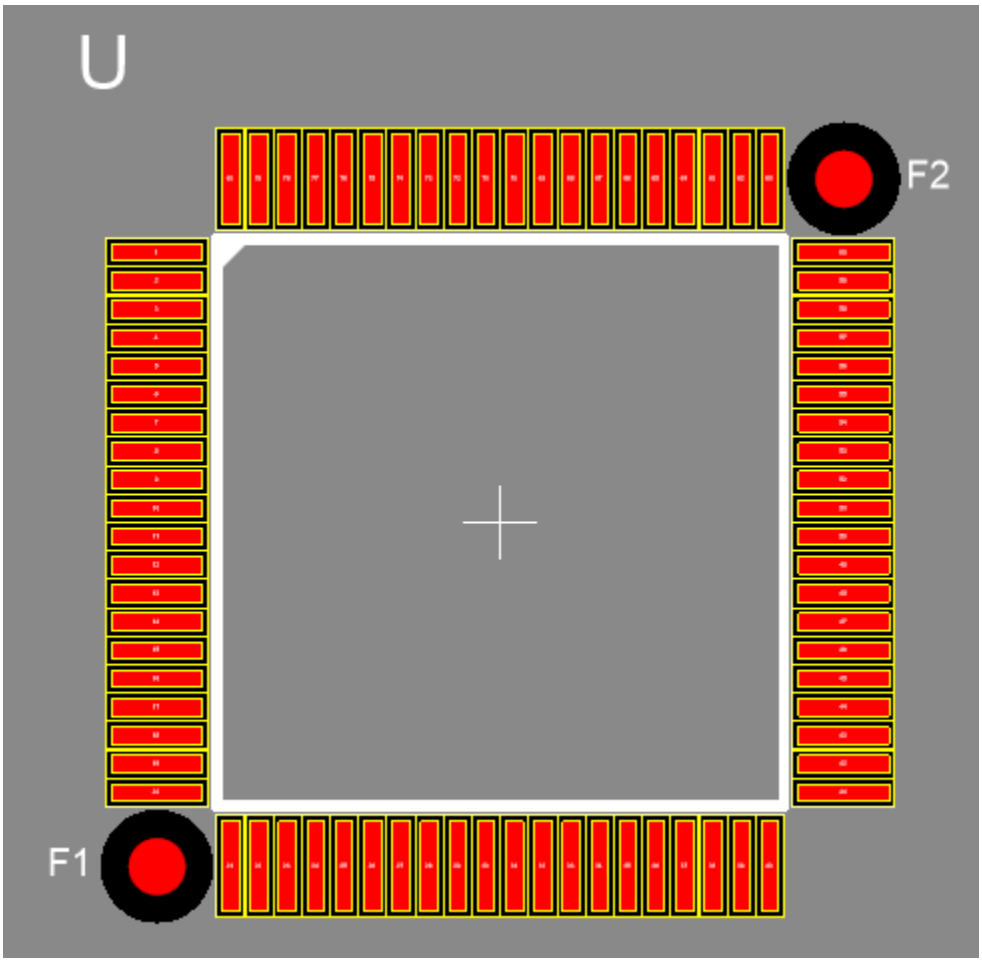
Using the Part Builder



A court QFP device with no fiducials



Check the 'Add Fiducial' checkbox to add local fiducials.



A QFP device with 2 fiducials

Adding a Fiducial Inside a Part Editor

To add a local fiducial to a footprint inside the part editor click on the fiducial button  in the add ribbon group and move the mouse cursor inside the viewport and click the left mouse button to position the fiducial.

Adding a Fiducial to a Placed Part

To add a local fiducial to a footprint inside the a project click on the fiducial button  in the add ribbon group and move the mouse cursor inside the viewport and click the left mouse button to position the fiducial.

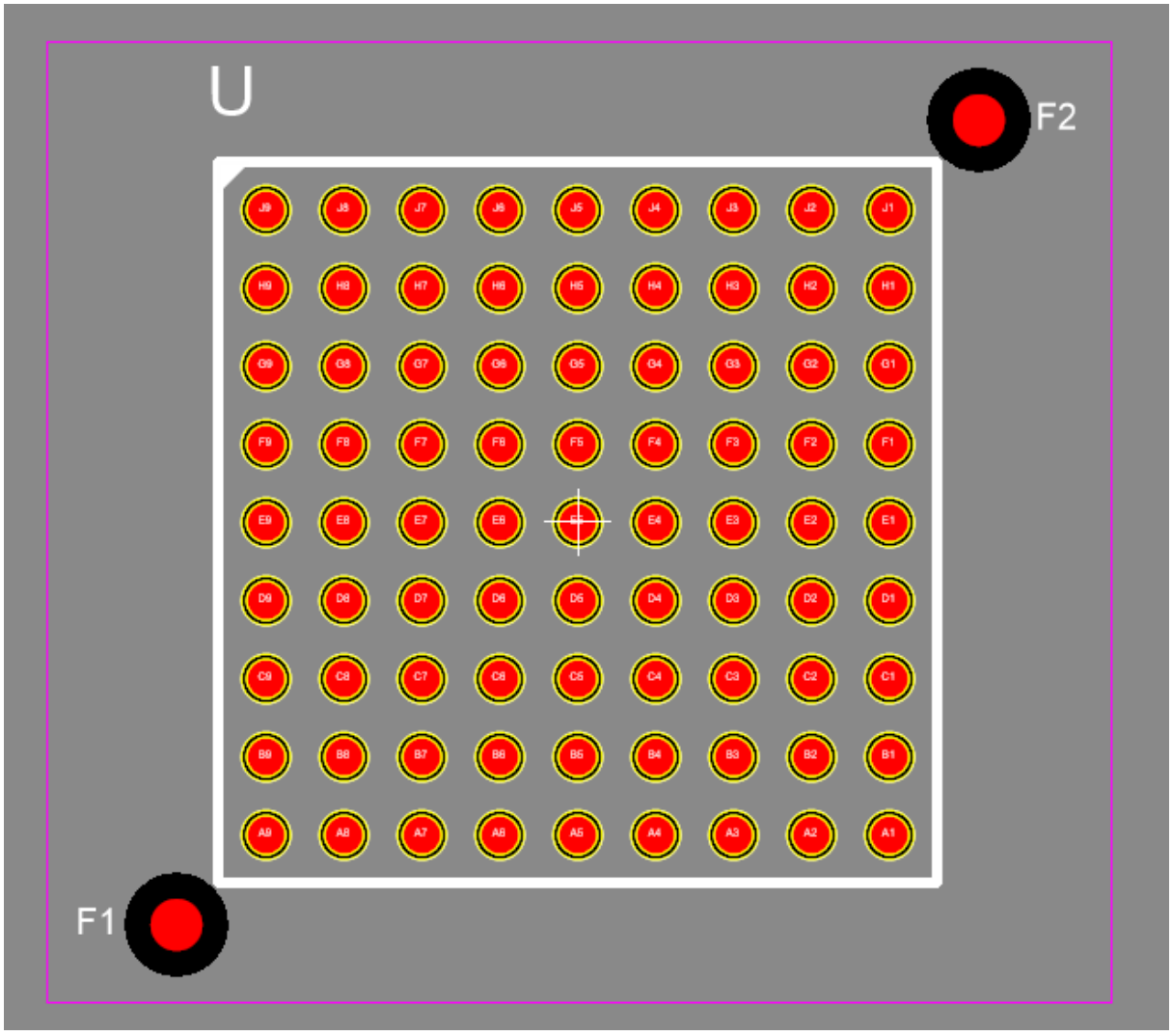
Next you need to place the fiducial inside the footprint. You can do this by first selecting the footprint and then, while holding down the shift key, selecting the

fiducial that you placed and then by clicking and selecting the group menu item. This will move the fiducial inside the footprint.

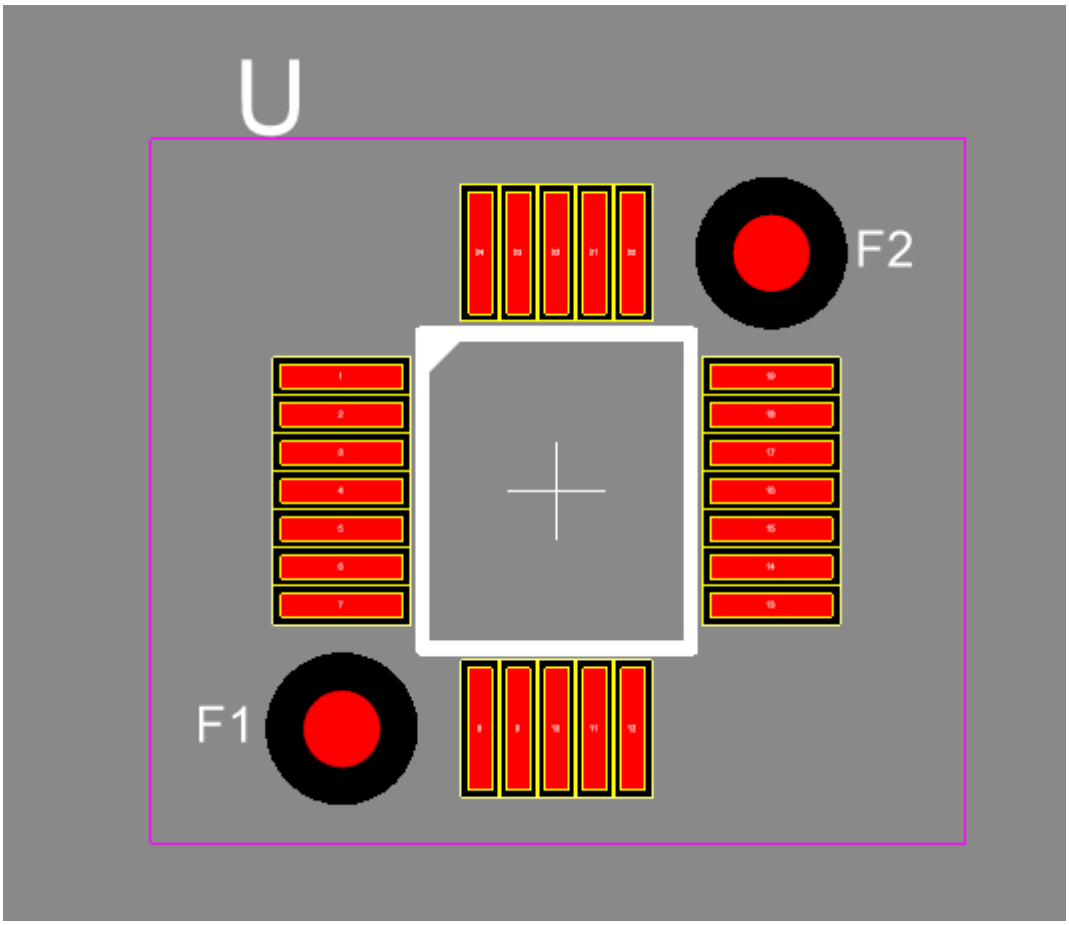
Using the Parametric Part Builder you can add local fiducials to BGAs,QFPs, SOICs and DIPs

Check the 

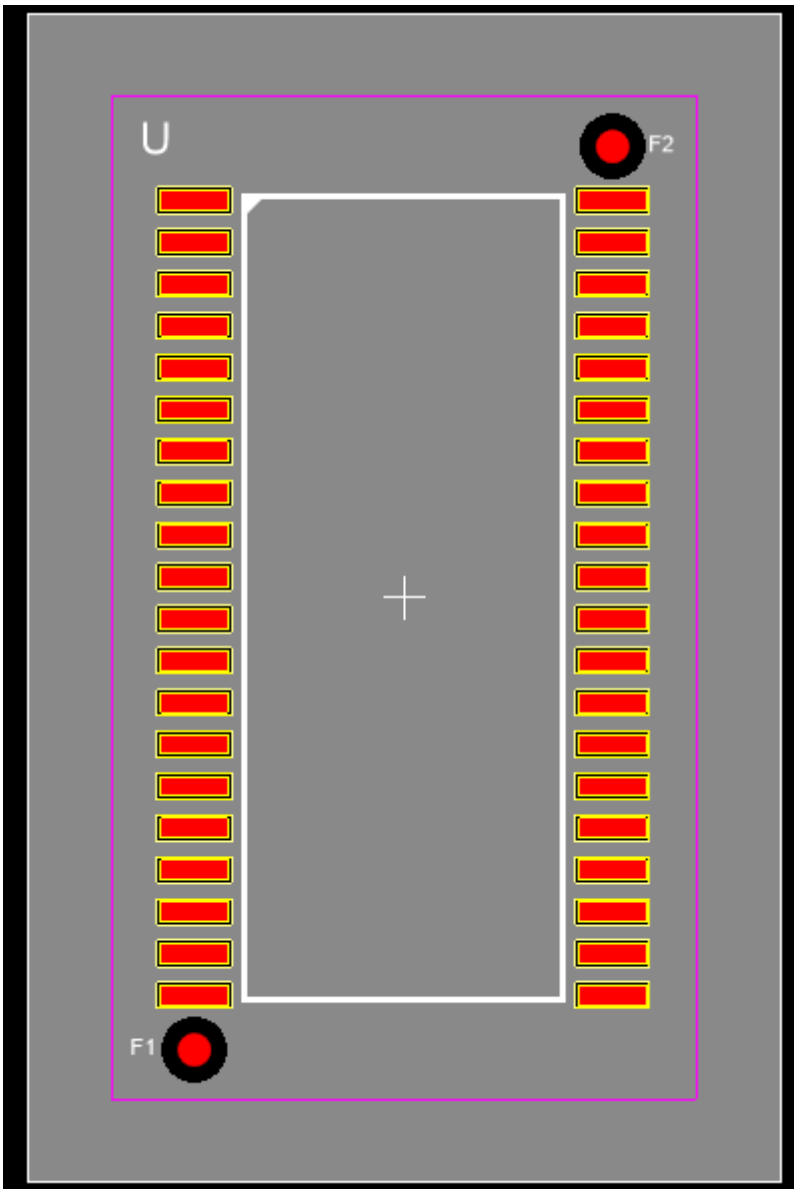
button in the Properties Panel



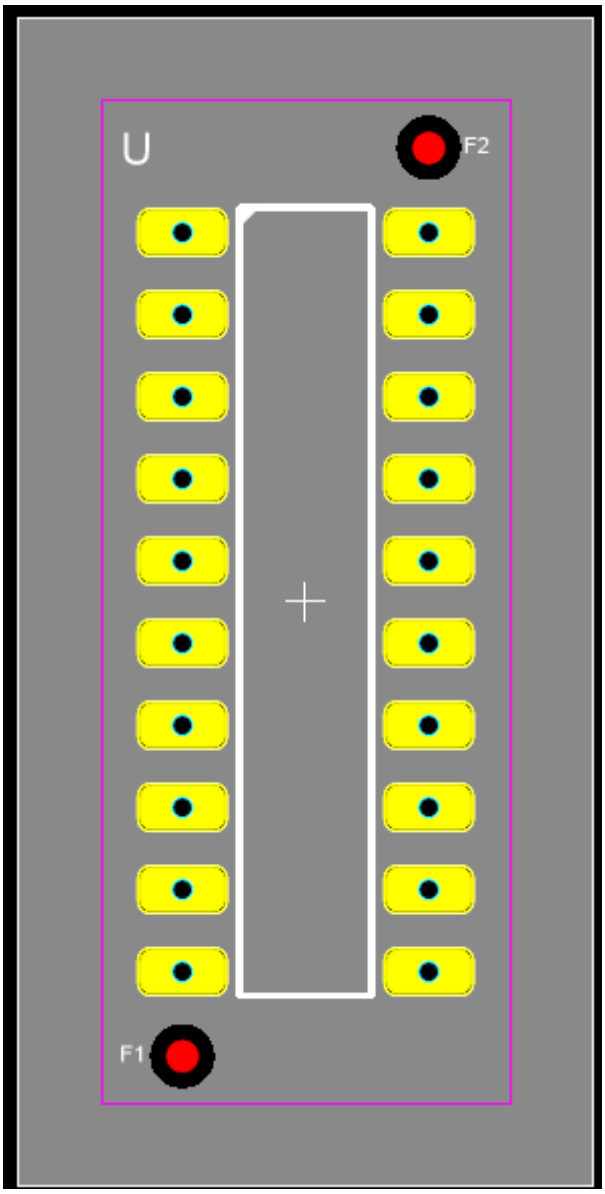
BGA



QFP

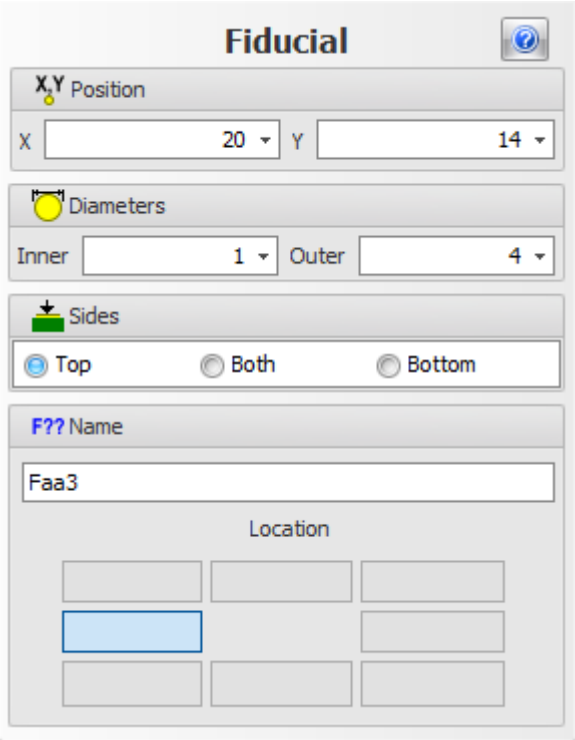


SOIC



DIP

1.2.6.11.26.4 Fiducial Properties Editor



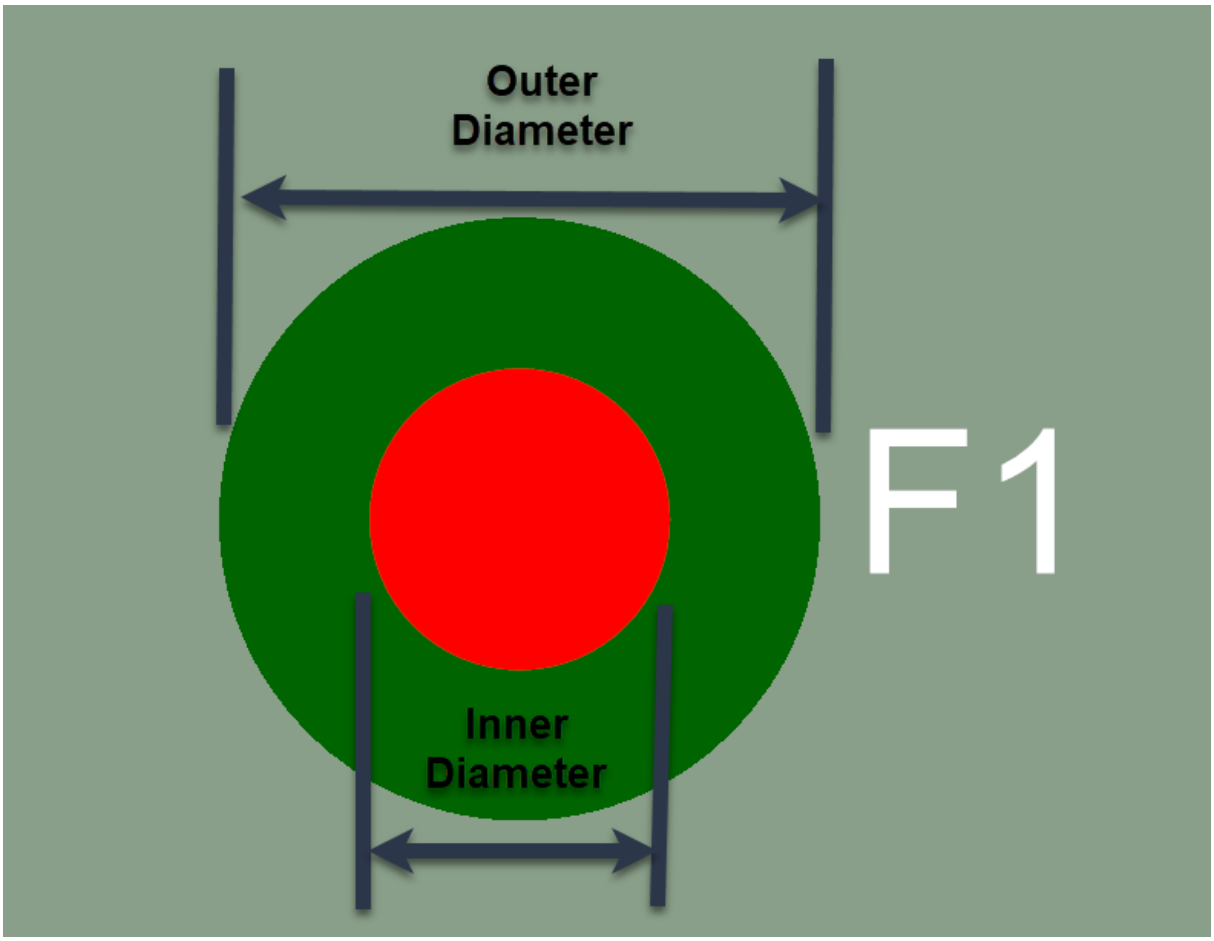
The fiducial properties editor

Clicking this button displays this help page.

Sets the x and y location of the center of the fiducial.

Sets the inner and outer diameter. The inner diameter is the diameter of the central copper circular area.

The outer diameter is the diameter of the circular clearance in the solder mask.



A typical fiducial showing both the inner and outer diameters

Sides

This sets the side **to** the PCB that the fiducial is on.

Top
This will place the fiducial on only the top side of the PCB.
Both
This will place the fiducial on both the top and the bottom of the PCB.
Bottom
This will place the fiducial on the bottom side of the PCB

F?? Name

You can optionally add a name to the fiducial and this will appear on the silkscreen layer at the side of the fiducial.

You can enter the full name of the fiducial. Obviously, if no name is entered then a name level will not appear the side of the fiducial.

Location

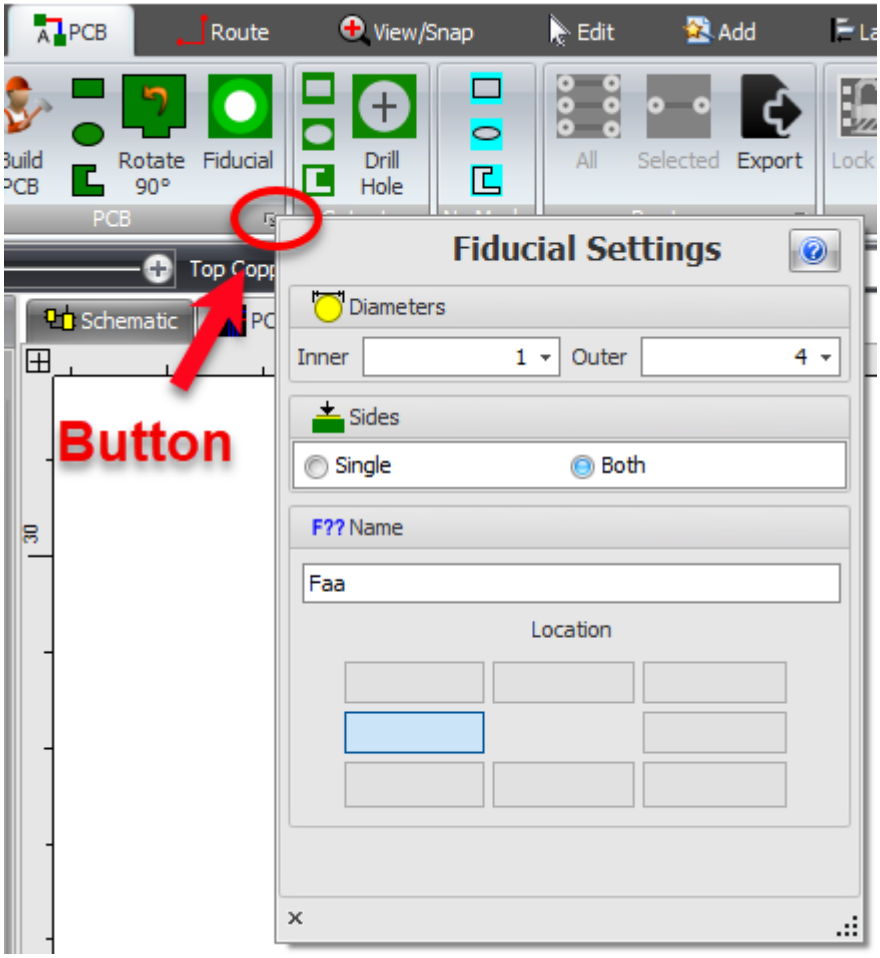
This sets the position of the fiducial label relative to the center of the fiducial. Click any of the eight rectangles to place the fiducial at the location relative to the center.

In the viewport with the fiducial selected, pressing the space key will change the location of the label and it will cycle around the locations in this control each time the space key is pressed..

1.2.6.11.26.5 Fiducial Settings Editor

The fiducial settings are the settings that will be used whenever a new fiducial is created.

You can set these default settings by clicking on the ribbon page group caption button and this will display the fiducial settings editor below.



The fiducial settings editor

Clicking this button displays this help page.

Sets the x and y location of the center of the fiducial.

Sets the inner and outer diameter. The inner diameter is the diameter of the central copper circular area.

The outer diameter is the diameter of the circular clearance in the solder mask.

This sets the side **to** the PCB that the fiducial is on.

Top
This will place the fiducial on only the top side of the PCB.
Both
This will place the fiducial on both the top and the bottom of the PCB.

Bottom
This will place the fiducial on the bottom side of the PCB

F?? Name

You can optionally set the prefix for all fiducials. For example you may this select **F** or **FD**. Whenever a prefix is added a number will be added to the end of the prefix to form the fiducial's name. If no prefix is defined then the fiducial will be simply a number

1.2.6.11.26.6 Fiducials in the Pick and Place File

Information for all fiducials is placed in the pick and place file. Example is shown below.

	A	B	C	D	E	F	G	H	I
1	Designator	Footprint	Center-X(Mil)	Center-Y(Mil)	Ref-X(Mil)	Ref-Y(Mil)	Layer	Rotation	Comment
2	Global Fiducial F1		0	2287.1					
3	Global Fiducial F2		0	0					
4	Global Fiducial F3		2938.9	0					
5	AD1		2370.8	177.5	2370.8	177.5	Top	90	
6	C1	100	680.8	1627.2	680.8	1627.2	Top	180	
7	C10	100	943.1	657.2	943.1	657.2	Top	0	
8	C11	4.7 50V	1092	431.9	1092	431.9	Top	90	
9	C12		1178.1	902.2	1178.1	902.2	Top	180	
10	C13	100	1940.8	462.5	1940.8	462.5	Top	180	
11	C2	100	680.1	1192.5	680.1	1192.5	Top	180	

An example pick and place file showing the fiducials in rows 2 to 4.

1.2.6.11.26.7 Renumbering All Fiducials

To renumber all the fiducial labels go to the tools menu and click on the **Renumber Fiducials** button 

in the utilities button group.
This will rename all the fiducials in a left-to-right and bottom to top order. So the lowest numbered fiducial would be at the bottom left the highest number will be at the top right.

1.2.6.11.27 Nets

A net connects two or more pads on a PCB. The connections are made using tracks, so a net appears as one of more tracks.

1.2.6.11.27.1 Adding Nets

To add a net to a PCB:

1. Click on the **Add→Add** Net button in the menu.
2. Move the mouse so it is over a pad, via or track and the cursor will turn to a green tick.

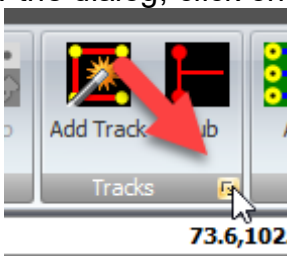
3. ***Left-click*** when you see the green tick to start a net.
4. Now move the mouse. As you move it a yellow line will be drawn from the source pad, via or track.
5. When the cursor goes over a valid end point for the net, the cursor will turn into a green tick.
6. ***Left-click*** when you see the cursor turn to green tick. The net will then optimize itself.

1.2.6.11.27.2 Editing Nets

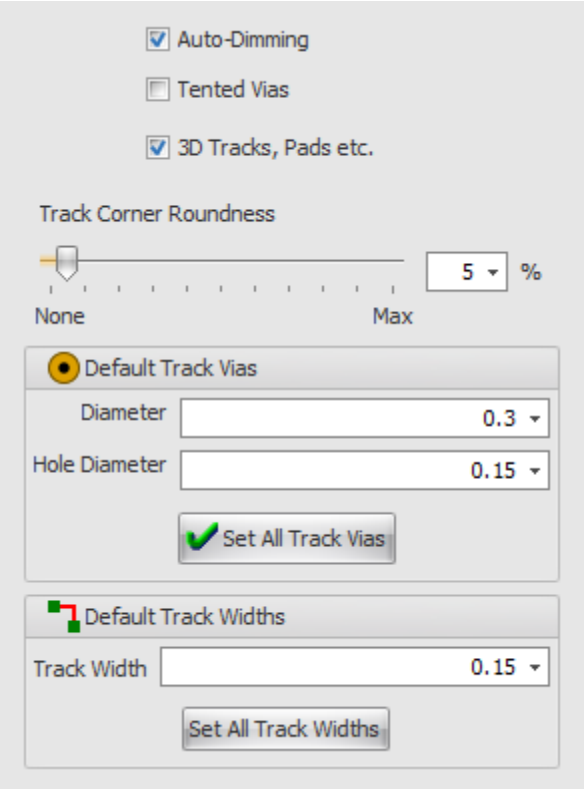
To edit a net click on one of it's tracks.

1.2.6.11.27.3 Track Settings

There are several settings that you can use for creating tracks.

To view the dialog, click on the small button at the bottom right of the Tracks button group. 

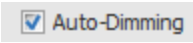
The track settings dialog is shown below.



Track Settings Popup

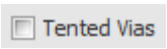
This dialog is also part of the PCB settings in the properties panel.

Auto-Dimming



Check this to have automatic dimming when manual routing. Auto dimming will dim non-selected tracks when you are routing a PCB, this makes it easier for you to do the routing.

Tented Vias



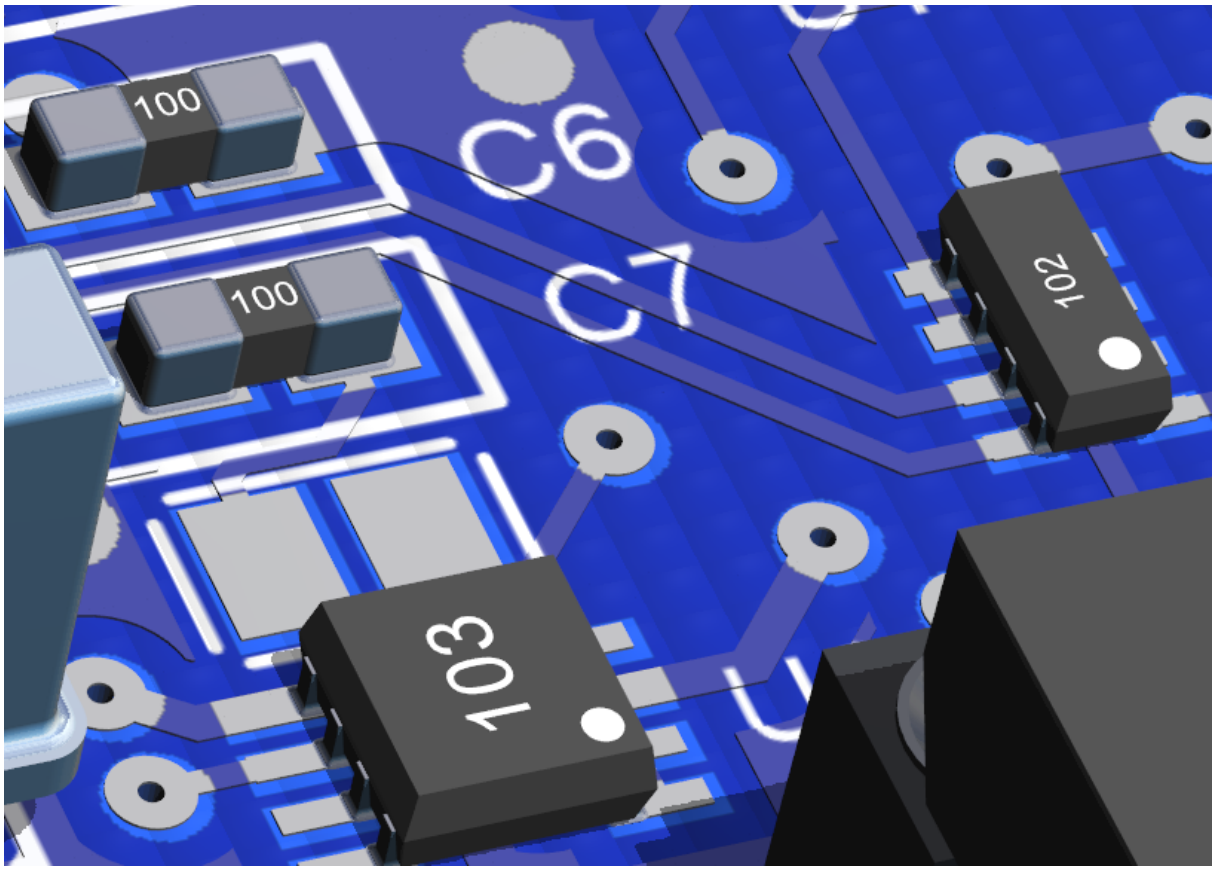
When checked all your vias will be tented.

The role of the PCB's solder-mask is to protect the copper traces from damage, oxidation, and solder bridging.

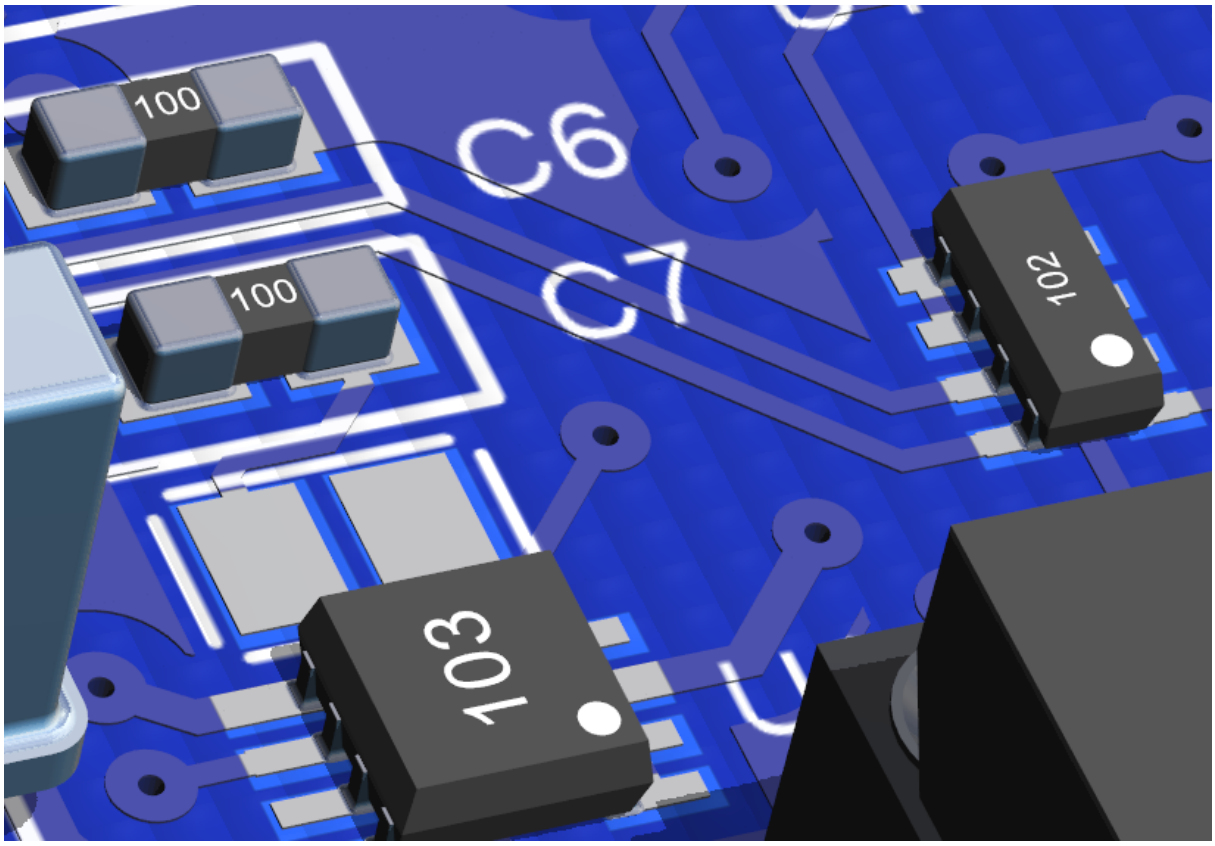
This protections also apply to the vias. Tented vias will be more resistant to physical damage and electrical shorts.

If your PCB design has through-hole components that will be wave soldered then a tented via will prevent the solder from flowing up into the via and over the other side of the PCB where it can potentially short out components.

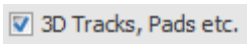
Vias that are also in close proximity to SMT pads should also be tented. This will prevent the solder paste from wicking into the via and creating a poor solder joint.



Non-Tented Vias (no solder mask on the vias)



Tented Vias (solder-mask covers the vias)

3D Tracks, Pads etc. 

Check to show true 3D tracks and pads in the 3D viewport. If not checked then tracks and pads are shown as images on the PCB; this speeds up drawing and is useful for slower machine. For the best quality use 3D tracks and pads.

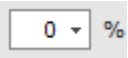
Track Corner Rounding

You can optionally round the corners of all you PCB tracks.

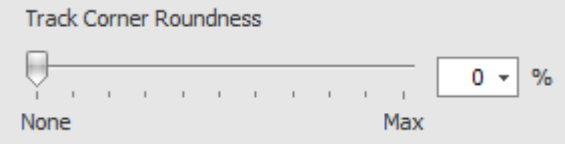
Automatic Rounded Track Corners

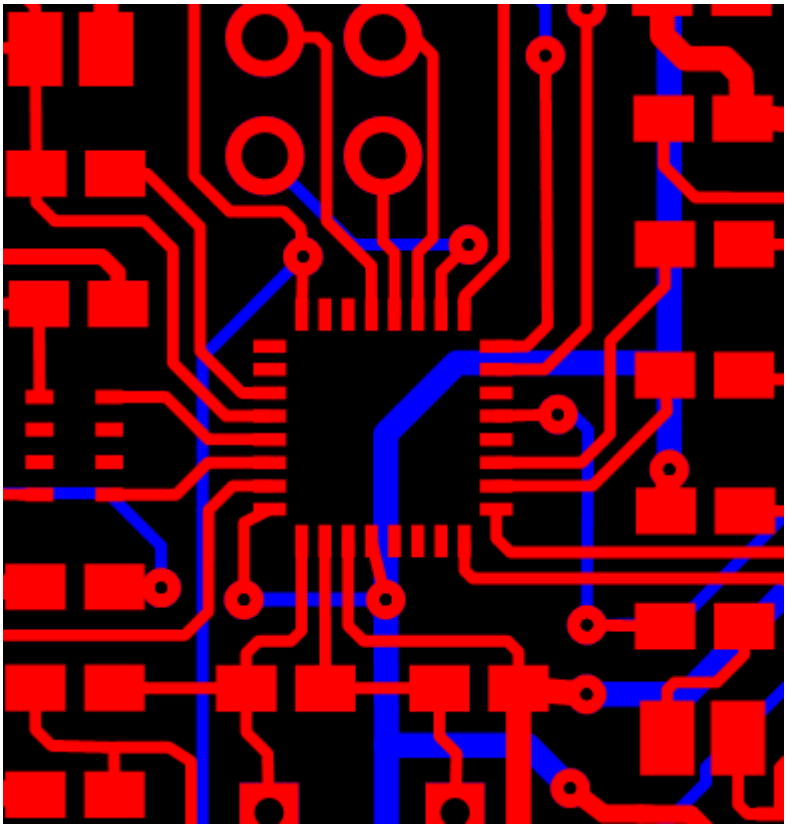
Drag the slider to the right ti increase rounding.

You can also enter the rounding percentage

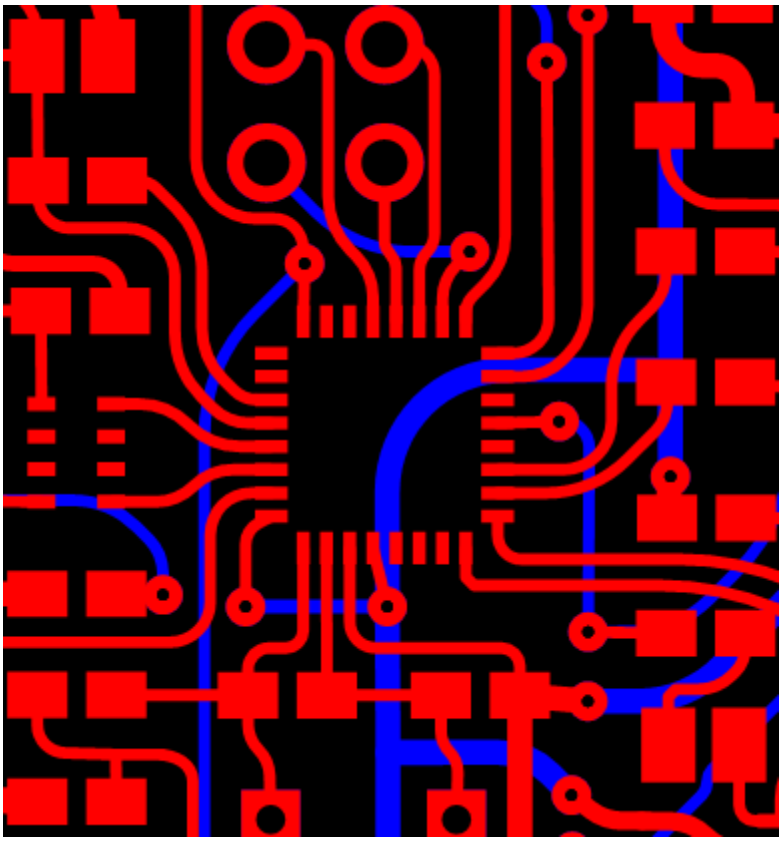


No rounding





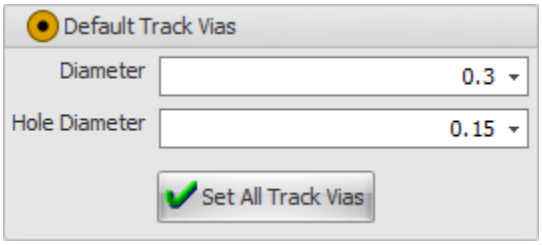
100% Rounding



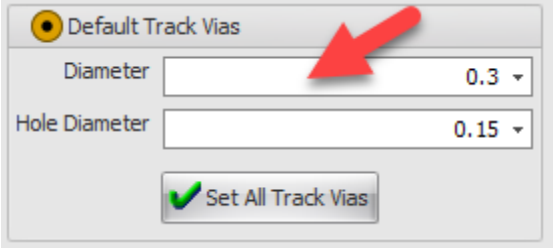
Default Track Vias

You can set the default track via diameter and the default track via hole diameter.

You can also set all track via sizes from the default sizes.

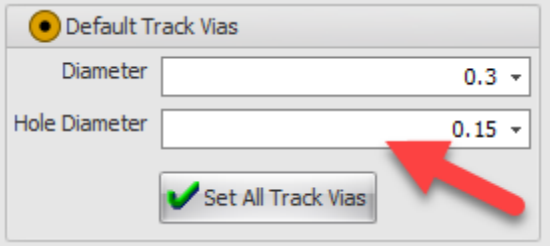


Default Track Via Diameter



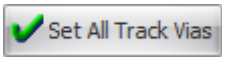
This sets the default via diameter. It is used to set the via diameter for newly created PCB nets. You can override this by editing either the PCB net of the associated schematic node using their respective properties panels.

Default Track Via Hole Diameter



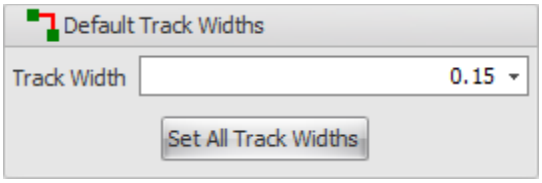
This sets the default via hole diameter. It is used to set the via hole diameter for newly created PCB nets. You can override this by editing either the PCB net of the associated schematic node using their respective properties panels.

Setting All Track Via Diameters and Track Via Hole Diameters

Click 

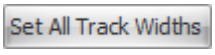
to set all track via diameters and hole diameters from the default via diameter and hole diameter. This will also set all net and node all track via diameters and hole diameters.

Default Track Width



You can set the default track width. 

It is used to set the via diameter for newly created PCB nets.

Click 

to set all track widths from the default track width.

1.2.6.11.28 Tracks

A track is a section of a net that is on a single layer. It starts and ends on a pad, a via or a track junction. A track junction is where 3 or more tracks that are all on the same layer meet.

A track consists of one or more straight segments.

1.2.6.11.28.1 Changing a track segments layer index

To change the layer for a track segment, first select it by ***left-clicking*** it and then either press either the ***Space*** key or press a ***number key***.

Obviously it is not possible to change the track segment's layer for single sided PCB.

Pressing the ***space key*** will send the current track segment backwards in the PCB layers. However if the track segment is on the bottom layer, then it will be cycled around to the top layer. For a 2-sided board pressing the space key appears to swap a track segment to the other side of the board.

When a layer segment is swapped, vias and/or track junctions will be added and/or removed as required.

Pressing a ***number key*** will move the selected track segment to the layer defined by the number entered or to the bottom layer if the number is greater than the number of electrical layers on the PCB.

e,g, pressing '***1***' moves the selected track segment to the top layer while pressing '***2'*** moved it to the second layer, in the case of a 2-layered PCB, to the bottom of the PCB.

Pressing ***'t'*** or ***'T'*** moves the track segment to the top layer

Pressing **'*b'*** or ***"B'*** moves the track segment to the top layer.

You can convert a track segment to a jumper on the top or the bottom side of the board. pressing the ***'H'*** to create a jumper on the bottom side of the board or the ***'J'*** key to create a jumper on the top side of the board.

1.2.6.11.28.2 Jumpers

A jumper is a short length of conductor just to jump over a routing obstacle on a PCB. There are uses, *rarely*, when it is impossible to finish a route due to an already routed track. A 3D model of the jumper is automatically created. Insulation is also added to the jumper if the jumper is long enough.

Adding a Jumper When Manual Routing

When you are manually routing a track:

1. Press the ***'J'*** key to start a jumper. AutoTRAX DEX will automatically add a via and start a jumper on the top side of the board.

2. Drag the mouse and the jumper's end point will follow the cursor. Press the ***'J'*** or ***Space*** key to end the jumper. AutoTRAX DEX will add another via to complete the jumper and return you to the layer you were on before you started adding the jumper.

Instead of pressing the ***'J'*** key you can start the jumper on the bottom side of the board by pressing the ***'H'*** key.

Editing a Track

To convert an existing track segment to a jumper:

1. Select the track segment by clicking on it.

2. Press the ***'J'*** key to convert it to a jumper on the top side of the board or press the ***'H'*** key to convert it to a jumper on the bottom side of the board.

Removing a Jumper

1. Select the track segment by clicking on it.

2. Press the ***Space*** bar to convert a jumper to a regular copper track.

Swapping the Jumpers Side

1. Select the jumper by clicking on it.

2. Press the ***'J'*** key to convert it to a jumper on the top side of the board. Press the ***'H'*** key to convert it to a jumper on the bottom side of the board.

1.2.6.11.29 Routing

There are two methods whereby you can route the tracks on your PCB.

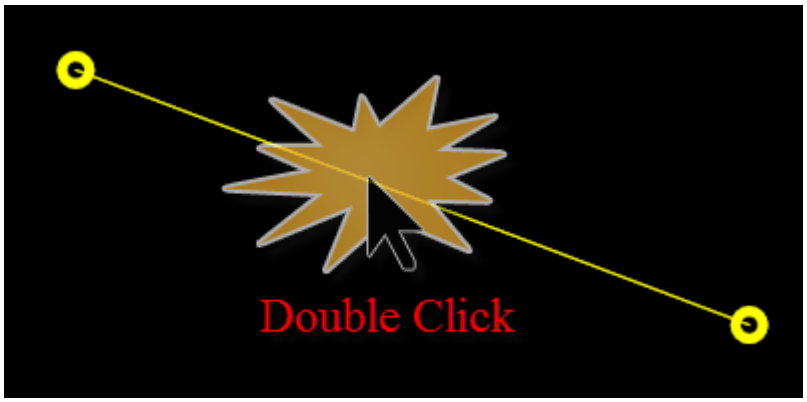
- Manual Routing
- Automatic Routing

1.2.6.11.29.1 Manual Routing

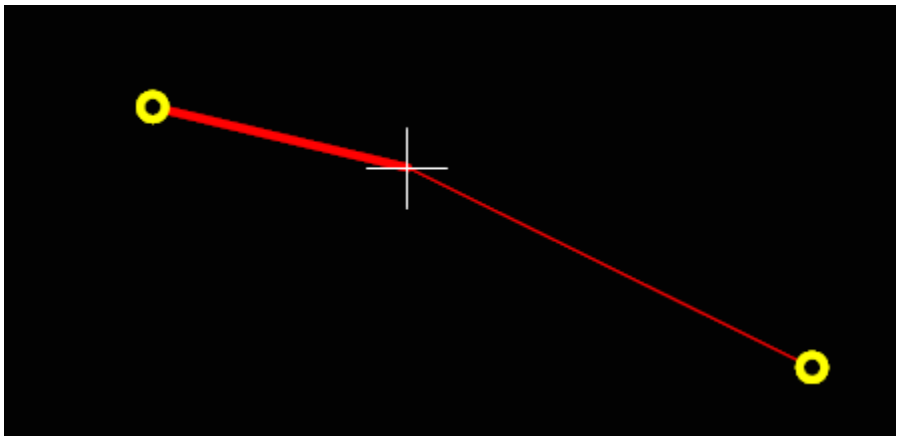
To manually route your PCB first ***double-click*** the PCB track.

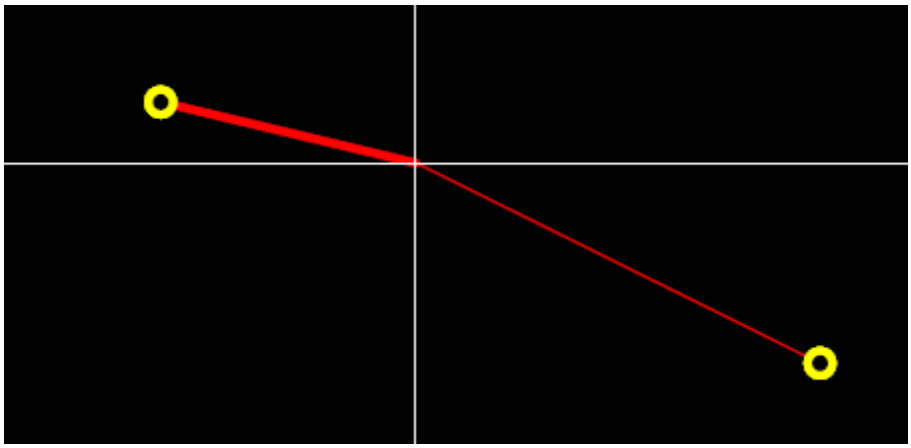
1.2.6.11.29.2 Manually Routing a Track

To start manually routing a track double click on any segment. Double click again or press the ESC key to end manual routing. The routing will automatically end when you reach the target (see below).

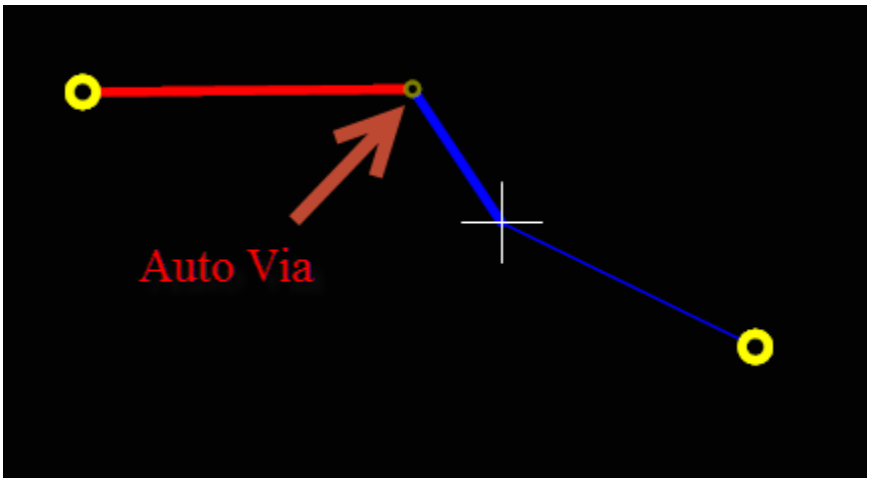


Double Click to start manual routing



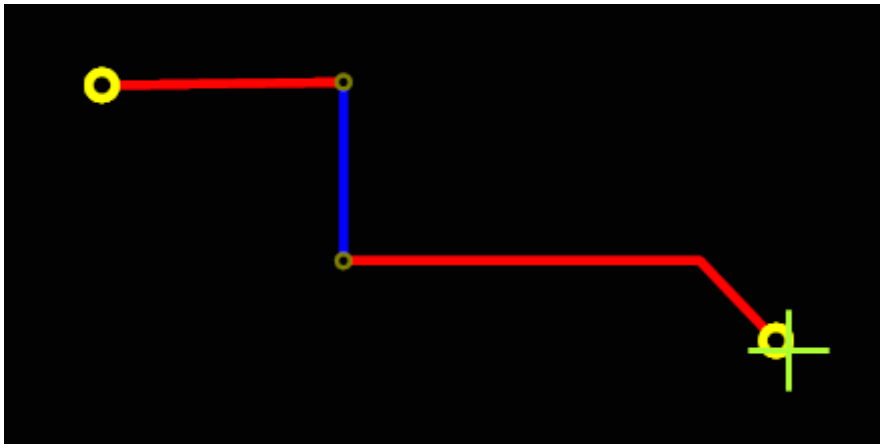


Routing starts from nearest vertex (pad/via/junction or corner). Normal cursor (left) Full cross cursor (right)



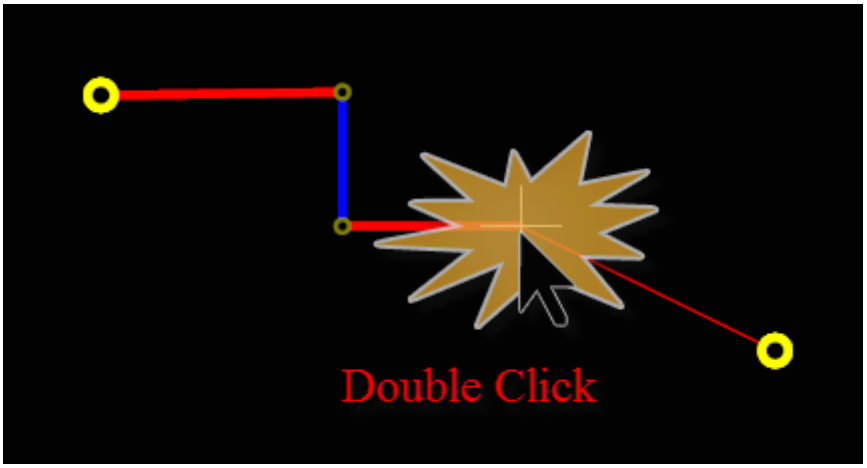
Pressing Space key automatically adds a via

(see more commands below)



Cross cursor turns green to indicate end.

Click to end the route or



Double click or press ESC to end partial routing

Keyboard Shortcuts

KEY	ACTION
Backspace	Undo the last manual route, e.g. undo a track layer change.
F12	Redo the last undone manual route, e.g. redo a track layer change.
Escape	End the manual routing.
Space	Change to the next layer and automatically add/remove vias.
+	Change up to the next layer and automatically add/remove vias, this will wrap round to the bottom layer if the current track layer if the top layer.
-	Move down one layer and automatically add/remove vias, this will wrap round to the top layer if the current track layer if the bottom layer.
J	Start or end a jumper
T	Change to the top layer, adding/removing vias if necessary
B	Change to the bottom layer, adding/removing vias if necessary
S	Toggle snap to grid. Turns it on/off

1.2.6.11.29.3 Electra

Electra is a Shape based Auto-routing powered by a multi-pass cost-based conflict reduction algorithm to find an optimal routing solution adapting to the natural flow of the nets while following advanced DFM and timing constraints.

Running Electra

Lean about running Electra.

The Electra View

Lean about the Electra viewport.

Initializing-Electra

Find out how to authorize Electra.

Using Electra Interactively

Lean how to use Electra interactively with AutoTRAX DEX.

Electra Router Settings

This details all settings for using Electra.

The Electra DO Commands

Here you will discover how to set the Electra route commands.

Importing Electra Route File

Lean about importing an Electra route file.

Route Report

Viewing the route report.

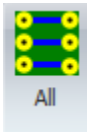
Differential pair routing

Electra can route differential pairs.

Length Constrained Auto-routing

Find out how to restrain and match track lengths.

View website...

To run the Electra auto-router click on the 

button in the PCB->Route menu.

You need to have the Use Electra option selected in the Router Settings.

You can set the Electra options using the Electra Router Settings

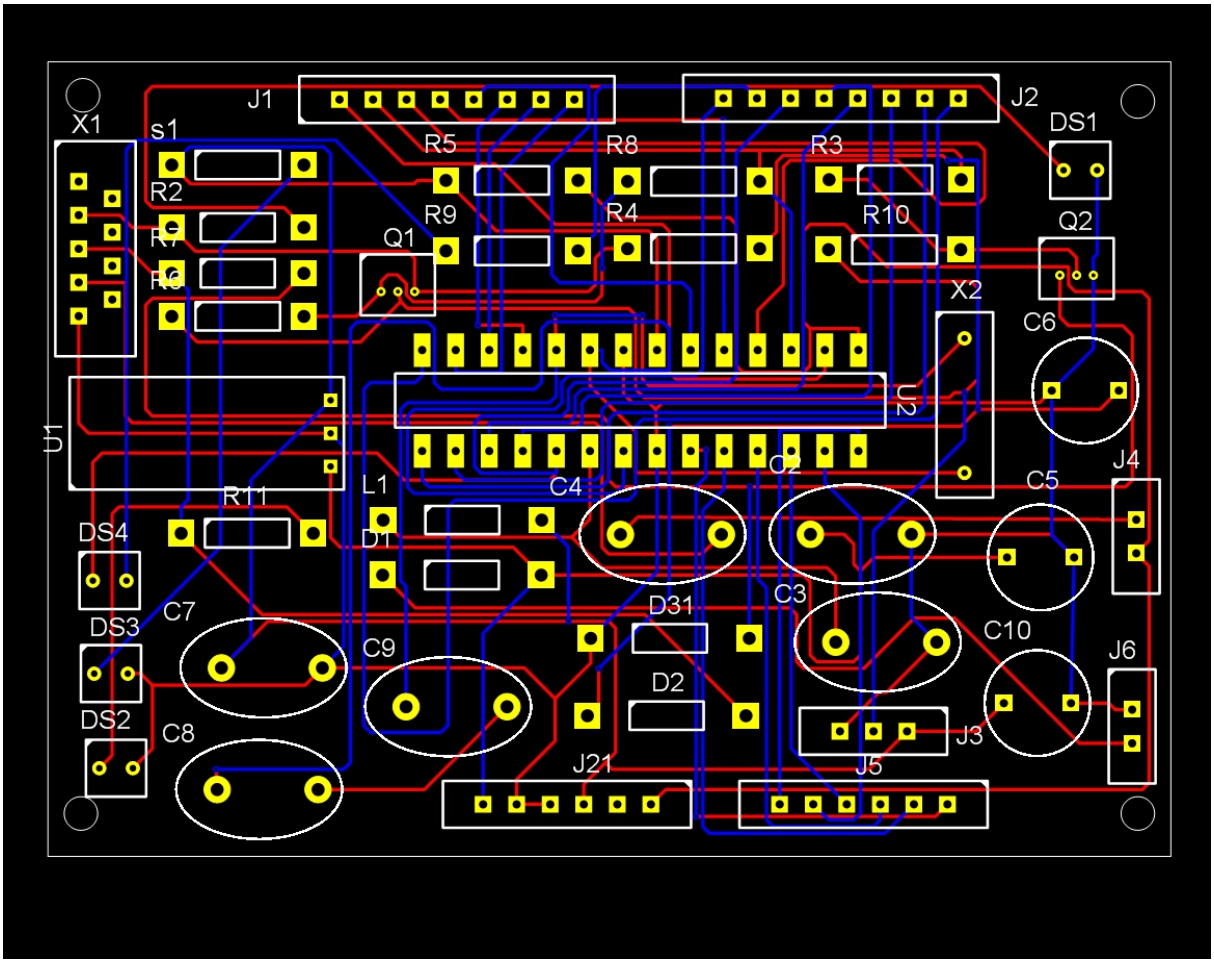
ELECTRA is a third party shape based PCB router that works well with AutoTRAX DEX.

It is a new generation of Shape-Based Auto routing software for PC boards. ELECTRA uses a multi-pass, cost-based conflict reduction algorithm to find a routing solution adapting to the natural flow of the nets. Adaptive routing algorithm is the only proven approach to reach a high completion rate. ELECTRA gives immediate feedback on the routing progress and conflict reduction rate.

AutoTRAX DEX, by default, does not come with an ELECTRA license. You will need to purchase a separate ELECTRA license. **Find out more...**

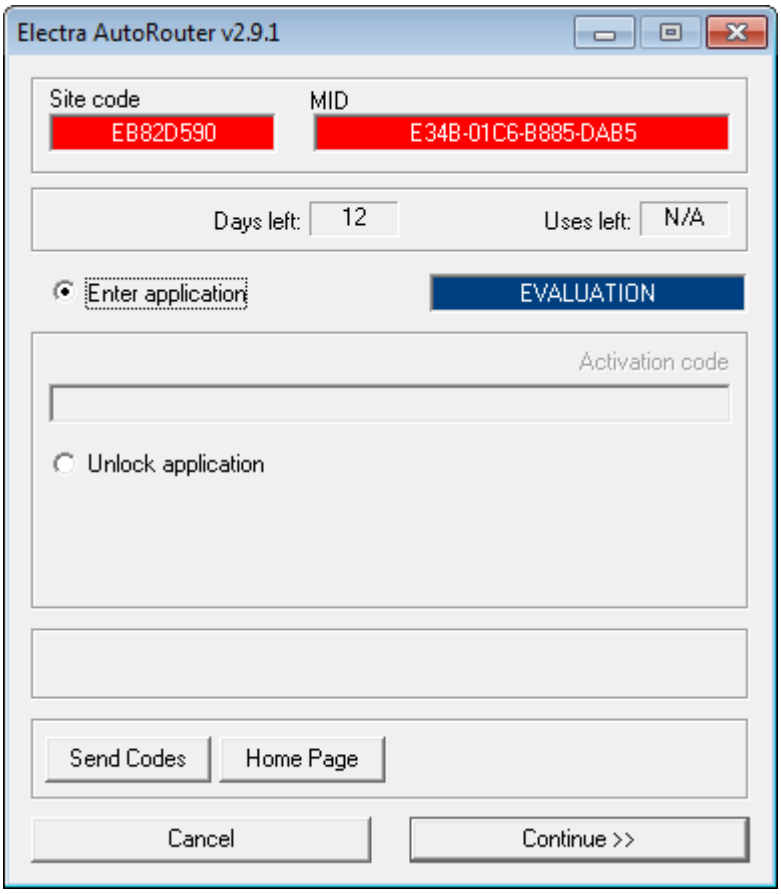
ELECTRA uses industry standard format by reading design file (DSN). Routing results are saved into standard route file format (RTE) or session file (SES).

Click to View the DSN Specification



Typical board routed with ELECTRA

When you first run Electra you have a 15 grace period before you have to buy it. It is the full version during this period and you will be prompted with the following dialog:

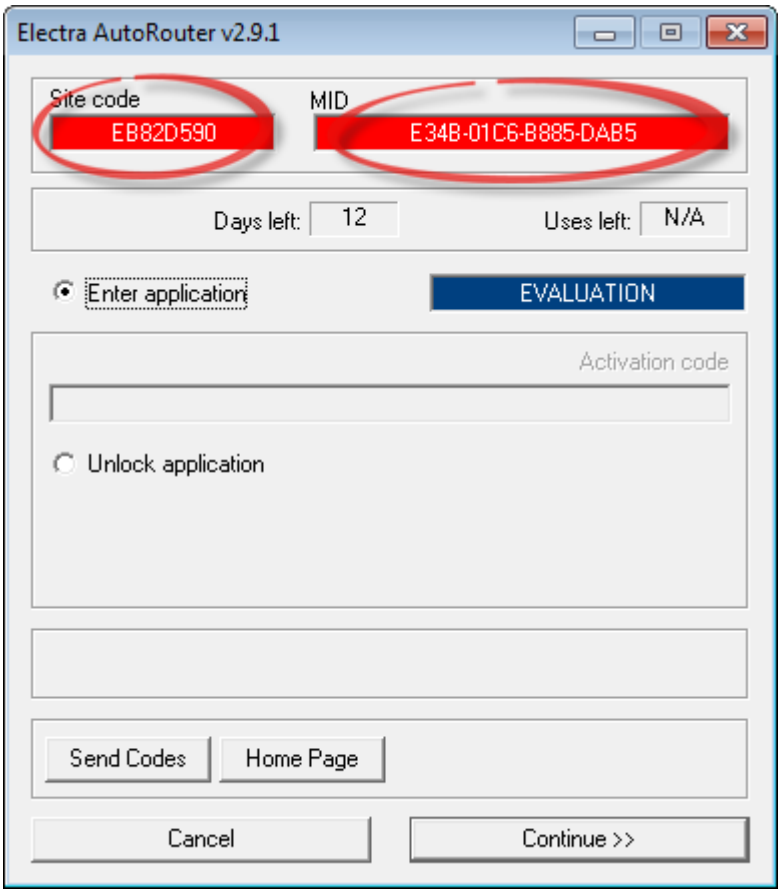


Click the  button to route your board.

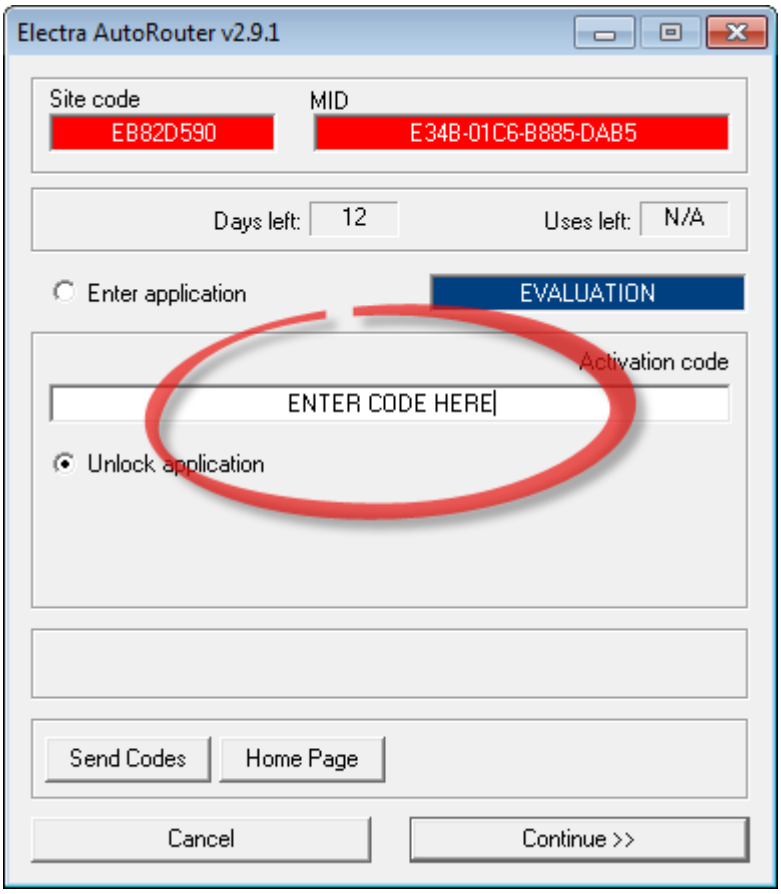
After the grace period you will no longer be able to run Electra for free.

If you wish to purchase Electra please go tohttps://konekt.com/wp/

After purchasing Electra you will be asked for the Site code and MID for Electra. These are shown below:



The actual values for your machine will be different from the values shown above. Once you have sent the details to us you will receive an activation code. To enter it click the Unlock application button and enter the code below:



Differential signaling is a method for electrically transmitting information using two complementary signals. The technique sends the same electrical signal as a differential pair of signals, each in its own conductor. The pair of conductors can be wires (typically twisted together) or traces on a circuit board. The receiving circuit responds to the electrical difference between the two signals, rather than the difference between a single wire and ground. The opposite technique is called single-ended signaling. Differential pairs are usually found on printed circuit boards, in twisted-pair and ribbon cables, and in connectors.

Differential signaling is often used in computers to reduce electromagnetic interference, because complete screening is not possible with micro-strips and chips in computers, due to geometric constraints and the fact that screening does not work at DC. If a DC power supply line and a low-voltage signal line share the same ground, the power current returning through the ground can induce a significant voltage in it. A low-resistance ground reduces this problem to some extent. A balanced pair of micro-strip lines is a convenient solution, because it does not need an additional PCB layer, as a strip line does. Because each line causes a matching image current in the ground plane, which is required anyway for supplying power, the pair looks like four lines and therefore has a shorter crosstalk distance than a

simple isolated pair. In fact, it behaves as well as a twisted pair. Low crosstalk is important when many lines are packed into a small space, as on a typical PCB.

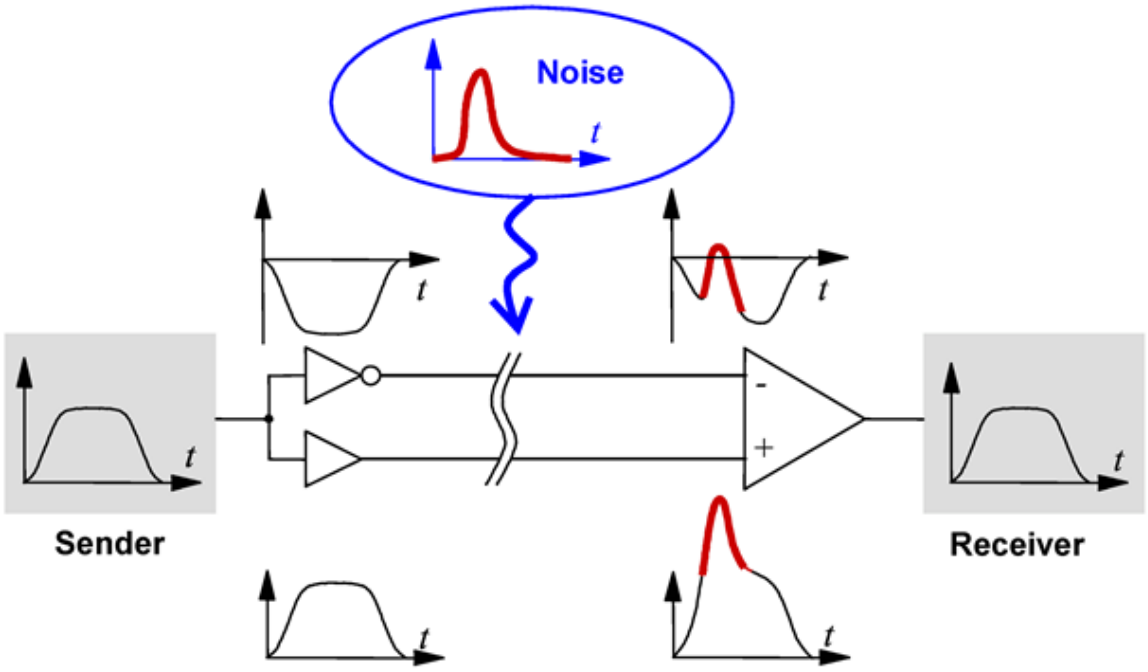
In single-ended signaling, the transmitter generates a single voltage that the receiver compares with a fixed reference voltage, both relative to a common ground connection shared by both ends. In many instances single-ended designs are not feasible. Another difficulty is the electromagnetic interference that can be generated by a single-ended signaling system that attempts to operate at high speed.

Provided that the source and receiver impedances in the differential signaling circuit are equal, external electromagnetic interference tends to affect both conductors identically. Since the receiving circuit only detects the difference between the wires, the technique resists electromagnetic noise compared to one conductor with an unpaired reference (ground). The technique works for both analog signaling, as in balanced audio—and digital signaling, as in RS-422, RS-485, Ethernet over twisted pair, PCI Express, DisplayPort, HDMI, and USB.

A differential signaling system is where a signal is transmitted down a pair of tightly coupled tracks with one of these carrying the signal and the other carrying an equal but opposite image of the signal. Differential signaling was developed to resolve situations where the logic reference ground of the signal source could not be well connected to the logic reference ground of the load. Differential signaling is inherently immune to common mode electrical noise, the most common interference artifact present in an electronic product. Differential signaling also minimizes electromagnetic interference (EMI) generated from the signal on the PCB.

The electronics industry, particularly in portable and mobile devices, continually strives to lower supply voltage to save power and reduce emitted electromagnetic radiation. A low supply voltage, however, reduces noise immunity. Differential signaling helps to reduce these problems because, for a given supply voltage, it provides twice the noise immunity of a single-ended system.

To see why, consider a single-ended digital system with supply voltage Vs. The high logic level is Vs, and the low logic level is 0 V. The difference between the two levels is therefore Vs-0 = Vs.



Now consider a differential system with the same supply voltage. The voltage difference in the high state, where one wire is at Vs , and the other at 0 V, is Vs-0 = Vs. The voltage difference in the low state, where the voltages on the wires are exchanged, is 0-Vs = -Vs The difference between high and low logic levels is therefore Vs-(-Vs) = 2Vs,. This is twice the difference of the single-ended system. If the voltage noise on one wire is uncorrelated to the noise on the other one, it takes twice as much noise to cause an error with the differential system as with the single-ended system. In other words, differential signaling doubles the noise immunity.

The technique minimizes electronic crosstalk and electromagnetic interference, both noise emission and noise acceptance, and can achieve a constant or known characteristic impedance, allowing impedance matching techniques important in a high-speed signal transmission line or high quality balanced line and balanced circuit audio signal path.

Resistance to electromagnetic interference

This advantage is not directly due to differential signaling itself, but to the common practice of transmitting differential signals on balanced lines. Single-ended signals are still resistant to interference if the lines are balanced and terminated by a differential amplifier.

Electra

You create differential pairs in the schematic, not in the PCB. The PCB nets will be automatically used to differential pair status.

Differential pair routing process starts by identifying gather points at the proximity of the fan out vias. The main part of the connections are then auto-routed between the gather points. The auto router applies the same multipass conflict reduction strategy to the differential pair nets and length constrained nets in order to reach the highest completion rate.

No T-junction or vias are used during routing of the differential pairs.

45 degree routing style can be predefined for the differential pair routing.

Differential pairs are treated as signal nets and take part into the adaptive auto-routing strategy with conflict resolution.

For min length and match length constraint, the router meanders routed wire to increase length by following an accordion pattern.

Includes the ability to define Signal Path containing connections within a net of a set of connections of different nets.

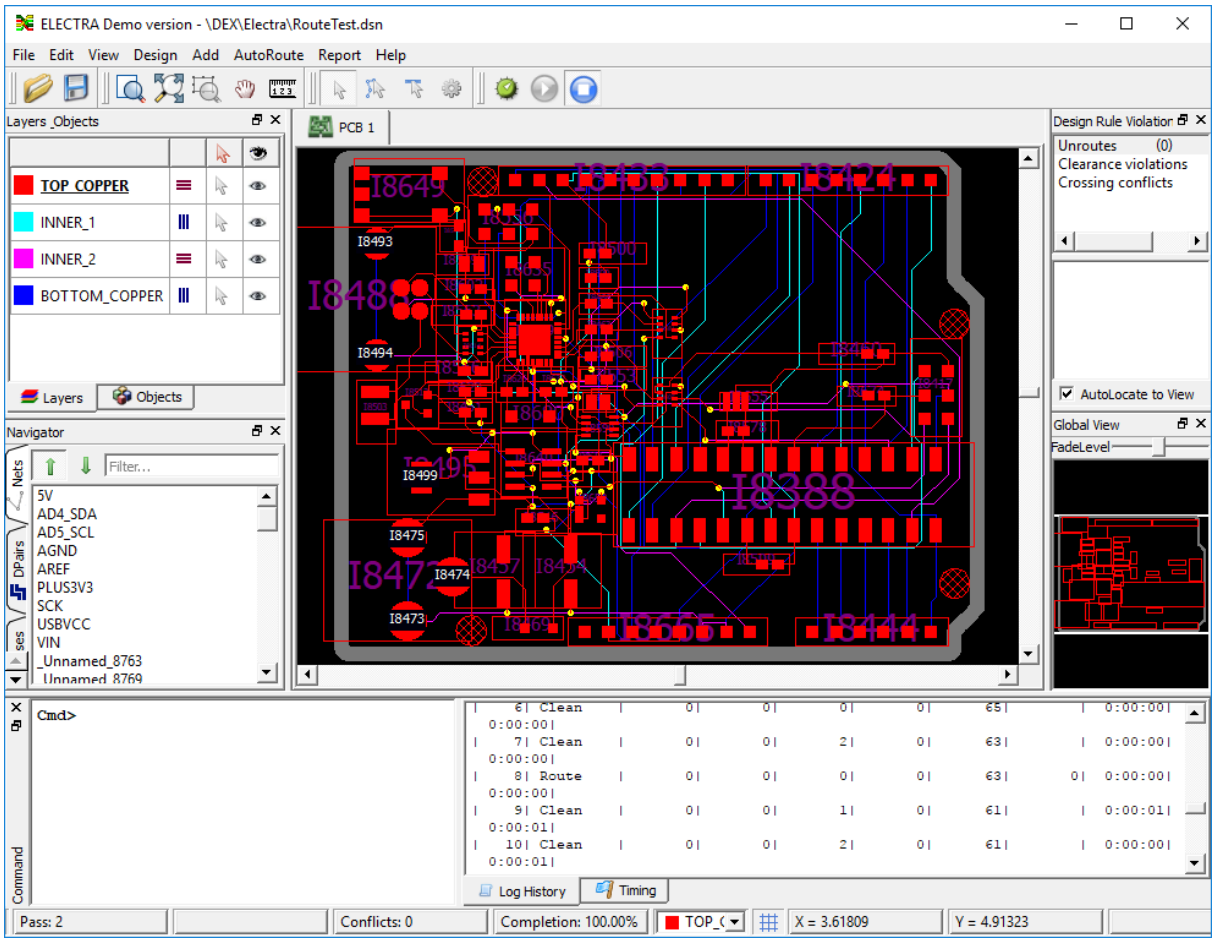
If you have checked then during routing the Electra application window will be displayed similar to the view below.

Please review the Electra manual on how to interact with Electra. *Click on the Help menu item at the top of the Window.*

Below you can see a list of the AutoTRAX DEX electrical layer by name at the upper left. The name are capitalized due to limitation in Electra.

Similarly, below the layer names you will see a list of AutoTRAX DEX net names. Again the name are capitalized due to limitation in Electra. If the AutoTRAX DEX net name is not set you will see it as _UNNAMED_<XXXX> where XXXX is a unique numeric identifier. Clock on the net name to see it in the Electra viewport.

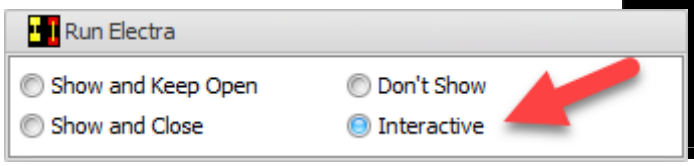
The Electra routed nets have the same color as in AutoTRAX DEX.



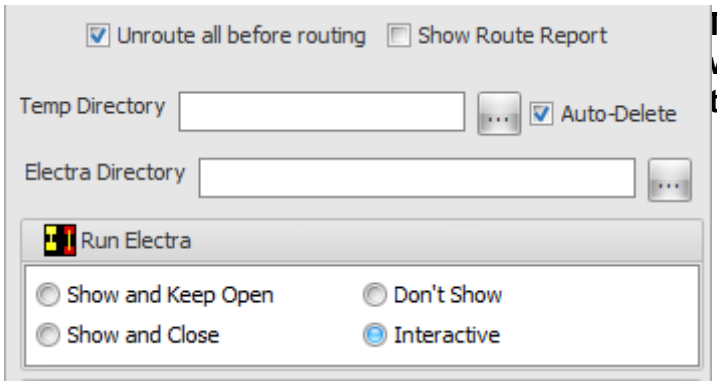
You can interactively route with Electra using AutoTRAX DEX.

To use Electra interactively check

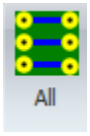
in Run Electra



at the top of the Electra Router Settings



This video show you how you can easily auto-route a PCB using TRAX and Electra. In particular it vs you an example of restricting ting of a net to a specific layer.

Next click on the 

button in the PCB->Route menu.

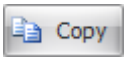
The main AutoTRAX DEX window will disappear and you will see the dialog below and Electra will start.

Do whatever you feel you need to do in Electra to route your PCB, see below.

You can click 

to minimize the above dialog.

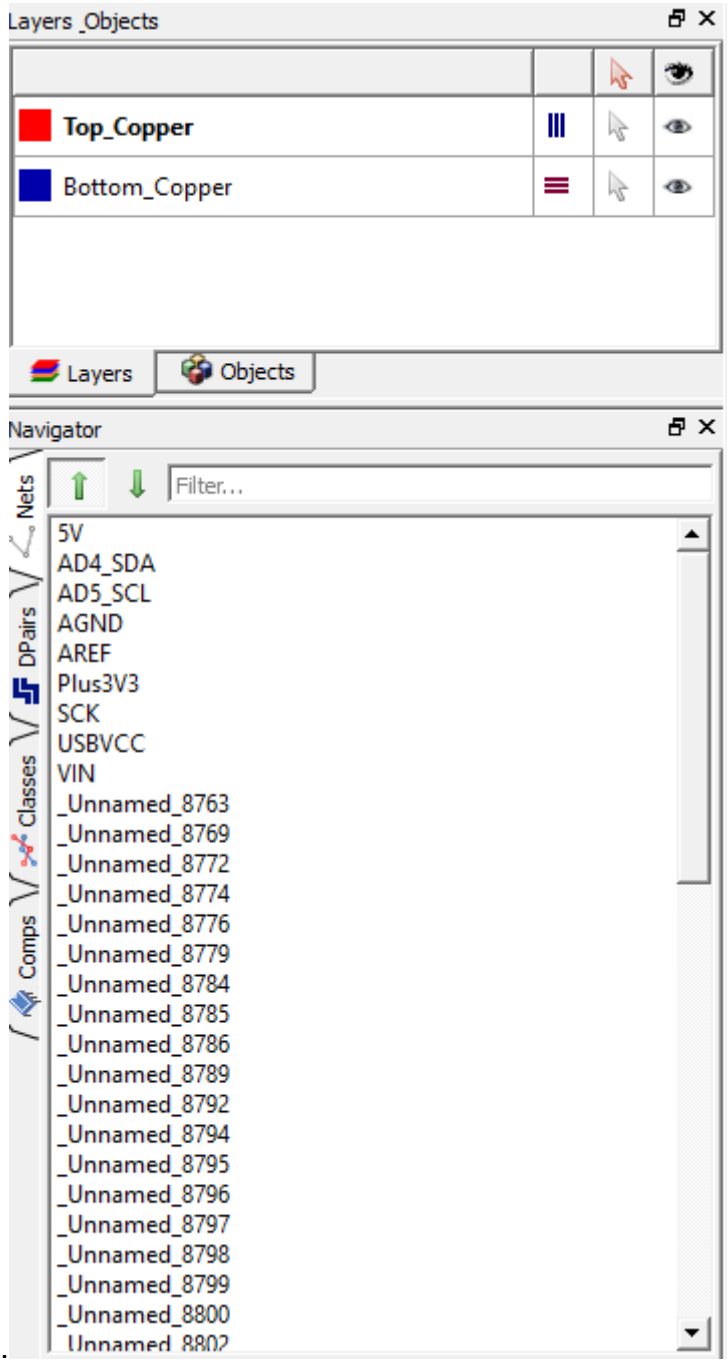
You can also close it and even close AutoTRAX DEX as so long as you have saved the project you can import the Electra route file.

Click 

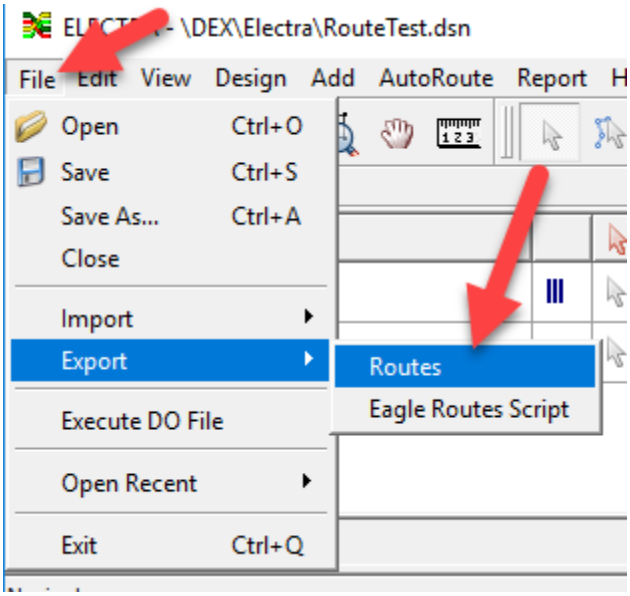
to copy the write routes command to the clipboard so you can paste it into the Electra command: window to export the Electra route file. As soon as you save the route file, the above dialog will automatically disappear and the AutoTRAX DEX main window will appear with the routed tracks.

Inside Electra

In Electra you will see the AutoTRAX DEX layers and named nets as shown below



After routing/editing your PCB in Electra you need to save your route file using the Electra File menu.



As soon as you save the route file, the above dialog will automatically disappear and the AutoTRAX DEX main window will appear with the routed tracks. If you export as a different name click on the 

in the above dialog to import the route information into AutoTRAX DEX.

The Electra router settings dialog is shown below.

Electra is installed by default in **C:\Program Files\AutoTRAX DEX Software\AutoTRAX DEX\Electra**

It is **C:\Program Files\AutoTRAX DEX Software\AutoTRAX DEX\Electra\Electra.exe**

This is the **full version** and will work for 15 days. You will then need to purchase it from https://konekt.com/c5/index.php

Unroute all before routing

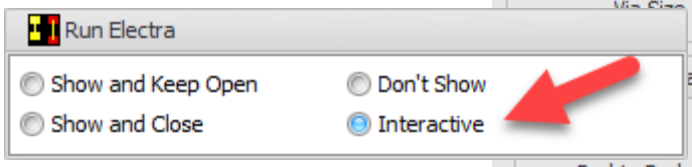
If checked the PCB will be totally unrouted and the Auto-route will be a full route. You are strongly advised to keep this checked.

Show Route-Report

Check Show Route Report to display a Route Report after routing.

Running Electra

There are 4 ways you can run Electra.



Show and Keep Open

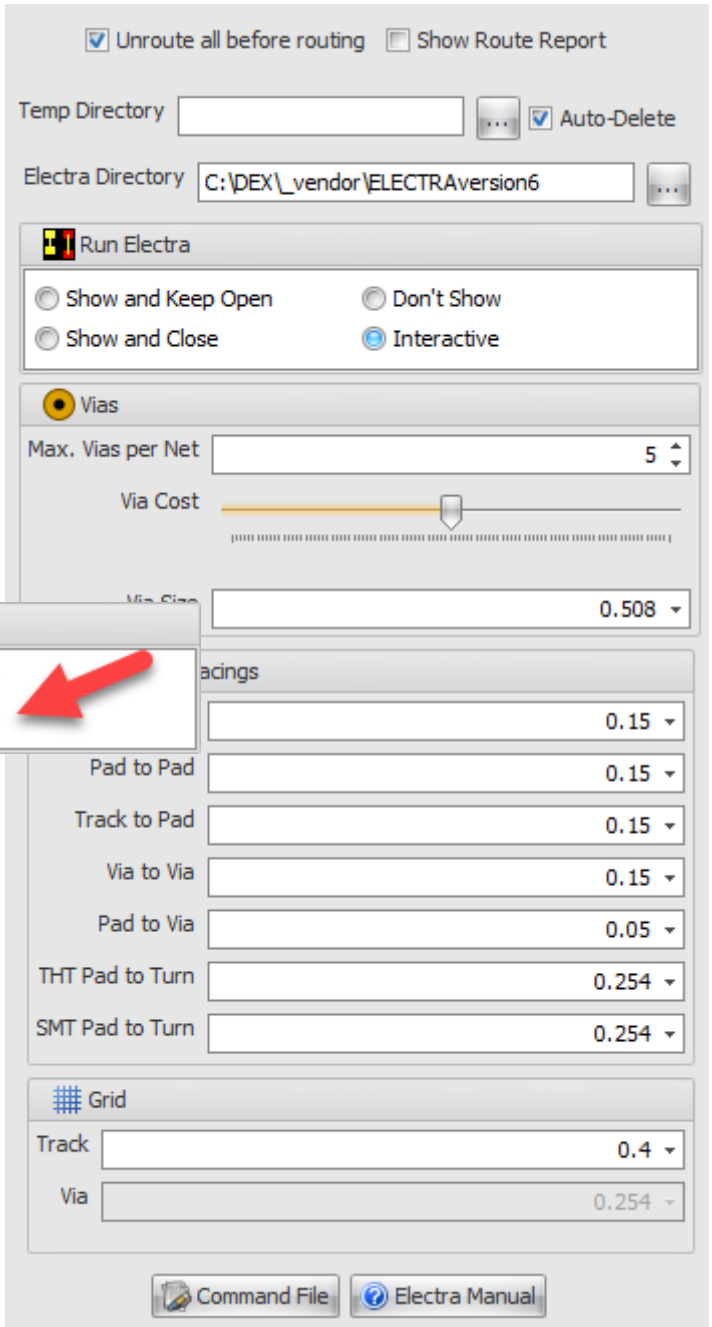
Electra will open, route and stay open. AutoTRAX DEX will automatically read in the Electra routes at the end of routing.

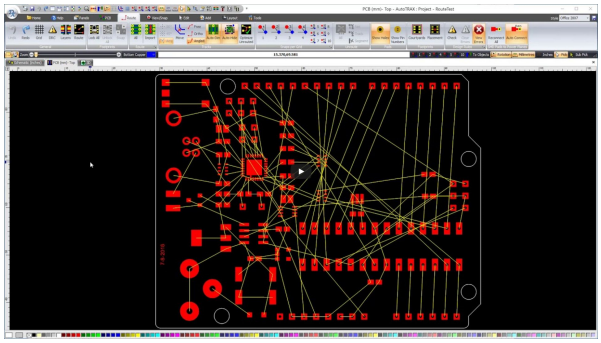
Show and Close

Electra will open, route and automatically close when routing is done. AutoTRAX DEX will automatically read in the Electra routes at the end routing.

Don't Show

Electra will not appear during routing but you will see the routing being done inside AutoTRAX DEX's CB viewport. ***This is not available when running the freeware version on Electra.***





Embedded Electra Routing Inside AutoTRAX

Interactive

1. This will open Electra with the current PCB but Electra will not automatically route the PCB.
2. You will need to route the PCB inside Electra using Electra commands
3. When you save the route file in Electra AutoTRAX DEX will automatically read be in. Optionally you can quit AutoTRAX DEX and import a later date.

This video show you how you can easily auto-route a PCB using AutoTRAX and Electra. In particular it shows you an example of restricting routing of a net to a specific layer.

Temp Directory

This is the directory used to store temporary files used by Electra during the routing process. Click the  button on the right to browse for a directory location.

Check the 

button to have AutoTRAX DEX automatically delete them after Electra has routed your PCB and AutoTRAX DEX has read in the results.

Electra uses the following files.

- **bestsave.rte** - intermediate routing results.

- **electraCommand.do** - Routing strategy file produced by AutoTRAX DEX.
- **projectName.dsn** - This is the file produced by AutoTRAX DEX to describe the circuit to Electra.
- **projectName.log** - Routing log file produced by Electra.
- **projectName.ses** - Session file produced by Electra.
- **projectName.sts** - Routing file produced by Electra.
- **projectName.rte** - This is the result of routing the PCB and is generated by Electra and read in by AutoTRAX DEX.

Electra Directory

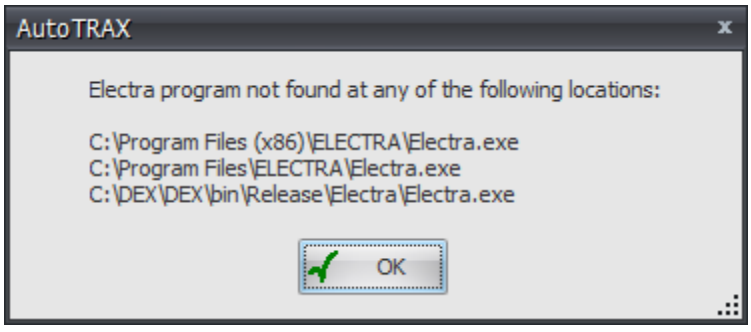
This is the directory that the Electra router is installed in to. AutoTRAX DEX comes complete with a full working copy of Electra. If you wish to use a different copy of Electra then enter the directory here. Click the  button on the right to browse for a directory location.

AutoTRAX DEX looks for the Electra program in the following directories:

1. The Electra directory you specify
2. C:\Program Files\ELECTRA
3. C:\Program Files (x86)\ELECTRA

If it does not find it in any of the about directories it tries to run the freeware version distributed with AutoTRAX DEX.

If it cannot find Electra in any of the above directories then it will display the following dialogue box:



Max Vias per Net

Enter the maximum number of vias to use for routing between any 2 pads in a net.

Via Cost

These is a relative via cost. Move the slider to increase the cost of using vias. Electra will then try 'harder' to not use vias.

Via Size

Enter the diameter to use for vias.

Minimum Spacings

Track to Track

This is the minimum track to track spacing.

Pad to Pad

This is the minimum pad to pad spacing.

Track to Pad

This is the minimum track to pad spacing.

Via to Via

This is the minimum via to via spacing.

Pad to Via

This is the minimum pad to via spacing.

TPH Pad to Turn

This is the minimum track distance from a TPH pad to the first corner.

SMT Pad to Turn

This is the minimum track distance from a surface mount pad to the first corner.

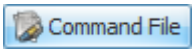
Grid

Track

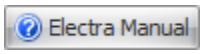
This is the track grid.

Via

This is the via grid.

Click on the 

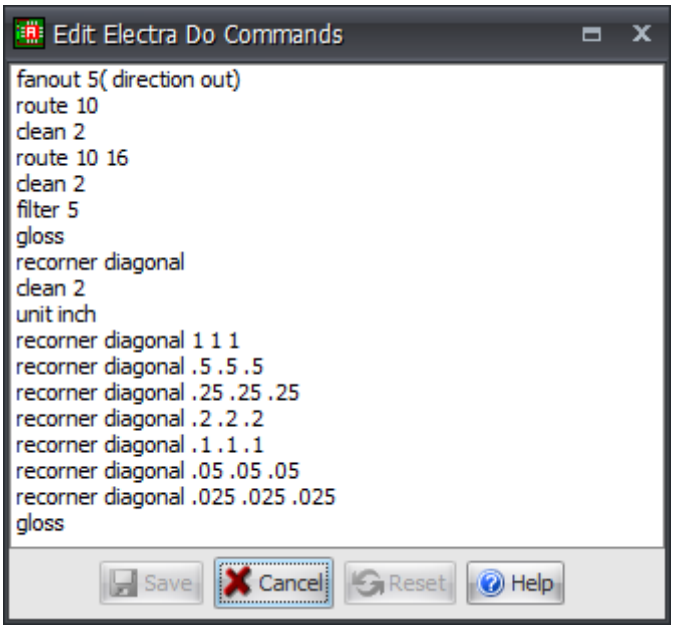
button to display the The Electra DO Commands

Click the 

to display the Electra manual.

You can set your own Electra route command from the Electra Router Settings

Click on the Command File button in Electra Router Settings to display the commands as shown below.



Click Save to save the Do file. This will then be used on all subsequent Electra routing.

Click the Reset to reset the commands to the factory settings.

bestsave

bestsave on | off

Controls automatic saving of the best routing solution during a multi-pass routing run. Routing result is saved into a file named “bestsave.rte” at the same location as the design file.

bus

bus [diagonal]

Invokes a “collinear pins” routing pass only on regular array of pins with collinear connections where the pins share a common X or Y coordinate. This is particularly effective on memory arrays. In this mode, the router will not generate conflicts, so rules must allow for sufficient space. By default traces are routed orthogonally, unless the diagonal option is specified.

check

This command can be used to run a DRC (design rules check) and visually tag the violations. This is used in particular when a rule is changed. A check is automatically invoked after every routing pass. The total number of violations is shown on the status line and visual feedback is added to the layout view to indicate conflict locations.

clean

clean [<passes>]

The clean command reroutes all the connections with higher costs settings and helps achieve

- Wire optimization with minimum bends
- Vias minimization
- Less off-center SMD pad entry
- Exit SMD pad on long edge

fanout

fanout [<passes>]

[(direction[in_out | in | out])]

[(pin_share [on | off])] [(via_share [on | off])]

{[(pin_type [active | signal | power | unused | all | single])]}

[(max_len <positive_dimension>)]

AutoRoutes short escape wires with a via from SMD pads. Recommended on SMD boards having more than 2 routing layers. Fanout direction can be set so that fanout vias are added inside SMD components, and/or outside. Fanout can be limited by pin type, for example pins connected to power nets only.

Examples:

5 fanout passes

fanout 5

Depth for blind & buried vias

fanout (depth opposite 2) (share_len 500)

5 (pin_type signal) (via_share on)

Sets via to microvia & grid 25

grid via 25 MICROVIA

Fanout using a grid of 25

fanout (via_grid 25)

filter

filter [<passes>]

At the end of a routing session you may end up with wires/vias in conflicts (crossing or too close to each other), so in order to remove these conflicts the router has to unroute the minimum number of connections that get into conflict, "Filter 5" will do that.

grid

grid [via <positive_value> [<via_id>] |

wire <positive_value> [<layer_name>]

Specifies wire and via grid spacing.

Examples:

use a routing grid of 8.333

grid wire 8.333

use a different routing grid 5 on layer 1

grid wire 5 layer 1

limit

limit [cross [<positive_integer> | -1] |

via [<positive_integer> | -1] |

bend [<positive_integer> | -1] |

way [<positive_dimension> | -1]]

The limit command sets absolute limit values to be applied to each connection. Control is provided to limit maximum allowed number of intersecting wire, number of vias per connection, number of bends, and the maximum distance of non preferred (wrong-way) routing. The range of limit for <positive_integer> is 0 through 255. You can set limit values, perform some routing passes, and return to the default system values by executing a limit command with a value of –1. If you don't supply limit values, computed default values are used by the autorouter.

Examples:

limit via 2

limit way 200

resets routing limit to default value

limit way –1

lock

lock net <name>

Prevents router from changing or deleting locked wires, Note that the router is not allowed to connect to locked wires.

miter

miter [diagonal] <pin_setback> <slant_setback> <tjunction setback> <bend_<start_setback> <end_setback>>

The miter command charges 90 degree wire corners to 135 degrees. It is performed on wire corners exiting pins and vias, as well as bend and slant wire configuration.

Pin_setback, slant_setback, tjunction and bend_setback options can be used to control which corner type is changed and you can override the default setback values as well:

miter pin .2

miter slant .2

miter bend .2

miter tjunction .2

In comparaison to the recorner command, the miter functionality adds control for tjunction and start/end setback value for the bends.

recorner

recorner [diagonal] <pin_setback> <slant_setback> <bend_setback>

The recorner command charges 90 degree wire corners to 135 degrees. It is performed on wire corners exiting pins and vias, as well as bend and slant wire configuration.

Pin_setback, slant_setback and bend_setback options can be used to control which corner type is changed and you can override the default setback values as well:

recorner pin .2

recorner slant .2

recorner bend .2

Pin option is for wires that are connected to pin or via.

Slant is two consecutive 90 degree bends

recorner diagonal .25 .25 .25

You can make multiple calls with decreasing values.

recorner diagonal .50 .50 .50

recorner diagonal .25 .25 .25

recorner diagonal .10 .10 .10

route

route [<passes>[<start_pass>]]

When the number of completed passes (n) is less than 15, the <start_pass> should be n+1.

When the number is higher than 15, then the <start_pass> should be 16.

status_file

Creates a status file with routing history and located in the design directory. The status filename is named after the design name, with the extension .sts.

tax

tax [way | cross | via | off_grid | off_center | side_exit |

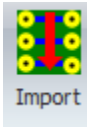
squeeze | layer <layer_name>] [<real> | positive_integer>]

This is an alternative method of using the cost command. This is the recommended way to adjust costing. The tax command applies a multiplier to control internal costing. The default value for tax is 1.

write

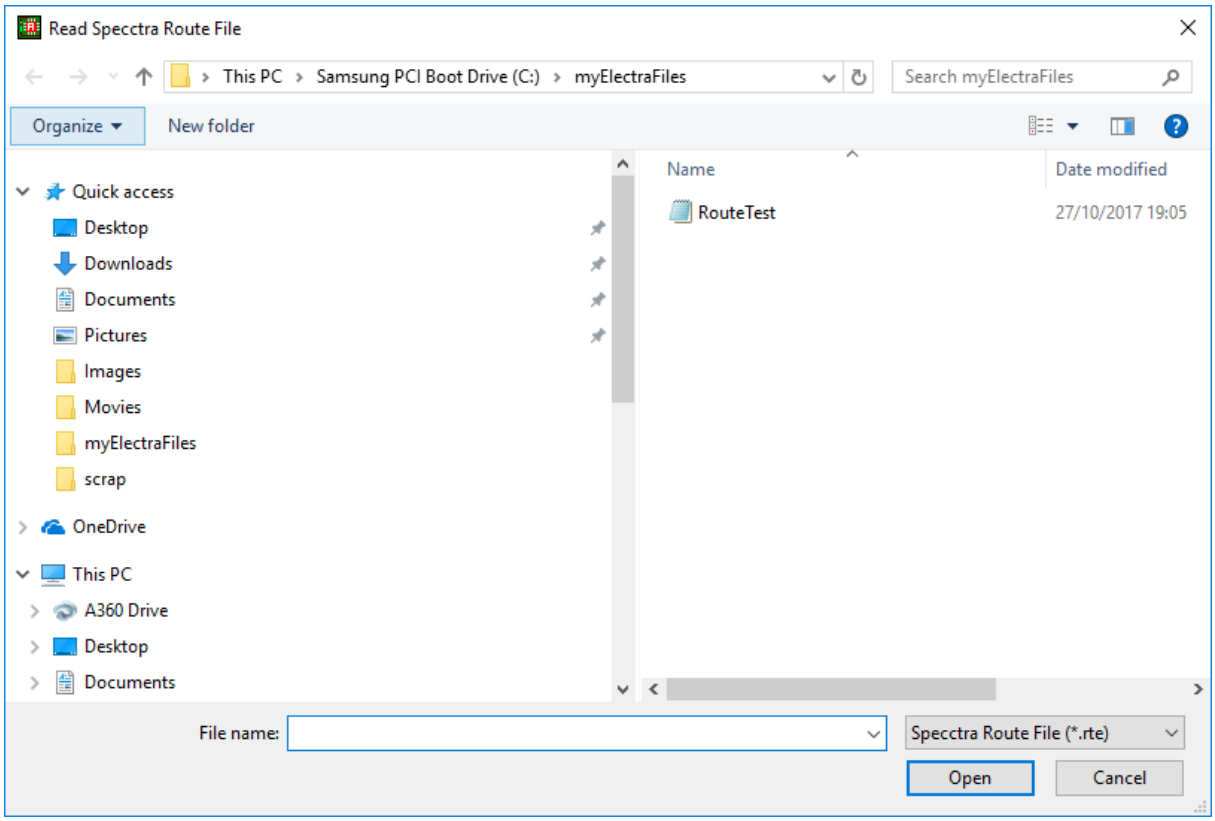
write [routes | session | script] [<filename>]

Command to save results to a routes file or a session file.The session file provides an integrated file to manage design data and routing data.

To import an Electra route files (*.rte) click on the 

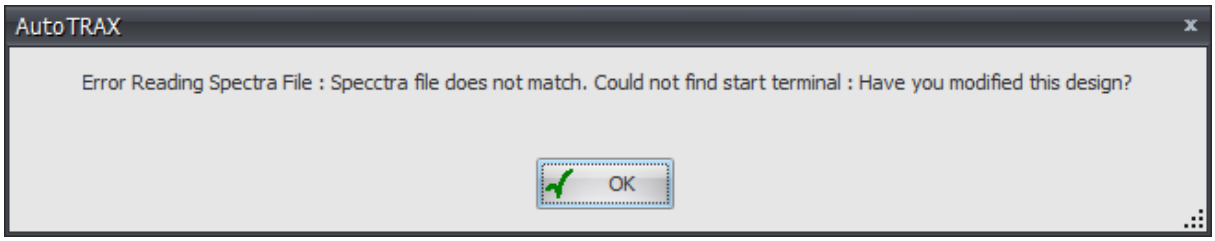
button in the Pcb->Route menu.

You will prompted for the route file. Enter the file name and the route information will be imported.

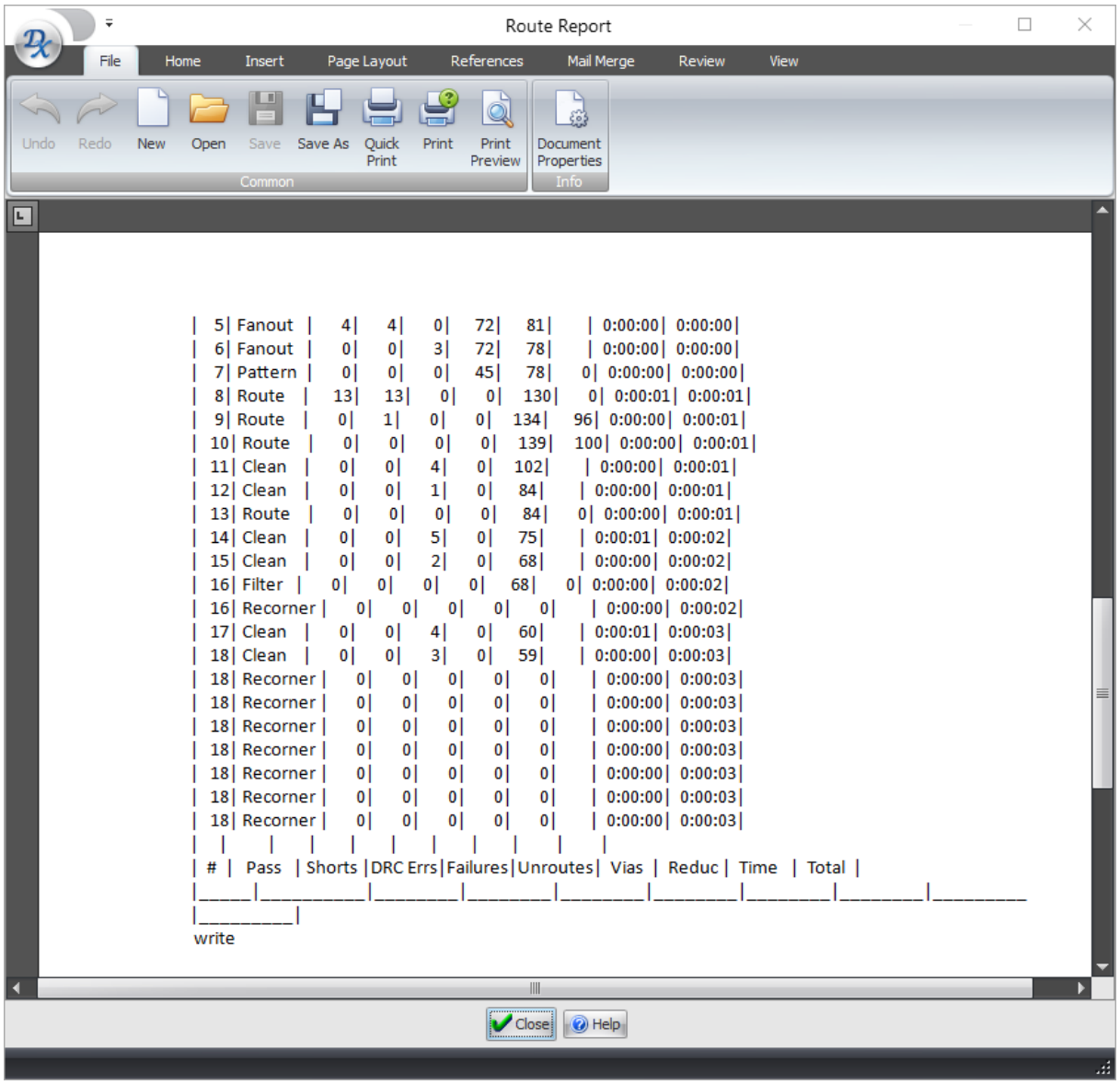


NOTE: The Electra route file **MUST** have been produced by the current state of the PCB, otherwise it will fail as Part Count/positions could differ and schematic routing could also differ (making a different set of nets in the PCB).

If it fails you will see as dialog like the one below. To fix it, reroute the PCB with Electra.



Check Show Route Report in the Electra Router Settings to display a route report after routing. Below is a typical route report.

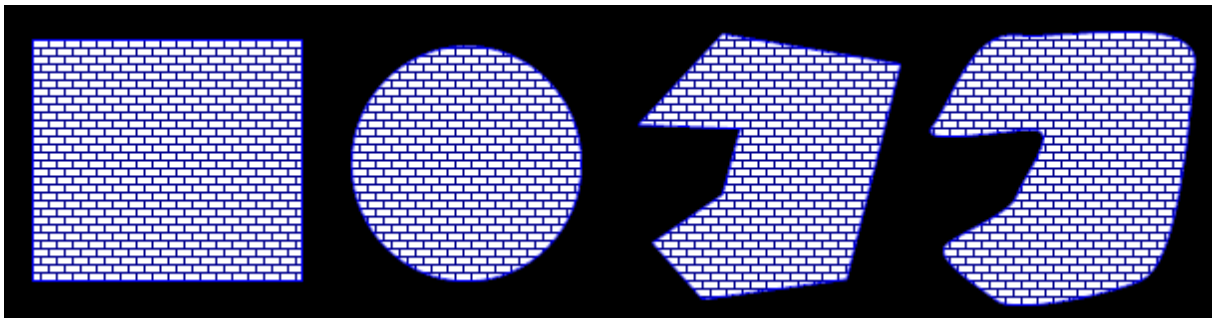


1.2.6.11.29.4 Automatic Routing

AutoTRAX DEX has an internal auto-router.

1.2.6.11.29.5 Keep Out Regions

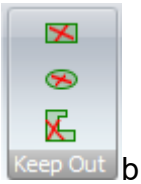
Keep-out regions are areas of a PCB that must remain free of electrical tracks and copper pour fill. You define these by adding keep-out regions to your board on the layer you wish to restrict



Keepout Regions

Adding Keepout Regions

Editing Keepout Regions

To add a keep out region click on one of the 

buttons in the **Add→Keepout** ribbon button group/

Click  to add a rectangular keep out region.

Click  to add an elliptical keep out region.

Click  to add a polygonal keep out region.

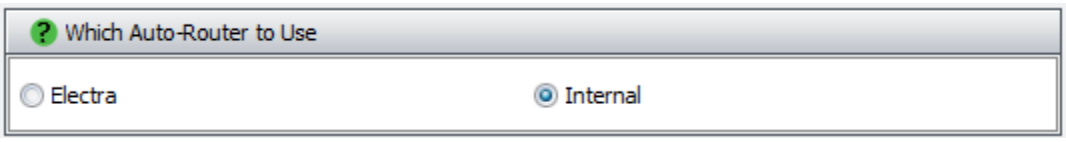
To edit a keepout first select it.

1.2.6.11.29.6 Router Settings

To set the auto-router/manual router settings click the small  button at the bottom right of **PCB→Route** ribbon button group.

Selecting The Auto-Router

To select which router to use for automatic routing check the appropriate control shown below.



Micro Vias

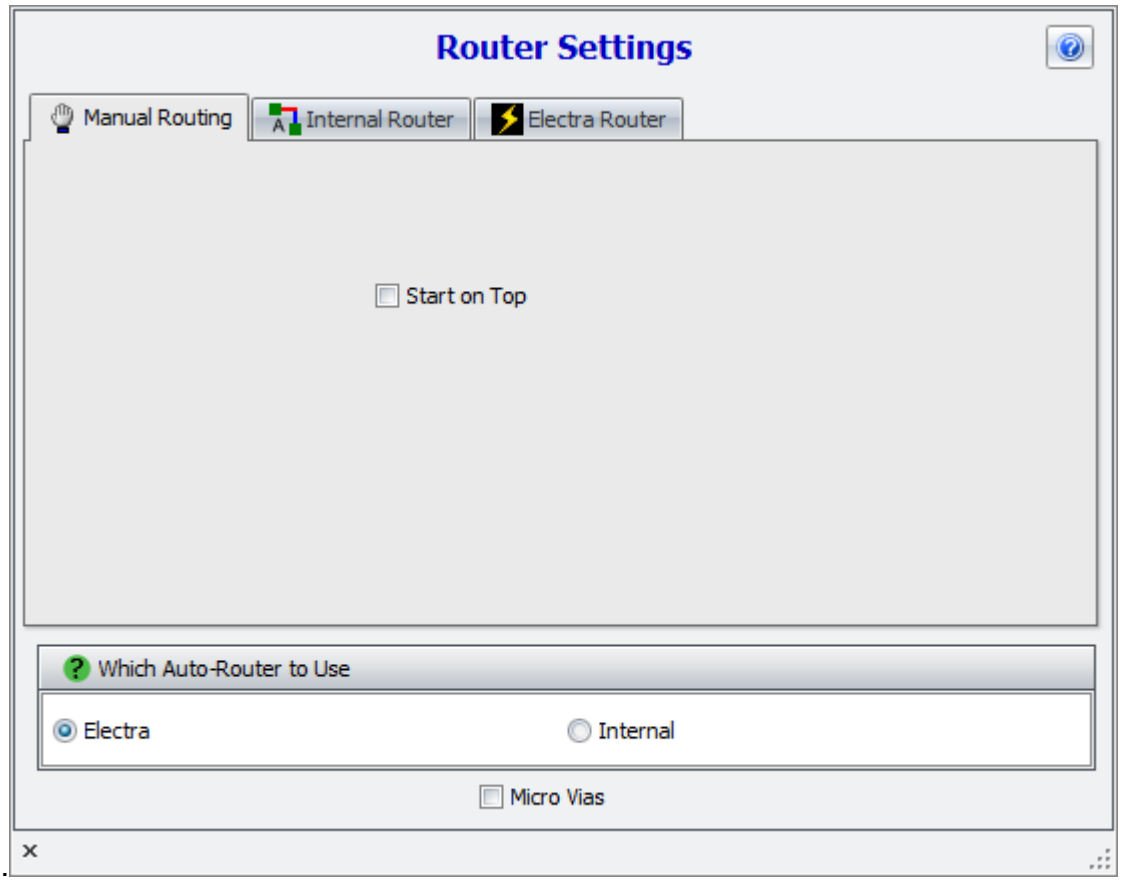
Check the Micro Vias box to enable micro vias.

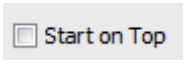
Router Specific Settings

There are 3 different tabs with each relating to a specific area of routing:

- Manual Router Settings
- Internal Router Settings

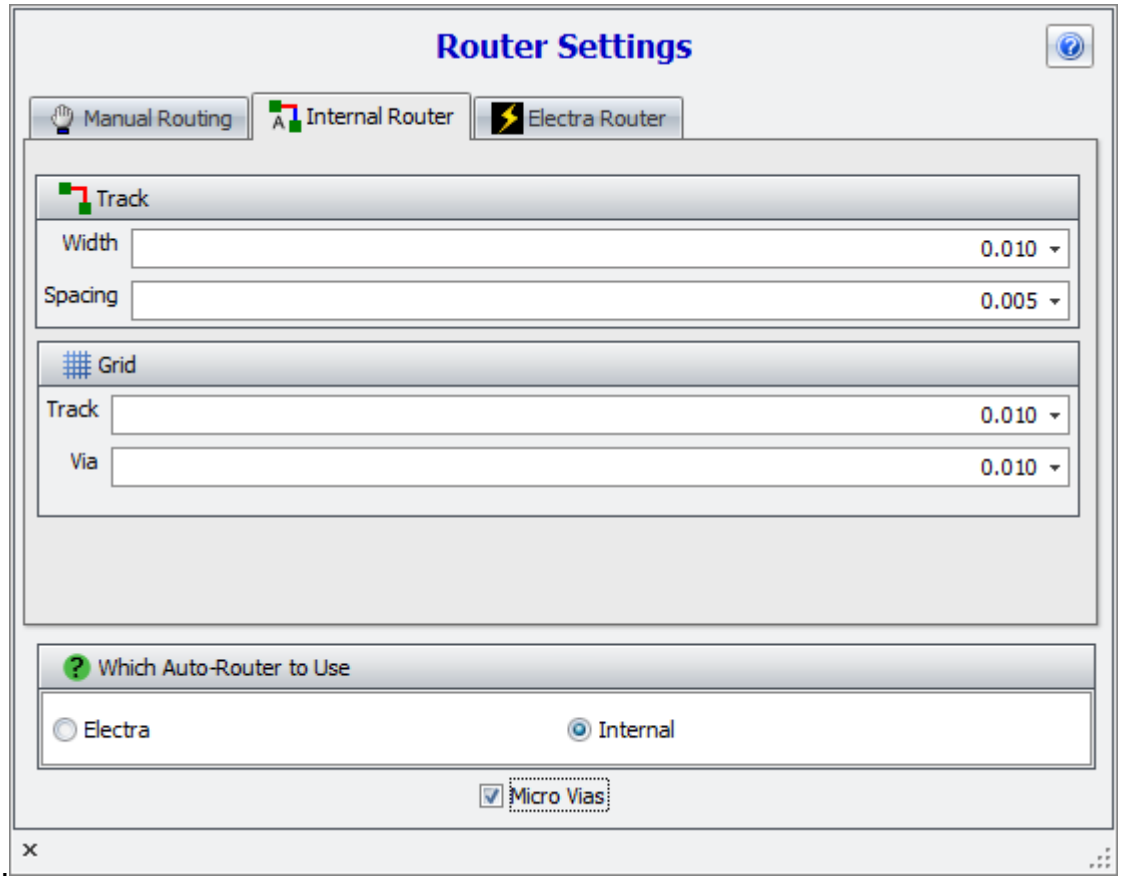
The manual router settings dialog is shown below



Check 

to start routing on the top layer. Uncheck to start routing on the bottom layer. This only applies to PCB with 2 or more electrical layers.

The internal router settings dialog is shown below



Track

Width - The default width for tracks. You can override this by setting the width for the track/net using the tracks properties box.

Spacing - The spacing between tracks. The internal router is a grid router and this is the size of the grid.

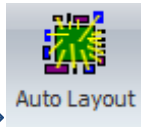
Grid

Track - The internal router is a grid router and this is the size of the grid for tracks.

Via - This is the size of the grid for vias.

1.2.6.11.29.7 Snap To Pad

1.2.6.11.30 Auto-Layout

To auto-layout your PCB click the Layout→PCB→ 

button.

1.2.6.11.30.1 An Insight into PCB Auto-Layout

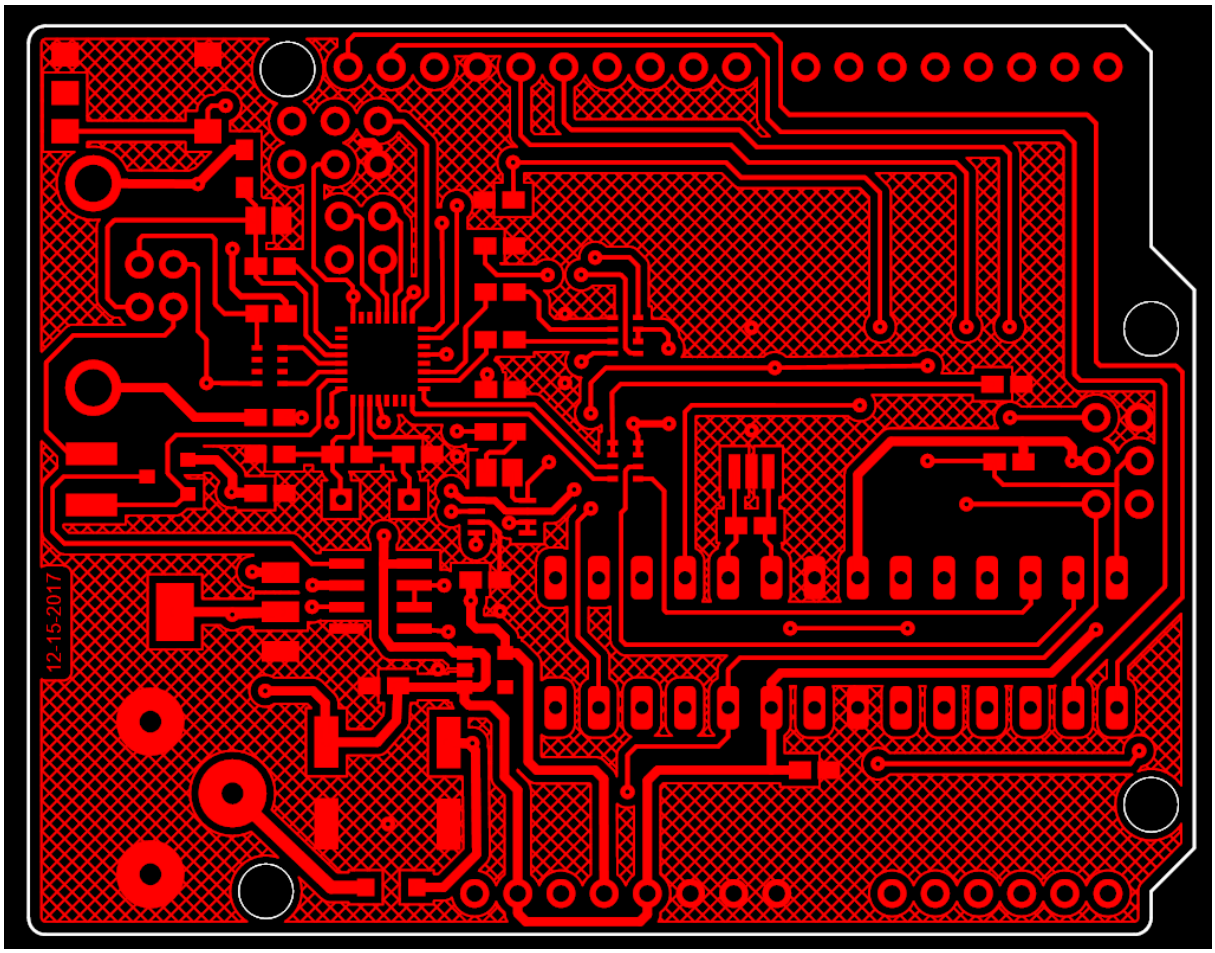
1.2.6.11.31 Copper Pour Regions

You can add copper pour areas to your PCB.

The term "copper pour" refers to an area on a printed circuit board filled with copper (the metal used to make connections in printed circuit boards). Copper pour is commonly used to create a power or ground plane. Another reason for using copper pour is to reduce the amount of etching fluid used during manufacturing.

A distinctive feature of copper pour is the backoff/margin (or stand-off) - a certain distance between the copper pour and any tracks or pads not belonging to the same electrical net. A copper pour therefore looks like it flows around other components, with the exception of pads which are connected to the copper pour using thermal connections.

While solid copper pour provides better resistive characteristics, hatched copper pour is used to balance the heat and open space on both sides of the board in order to avoid warping of certain substrates. Heating might cause gas bubbles between solid copper pour and certain substrates. Furthermore, it might be possible to adjust the impedance of high frequency traces by using hatched copper pour in order to reach better signal quality.



Copper Pour Area

Adding Copper Pour Regions

Editing Copper Pour Regions

Copper Pour Settings

1.2.6.11.31.1 What is PCB Copper Pour

PCB copper pour, also known as copper fill or copper plane, refers to the process of filling empty areas on a printed circuit board (PCB) with copper to create a continuous, solid copper region. This copper region is often used as a ground plane or power plane, providing a low impedance path for return currents and a stable voltage reference for components.

The copper pour is typically done on inner layers or additional layers of a multilayer PCB, where the surface is not occupied by traces or components. Copper pour can also be used on the top and bottom layers of the PCB in certain cases.

Here's how the copper pour process works

Creating Copper Regions: In PCB design software, the designer defines areas on the PCB that should be filled with copper. These areas are known as copper pour regions.

Connecting to Nets: Copper pour regions can be connected to specific nets, such as a ground net or a power net. This connection ensures that the copper pour is electrically tied to the desired net, providing a conductive path for the ground or power signal.

Adjusting Parameters: The designer can adjust parameters for the copper pour, such as the thickness of the copper fill, the distance between the copper and nearby traces or components (clearance), and any required thermal relief connections to vias and pads.

Filling Empty Spaces: Once the copper pour regions are defined, the PCB design software automatically fills the defined areas with copper, creating a continuous copper plane on the selected layers.

Benefits of PCB Copper Pour

Ground and Power Planes: Copper pour is commonly used to create ground planes and power planes, which help improve signal integrity, reduce noise, and provide stable voltage references.

Heat Dissipation: Copper is an excellent conductor of heat. In some cases, copper pour can be used to dissipate heat generated by high-power components or power traces.

Reduction of Impedance: Copper pour helps reduce the impedance of signal traces by providing a low-resistance return path for return currents.

EMI Reduction: By using copper pour to create solid planes, electromagnetic interference (EMI) can be reduced, as the continuous copper regions act as shields, reducing radiated emissions and susceptibility.

Enhanced Manufacturability: Copper pour can improve manufacturability by reducing the number of isolated copper islands on the PCB, which simplifies the manufacturing and etching process.

However, it's essential to carefully design the copper pour to avoid potential issues such as clearance violations, unintentional shorts, or interference with signal traces. Proper clearance rules, thermal reliefs, and polygon stitching are some of the techniques used to mitigate these issues and ensure the success of the copper pour.

1.2.6.11.31.2 Adding Copper Pour Regions

To add a copper pour areas click on one of the 3 buttons 

in the **Add→Pour** ribbon button group.

There are 3 basic shape type.

1. Rectangular
2. Elliptical/Circular
3. Polygonal.

1.2.6.11.31.3 PCB Copper Pour and Warped PCBs

PCB copper pour can sometimes contribute to the warping or bowing of printed circuit boards (PCBs), especially in multilayer designs. Warping occurs when the PCB's flatness is compromised, leading to a non-planar surface. There are a few factors related to copper pour that can contribute to this issue:

Thermal Imbalance during Manufacturing: During the PCB manufacturing process, temperature variations can occur due to the difference in thermal conductivity between the copper pour areas and the non-copper areas. When the PCB goes through processes like solder mask application, soldering, or reflow, the uneven distribution of heat can cause differential expansion and contraction, leading to warping.

Insufficient Copper-Pour Relief: Copper pour regions can be connected to ground planes or power planes, and they also create larger areas of copper on the PCB. If the copper pour is too extensive and does not have adequate relief, it can cause stress during manufacturing and thermal cycles, leading to warping.

Uneven Copper Distribution: In some cases, copper pour regions might have uneven copper distribution or excessive copper thickness. This uneven distribution can lead to varying thermal expansion rates, which contribute to warping.

Imbalance in Copper-Pour Coverage: Uneven or unbalanced copper-pour coverage on different layers of a multilayer PCB can create stress imbalances and lead to warping.

To mitigate warping caused by copper pour, PCB designers and manufacturers can employ several strategies:

Optimize Copper-Pour Area and Distribution: Careful consideration should be given to the size and location of copper pour regions. Designers can strategically create smaller copper islands instead of large contiguous copper pours to balance thermal effects.

Thermal Relief: Applying thermal relief to copper pour connections can help reduce the mechanical stress caused by temperature variations during manufacturing and soldering processes.

Balanced Copper-Pour Distribution: Ensuring a balanced distribution of copper pour on different layers of a multilayer PCB can help maintain mechanical stability.

Controlled Impedance Design: Designing controlled impedance traces and properly managing layer stackups can help reduce the impact of warping.

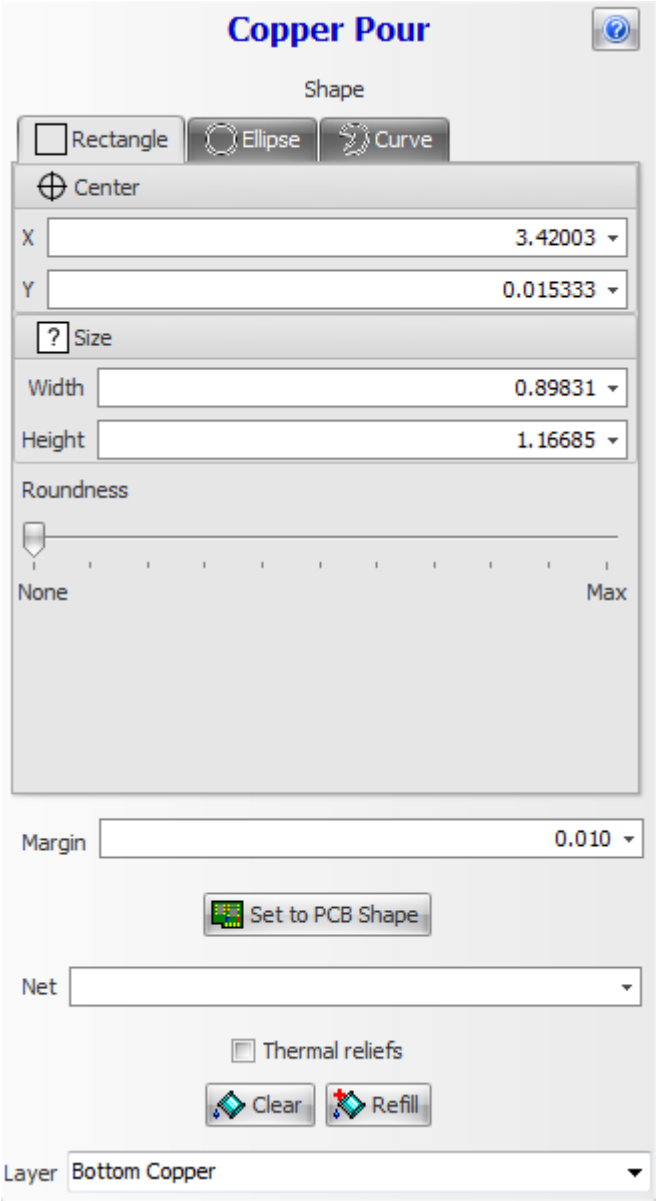
Consideration of PCB Thickness and Material Properties: The choice of PCB thickness and material properties, including the type of substrate used, can influence warping tendencies. Selecting appropriate materials and thicknesses can help address this issue.

Manufacturing Processes: Manufacturers can adopt controlled heating and cooling processes during PCB assembly to minimize thermal stress and prevent warping.

By employing these strategies, PCB designers and manufacturers can minimize the risk of warping and ensure that the final PCBs meet the required dimensional tolerances and maintain their flatness throughout their lifecycle. Collaboration between design and manufacturing teams is essential to achieve the best results.

1.2.6.11.31.4 Editing Copper Pour Regions

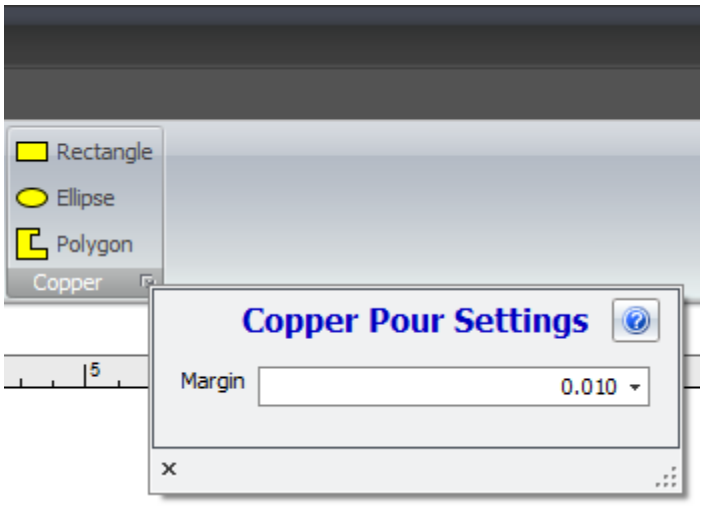
To edit a copper pour, use the copper pour's properties dialog in the Properties panel.



Copper Pour Properties

1.2.6.11.31.5 Copper Pour Settings

You can set the initial setting of copper pours by clicking on the small button at the lower left of the **Add→Copper** ribbon button group.



1.2.6.11.32 The Solder Mask

Solder mask or solder resist is a thin lacquer-like layer of polymer that is usually applied to the copper traces of a printed circuit board (PCB) for protection against oxidation and to prevent solder bridges from forming between closely spaced solder pads. A solder bridge is an unintended electrical connection between two conductors by means of a small blob of solder. PCBs use solder masks to prevent this from happening. Solder mask is not always used for hand soldered assemblies, but is essential for mass produced boards that are soldered automatically using reflow or solder bath techniques. Once applied, openings must be made in the solder mask wherever components are soldered, which is accomplished using photolithography. Solder mask is traditionally green but is now available in many colors.

Solder mask comes in different media depending upon the demands of the application. The lowest-cost solder mask is epoxy liquid that is silk-screened through the pattern onto the PCB. Other types are the liquid photoimageable solder mask (LPSM) inks and dry film photoimageable solder mask (DFSM). LPSM can be silkscreened or sprayed on the PCB, exposed to the pattern and developed to provide openings in the pattern for parts to be soldered to the copper pads. DFSM is vacuum laminated on the PCB then exposed and developed. All three processes go through a thermal cure of some type after the pattern is defined.

In electronic design automation, the solder mask is treated as a layer of the printed circuit board, and is described as a Gerber file like any other layer, such as the copper and silkscreen layers.

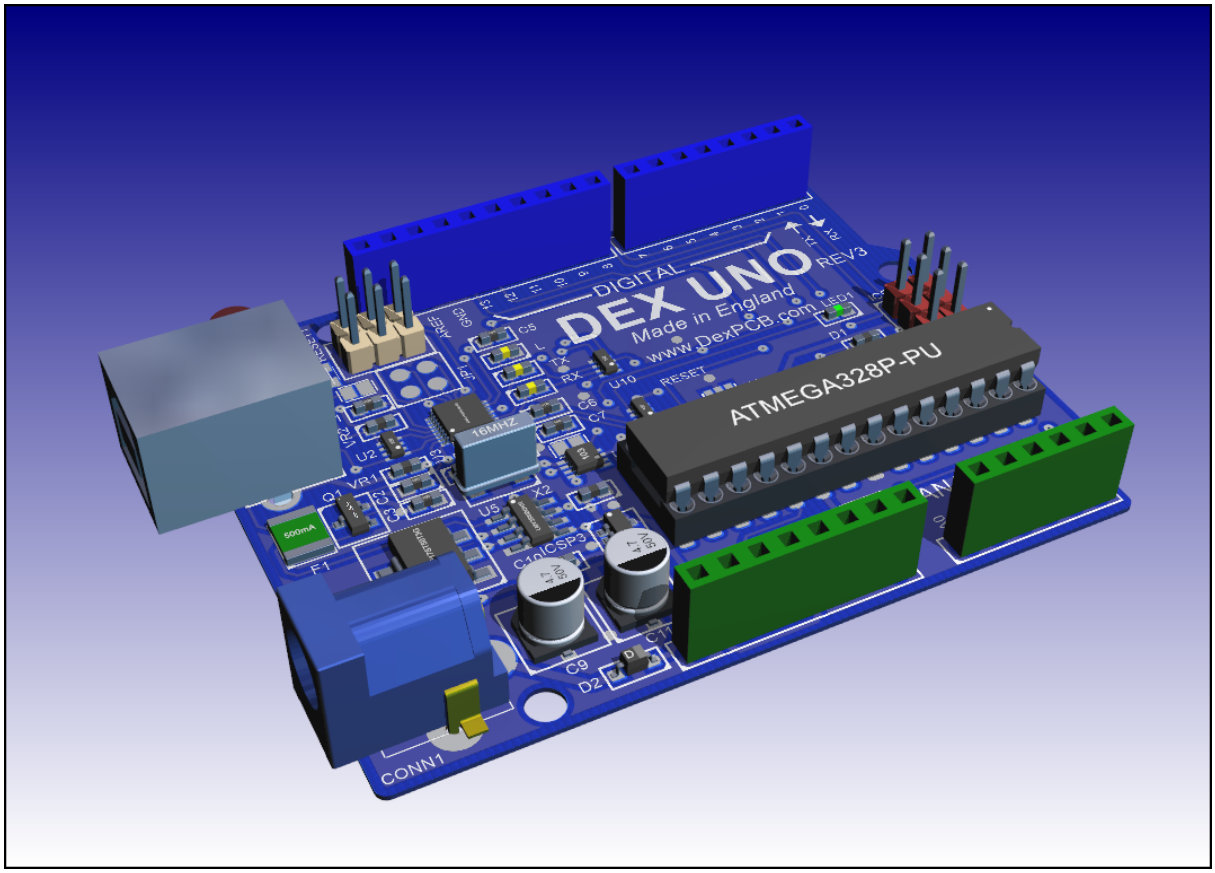
The solder mask is generated by default and covers all the PCB except pads and no solder mask areas. **There is no explicit Solder Mask layer in AutoTRAX DEX**. The mask is automatically generated for the top and bottom sides of the PCB. Normally you would not need to make any changes yourself.

Color Me Beautiful

For a seemingly endless time, solder-mask has been green and only green. But all that has started to change.

There is a trend towards clear or colorful electronics, in everything from toasters to blenders and from phones to computers. With OEMs trending toward clear appliances and electronics, boards are taking on much more of a personality. Any marketing professional will tell you if you're going to see the guts of something, it better look interesting and it had better be cool.

Today there are a variety of commercially available colors from which to choose. Alternative colors certainly aren't new. But the past several years have seen a growing use of colors other than green, primarily driven by assemblers and OEMs.

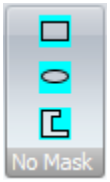


1.2.6.11.32.1 Adding Solder Mask Cutouts

In addition to the automatic cutouts for pads, you can add mask cutouts.

Rectangular, elliptical and polygonal solder mask cutout

You can add solder mask cutout using the buttons in the Add->No Mask button group.



Adding a rectangular solder mask cutout

To add a rectangular no solder mask area click on .

1. Move the cursor to the top left or bottom right corner for the cutout.
2. Click the left mouse button and drag the mouse to define the cutout or just hold down the left mouse button and drag.
   a. If you hold down the Shift key the cutout rectangle will be centered at the position you first clicked.
   b. If you hold down the Ctrl key the cutout rectangle will be square.
   c. You can hold down both the Shift and Ctrl key to create a square cutout centered at the original left click.
3. Click again to end adding the cutout or release the left mouse button if you started by holding down and dragging.

Adding an elliptical or circular solder mask cutout

To add an elliptical or circular no solder mask area click on .

4. Move the cursor to the top left or bottom right corner for the cutout.
5. Click the left mouse button and drag the mouse to define the cutout or just hold down the left mouse button and drag.
   a. If you hold down the Shift key the cutout ellipse/circle will be centered at the position you first clicked.

b. If you hold down the Ctrl key the cutout will be a circle.

c. You can hold down both the Shift and Ctrl key to create a circle cutout centered at the original left click.

6. Click again to end adding the cutout or release the left mouse button if you started by holding down and dragging.

Adding a polygonal solder mask cutout

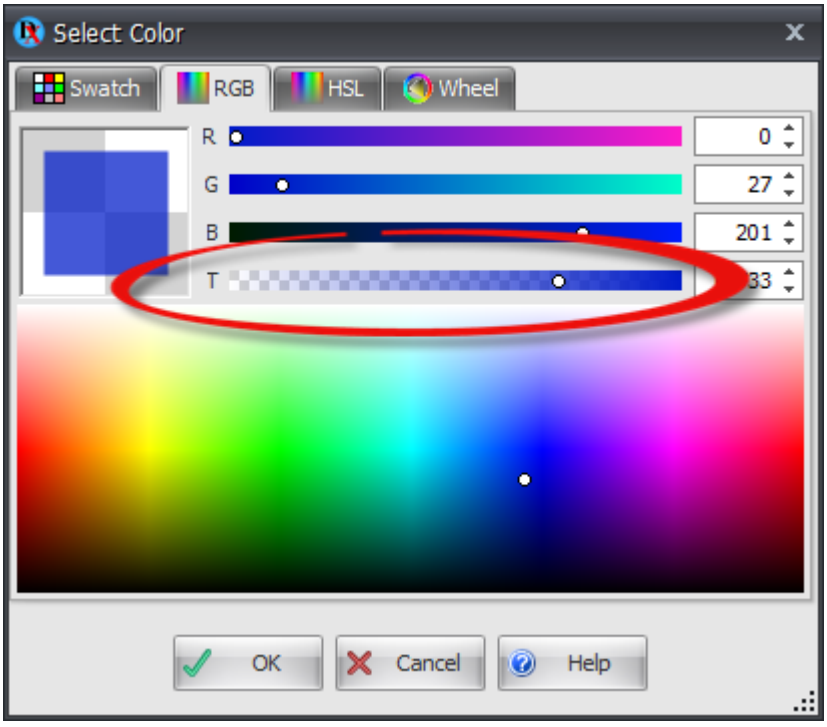
To add an elliptical or circular no solder mask area click on .

1. Move the cursor to the first corner of the cutout and click the left mouse button.
2. Now drag the cursor to define the first edge and click to start the second edge.
3. Repeated drag and click to define additional edge.
4. Double click the complete the polygonal cutout.

1.2.6.11.32.2 Editing Solder Masks

Setting the color of the solder mask

To set the color of the solder mask click on the Color button. The color selector dialog will appear. Make sure to make it semi-transparent so you can see through it when it is displayed. You can make it semi-transparent by dragging the T slider shown below.



Solder mask color

Removing a pad's cutout

Normally a pad generates its own solder mask cutout. However you may want to define a custom no solder mask area and do not want the pad to automatically create a no solder mask region. You do this by selecting the pad and unchecking the Add Solder Cutouts checkbox in the pads properties panel.

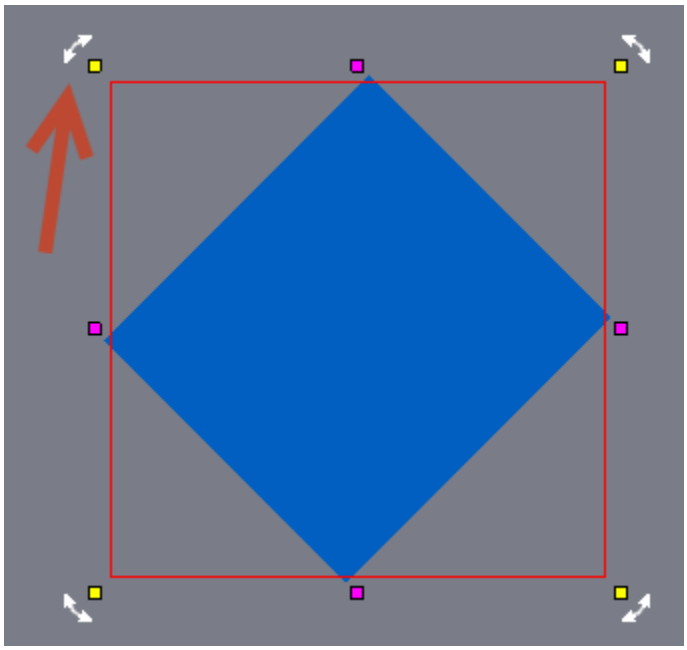
Adding a custom soldermask cutout

You can add a custom solder mask cutout to the top or bottom side of the PCB. To add one you must have either the top copper layer or the bottom copper layer selected. You can do this using the layers panel.

A custom solder mask cutout can be rectangular (with optional rounded corners) or circular/elliptical or polygonal/curved.

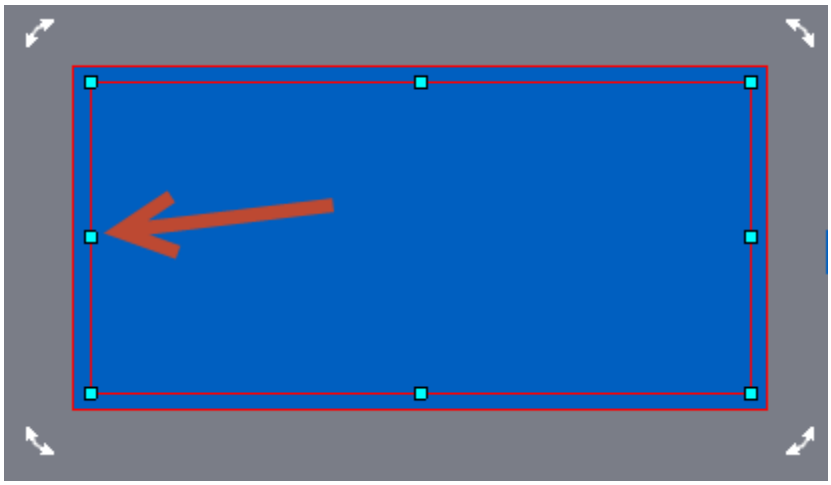
Rotating a soldermask cutout

1. To rotate a cutout select the cutout by clicking on one of its edges.
2. Drag one of the corner manipulators to rotate it. See rotation snap and rotating objects for more.



Scaling a soldermask cutout

1. To cutout a cutout select the cutout by clicking on one of its edges.
2. Drag one of the edge manipulators to scale it. See scaling objects for more.



Moving a soldermask cutout

To move a soldermask cutout, hold down the left mouse key over the cutouts edge and drag it.

Deleting a soldermask cutout

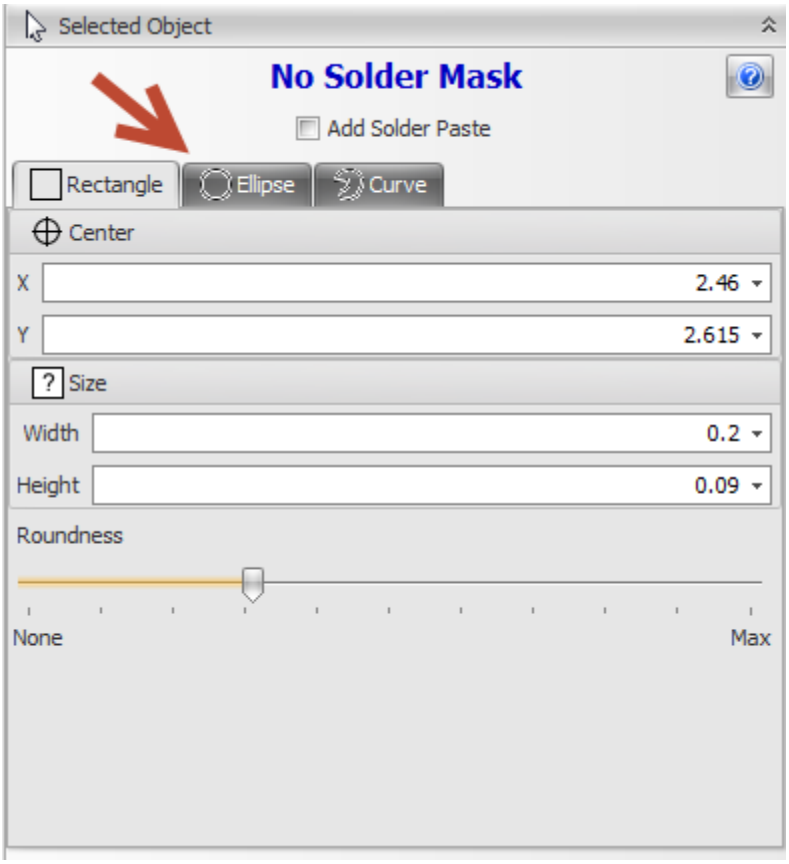
Select the cutout and press the **DEL** key.

1.2.6.11.32.3 Changing a Solder Mask Cutout's Shape

Solder mask cutouts come in 3 different forms. Rectangular, elliptical and polygonal/curved.

To change the shape first select the cutout.

To switch between rectangular, elliptical and polygonal/curved cutouts, click on the top tab.

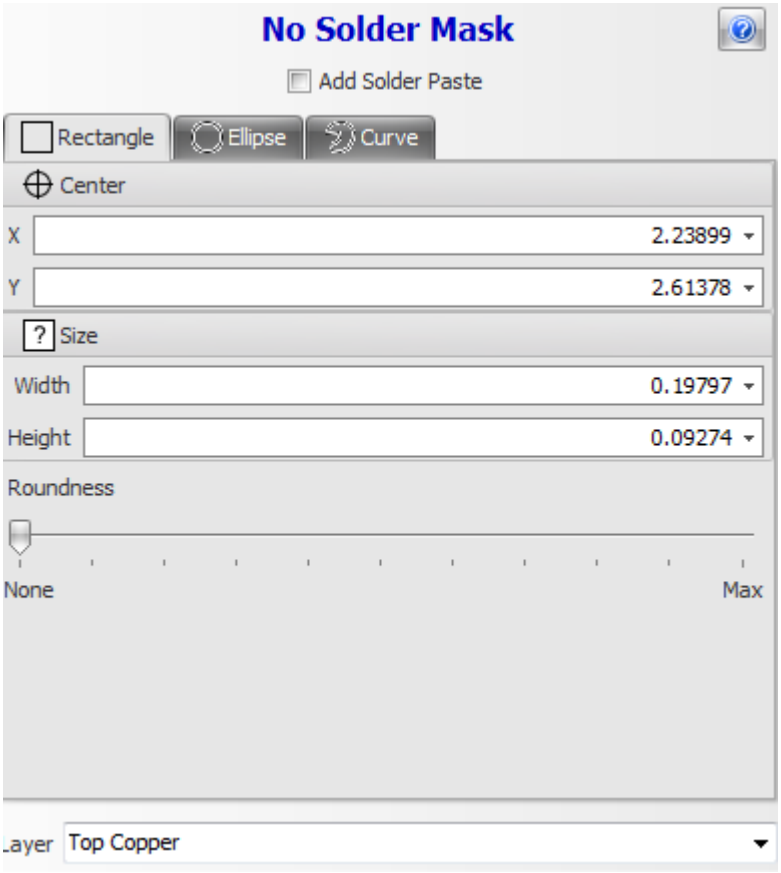


Solder mask cutouts can be rectangular. To change to rectangular cutout click on the Rectangle tab in the cutouts properties in the properties panel.

Rectangular cutout can have rounded corners.

without and with rounded corners

To edit the parameters of a cutout, first select it. The properties panel will display the properties for the selected cutout.



X

The x coordinate of the center of the cutout

Y

The y coordinate of the center of the cutout

Width

The width of the cutout.

Height

The height of the cutout.

Roundness

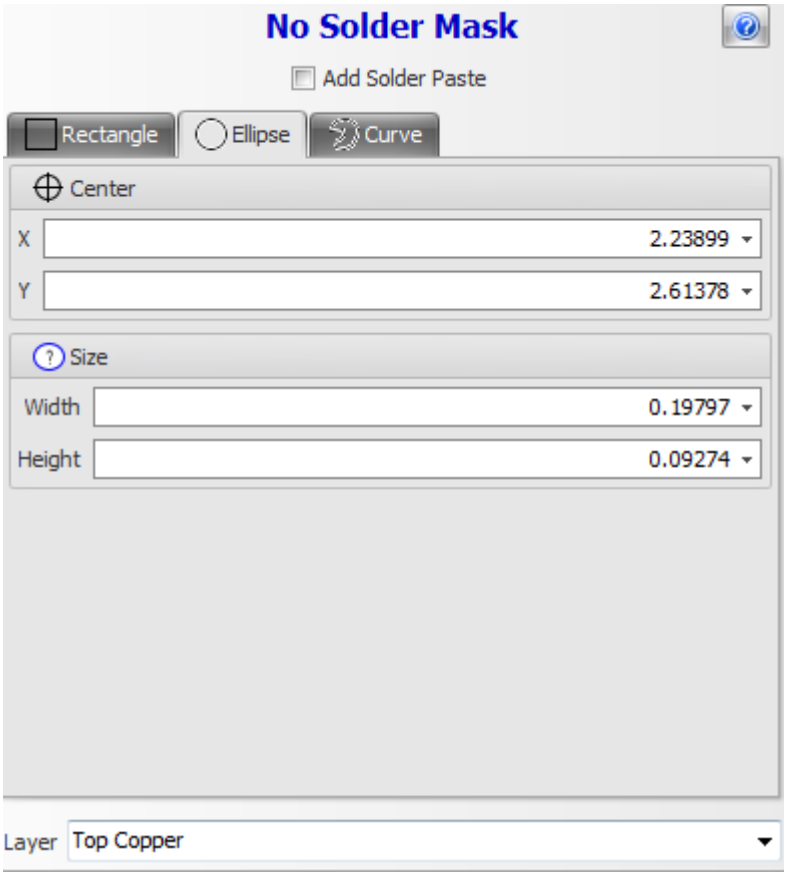
Drag to from left to right to increase the rounding of the corners.

Layer

Select the layer for the cutout

Solder mask cutouts can be elliptical or circular. To change to an elliptical/circular cutout click on the Ellipse tab in the cutouts properties in the properties panel.

To edit the parameters of a cutout, first select it. The properties panel will display the properties for the selected cutout.



X

The x coordinate of the center of the cutout

Y

The y coordinate of the center of the cutout

Width

The width of the cutout.

Height

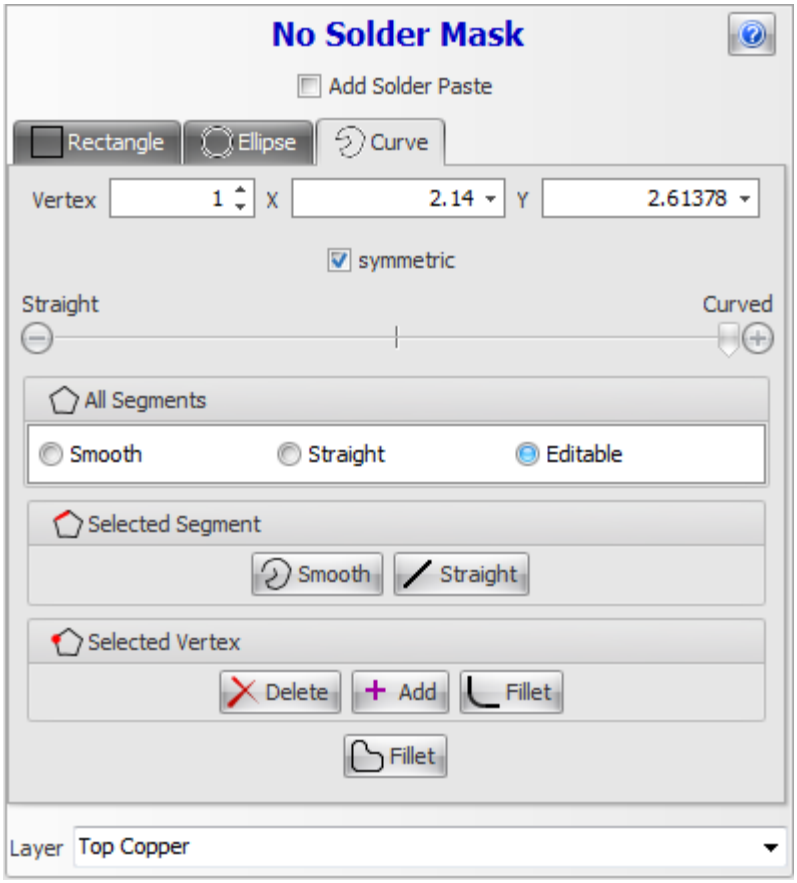
The height of the cutout.

Layer

Select the layer for the cutout

Solder mask cutouts can be polygonal with optional curved edges. To change to curved cutout click on the Curve tab in the cutouts properties in the properties panel.

To edit the parameters of a cutout, first select it. The properties panel will display the properties for the selected cutout.



Click the  button to show this help topic.

Vertex

This is the selected vertex/segment. The selected vertex is drawn as a solid red square. The selected segment id drawn in red.

X

The X coordinate of the selected vertex.

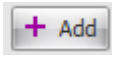
Y

The Y coordinate of the selected vertex.

Tension

Slide the 

slider to change the 'curvature' of the curve.

Click 

to add a vertex.

Click 

to delete the selected vertex.

All Segments

Check the Smooth button to smooth the entire curve.

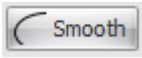
Check the Straight button to straighten the entire curve.

Check the Editable button to make the entire curve editable.

Segment

Click the 

button to straighten the selected segment.

Click the 

button to smooth the selected segment.

Layer

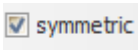
Use this drop-down to set the layer for the rectangle.

Click 

to fillet the polygon

Editable Curves

Drag any of the control points 
to change the shape.

Uncheck the 

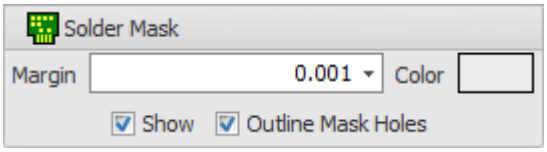
button to make the control points for the vertex independent of each other.

1.2.6.11.32.4 Setting The Solder Mask Margin

You set the margin or soldermask expansion in the Solder Mask properties in the Layers panel. The margin is the gap between the pad and the solder mask. A value of 2-4 mils is usual.

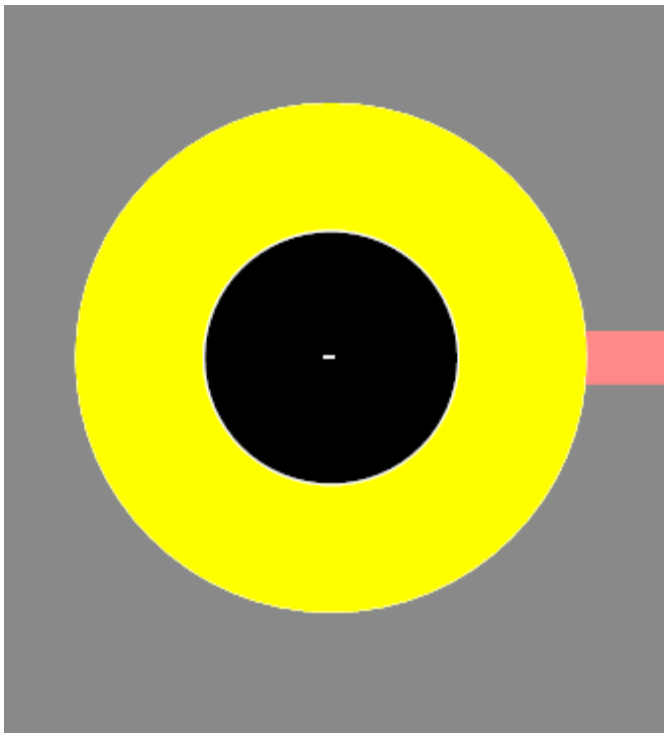
The reason you have a margin > 0 is that during manufacture the layers never match up perfectly. The actual hole in the solder mask may be slightly less than what you specified, and that hole is always be placed in a slightly different location than what you specified. If your solder mask expansion is too small, then these misalignments cause the solder mask to partially or completely overlap SMT pads and through-hole pads.

With some designs, you want to make sure that the solder mask never overlaps SMD pads. This is especially critical with QFN or LGA packages where the contacts of the parts do not stick out over the plastic molding or with parts that have a very fine pitch: Even small registration issues would cause the solderable areas of the pads to be reduced.

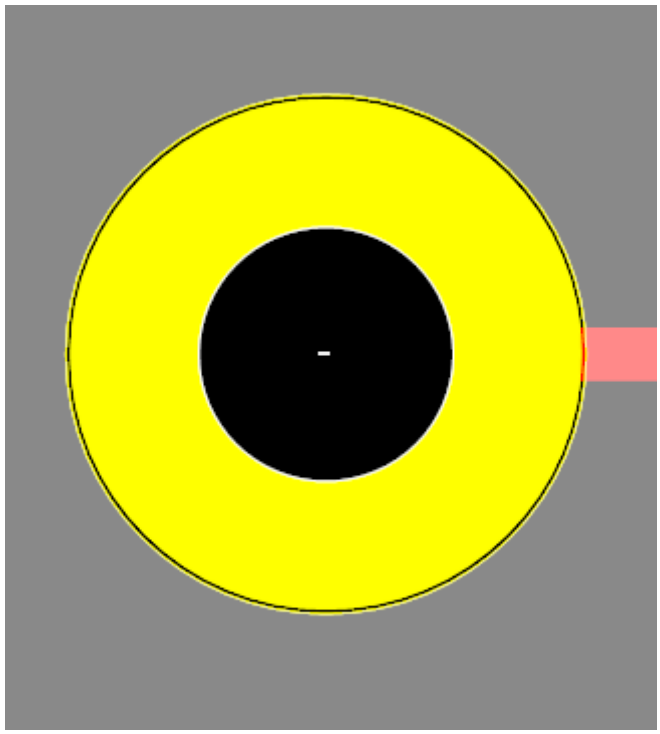


The solder mask properties

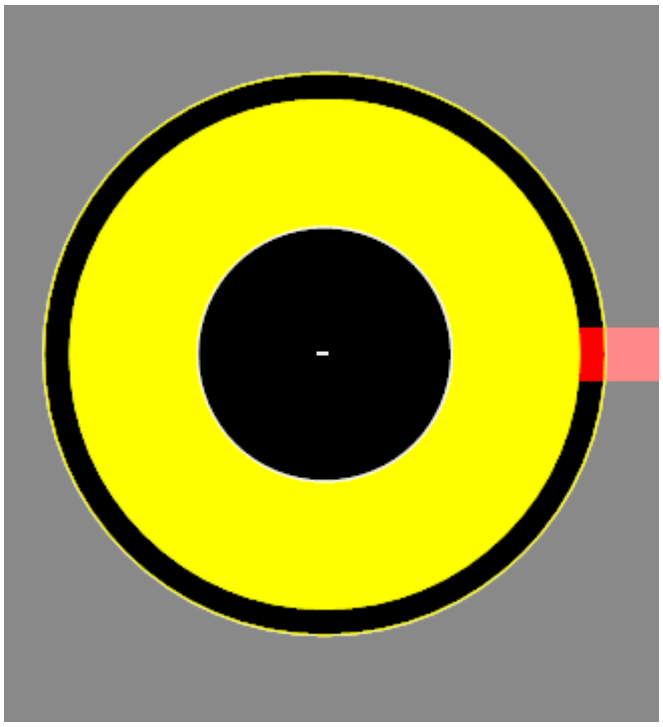
Below is a 100 Mil diameter pad with 3 different margins.



No margin

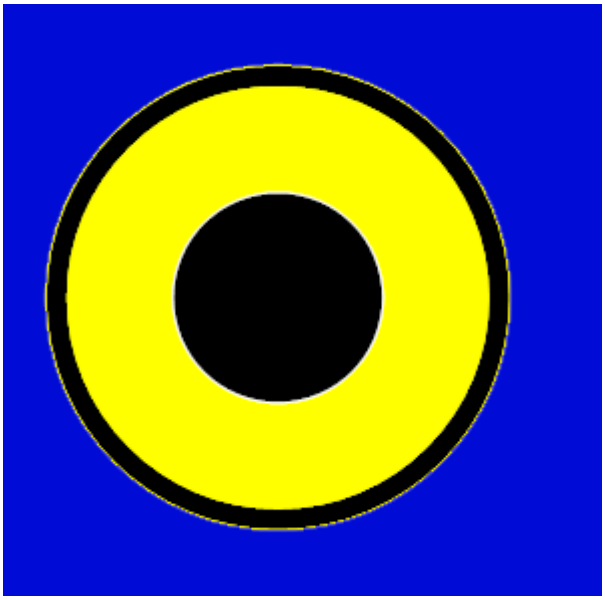


1 mil margin

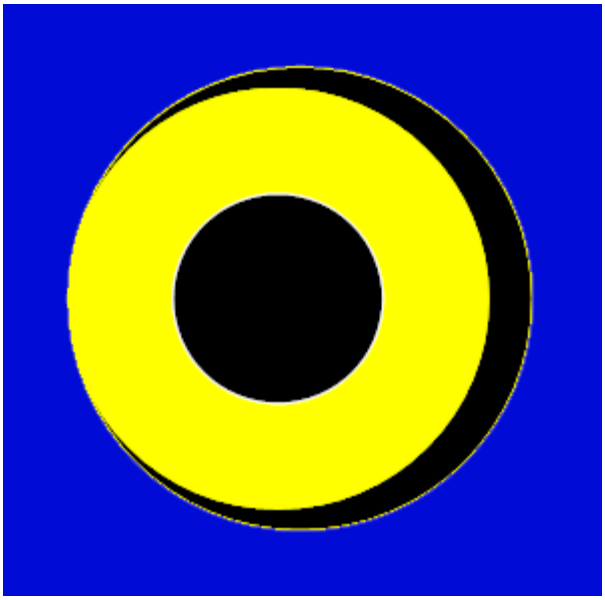


5 mil margin

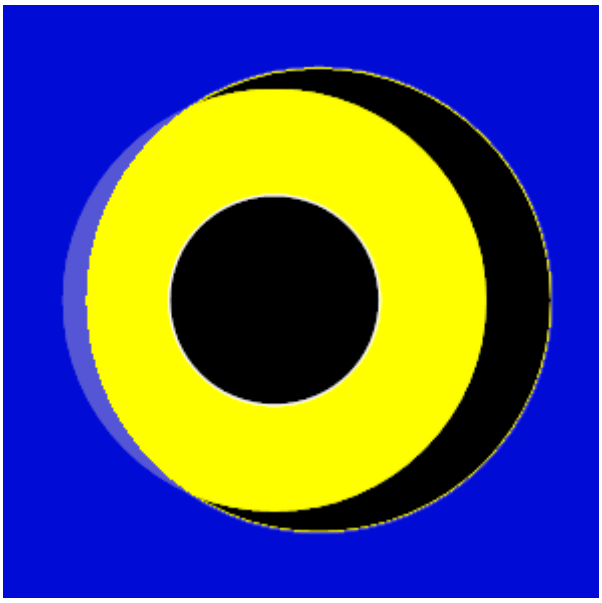
Below is what happens if the soldermask is misaligned. (Solder mask set to blue). So, the margin is to allow for manufacturing error and to ensure that all of the pad will have solder.



Perfect

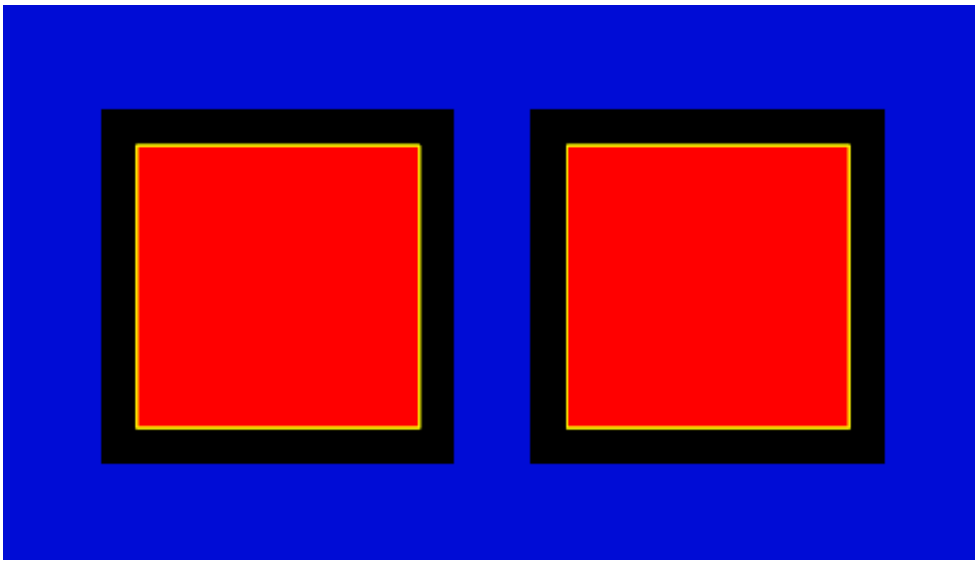


Misaligned but acceptable

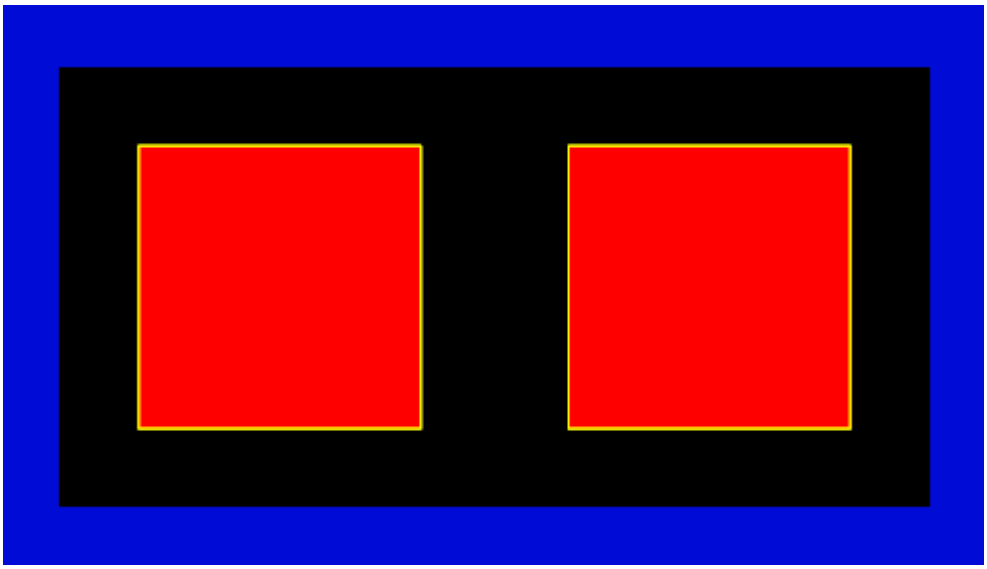


Misaligned and not acceptable

However, you do not want to make the margin too large because it could remove the solder mask between pad and risk solder bridging.



OK

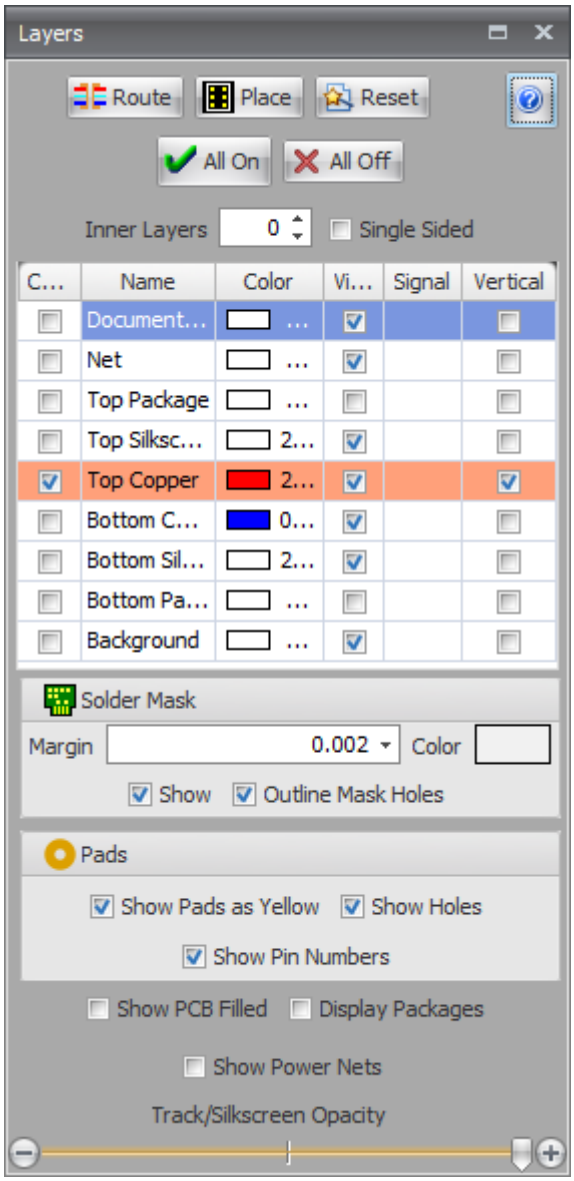


Margin too large. No solder mask between pads.

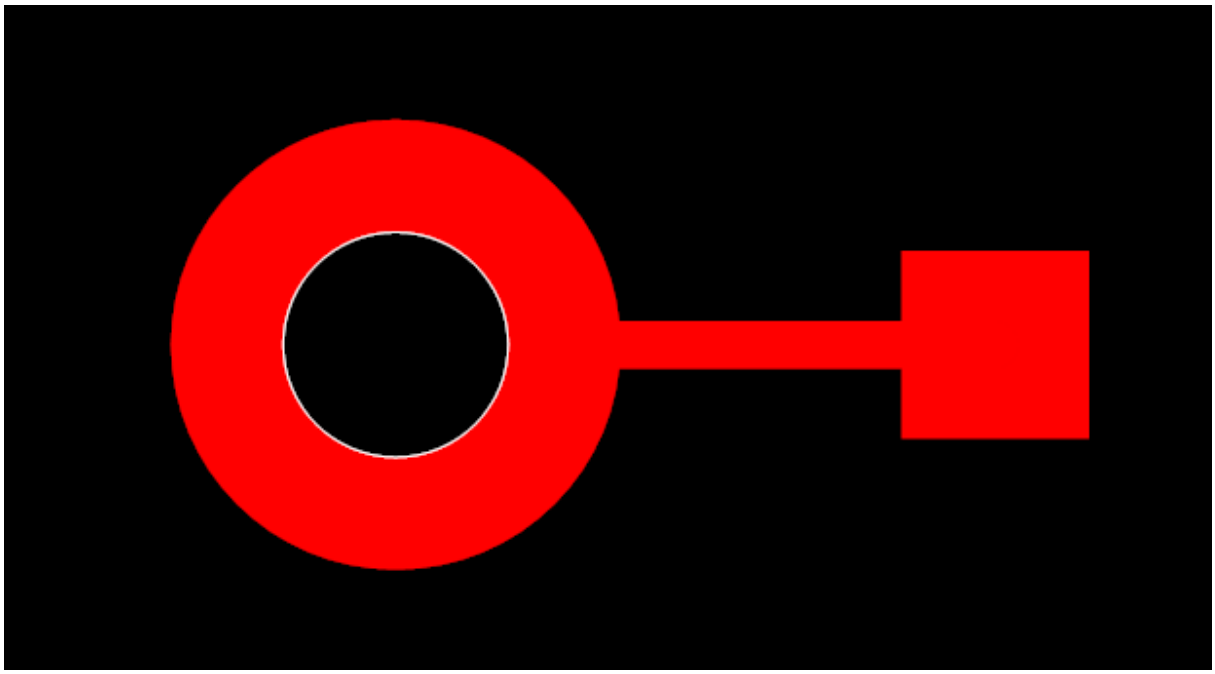
1.2.6.11.32.5 Solder Mask Viewing Options

Hiding and displaying the solder mask

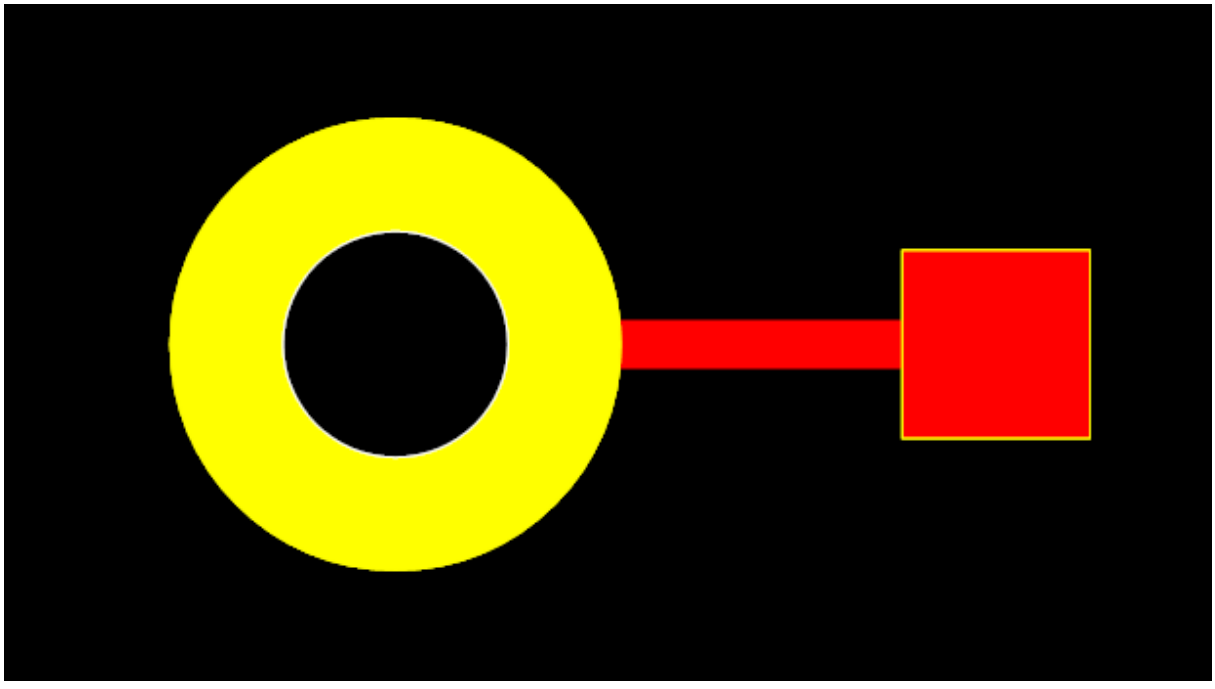
The solder mask is normally not visible in the PCB/Footprint view. However you can show/hide the mask by checking the Show in the solder mask properties in the layers panel.



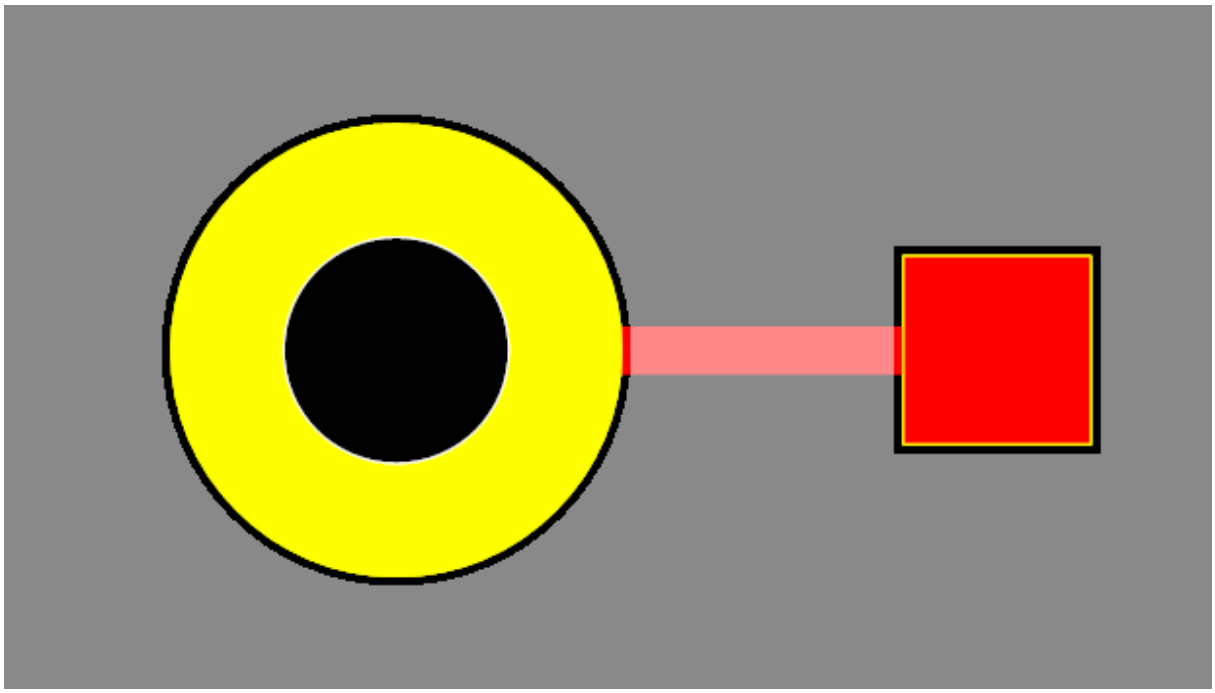
The Layers panel



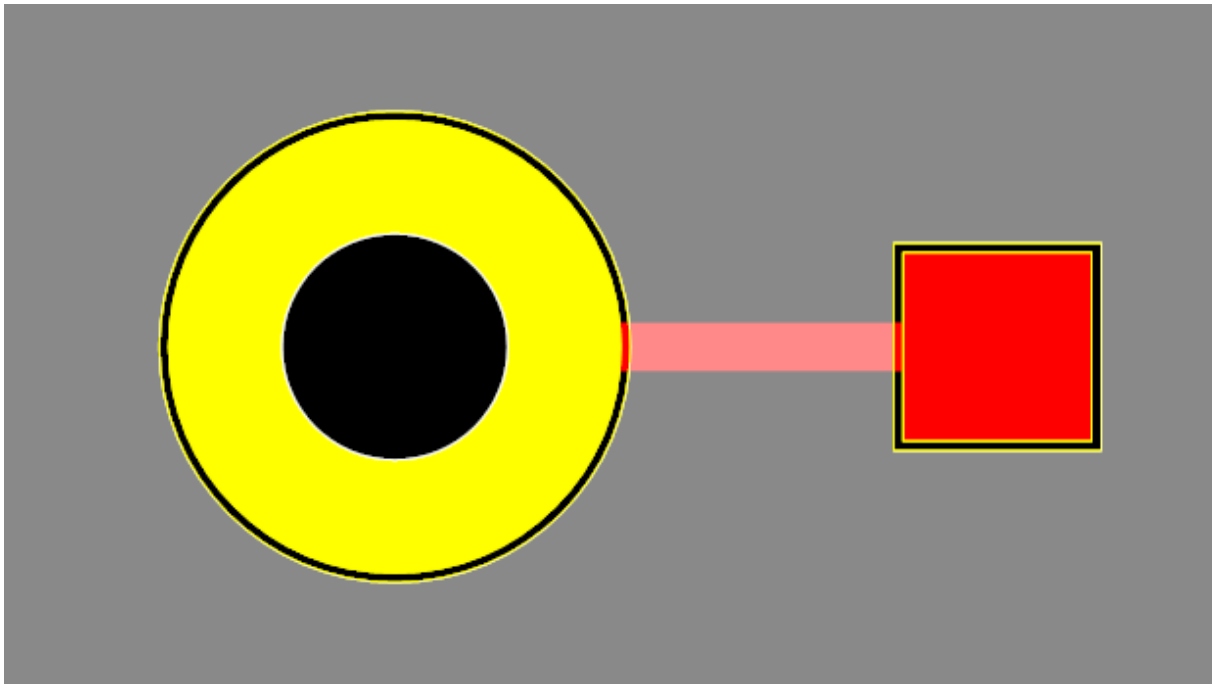
A TPH pad on the left connected to a SMT pad on the right



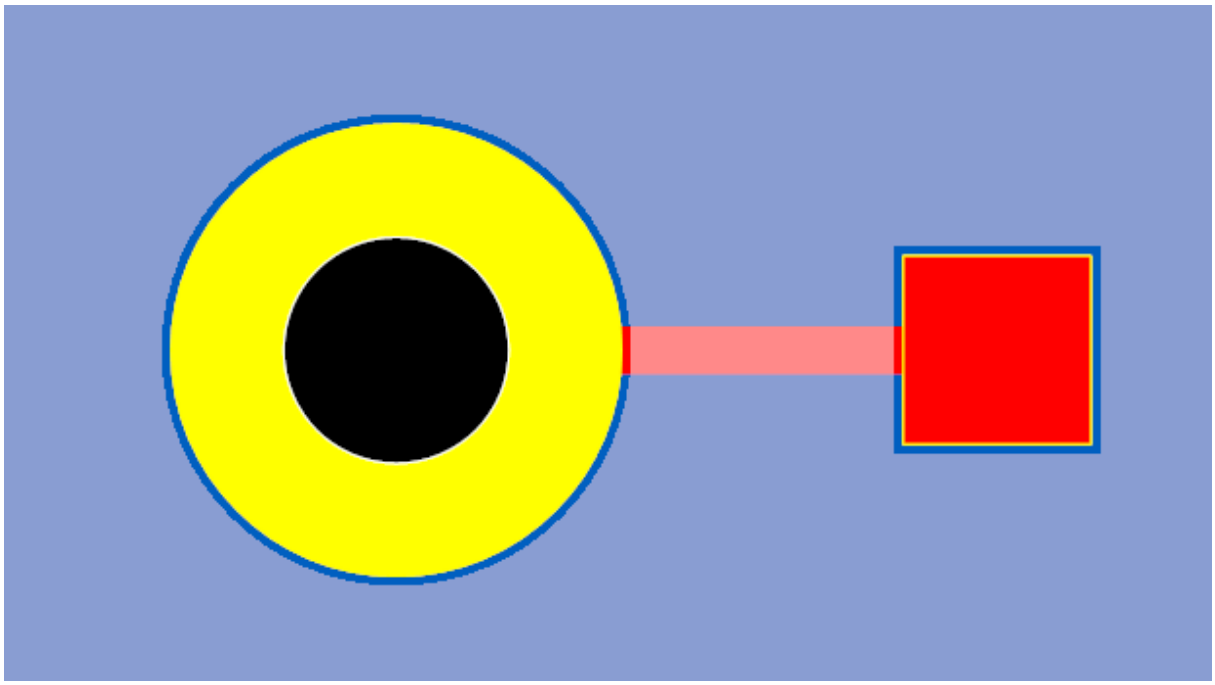
Show pads as yellow. SMT pads have yellow border



Solder mask visible. Color gray with semi-transparency



Outline mask holes checked. Yellow outline for cutouts



Show PCB filled

1.2.6.11.32.6 Adding Solder Paste to a Solder Mask Cutout

Normally a solder mask cutout will not have a solder paste applied.

To add solder paste, check 

1.2.6.11.32.7 Adding Solder Mask Cutout to Footprints

You can add solder mask cutouts to footprints. When the part if placed on the PCB, the cutouts in the footprint will be added to the PCB. Swapping the side of a footprint will swap the side for all the cutouts.

1.2.6.11.32.8 Pad Solder Mask Cutout

By default, pads automatically add cutouts to the top/bottom soldermask as required. The shape of the cutout is automatically adjusted to the pads shape.

You can turn off the automatic cutout generation by unchecking 

in the pads properties panel.

You would do this is you wanted to replace the automatic cutout with a custom one of your own design.

1.2.6.11.32.9 Solder Mask Cutouts by Combining Shapes

You can combine 2 or more shapes into a soldermask cutout. See combing shapes.

To combine graphics objects into a cutout first select them and then click the

Edit→Combine→ 

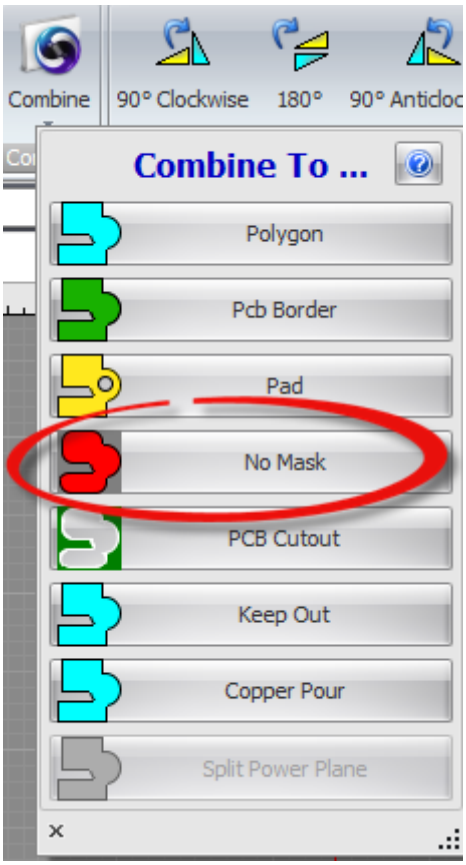
button. This button will only be enables if you have selected objects than can be combined. These include:

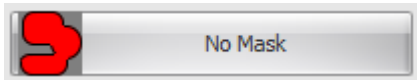
- Rectangles
- Ellipses
- Polygons
- Closed Curves
- Solder Mask Cutouts

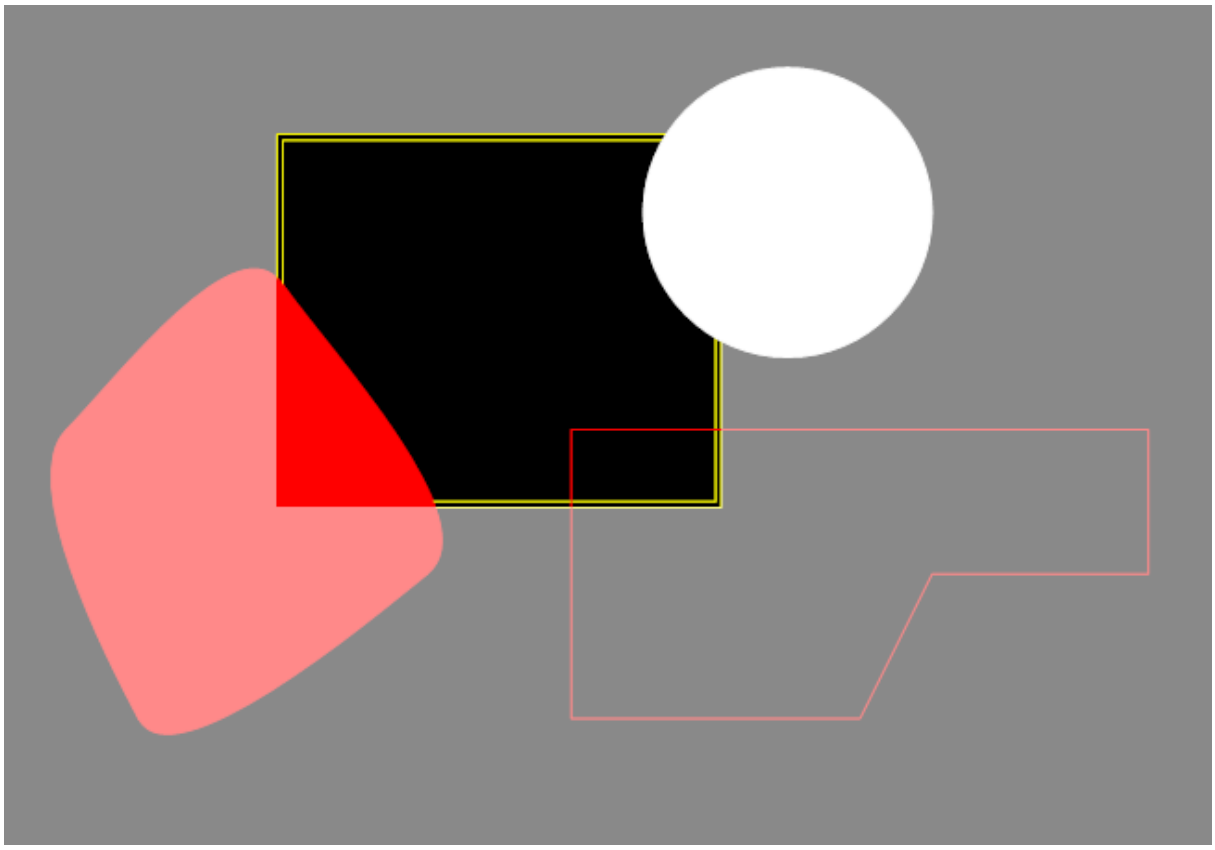
Select the objects to combine.

Click the Edit→Combine→ 

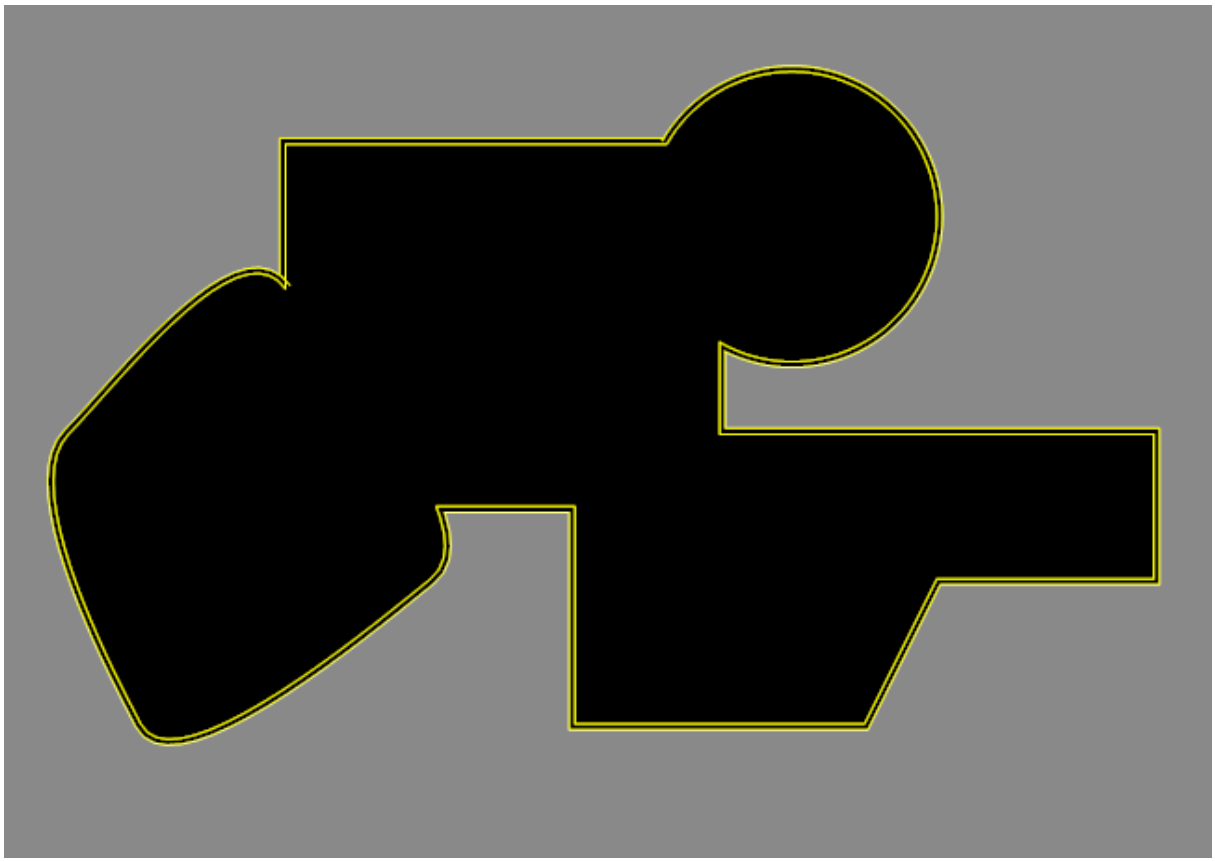
button



Click on  in the drop down.



Several shapes on different layer to combine together to make a cutout



The final cutout

1.2.6.11.33 Adding Solder Paste

AutoTRAX DEX can automatically add solder paste for you.

You can optionally and your own custom solder paste areas.

SMT pads

Surface mounted pads automatically add a solder paste area. You can turn off the automatically added solder paste using the pad's properties panel.

TPH pads

Thru-plated hole (TPH) pads do not have solder paste automatically added.

No mask cutouts

No mask cutouts normally do not have solder paste but you can optionally have automatically generated solder paste added to the no mask cutouts.

Custom solder paste areas

Finally, you can add your own custom solder paste areas by adding any graphics to either the top solder mask layer or the bottom solder mask layer.

Layers panel

in the layers panel you will see to solder mask areas. The first one is above the top copper layer and the second one is below the bottom copper layer.

Normally this solder mask layers are off but you can easily turn them on using the layers panel.

1.2.6.11.33.1 Solder Paste

When generating Gerber files, AutoTRAX DEX will automatically generate up to 2 additional file for the top and bottom layer solder paste stencils, but only if there are SMT pads on the top or bottom side. You can optionally add solder mask areas to no mask areas and also add your own custom no mask shapes. No file is generated for a side if there are Solder paste areas on that side.

No holes are output for TPH pads.

The 2 files with a (.gbr) extension are called:

- Top Solder Paste
- Bottom Solder Paste

Solder paste (or solder cream) is used to connect the leads of surface mount integrated chip packages to attachment points (pads/lands) in the circuit patterns on a printed circuit board. The paste is typically applied to the pads using a stencil to "print" the paste, although other methods, like dispensing from a tube, are also used.

A majority of the defects in circuit-board assembly are caused due to issues in the solder-paste printing process or due to defects in the solder paste. There are many different types of defects—too much solder, and the solder melts and connects too many wires (bridging), resulting in a short circuit. Insufficient amounts of paste result in incomplete circuits. Head-in-pillow defects, or incomplete coalescence of ball grid array (BGA) sphere and solder paste deposit, is a failure mode that has seen increased frequency since the transition to lead-free soldering. Often missed during inspection, a head-in-pillow (HIP) defect appears like a small head resting on a pillow with a visible separation in the solder joint at the interface of the BGA sphere and paste deposit. An electronics manufacturer needs experience with the printing process, specifically the paste characteristics, to avoid costly re-work on the

assemblies. The paste's physical characteristics, like viscosity and flux levels, need to be monitored periodically by performing in-house tests.

Solder paste is typically used in a screen-printing process, in which paste is deposited over a stainless steel or polyester mask to create the desired pattern on a printed circuit board. The paste may be dispensed pneumatically, by pin transfer (where a grid of pins is dipped in solder paste and then applied to the board), or by jet printing (where the paste is sprayed on the pads through nozzles, like an ink-jet printer).

As well as forming the solder joint itself, the paste carrier/flux must have sufficient tackiness to hold the components while the assembly passes through the various manufacturing processes, perhaps moved around the factory.

Printing is followed by preheating and reflow (melting).

The paste manufacturer will suggest a suitable reflow temperature profile to suit their individual paste; however, one can expend too much energy on this. The main requirement is a gentle rise in temperature to prevent explosive expansion ("solder balling"), yet activate the flux. Thereafter, the solder melts. The time in this area is known as Time Above Liquidus. A reasonably rapid cool-down period is required after this time.

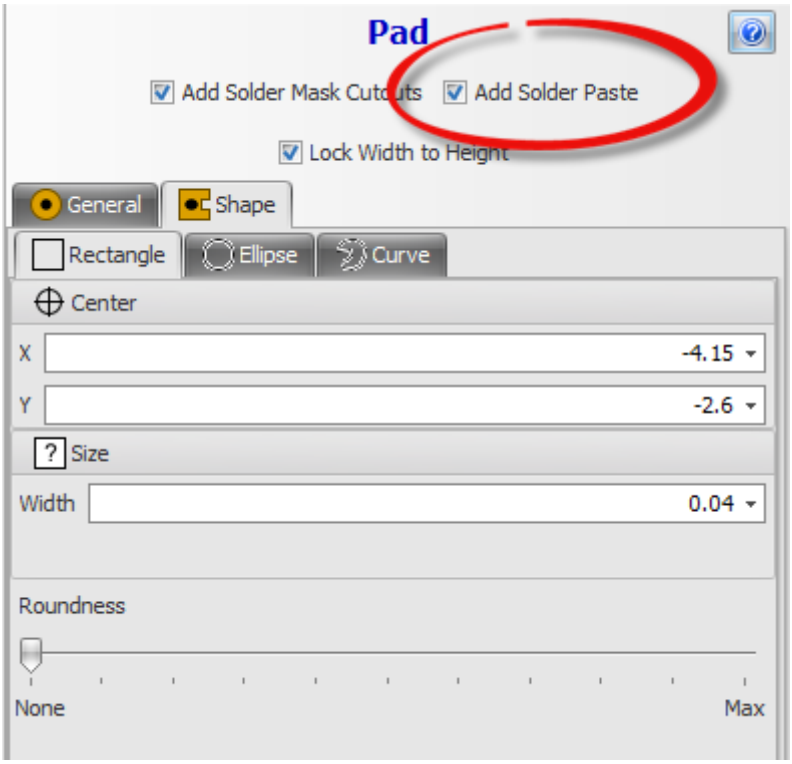
A good tin/lead solder joint will be shiny and relatively concave. This will be less so with lead-free solders.

As with all fluxes used in electronics, residues left behind may be harmful to the circuit, and standards (e.g., J-std, JIS, IPC) exist to measure the safety of the residues left behind.

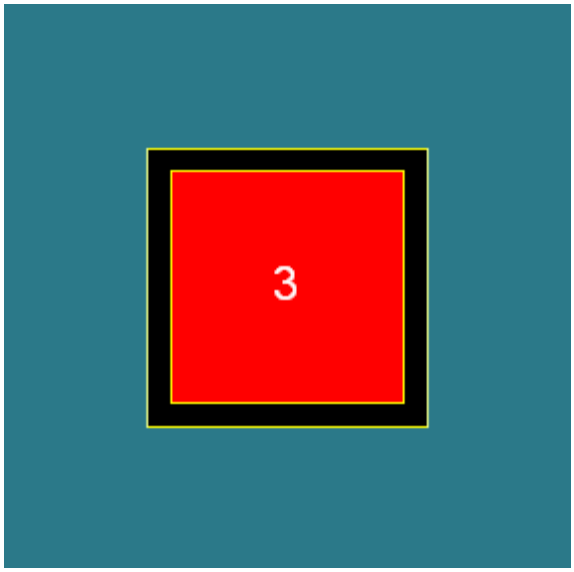
In most countries, "no-clean" solder pastes are the most common; in the United States, water-soluble pastes (which have compulsory cleaning requirements) are common.

1.2.6.11.33.2 Adding Solder paste to SMT pads

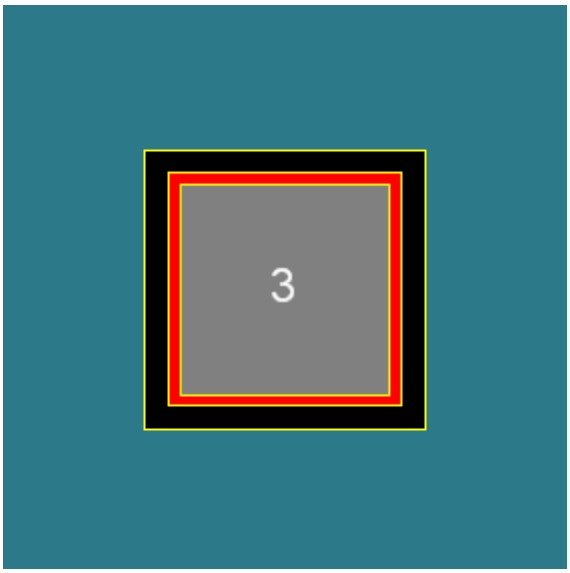
By default SMT pads automatically add solder paste. However, you can disable this using the pads properties panel, see below.



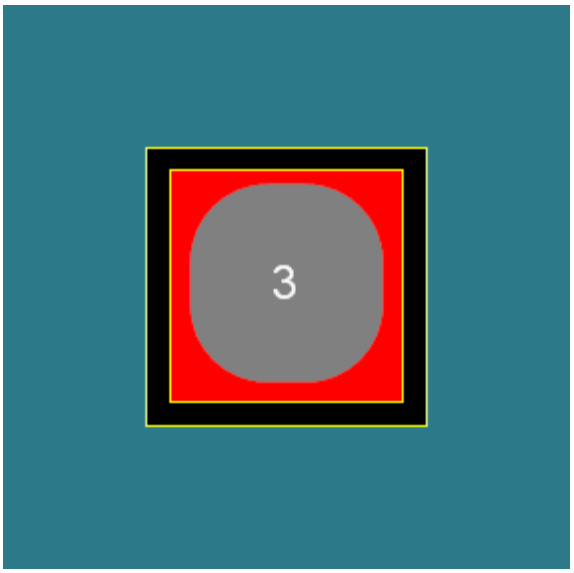
Enabling and disabling automatic solder paste generation



Without solder paste



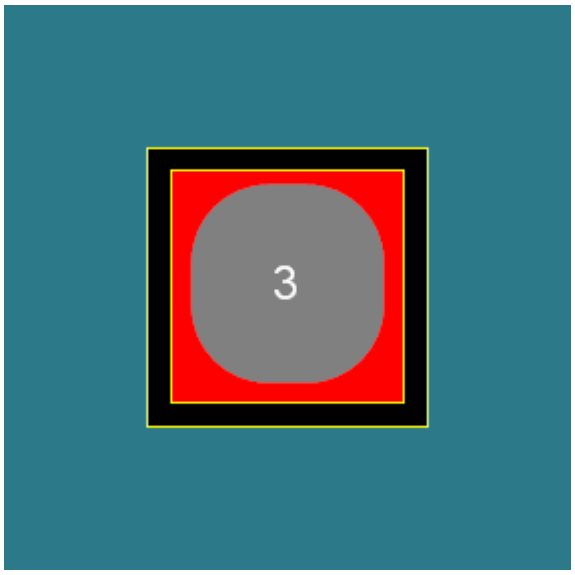
With auto-generated solder paste



Auto-generated solder paste disabled and custom shape added to solder paste layer

1.2.6.11.33.3 Adding Solder paste to no mask areas

To add solder paste to a no mask area first select either the top or bottom solder paste layer and add a graphic shape to that layer. The graphics shape will generate a solder paste area.

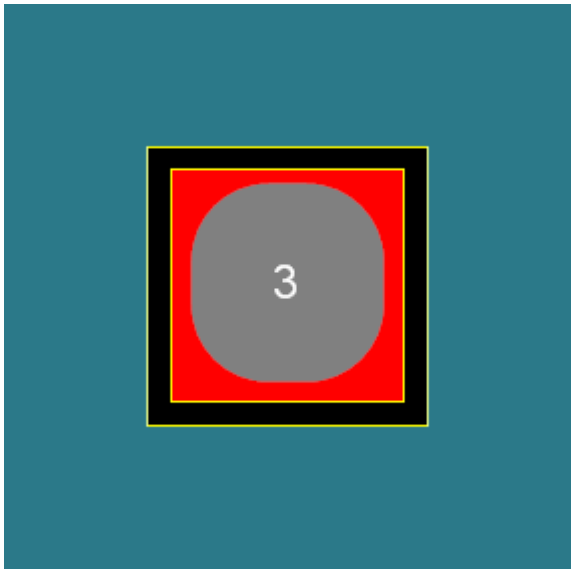


Auto-generated solder paste disabled and

custom shape added to solder paste layer

1.2.6.11.33.4 Adding custom solder paste areas

To add solder paste to a no mask area first select either the top or bottom solder paste layer and add a graphic shape to that layer. The graphics shape will generate a solder paste area.

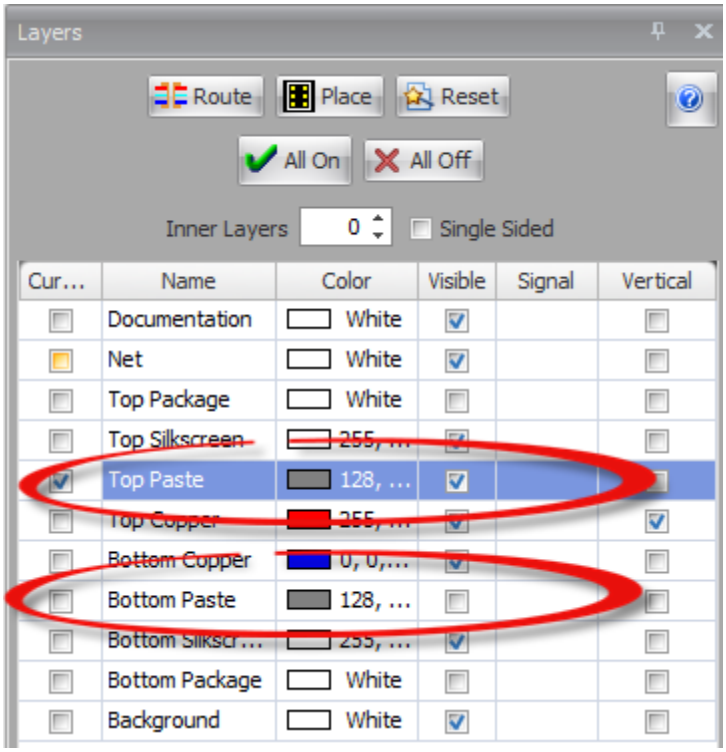


Auto-generated solder paste disabled and

custom shape added to solder paste layer

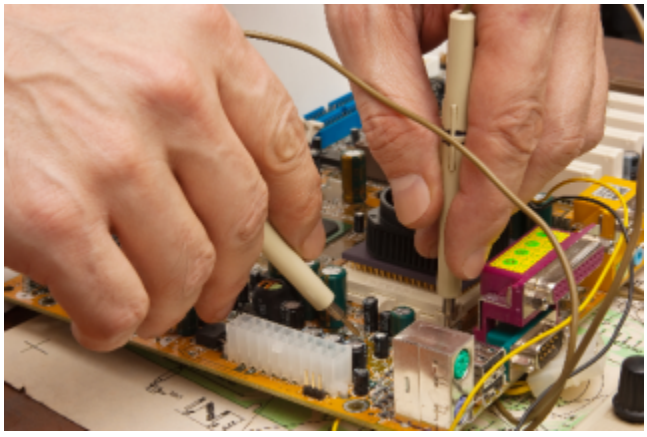
1.2.6.11.33.5 Viewing the solder paste

To view the solder paste set either the top and/or bottom solder paste layers visible.



The top and bottom solder paste layers

1.2.6.11.34 Checking Your PCB Design



To check your design use the Design Checker panel.

You can Export and Import the design rules.

The Design Rules Checker tests for:

- Unrouted Tracks

- Minimum Track Width
- Track Intersections
- Track to Track Clearance
- Track to Pad Clearance
- Track to Via Clearance
- Track to Hole Clearance
- Track to Cutout Clearance
- Track to Keepout Clearance
- Track to Border Clearance

- Minimum Pad hole Diameters
- Minimum Pad Annular Ring Size
- Pad to Pad Clearance
- Pad to Via Clearance
- Pad to Hole Clearance
- Pad to Cutout Clearance
- Pad to Border Clearance

- Minimum Via Hole Diameters
- Minimum Via Annular Ring Size
- Via to Via Clearance
- Via to Hole Clearance
- Via to Cutout Clearance
- Via to Keepout Clearance
- Via to Border Clearance

- Hole to Hole Clearance
- Hole to Cutout Clearance

- Hole to Border Clearance

- Cutout to Cutout Clearance
- Cutout to Border Clearance

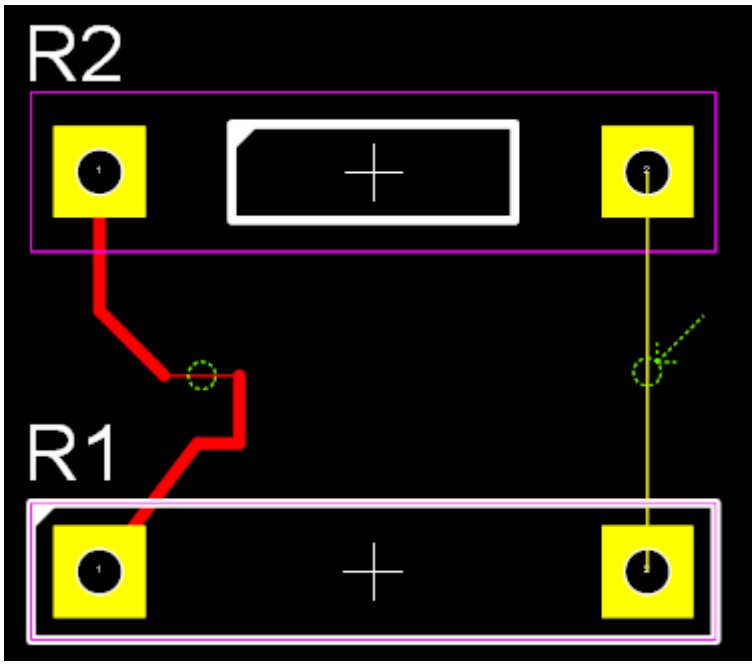
- Courtyard to Courtyard Clearance
- Courtyard to Holes Clearance
- Courtyard to Cutouts Clearance
- Courtyard to Border Clearance

- Silkscreen to Pad Clearance

Each test can be optionally turned on or off. All tests are on by default.

1.2.6.11.34.1 Unrouted Tracks

All unrouted tracked segments will be marked as unrouted.

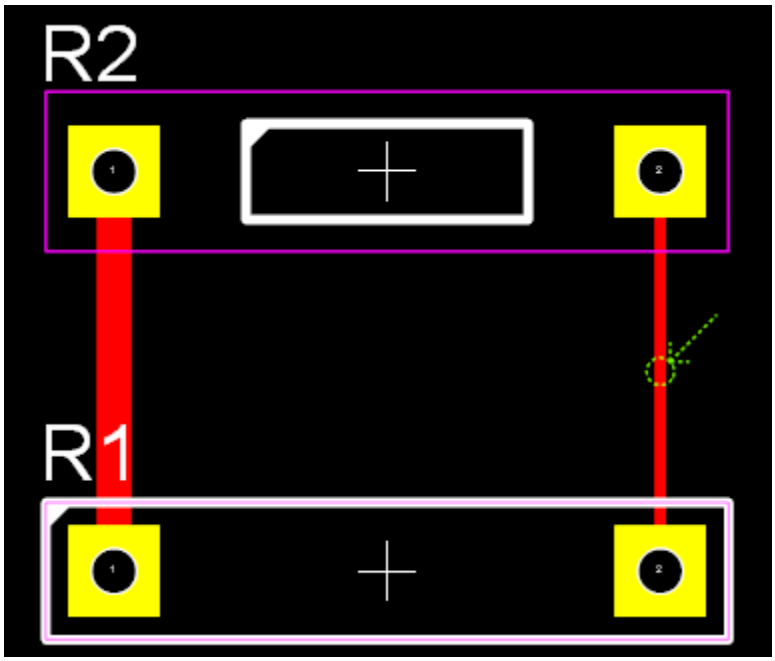


Unrouted track - marked with green circle (and arrow if error selected)

1.2.6.11.34.2 Minimum Track Width

All routed track segments whose width is less than the minimum track width will be marked.

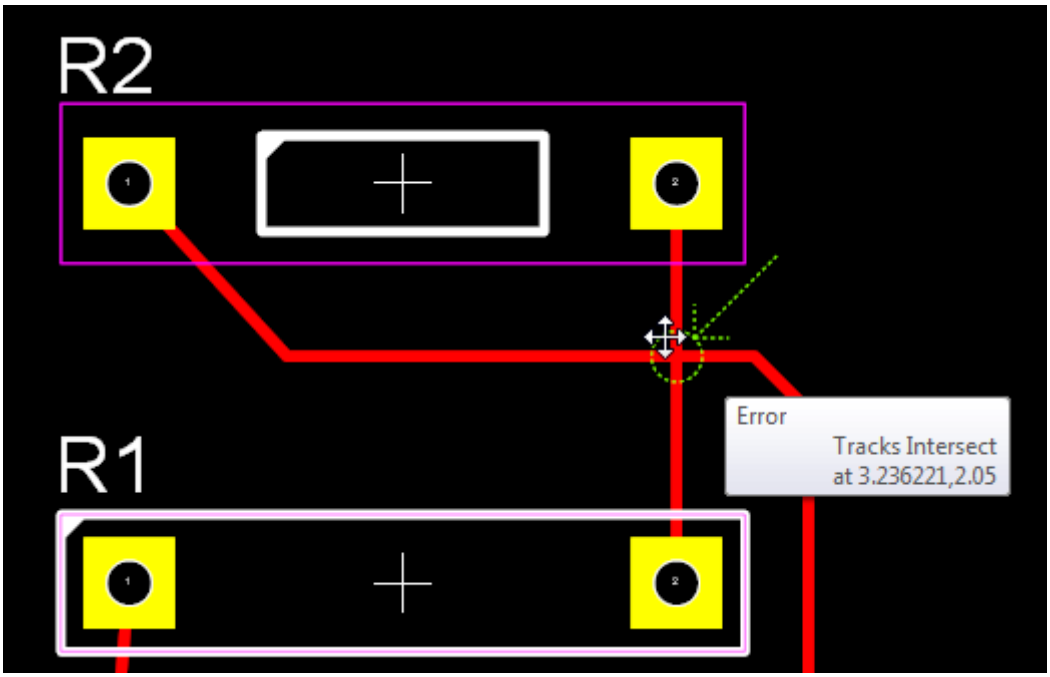
Move the mouse over the marker to display a tooltip showing the width and the minimum track width.



Track width minimum track width - marked with green circle (and arrow if error selected)

1.2.6.11.34.3 Track Intersections

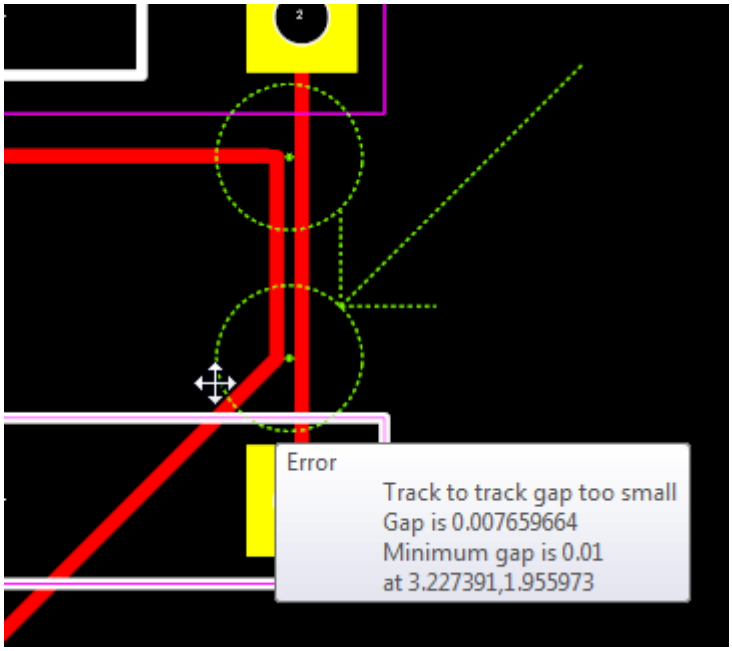
All tracks that intersect another track segments will be marked as shown below.



Intersecting tracks - marked with green circle (and arrow if error selected)

1.2.6.11.34.4 Track to Track Clearance

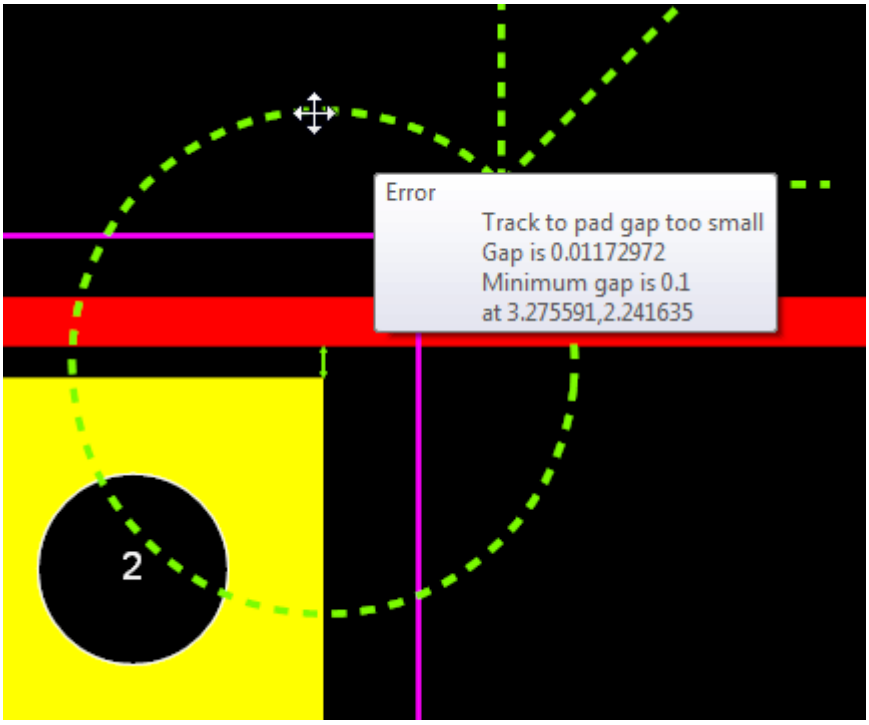
All track to track spacings that are less than the minimum track to track clearance will be marked as shown below.



Tracks too close together - marked with green circle (and arrow if error selected)

1.2.6.11.34.5 Track to Pad Clearance

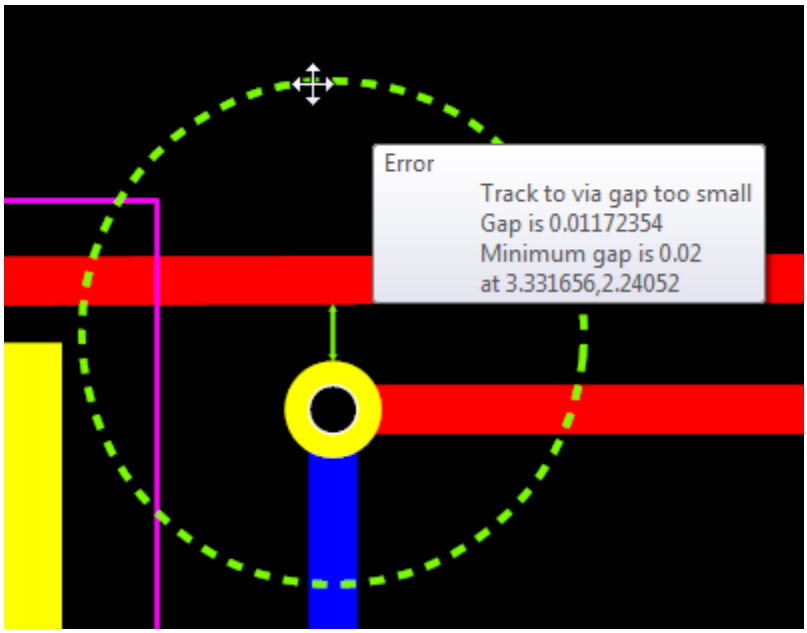
All track to pad spacings that are less than the minimum track to pad clearance will be marked as shown below.



Track too close to a pad - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.6 Track to Via Clearance

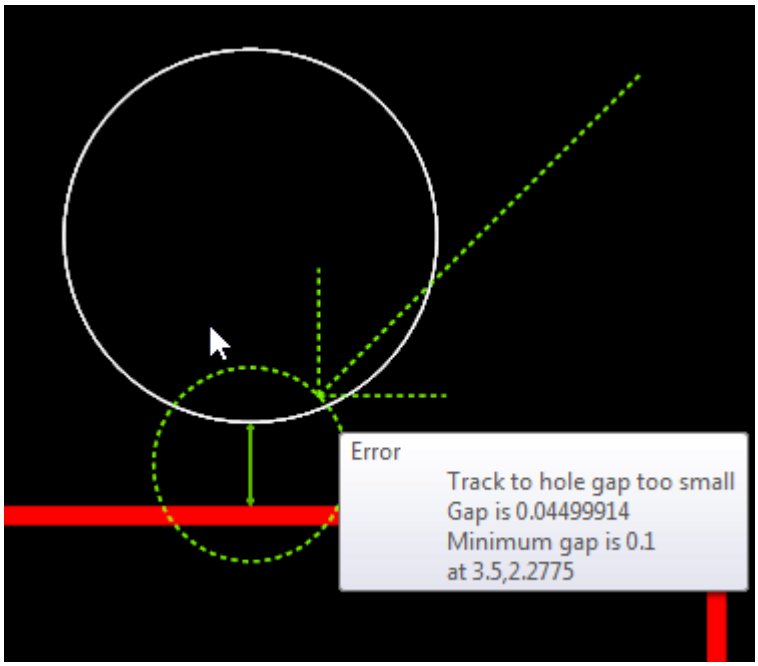
All track to via spacings that are less than the minimum track to via pad clearance will be marked as shown below.



Track too close to a via - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.7 Track to Hole Clearance

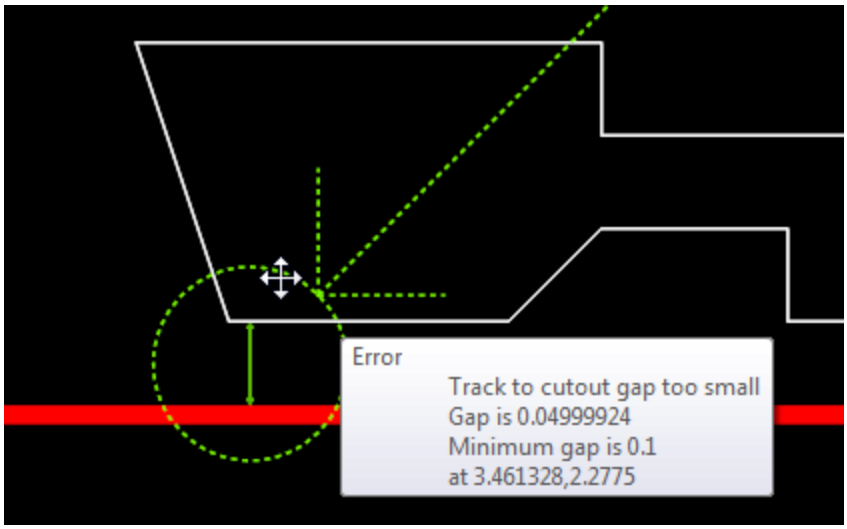
All track to PCB hole (not pad or via holes) spacings that are less than the minimum track to hole pad clearance will be marked as shown below.



Track too close to a hole - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.8 Track to Cutout Clearance

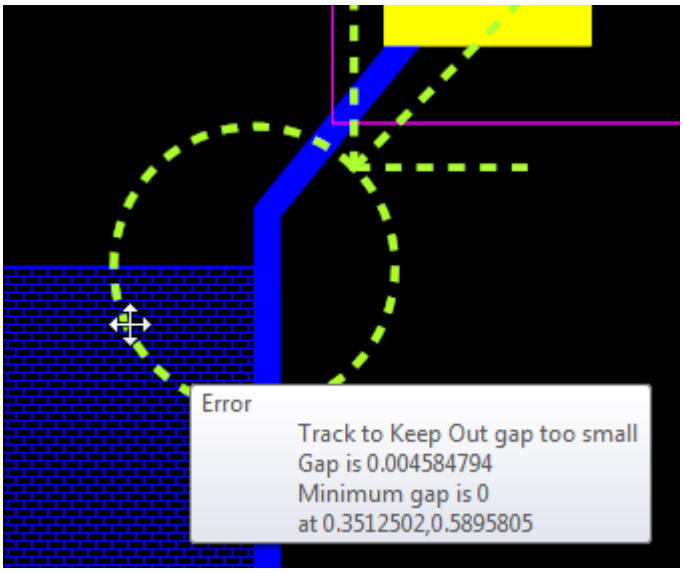
All track to PCB cutout spacings that are less than the minimum track to cutout clearance will be marked as shown below.



Track too close to a cutout - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.9 Track to Keepout Clearance

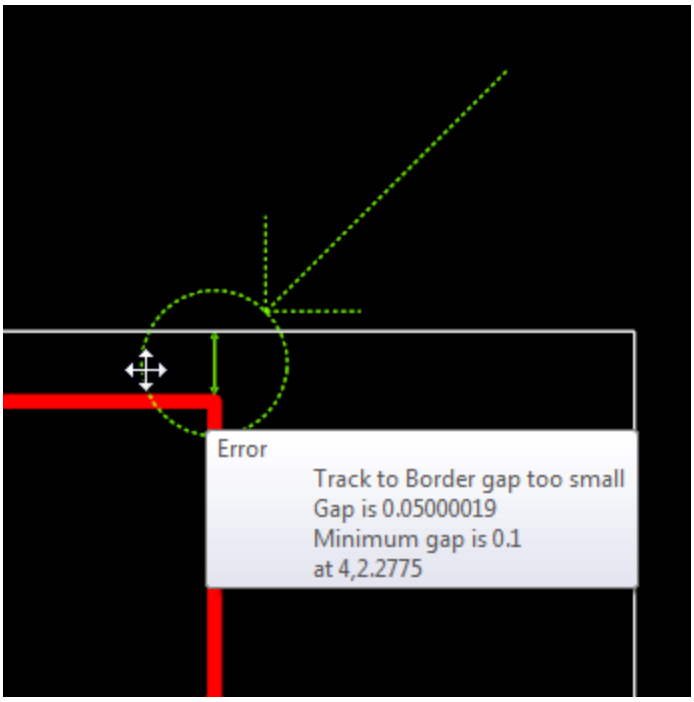
All track to PCB keepout spacings that are less than the minimum track to cutout keepout will be marked as shown below.



Track too close to a keepout - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.10 Track to Border Clearance

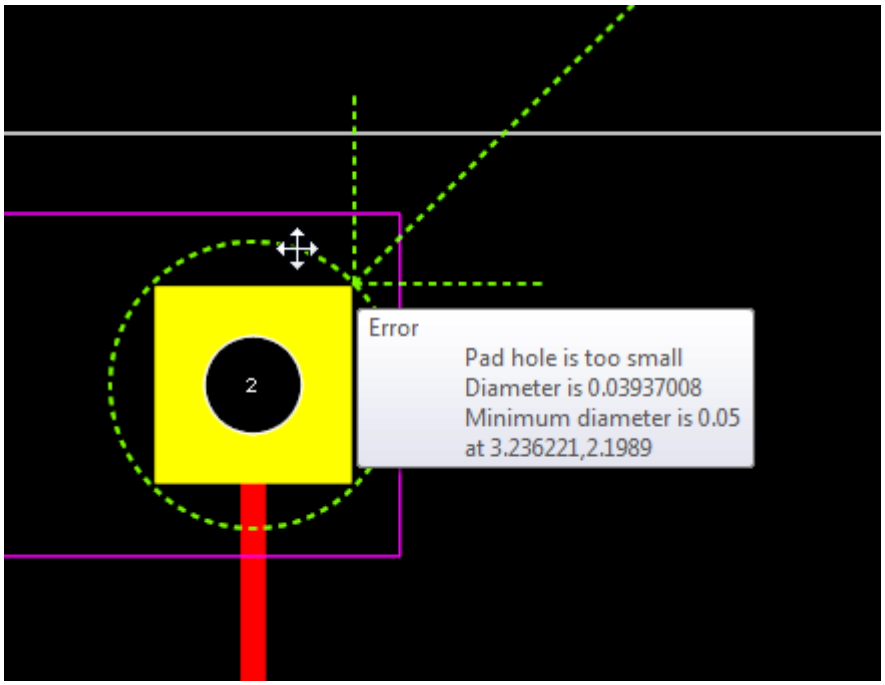
All track to PCB border spacings that are less than the minimum track to PCB border clearance will be marked as shown below.



Track too close to the PCB border - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.11 Minimum Pad hole Diameters

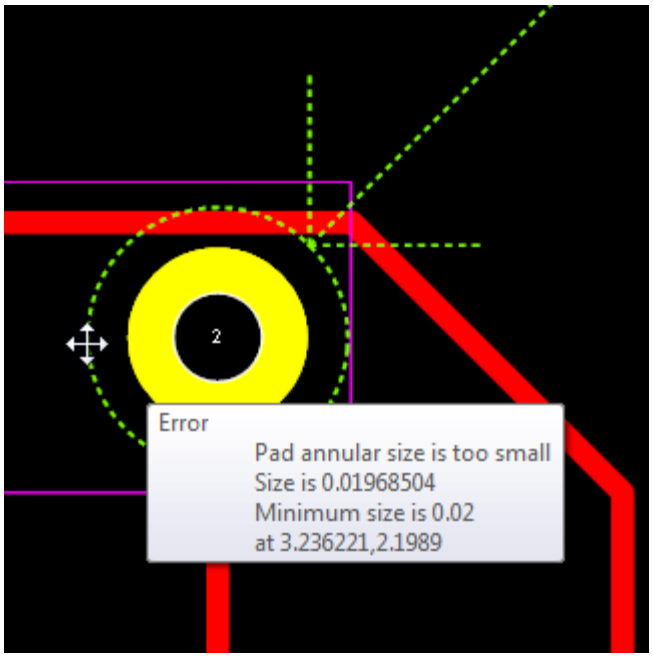
All TPH pads whose holes are less than the minimum hole diameter will be marked as below.



Pad hole too small - marked with green circle (and arrow if error selected)

1.2.6.11.34.12 Minimum Pad Annular Ring Size

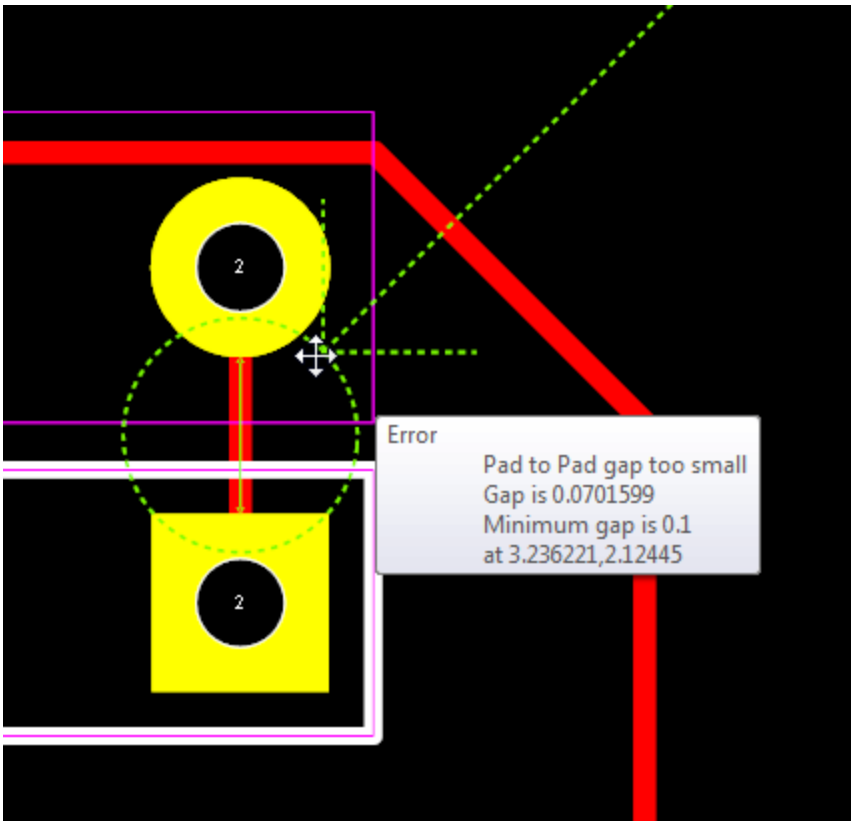
All pads whose copper dimensions are less than the minimum pad annular ring will be marked as below.



Pad annular ring too small - marked with green circle (and arrow if error selected)

1.2.6.11.34.13 Pad to Pad Clearance

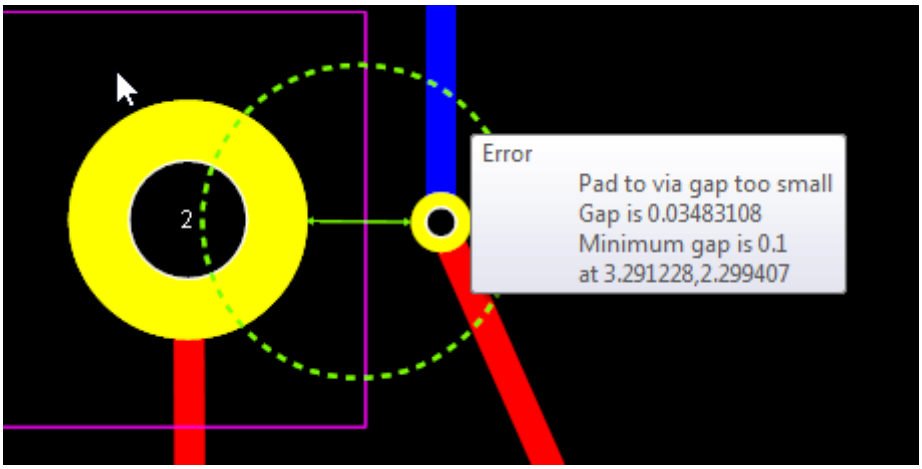
All pad to pad spacings that are less than the minimum pad to pad clearance will be marked as shown below.



Pads too close together - marked with green circle (and arrow if error selected)

1.2.6.11.34.14 Pad to Via Clearance

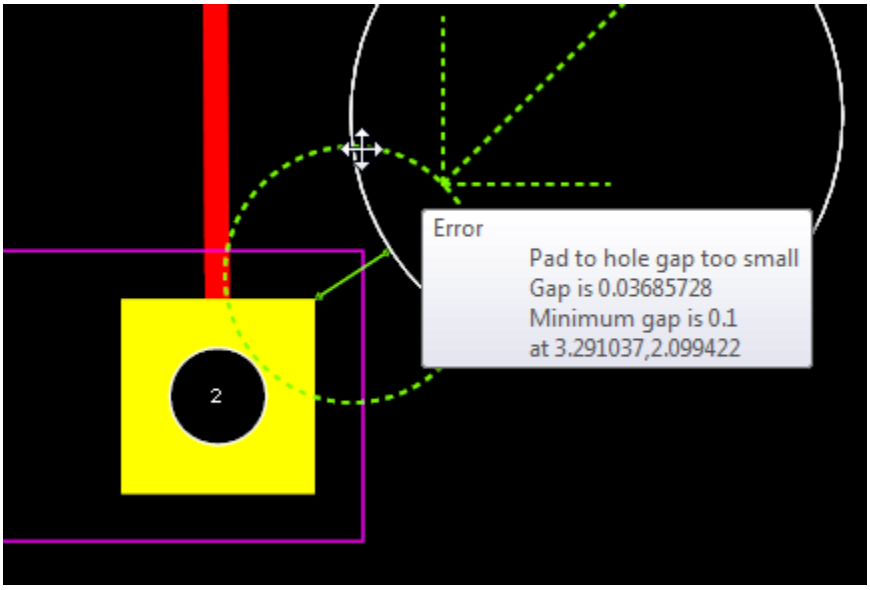
All pad to via spacings that are less than the minimum pad to via clearance will be marked as shown below.



Pads too close to a via - marked with green circle (and arrow if error selected)

1.2.6.11.34.15 Pad to Hole Clearance

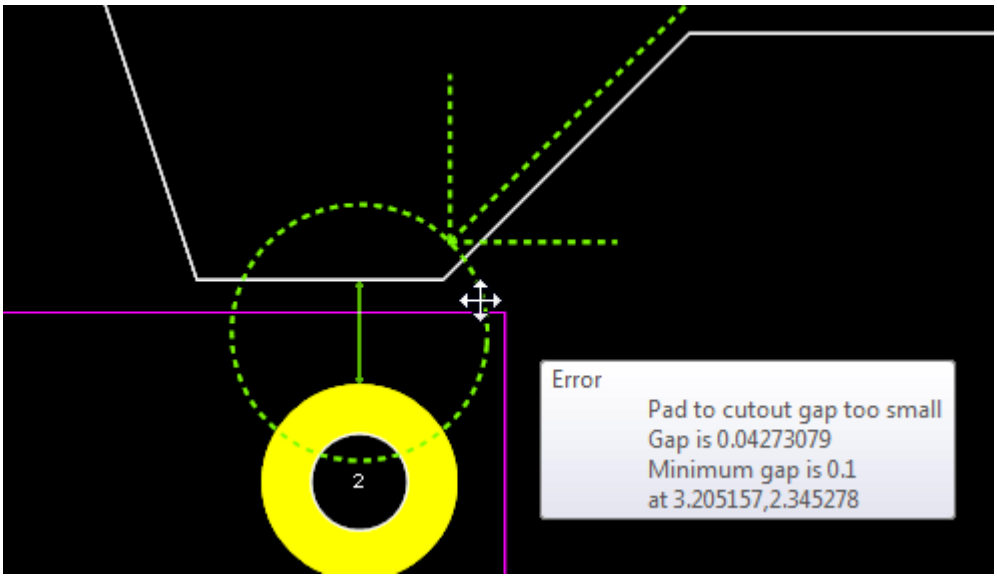
All pad to hole (not pad or via holes) spacings that are less than the minimum pad to hole clearance will be marked as shown below.



Pads too close to a hole - marked with green circle (and arrow if error selected)

1.2.6.11.34.16 Pad to Cutout Clearance

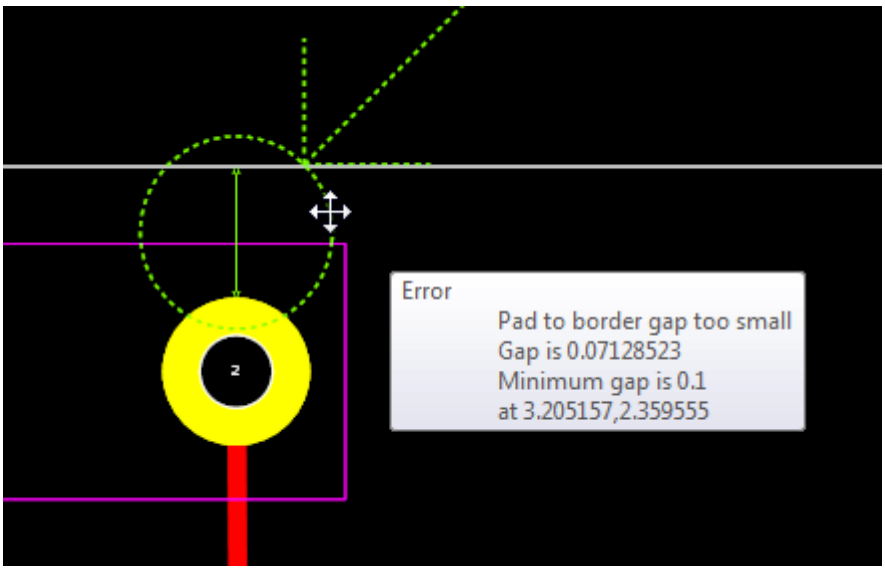
All pad to cutout spacings that are less than the minimum pad to cutout clearance will be marked as shown below.



Pads too close to a cutout - marked with green circle (and arrow if error selected)

1.2.6.11.34.17 Pad to Border Clearance

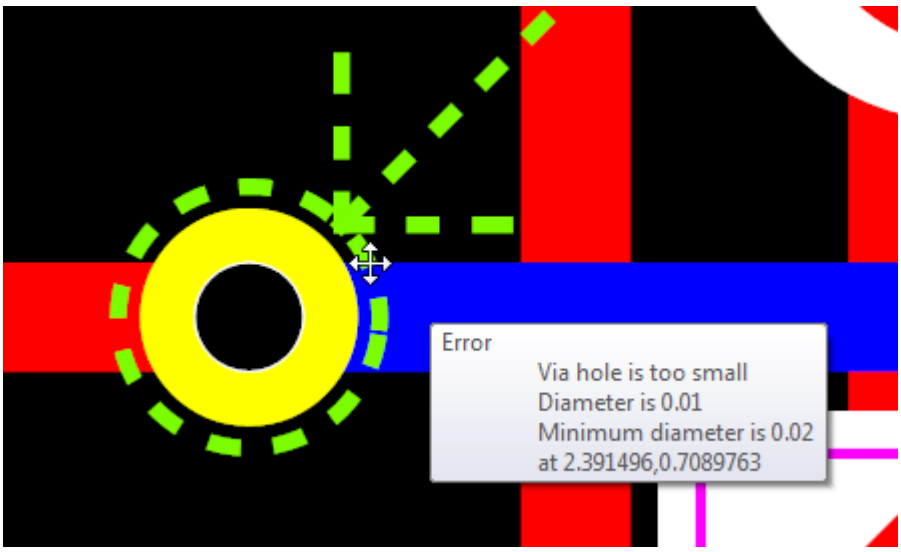
All pad to PCB border spacings that are less than the minimum pad to PCB border clearance will be marked as shown below.



Pad too close to the PCB border - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.18 Minimum Via Hole Diameters

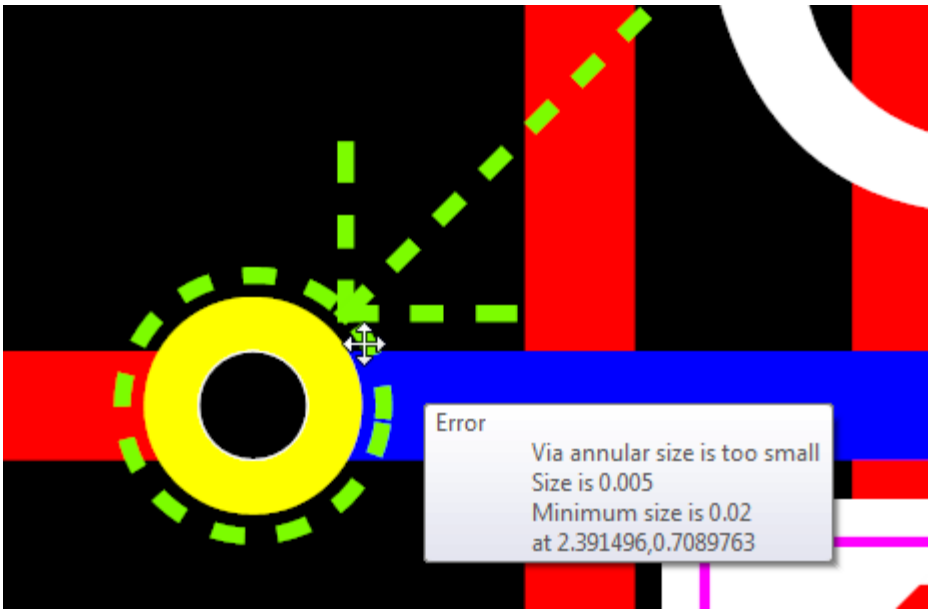
All vias whose holes are less than the minimum via hole diameter will be marked as below.



Via hole too small - marked with green circle (and arrow if error selected)

1.2.6.11.34.19 Minimum Via Annular Ring Size

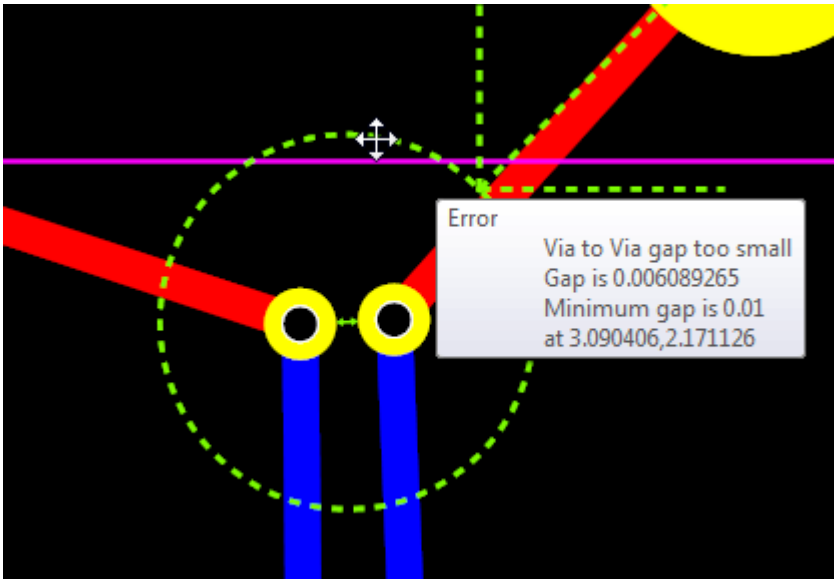
All vias whose copper dimensions are less than the minimum via annular ring will be marked as below.



Via annular ring too small - marked with green circle (and arrow if error selected)

1.2.6.11.34.20 Via to Via Clearance

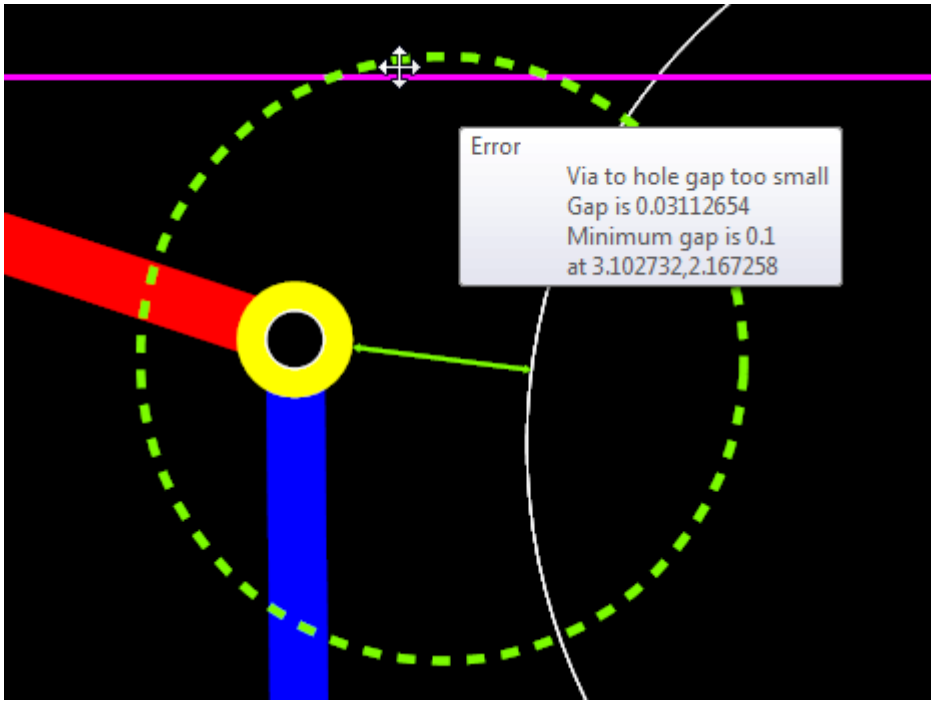
.All via to via spacings that are less than the minimum via to via clearance will be marked as shown below.



Via too close together - marked with green circle (and arrow if error selected)

1.2.6.11.34.21 Via to Hole Clearance

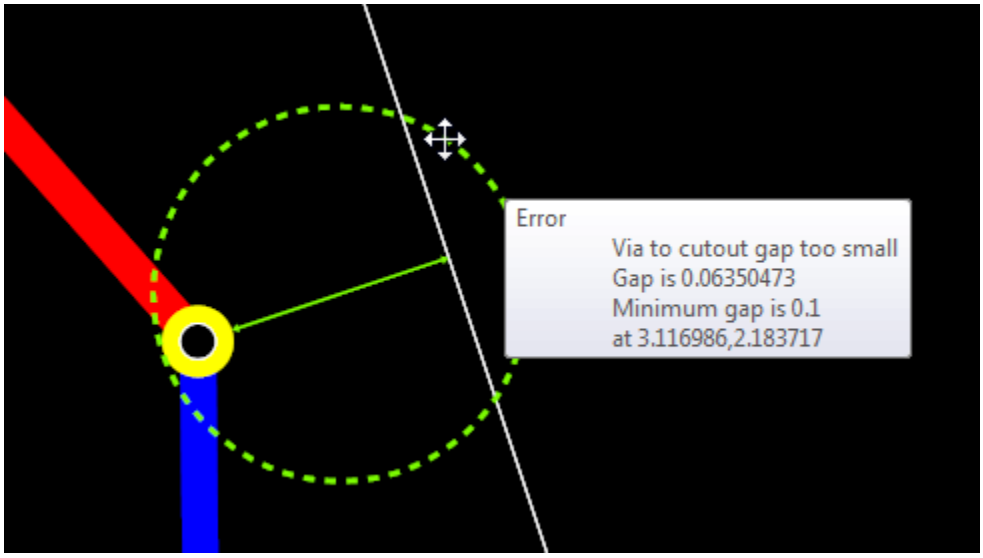
All via to hole (not pad or via holes) spacings that are less than the minimum via to hole clearance will be marked as shown below.



Via too close to a hole - marked with green circle (and arrow if error selected)

1.2.6.11.34.22 Via to Cutout Clearance

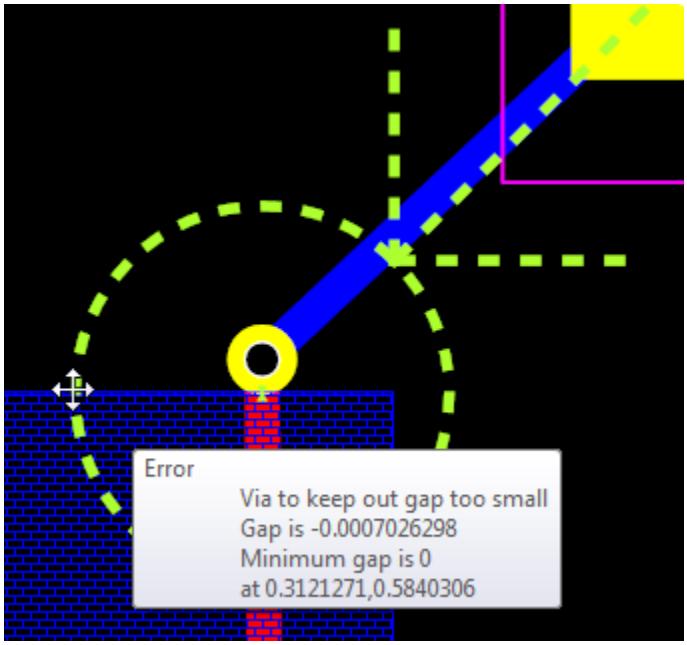
All via to cutout spacings that are less than the minimum via to cutout clearance will be marked as shown below.



Via too close to a cutout - marked with green circle (and arrow if error selected)

1.2.6.11.34.23 Via to Keep Out Clearance

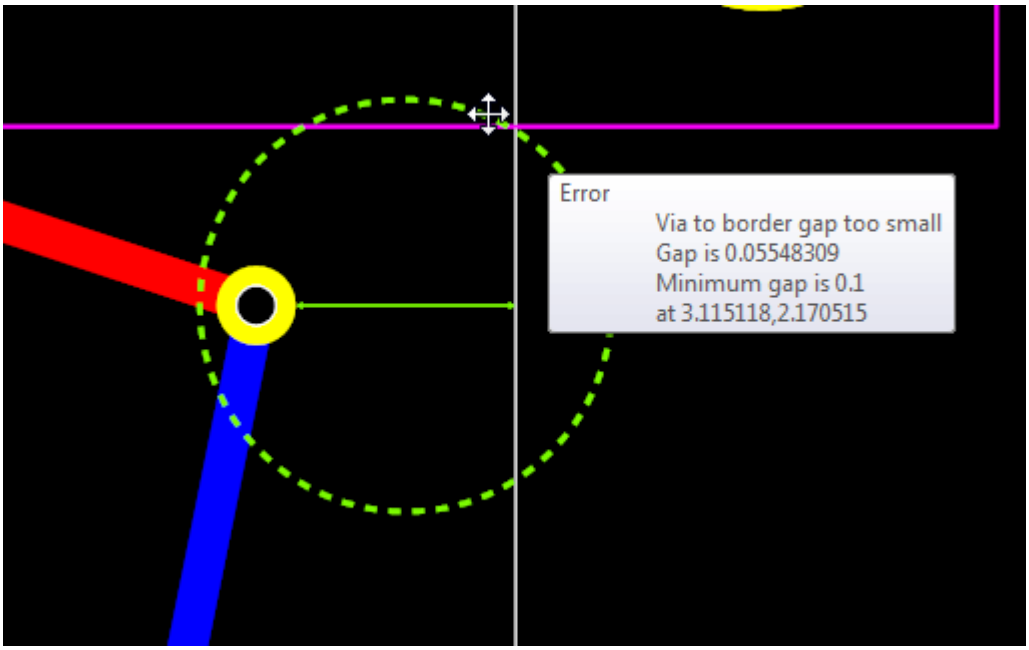
All via to keepout spacings that are less than the minimum via to cutout clearance will be marked as shown below.



Via too close to a keepout - marked with green circle (and arrow if error selected)

1.2.6.11.34.24 Via to Border Clearance

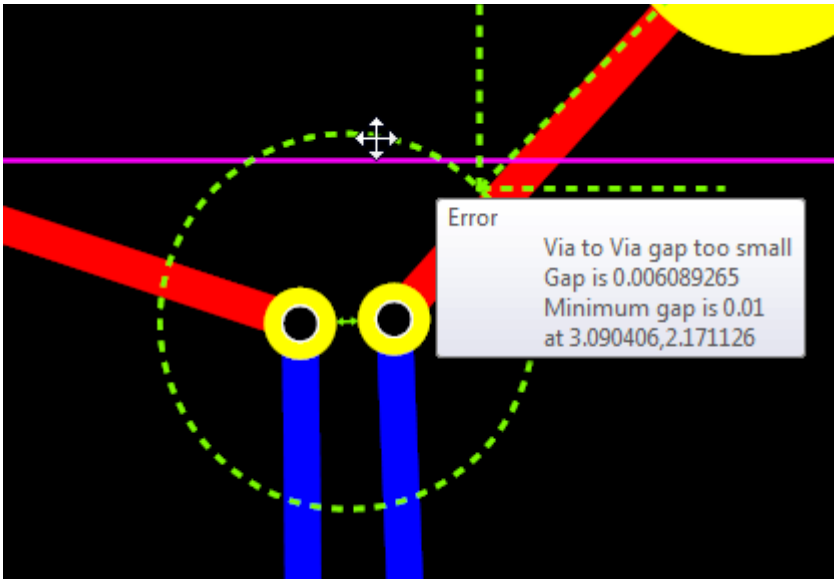
All via to PCB border spacings that are less than the minimum via to PCB border clearance will be marked as shown below.



Via too close to the PCB border - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.25 Hole to Hole Clearance

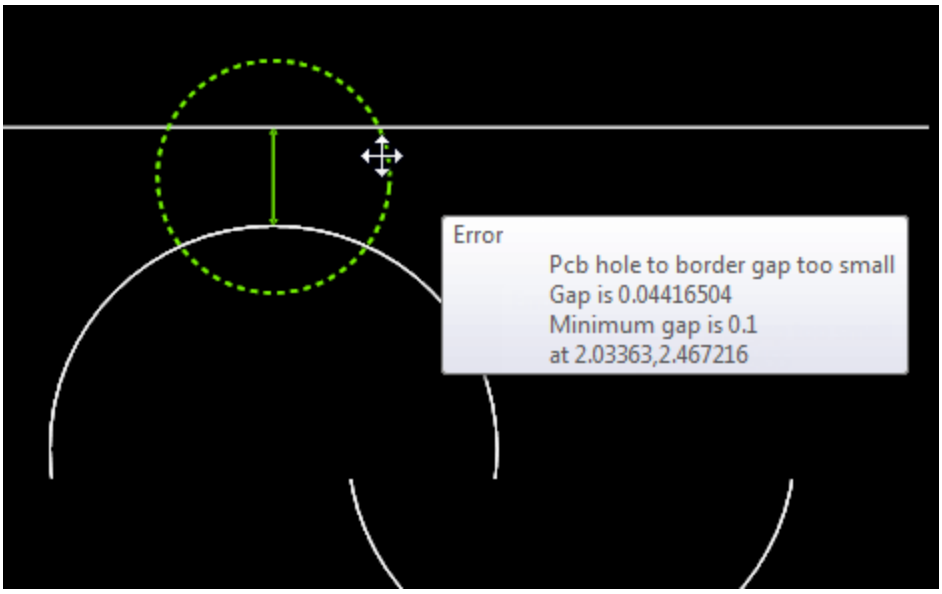
.All hole to hole spacing that are less than the minimum hole to hole clearance will be marked as shown below.



Hole too close together - marked with green circle (and arrow if error selected)

1.2.6.11.34.26 Hole to Cutout Clearance

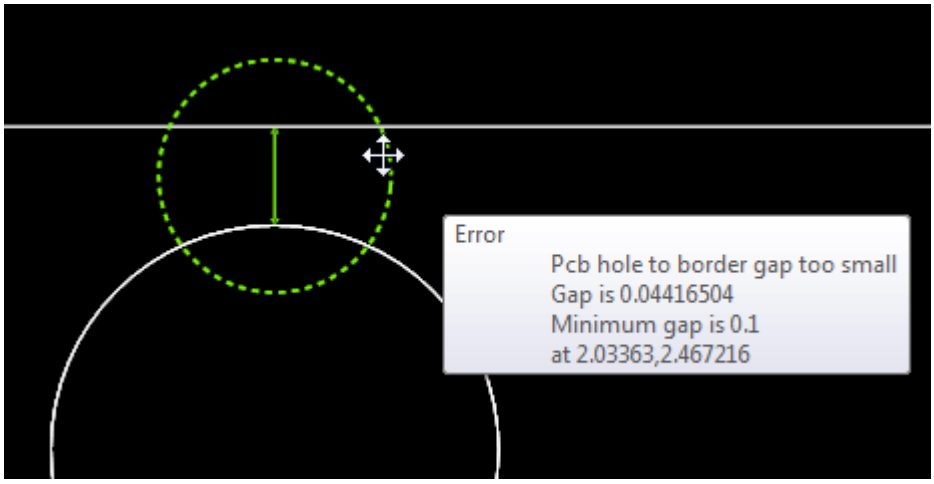
All hole to cutout spacings that are less than the minimum hole to cutout clearance will be marked as shown below.



Hole too close to a cutout - marked with green circle (and arrow if error selected)

1.2.6.11.34.27 Hole to Border Clearance

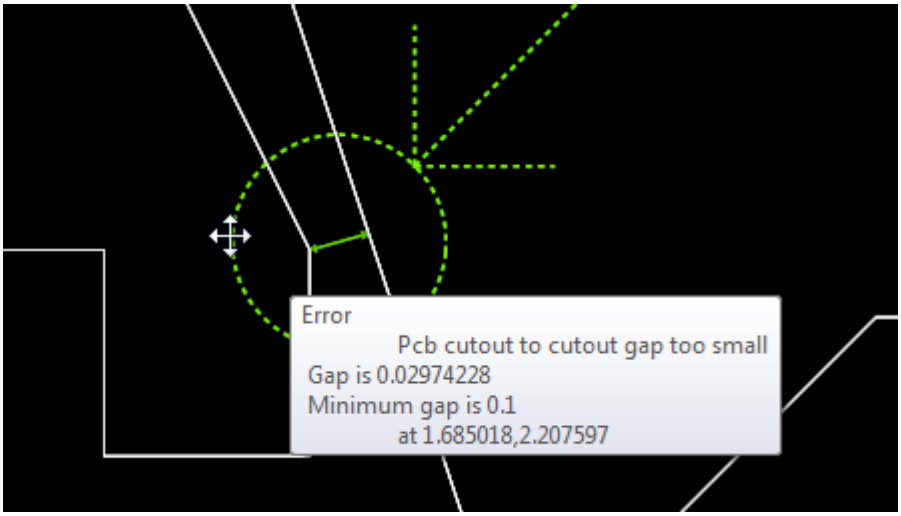
All hole to PCB border spacings that are less than the minimum hole to PCB border clearance will be marked as shown below.



Hole too close to the PCB border - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.28 Cutout to Cutout Clearance

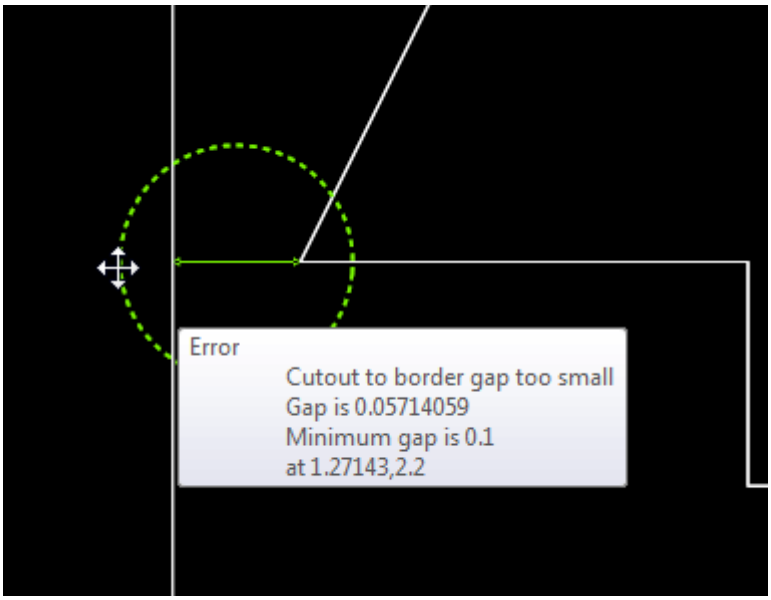
.All cutout to cutout spacings that are less than the minimum cutout to cutout clearance will be marked as shown below.



Cutouts too close together - marked with green circle (and arrow if error selected)

1.2.6.11.34.29 Cutout to Border Clearance

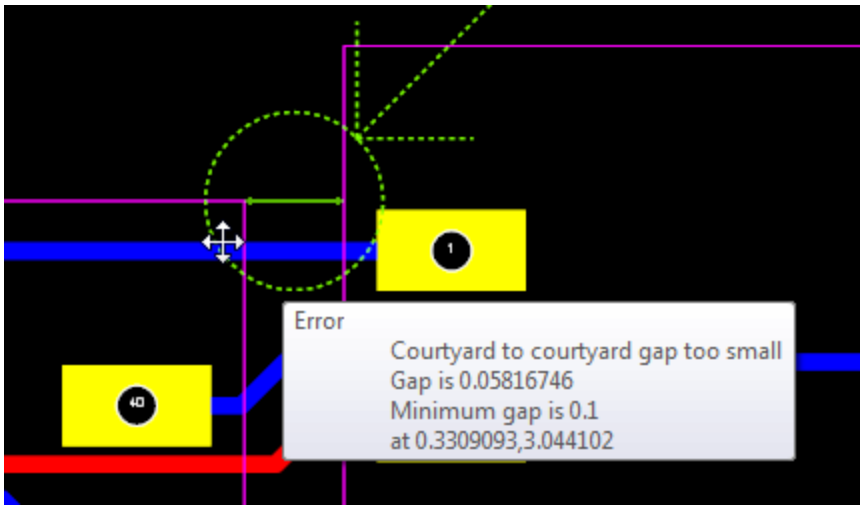
All cutout to PCB border spacings that are less than the minimum cutout to PCB border clearance will be marked as shown below.



Cutout too close to the PCB border - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.30 Courtyard to Courtyard Clearance

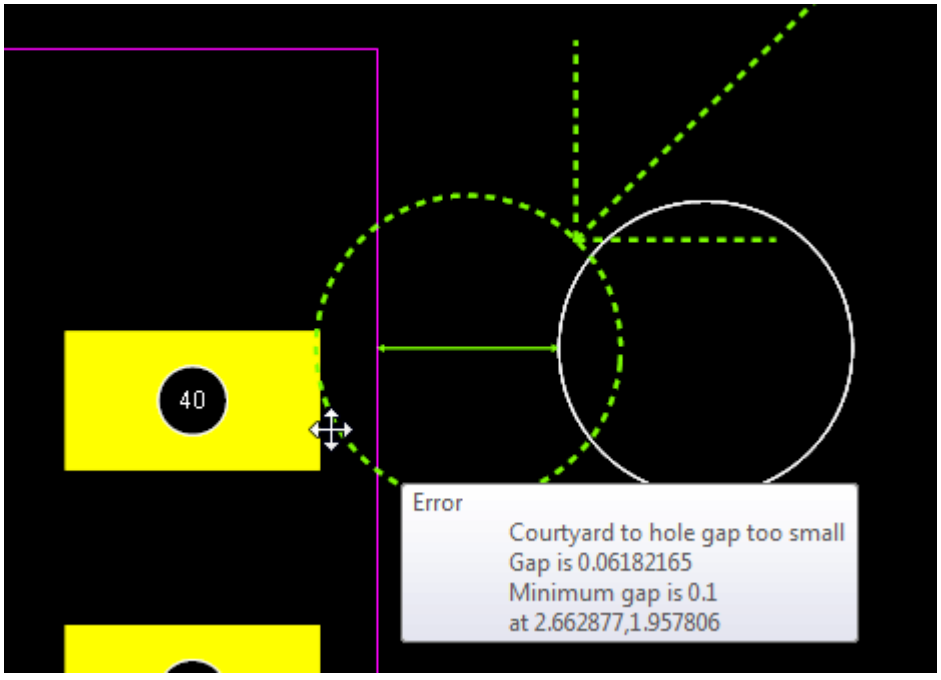
All courtyard to courtyard spacings that are less than the minimum courtyard to courtyard clearance will be marked as shown below.



Courtyard too close to a hole - marked with green circle (and arrow if error selected)

1.2.6.11.34.31 Courtyard to Holes Clearance

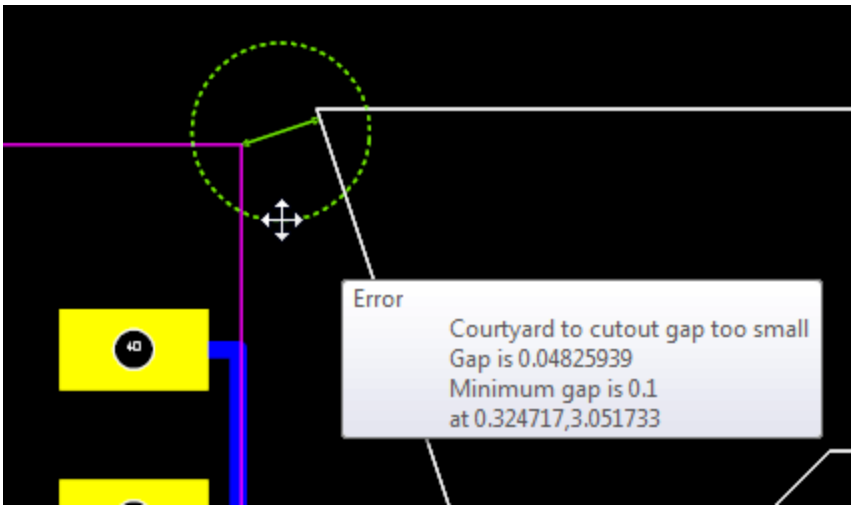
All courtyard to hole (not pad or via holes) spacings that are less than the minimum courtyard to hole clearance will be marked as shown below.



Courtyard too close to a hole - marked with green circle (and arrow if error selected)

1.2.6.11.34.32 Courtyard to Cutouts Clearance

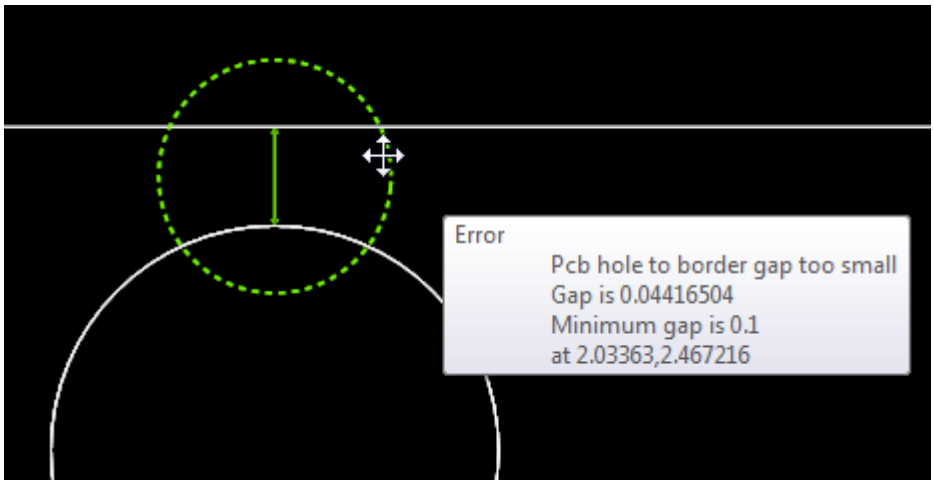
All courtyard to cutout spacings that are less than the minimum courtyard to cutout clearance will be marked as shown below.



Courtyard too close to a cutout - marked with green circle (and arrow if error selected)

1.2.6.11.34.33 Courtyard to Border Clearance

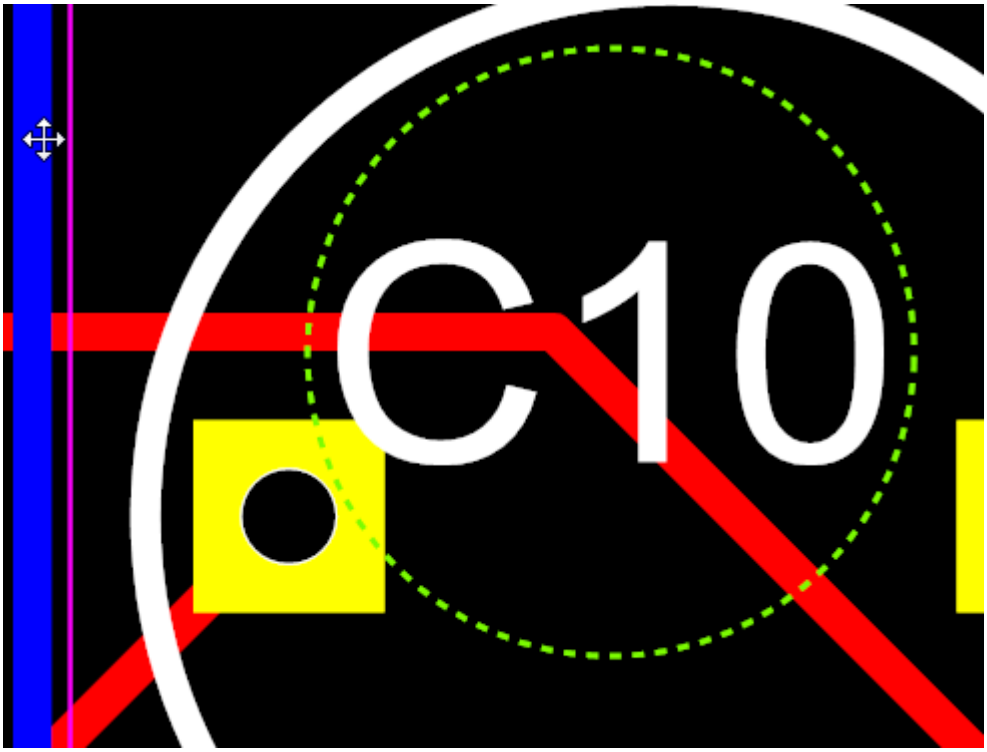
All courtyard to PCB border spacings that are less than the minimum courtyard to PCB border clearance will be marked as shown below.



Courtyard too close to the PCB border - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.34.34 Silkscreen to Pad Clearance

All silkscreen to pad spacings that are less than the minimum pad to PCB border clearance will be marked as shown below.



Silkscreen too close to the pad - marked with green circle and line segment showing the distance (and arrow if error selected)

1.2.6.11.35 3D

1.2.6.11.35.1 Adding 3D objects

There are 2 ways to add 3D objects to a PCB or footprint.

The first is using the **Add→3D** menu item.

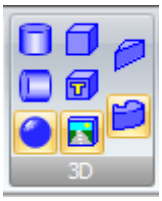
- **Adding Internal 3D Objects**

The second is to import 3D from an external file.

- **Adding Internal 3D Objects**

1.2.6.11.35.2 Adding Internal 3D Objects

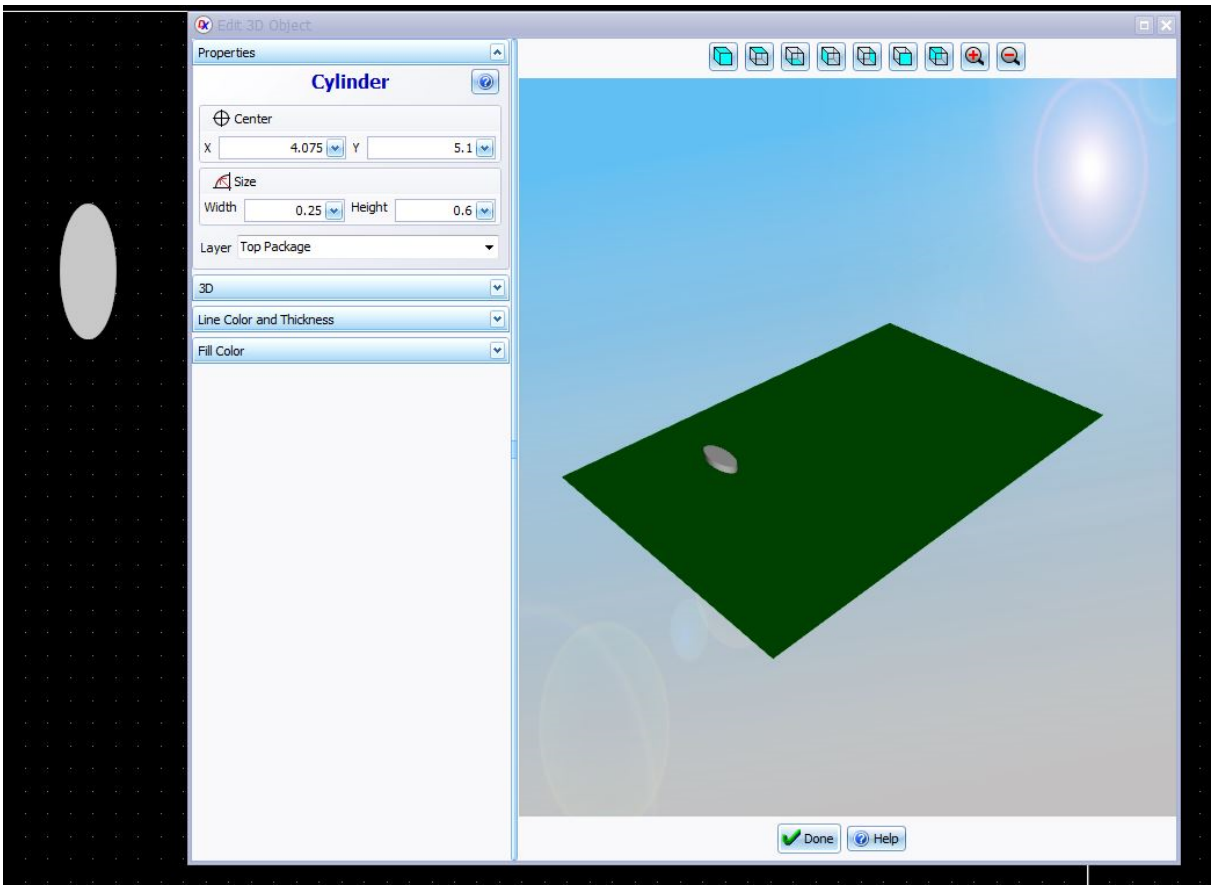
Using the Add→3D menu item



Use the button group in the **Add→3D** ribbon button group when editing the PCB.

1. Select either the top or bottom layer as the current layer.
2. Add your graphics.
3. Set the height and base of the graphic using the properties panel of the Edit 3D Object dialog that opens and click Done when finished.

See the example of adding a cylinder to the PCB below...



Adding a cylinder to a PCB using the 3D object tools from the Add→3D ribbon button Group

1.2.6.11.35.3 Adding External 3D Objects

You can add 3D from an external file to both a PCB and a Footprint

Click on the **Tools→Import→**  button.

The import 3D file will be shown as shown below.

Select a file and click  to add it to the PCB.

The 3D model will be imported and embedded in a 3D Model.

Currently AutoTRAX DEX will only import XGL files.

The XGL file format is designed to represent 3D information for the purpose of visualization. It attempts to capture all of the 3D information that can be rendered by SGI's OpenGL rendering library. It uses XML 1.0 syntax. These features make XGL the ideal format to use when data must be exchanged between two graphics systems for the purposes of visualization.

Autodesk Inventor will export to XGL.

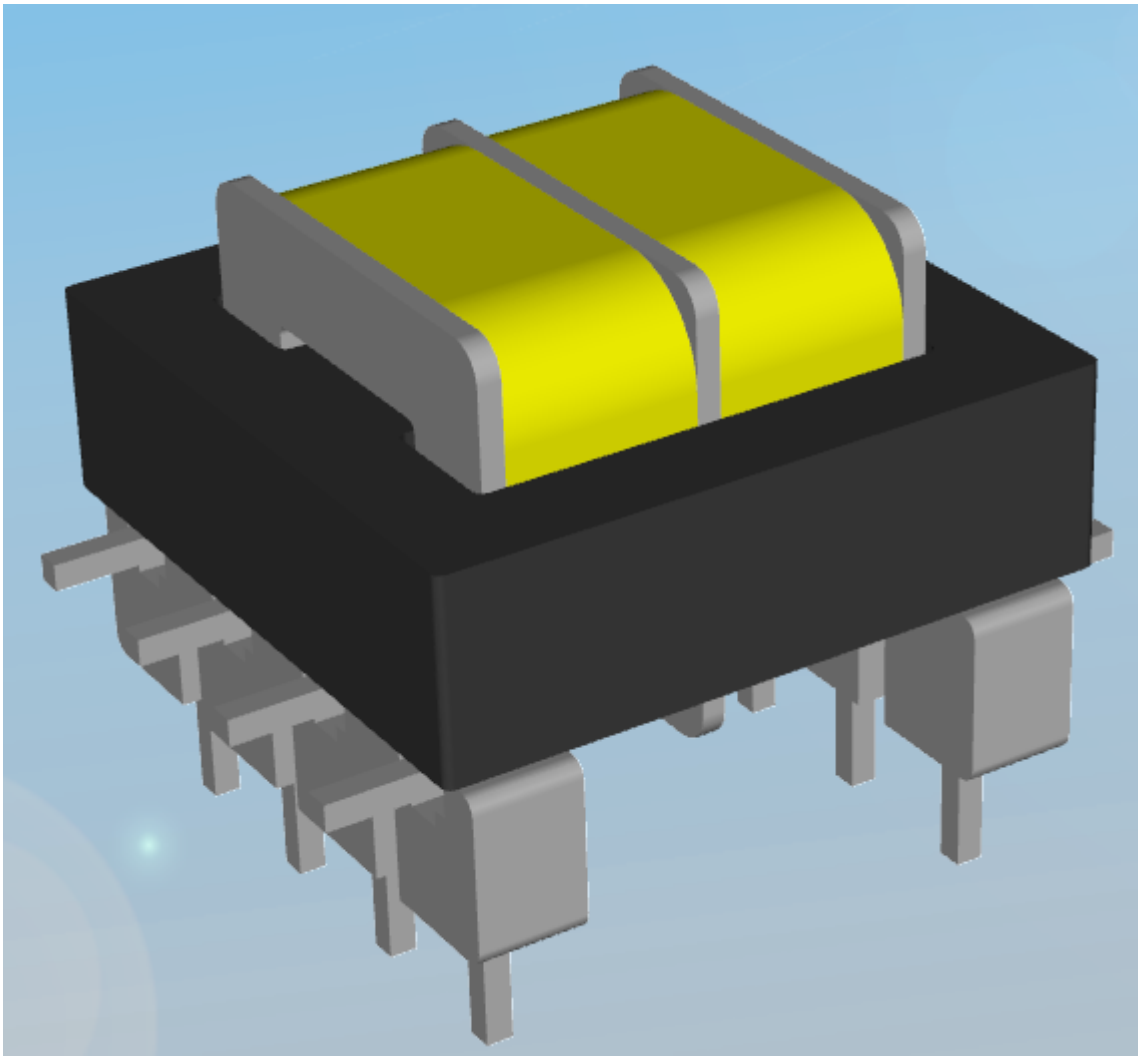
You can find loads of **free** 3D models at https://www.3dcontentcentral.com/

To turn them into XGL files:

1. Download the 3D file from https://www.3dcontentcentral.com/ as a .SAT file.
2. Start Autodesk Inventor.
3. Open the downloaded .SAT file.
4. Export to a XGL file.
5. Import it into AutoTRAX DEX.

1.2.6.11.35.4 3D Models

A 3D model is a 3D object imported from an external source.



A Transformer 3D model

To set the vertical height of the object use its properties dialog in the The Properties Panel

Drag the  thumb-wheel to the left or right to raise or lower the object in above/below the PCB.

1.2.6.11.36 Creating Complex Shapes

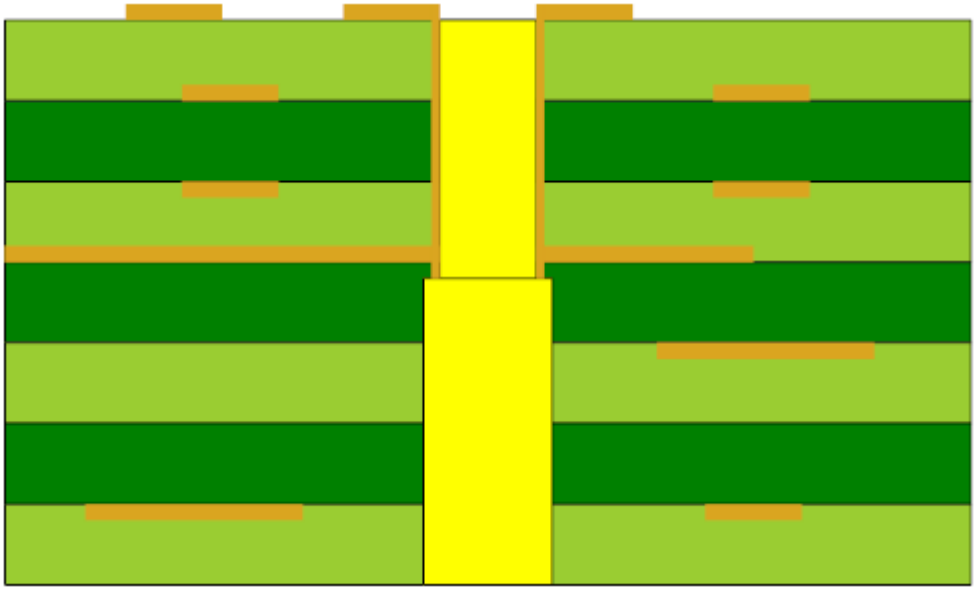
You can create a polygon from a collection of connected shapes.

In addition, on PCBs and Footprints you can create any of the following from a collection of connected shapes.

- PCB Border
- PCB Cutout
- Keepout region
- Copper Pour region
- TPH polygonal pad.

1.2.6.11.37 Drilling

Backdrill and Controlled Depth Drilling



The backdrill process removes stubs from plated-through-holes (vias). Stubs are the unnecessary /unused portions of vias, which extend further than the last connected inner layer.

Stubs can lead to reflections, as well as disturbances of capacity, inductance and impedance. This discontinuity errors become critical with increasing propagation speed.

Back-planes and thick Printed Circuit Boards in particular, can endure significant signal integrity disturbances through stubs. For High Frequency PCBs (e.g. with Impedance control), the application of backdrilling, as well as the application of blind and buried vias, can be part of the solution.

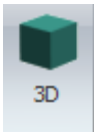
Backdrill can be applied to any type of circuit board where stubs cause signal integrity degradation, with minimal design and layout considerations. In contrast, when using blind vias, the aspect ratio has to be kept in mind.

Advantages of Backdrill

- Reduced deterministic jitter
- Lower bit error rate (BER)
- Less signal attenuation with improved impedance matching
- Increased channel bandwidth
- Increased data rates
- Reduced EMI radiation from the stubs
- Reduced excitation of resonance modes
- Reduced via-to-via crosstalk
- Aspect ratio can be neglected (in contrast to blind vias)

1.2.6.11.38 Viewing the PCB in 3D

To view the PCB or part in 3D:

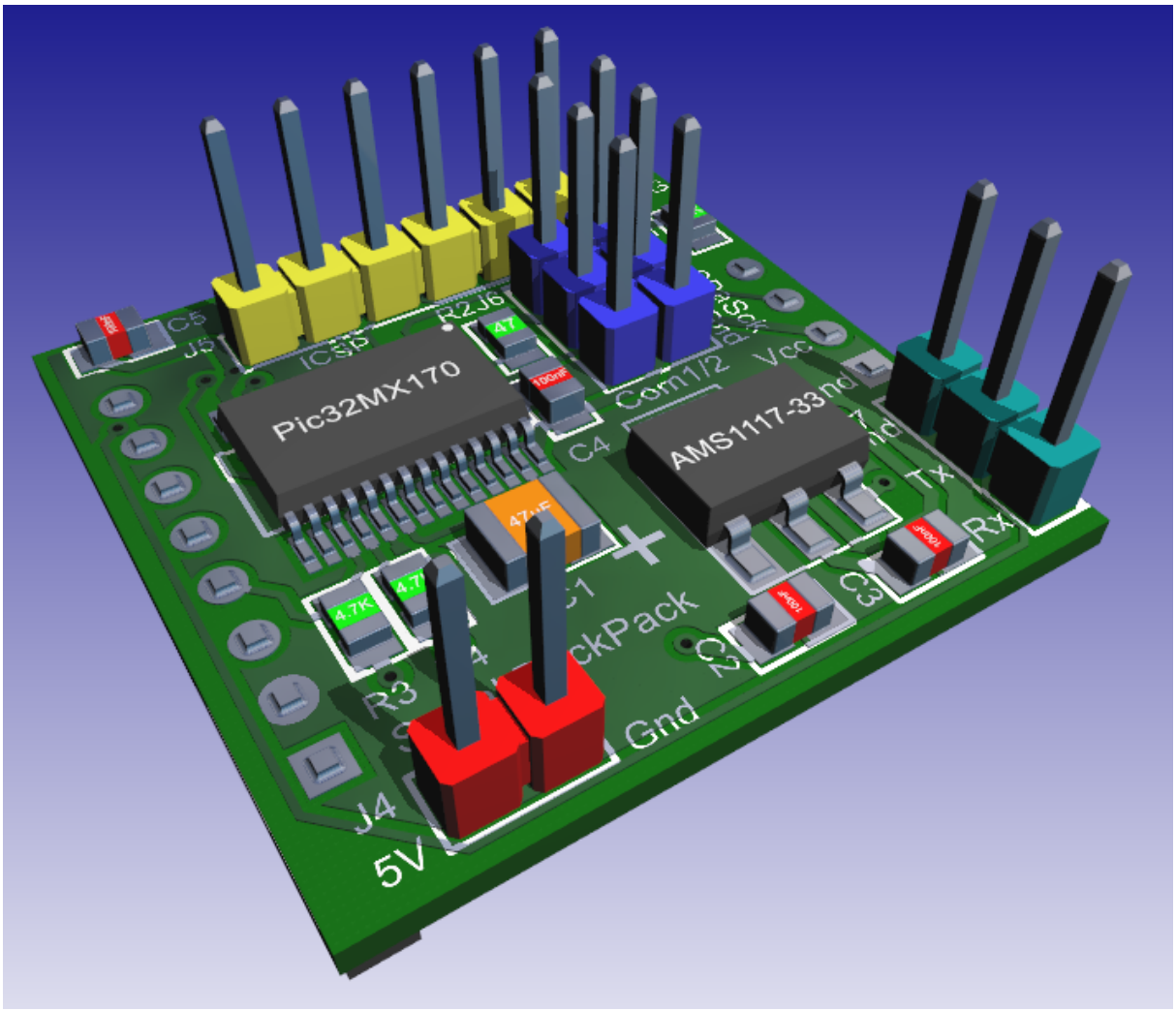
Click on the 

button in the Panels→Window button group or the **PCB→Panels** button group.

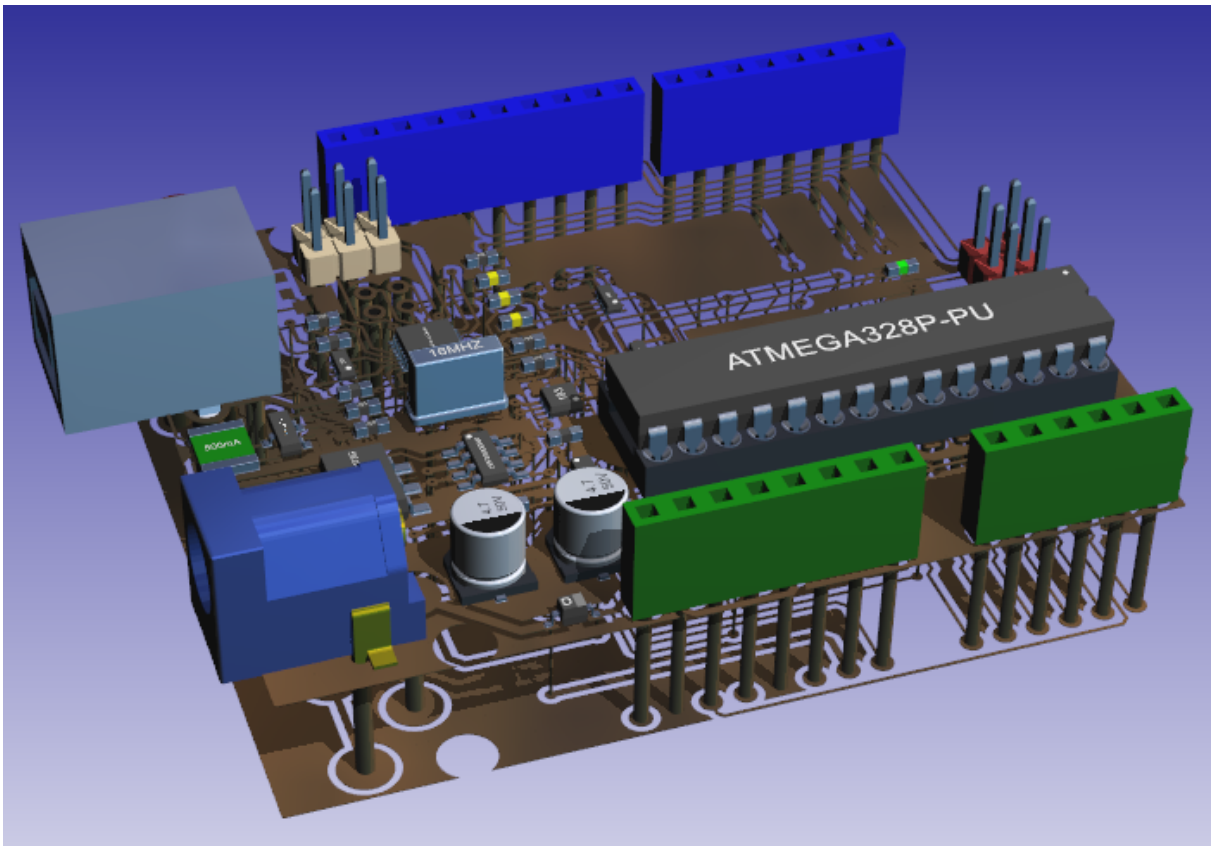
Alternately ***right-click*** in the PCB view and click on the **View in 3D** menu item.

The 3D view includes:

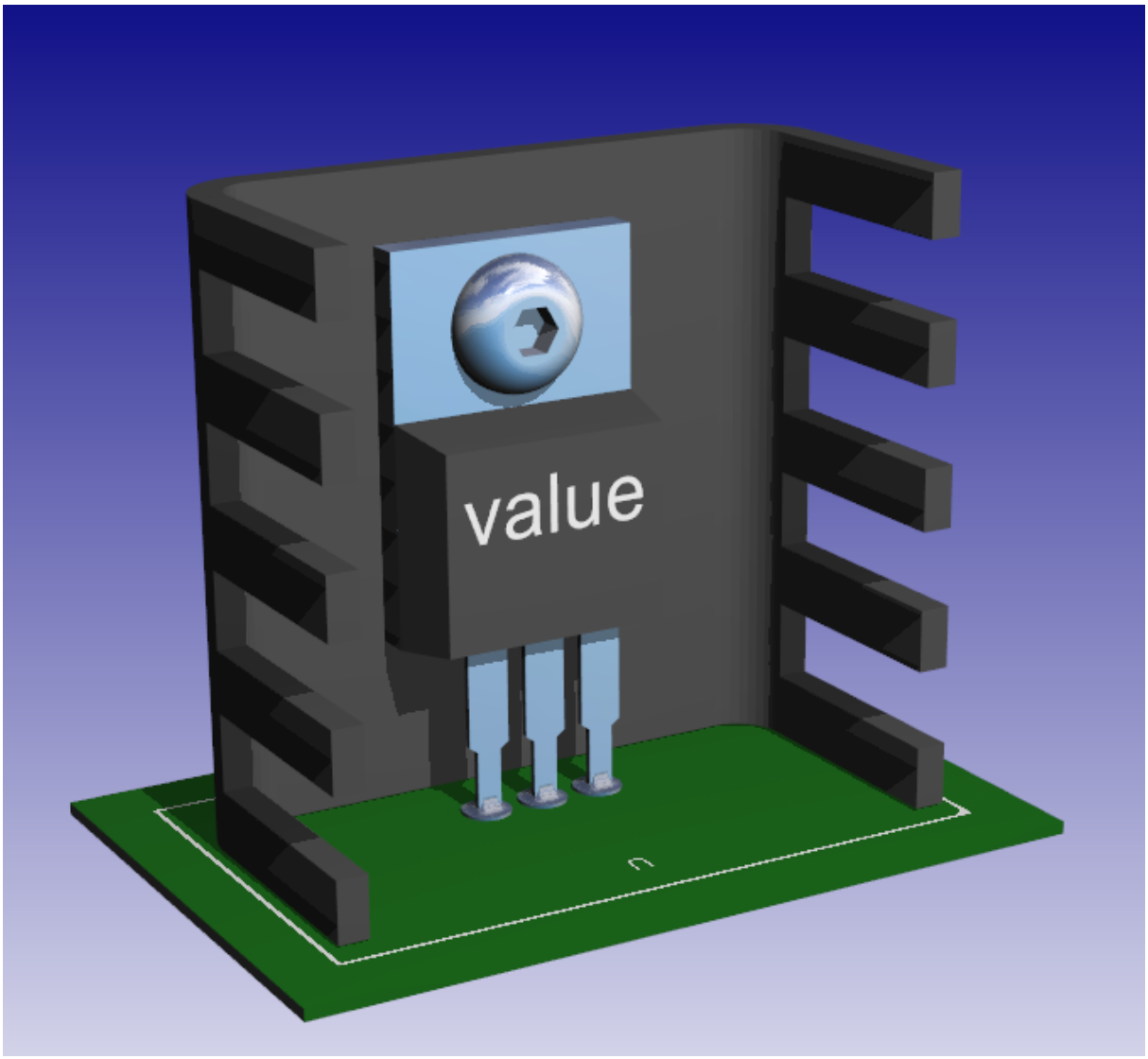
- Real-time shadows
- Reflective surfaces
- Texture maps and images
- Surface bump textures
- Automatic solder added
- 3D copper tracks
- Parametrically generated 3D parts
- Exploded PCB view



3D view of a PCB



3D Exploded View



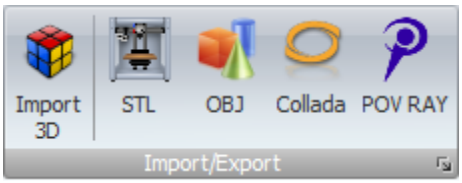
3D view of a part

1.2.6.11.38.1 The 3D Viewport Menu Commands

You'll find several useful commands for controlling the views and layouts of the 3D viewport in the top ribbon menu under the 3D ribbon tab.

The import/export ribbon group contains several useful commands for importing and exporting 3-D.

Import/Export



Import

Click on the 

button to import 3D into your design.

Find Out More...

Export

STL

Click on the 

button to export a STL file.

STL (an abbreviation of "stereolithography") is a file format native to the stereolithography CAD software created by 3D Systems.The STL format specifies both ASCII and binary representations. Binary files are more common, since they are more compact.An STL file describes a raw, unstructured triangulated surface by the unit normal and vertices (ordered by the right-hand rule) of the triangles using a three-dimensional Cartesian coordinate system. In the original specification, all STL coordinates were required to be positive numbers, but this restriction is no longer enforced and negative coordinates are commonly encountered in STL files today. STL files contain no scale information, and the units are arbitrary.

OBJ

Click the 

will export the 3-D to a Wavefront OBJ file.

OBJ (or .OBJ) is a geometry definition file format first developed by Wavefront Technologies for its Advanced Visualizer animation package. The file format is open and has been adopted by other 3D graphics application vendors.

The OBJ file format is a simple data-format that represents 3D geometry alone — namely, the position of each vertex, the UV position of each texture coordinate vertex, vertex normals, and the faces that make each polygon defined as a list of vertices, and texture vertices. Vertices are stored in a counter-clockwise order by default, making explicit declaration of face normals unnecessary. OBJ coordinates have no units, but OBJ files can contain scale information in a human readable comment line.

Collada

The 

will export the 3-D to a Collada file.

COLLADA (COLLAborative Design Activity) is an interchange file format for interactive 3D applications. It is managed by the nonprofit technology consortium, the Khronos Group, and has been adopted by ISO as a publicly available specification.

POVRAY

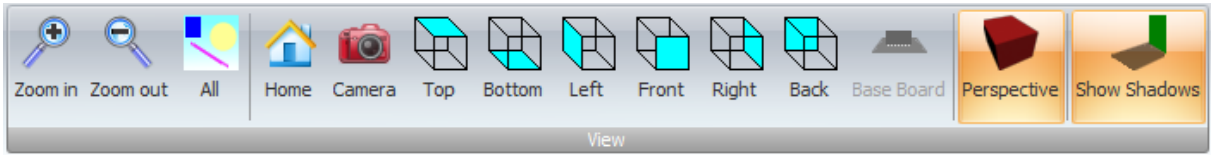
The 

button will create a POV RAY photorealistic render file.

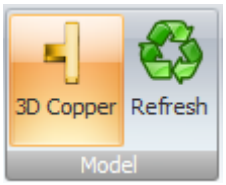
If you have POV RAY installed then the POV RAY program will start and render your file in 3D. The Persistence of Vision Ray Tracer, or POV-Ray, is a ray tracing program which generates images from a text-based scene description, and is available for a variety of computer platforms.

Find Out More...

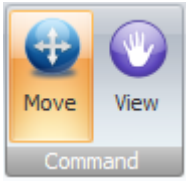
Views



Model

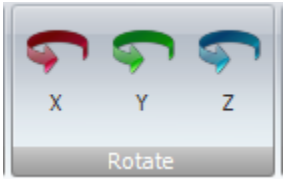


Command

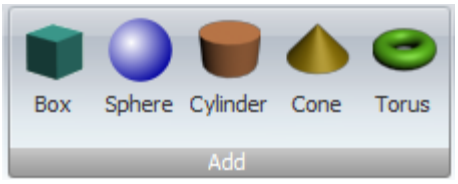


Pan

Rotate



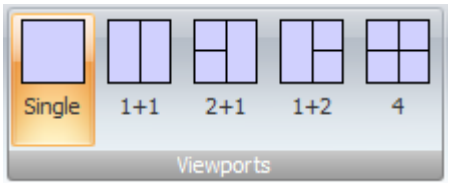
Add



Find Out More...

Multiple Viewports

You can have anywhere from 1 to 4 views into the same scene. Click on any of the five buttons in the viewport's menu to select the view configuration. The pattern in the buttons show you the layout of the individual views

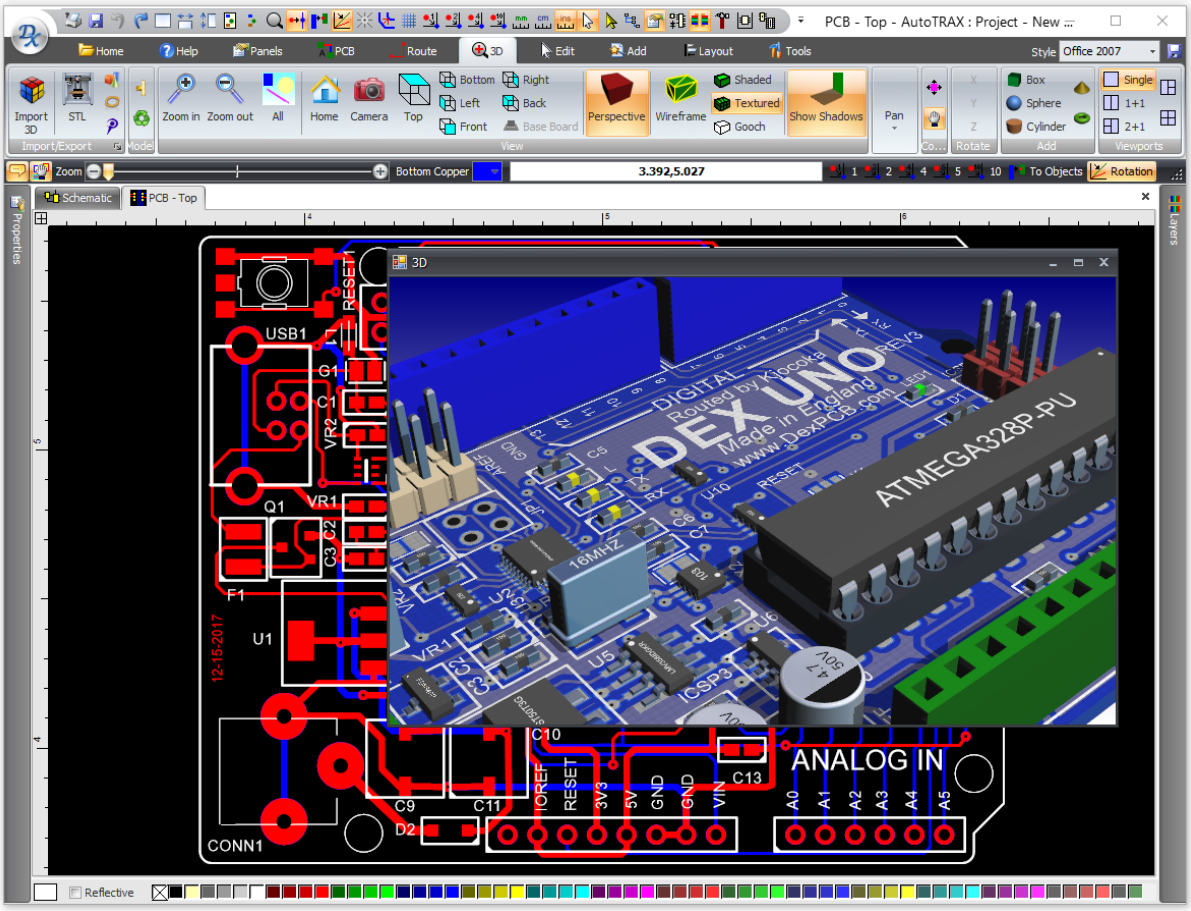


Find Out More...

1.2.6.11.38.2 Floating the 3D Viewport

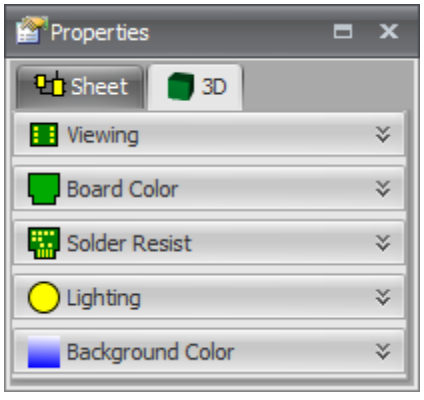
To float the 3D viewport double click on the 3D viewport's name tab. You can also hold down the left mouse button over the name tab and drag it, the viewport will then float when you release the left mouse button.. You can then re-size and move the 3D viewport. You can even move it to another screen. If you save the design, then the next time the design is opened the floating 3D viewport will appear.

Floating 3D Viewport



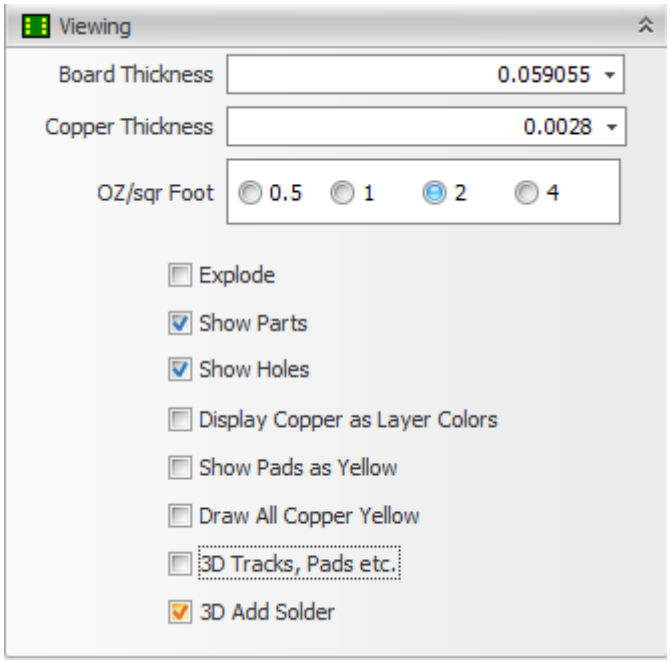
1.2.6.11.38.3 3D View Control

You can set several viewing options for 3D using the 3D tab in the properties panel. Click on to expand the control.



Viewing

This set of controls set the visibility of various parts of the PCB and also the thickness of the PCB.

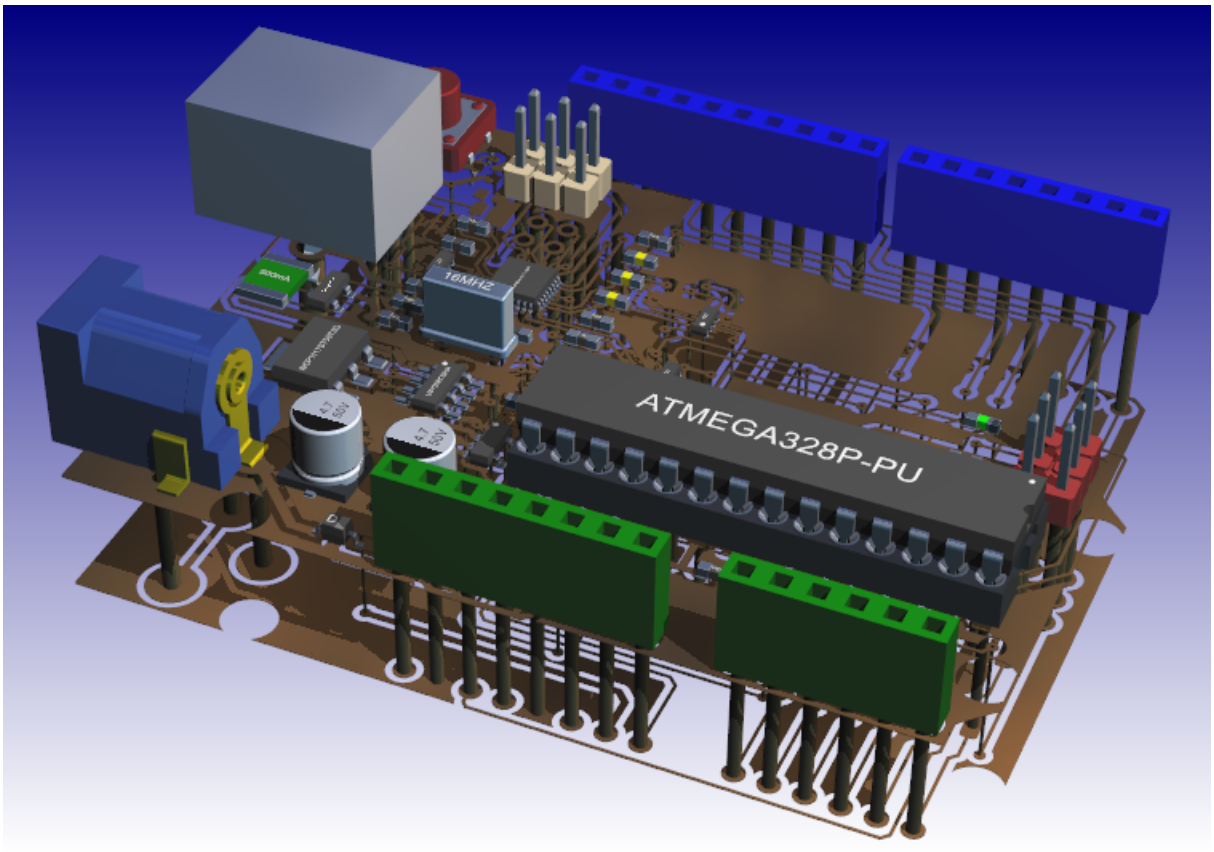


Board Thickness. This is the total thickness of the board including the copper layers on the upper and lower surfaces.

Copper Thickness. This is the thickness of the copper on the top and bottom side of the PCB.

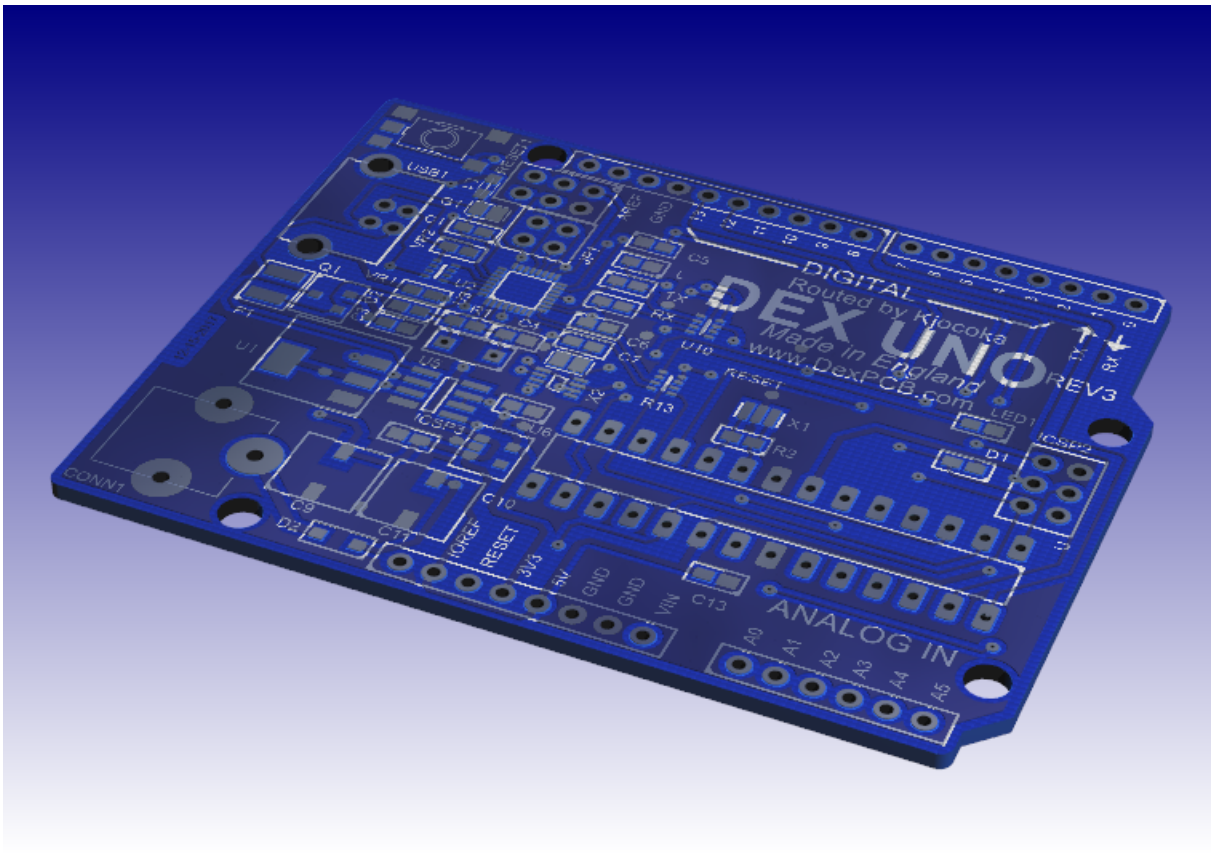
Explode. Check this to create an exploded view of the PCB. This will show you the internals of the PCB.

Exploded View of the PCB



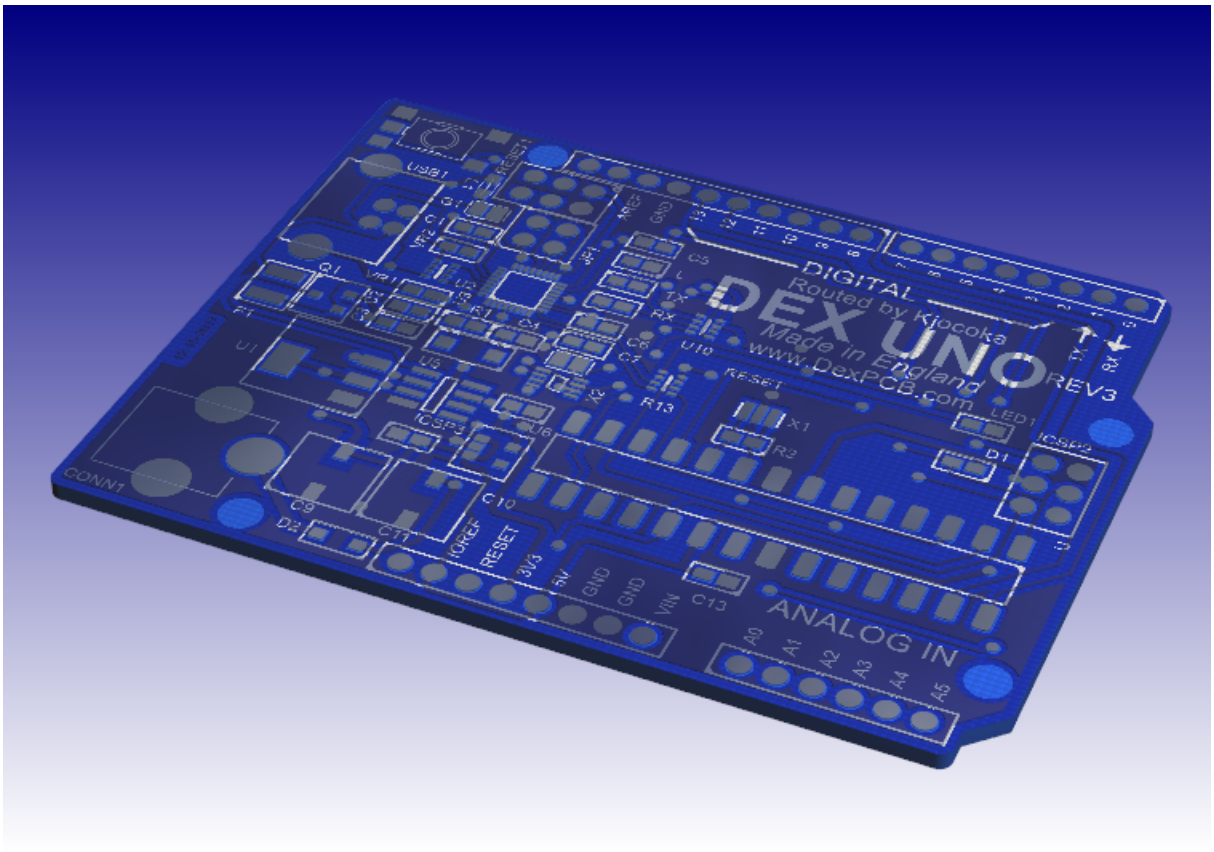
Show Parts. Uncheck this to hide the parts and show only the PCB.

Show Parts Unchecked

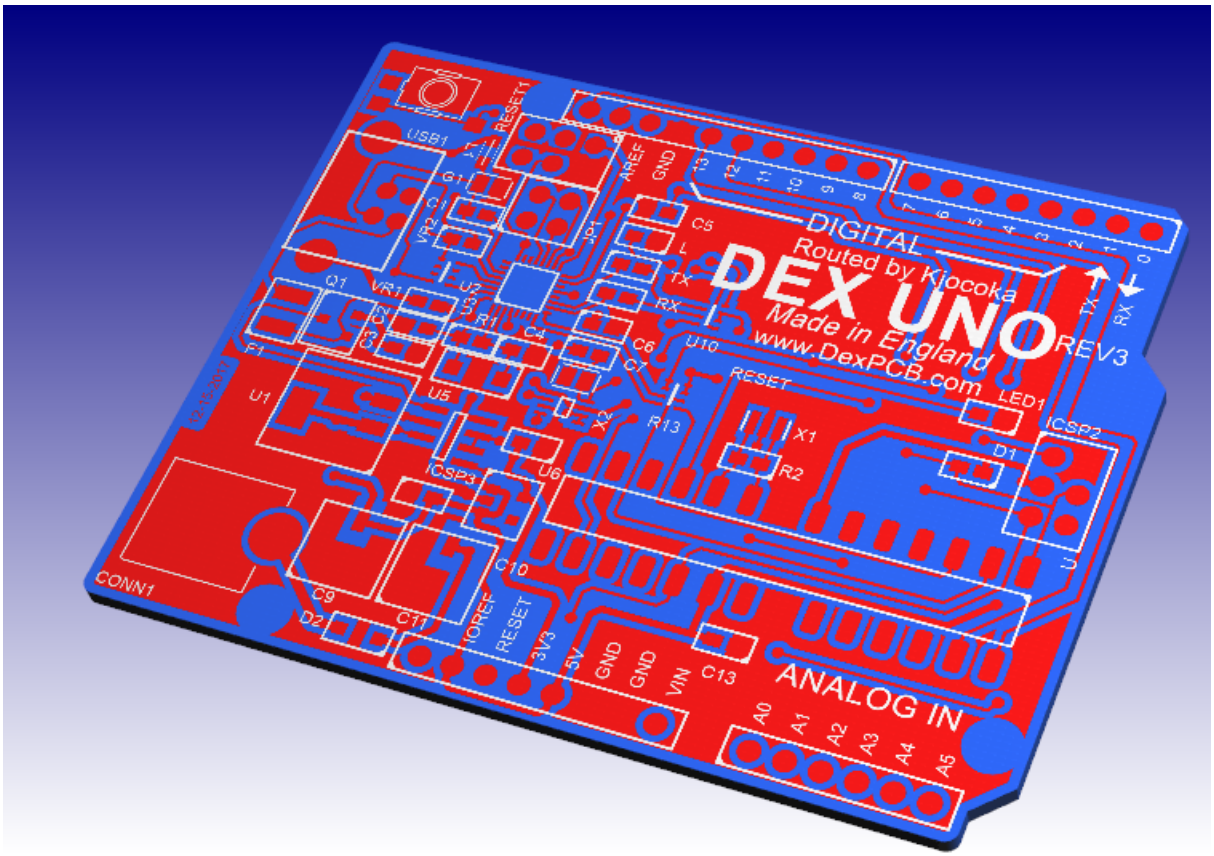


Show Holes. Check this to show the PCB holes. If unchecked the holes will not be visible.

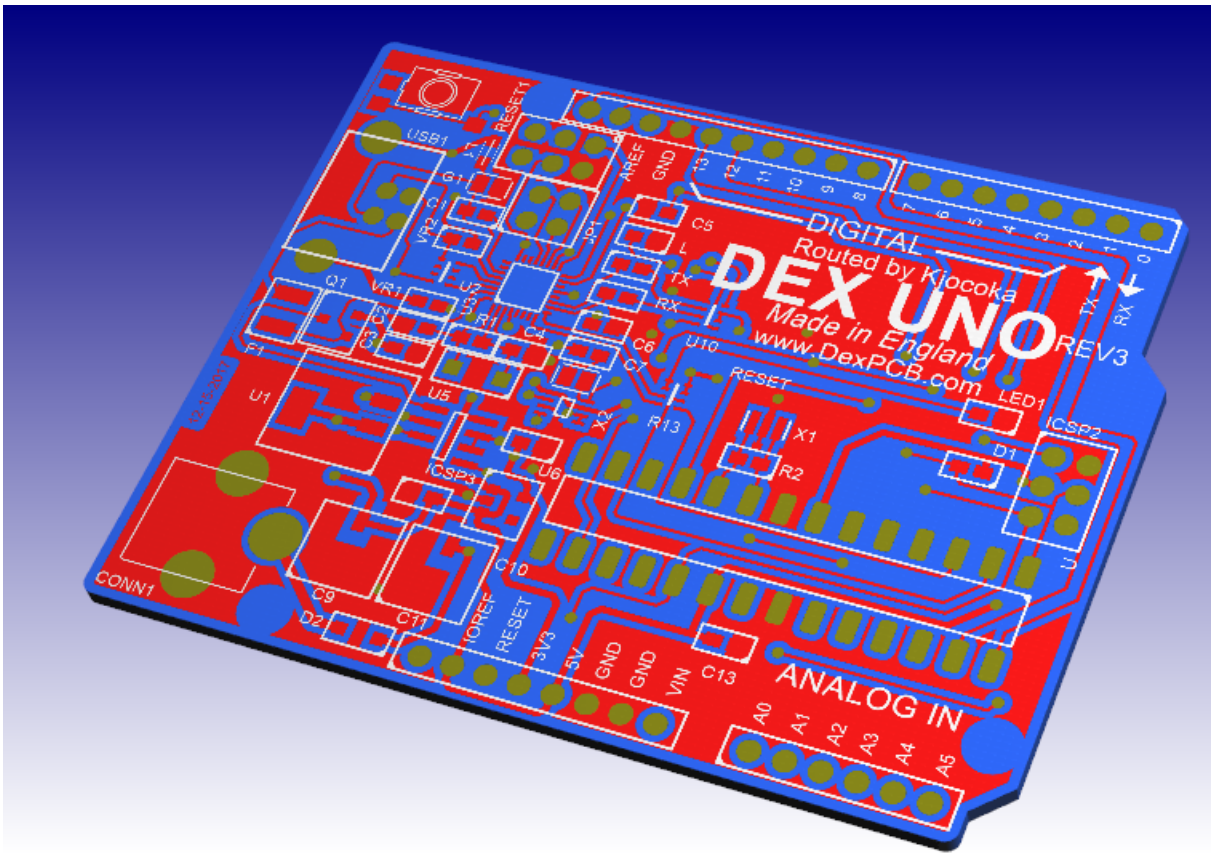
Show Parts and Show Holes Unchecked



Display Copper as Layer Color. Check this to display the copper with the same color as the color in the to the PCB view.

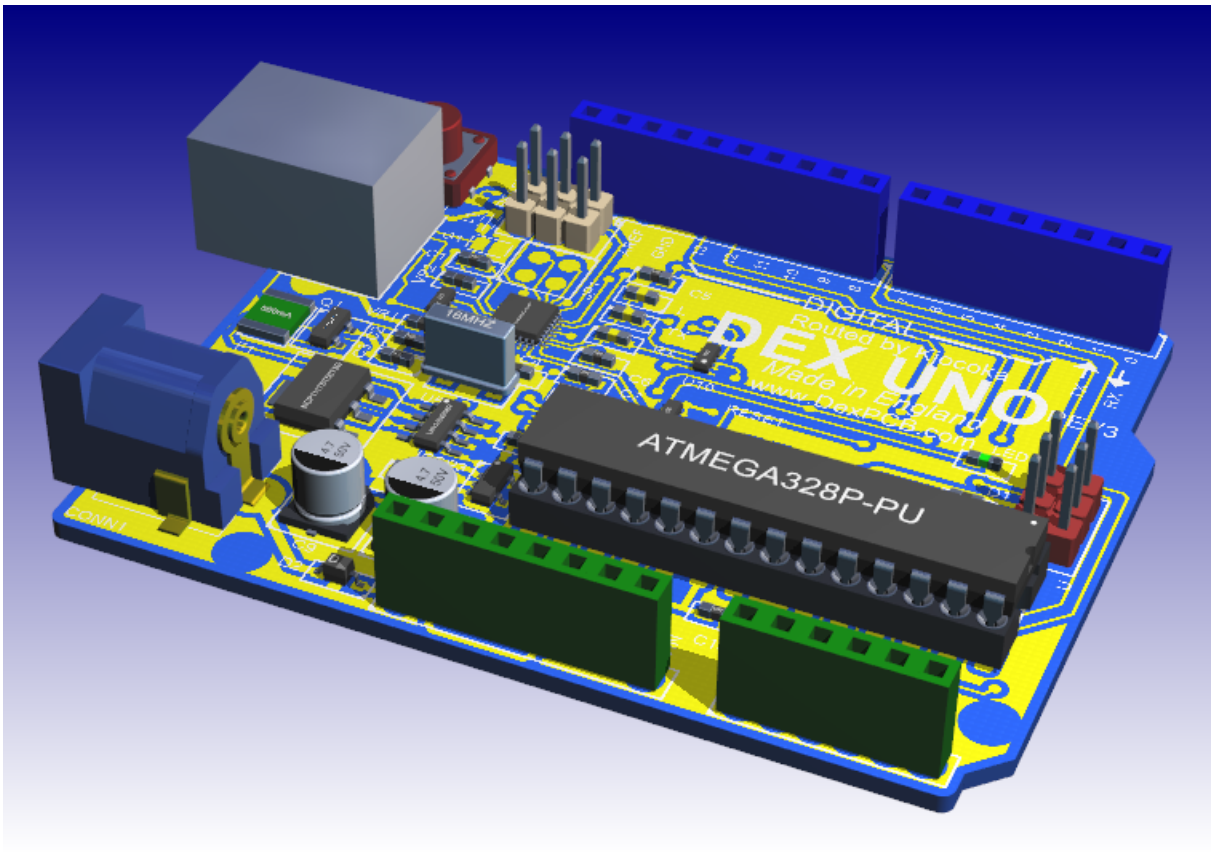


Show Pads as Yellow. Check this to show all pads colored yellow.



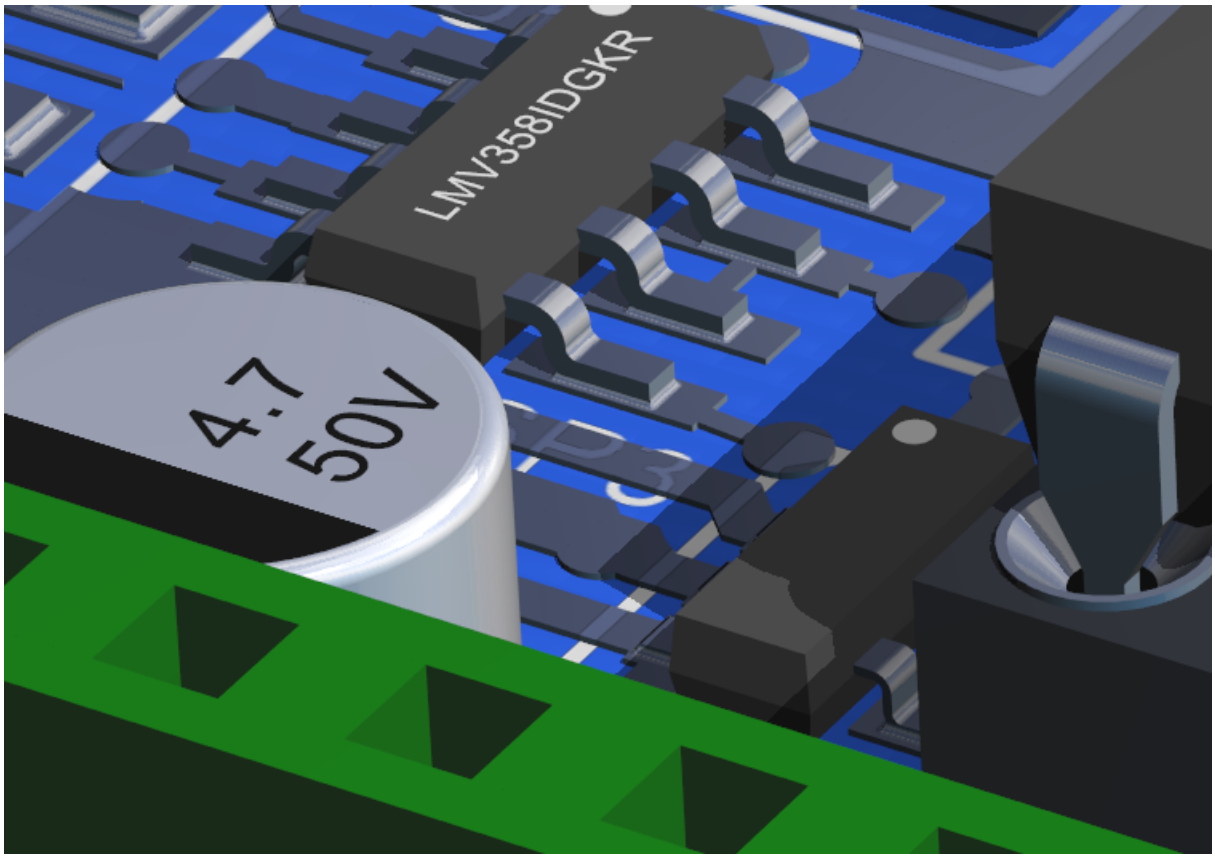
Draw All Copper Yellow. Check this to show all copper colored yellow.

All copper on the top and bottom of the PCB drawn as yellow.

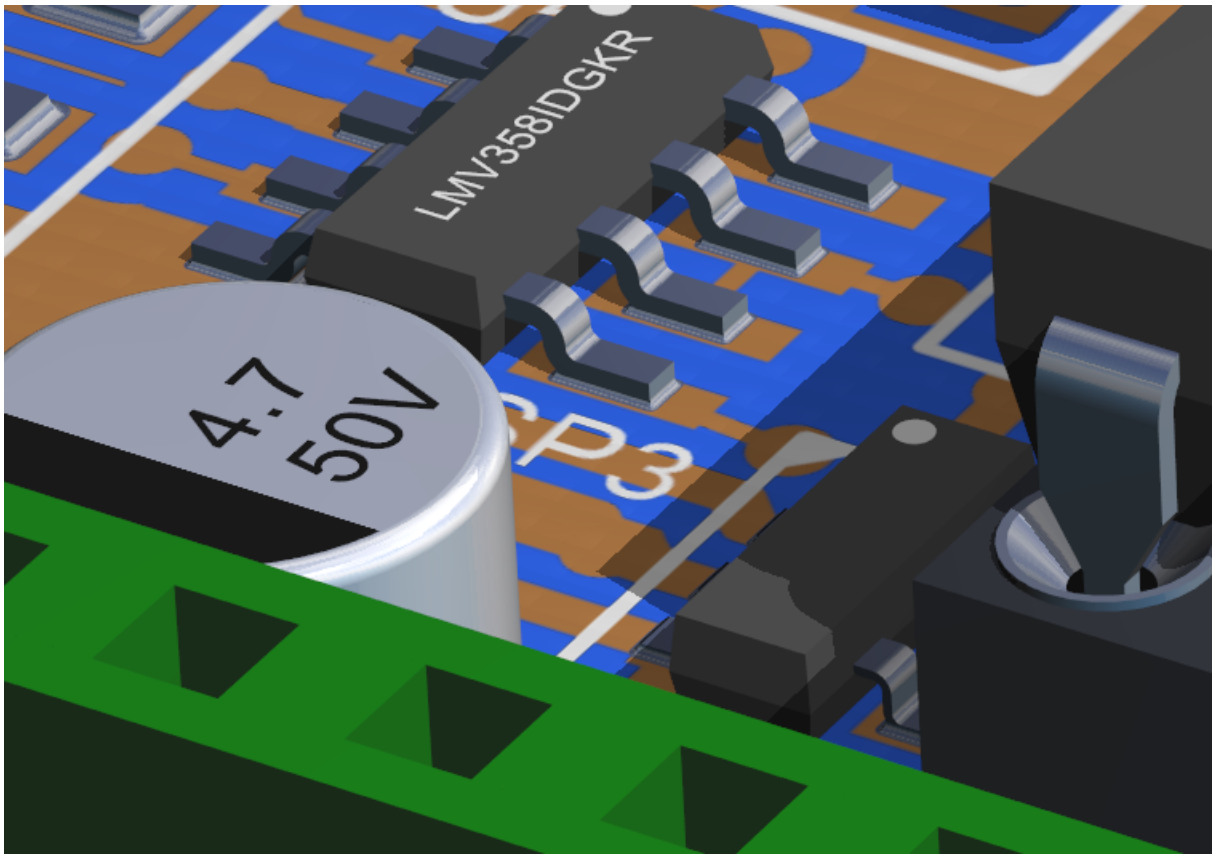


3D Tracks, Pads etc. Check to show the tracks, pads and other copper areas as 3-D. If unchecked then they will be added as texture to the surface of the PCB and will draw a lot faster but the quality will not be as good.

3D Copper

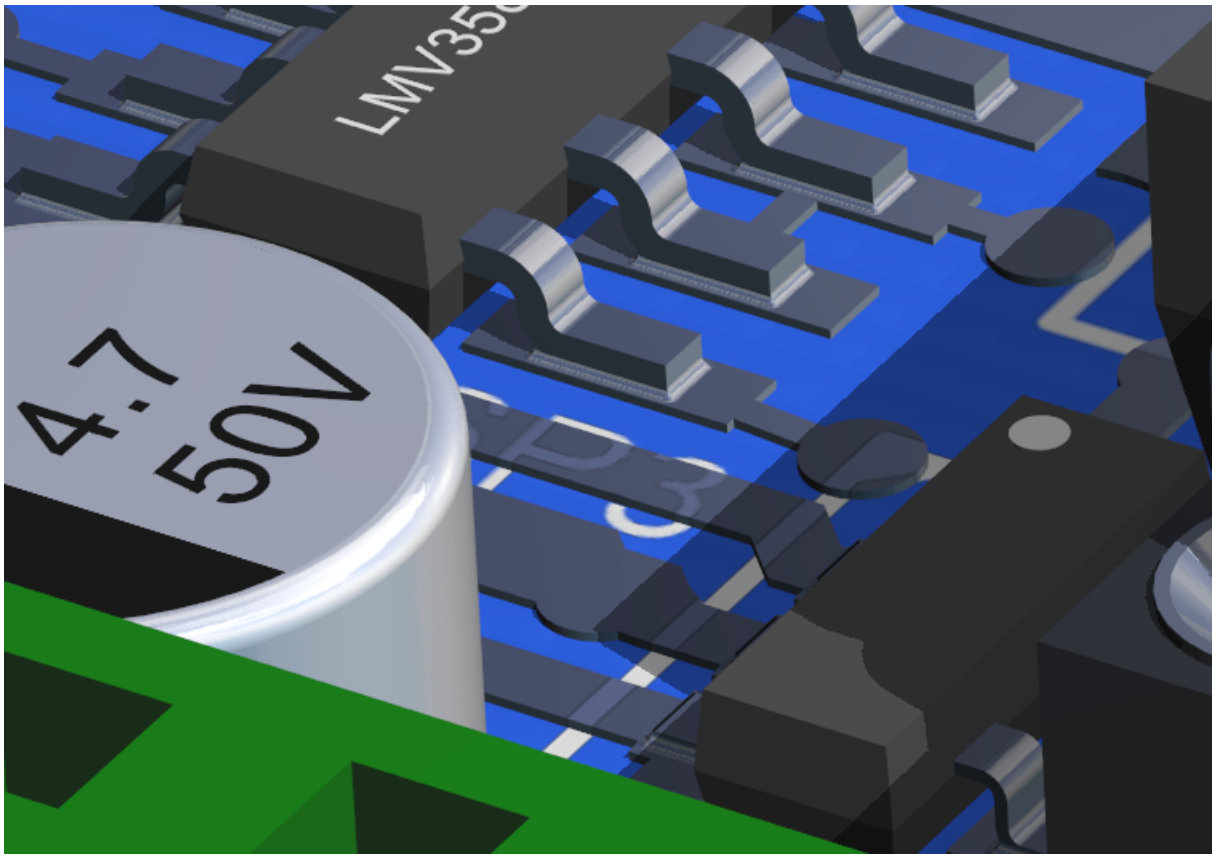


2D Copper

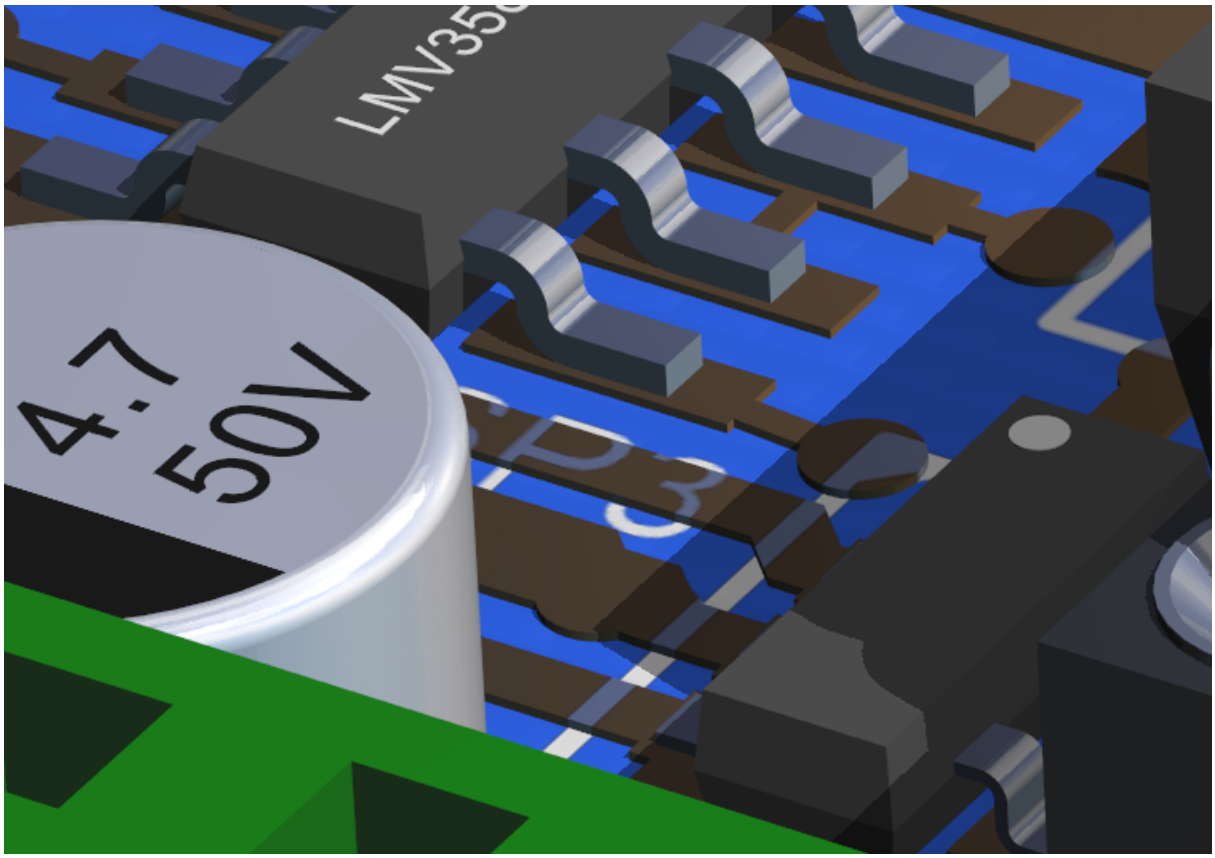


3D Add Solder. Checking this will add 3-D solder to all pad connections.

3D Solder

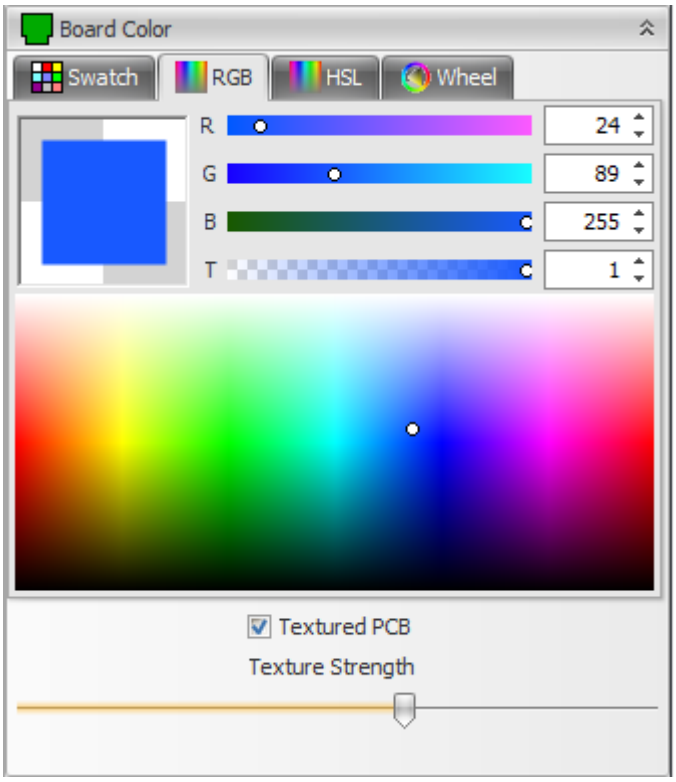


No 3D Solder

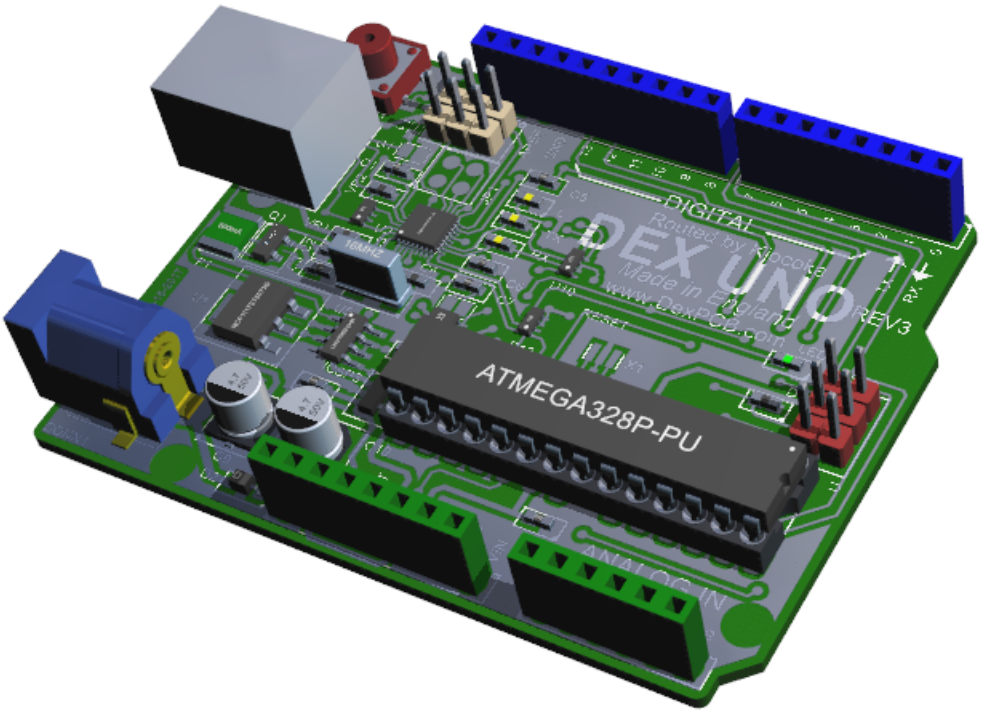


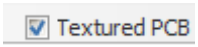
Board Color

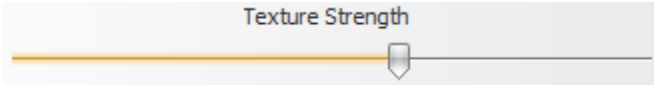
This sets the color of the board material. It does not set the color of the solder mask.



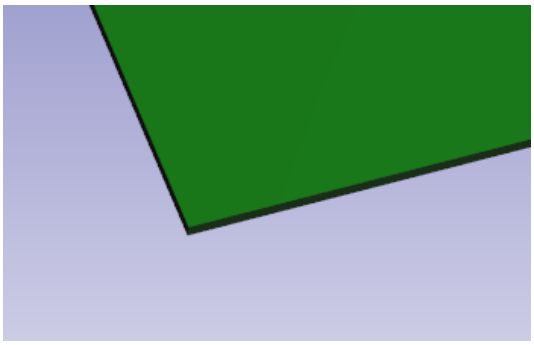
Green Board



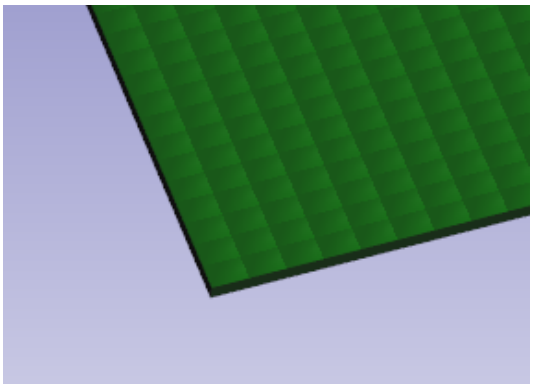
Check  to give the board a glass fiber type weave.

Drag the  slider to enhance the appearance of the weave.

No Weave

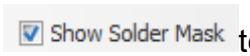


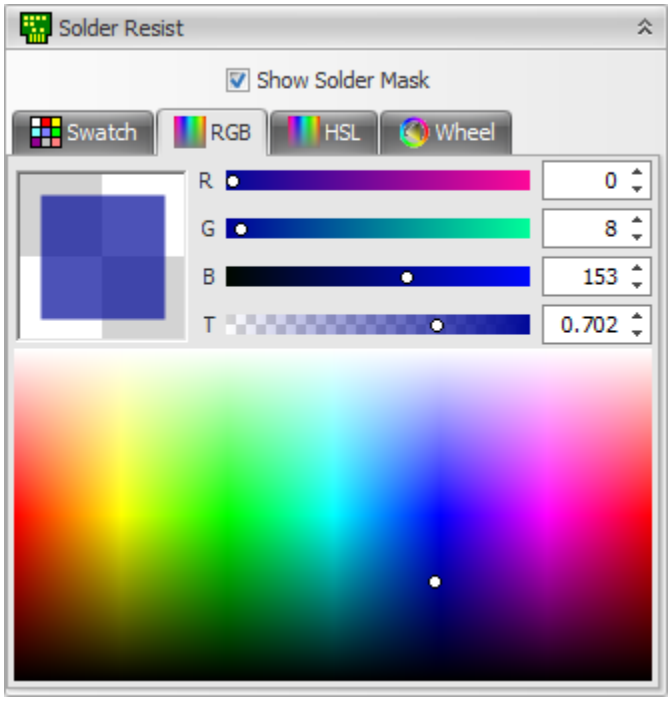
Enhanced Weave



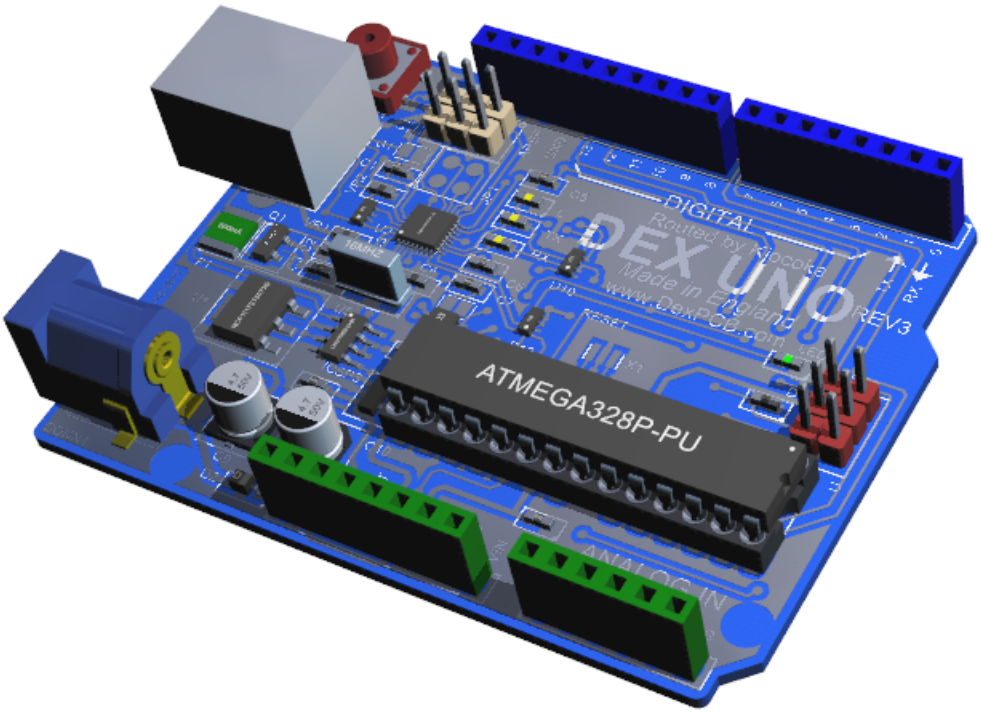
Solder Mask

This sets the visibility and color of the solder mask.

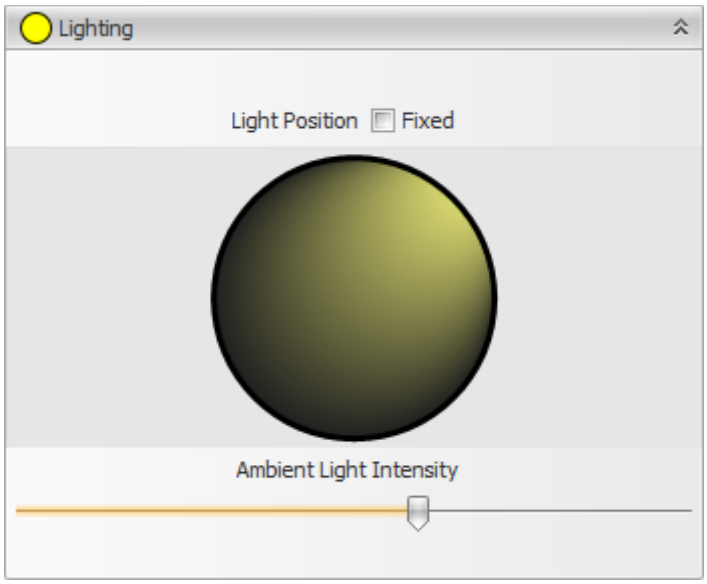
Check  to show the solder mask.

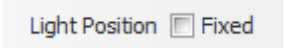


No Solder Mask



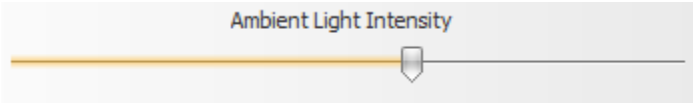
Lighting



Check  to fix the position of the light relative to the PCB. If unchecked then the light is attached to the camera (Your viewing position) and move when you alter the viewpoint.

Hold down the left mouse button and drag it over the light track ball

 to set the position of the light.

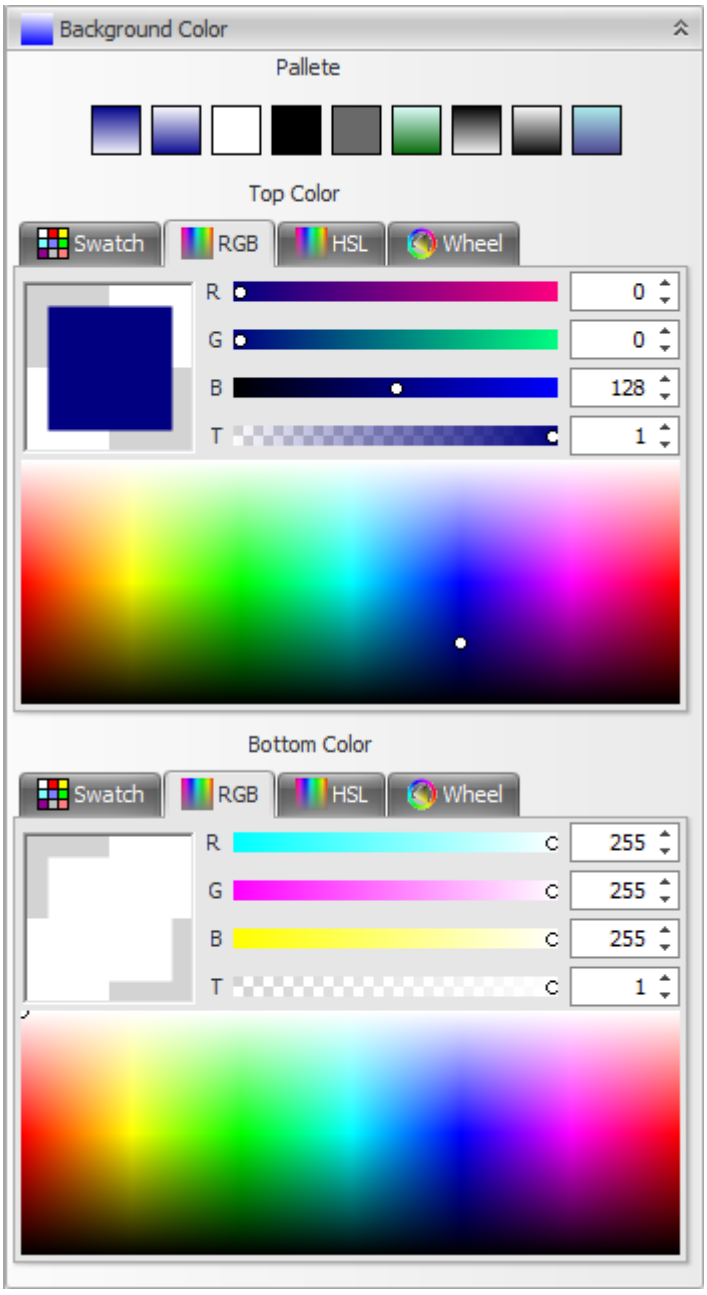
Drag the 

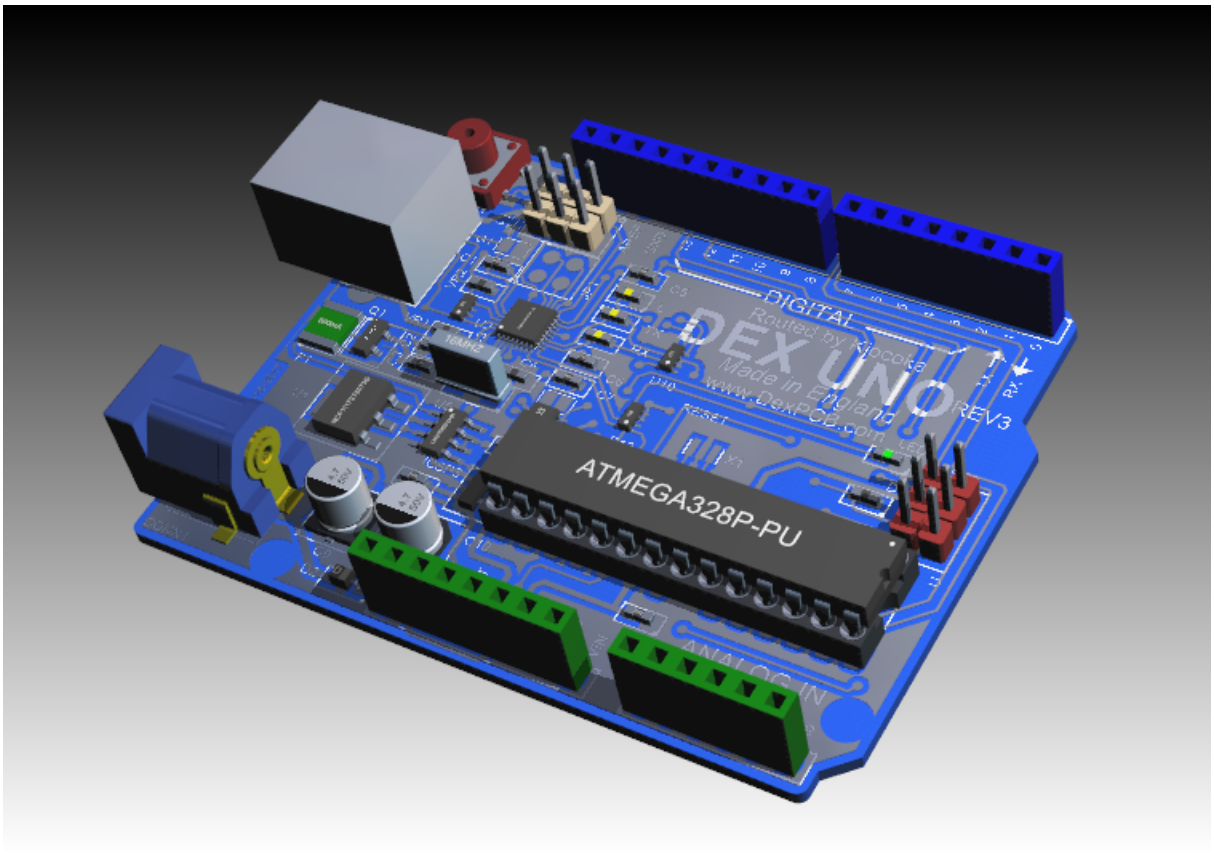
slider to the right to increase the level of background/fill lighting.

Background Color

The 3D background is a vertical gradient from a top color to a bottom color.

At the top of the Background Color expansion group is a palette of preset gradient. Click on any of the





1.2.6.11.38.4 The 3D Viewport

You'll find several useful commands for controlling the views and layouts of the 3D viewport in the top ribbon menu under the 3D ribbon tab.

The import/export ribbon group contains several useful commands for importing and exporting 3-D.

Import/Export

Import

Click on the 

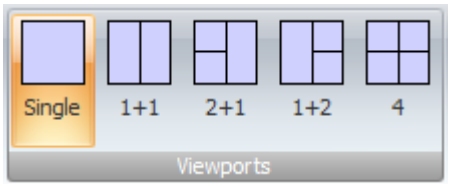
button to import 3D into your design. Read More...

Export

Read More...

Multiple Viewports

You can have anywhere from 1 to 4 views into the same scene. Click on any of the five buttons in the viewport's menu to select the view configuration. The pattern in the buttons show you the layout of the individual views



Read More...

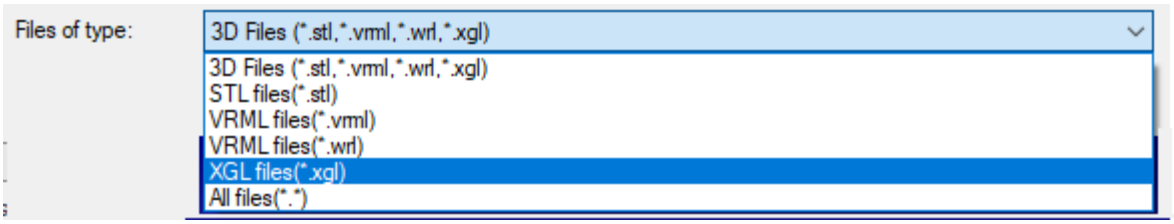
Click on the Click on the 

to display the Import 3D Dialog.

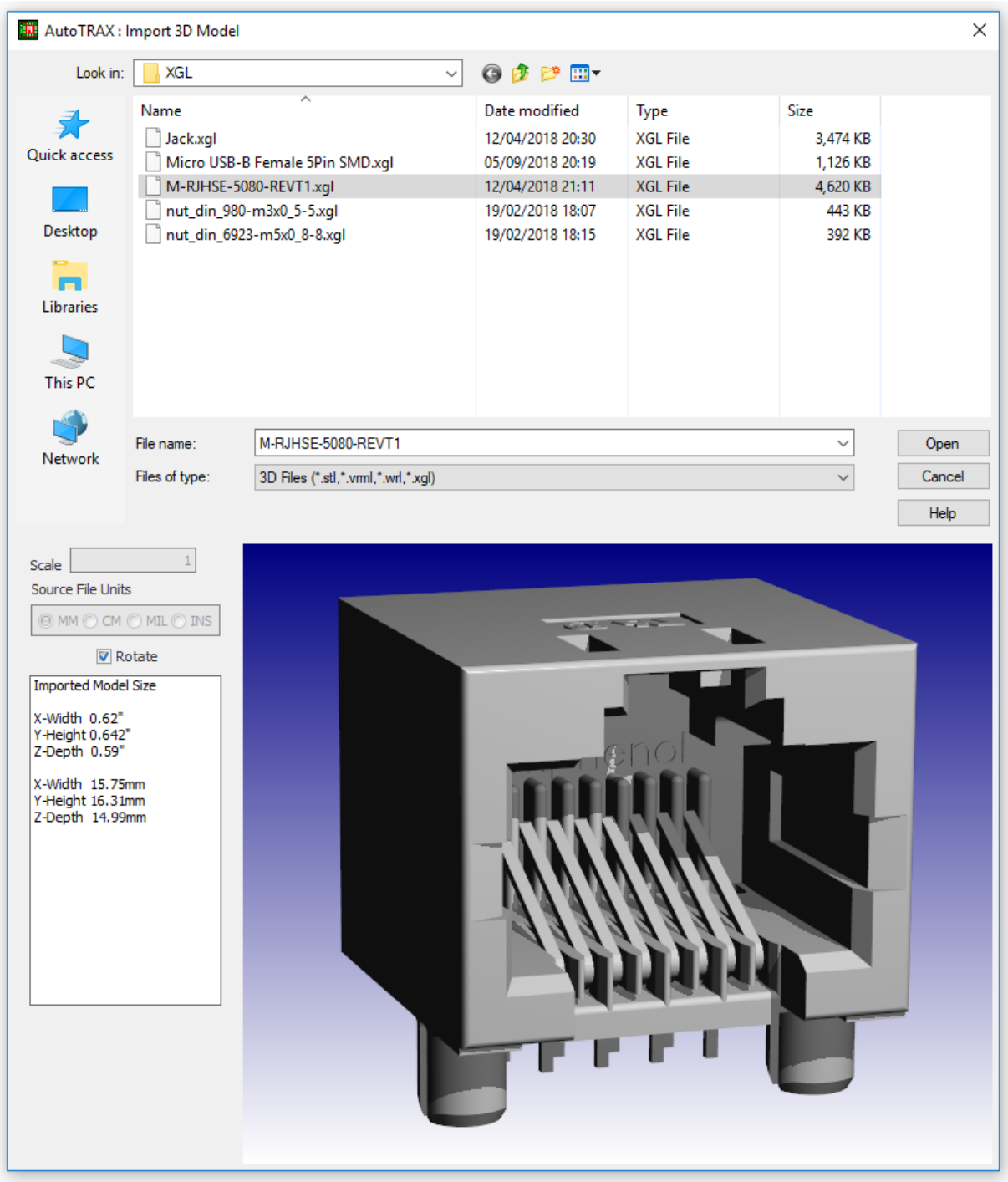
You can import any of the following 3-D file format types:

- **STL** This format was designed for 3-D printing and unfortunately only contains white objects as no material properties are defined in STL files. If possible you should use a different file format such as XGL. Find Out More...
- **VRML** Both VRML version 1 and version 2 are fully supported.Find Out More...
- **XGL** This is the recommended file format as it can contain material and colors.

Drop-down list of supported file formats

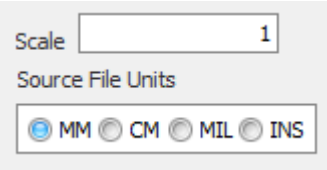


The Import 3D Dialog



Units and Scale

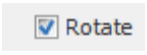
Both VRML and STL files do not contains units, so you may have to set the import scale. XGL files have units defined so you do not need to define units or scale.



Rotate

Different source define different direction for the vertical direction (distance from ground)

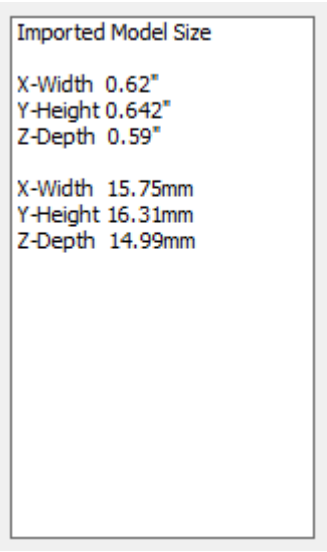
AutoTRAX DEX designs the UP direction as the +Z However others use the +Y.

Check the 

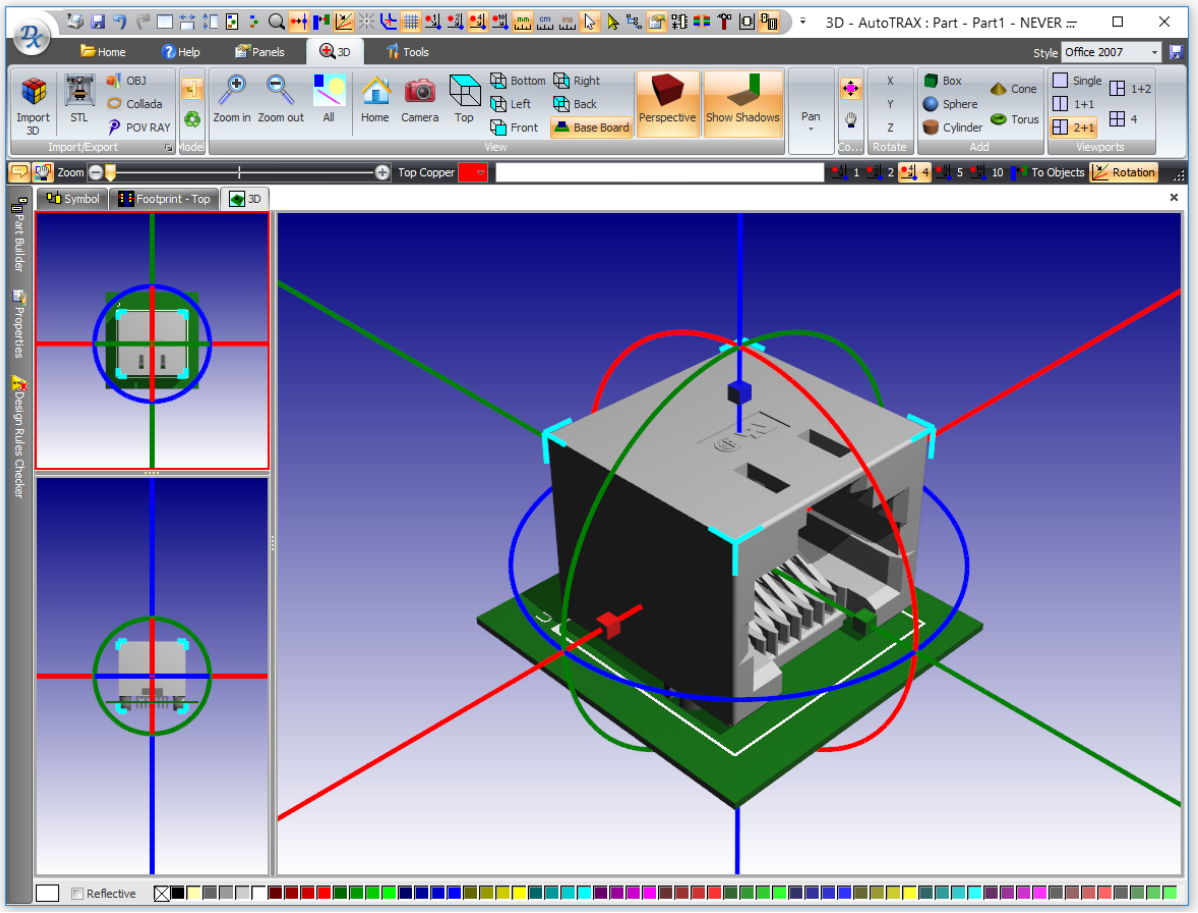
checkbox to rotate the model 90° to switch between Z up and Y up.

Import Size

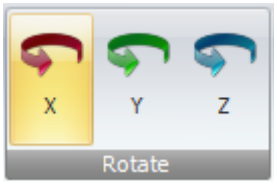
The import size is shown in the information box at the lower left. of the Import 3D Dialog.



3D Model Imported



When you import the model, the position or rotation may not be to your liking.

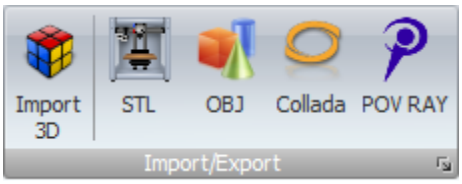
To rotate the model about an axis, select it and click on any of the 3 axis rotate button. 

To move the imported model, select it and hold down the left mouse button and drag on either the red, green or blue axis in the 3D viewport.

Drag the red, green or blue boxes to scale the model.

Drag on the red, green or blue circles to rotate the model.

The import/export ribbon group contains several useful commands for exporting 3-D.



Click on the 

button to export a STL file.

Exporting a STL File

Click the 

will export the 3-D to a Wavefront OBJ file.

Export a Wavefront OBJ File

The 

will export the 3-D to a Collada file.

Exporting a Collada 3D File

The 

button will create a POV RAY photo-realistic render file. If you have POV RAY installed then the POV RAY program will start and render your file in 3D.

Exporting and Rendering with POVRAY

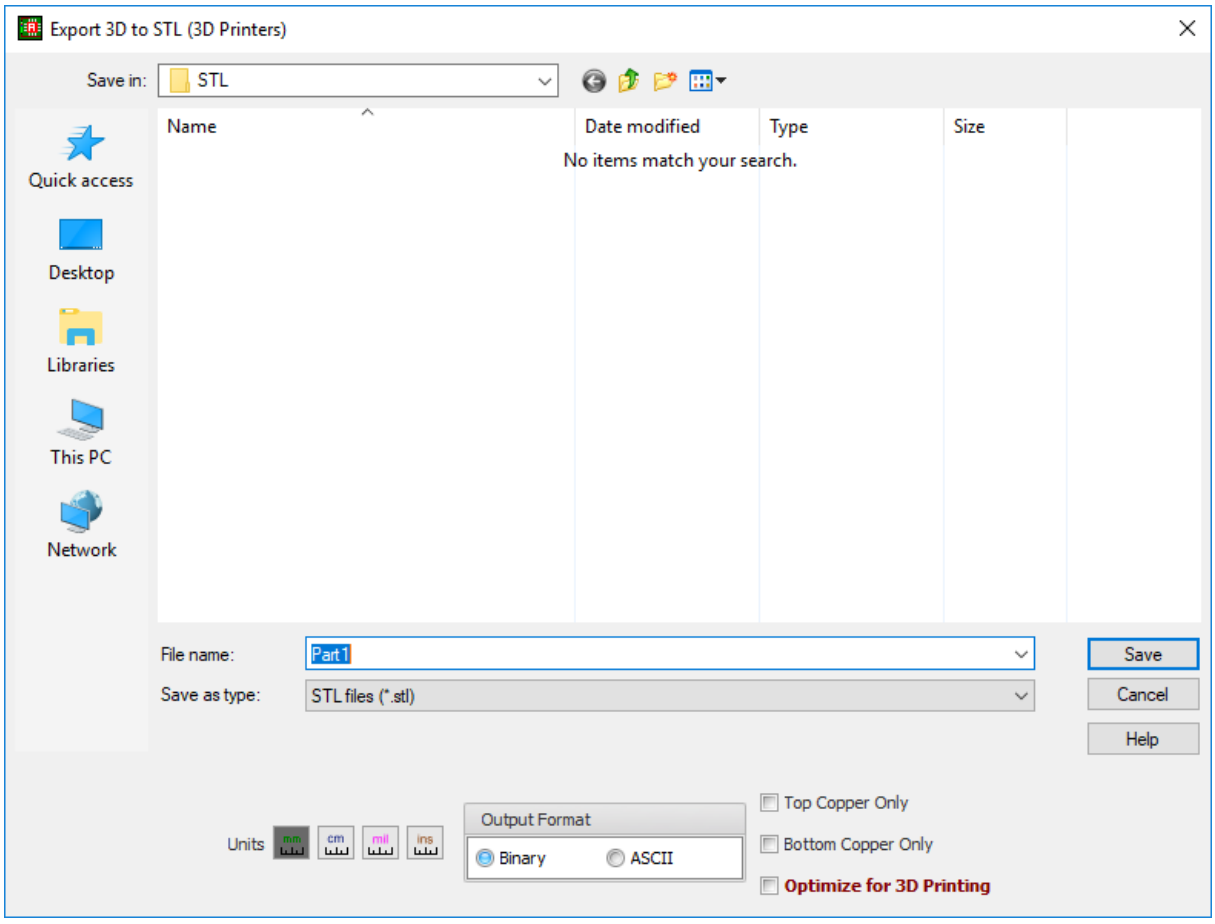
STL (an abbreviation of "stereolithography") is a file format native to the stereolithography CAD software created by 3D Systems. STL has several after-the-fact backronyms such as "Standard Triangle Language" and "Standard Tessellation Language". This file format is supported by many other software packages; it is widely used for rapid prototyping, 3D printing and computer-aided manufacturing.[5] STL files describe only the surface geometry of a three-dimensional object without any representation of color, texture or other common CAD model attributes. The STL format specifies both ASCII and binary representations. Binary files are more common, since they are more compact.

An STL file describes a raw, unstructured triangulated surface by the unit normal and vertices (ordered by the right-hand rule) of the triangles using a three-dimensional Cartesian coordinate system. In the original specification, all STL

coordinates were required to be positive numbers, but this restriction is no longer enforced and negative coordinates are commonly encountered in STL files today. STL files contain no scale information, and the units are arbitrary.

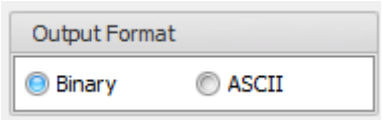
Find Out More...

The Export to STL Dialog



Set the units by click on the appropriate unit button 

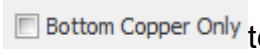
.

Check the appropriate radio button 

to set the output format.

Check 

to export only the top copper.

Check 

to export only the bottom copper.

Check 

to optimize the 3D output for printing.

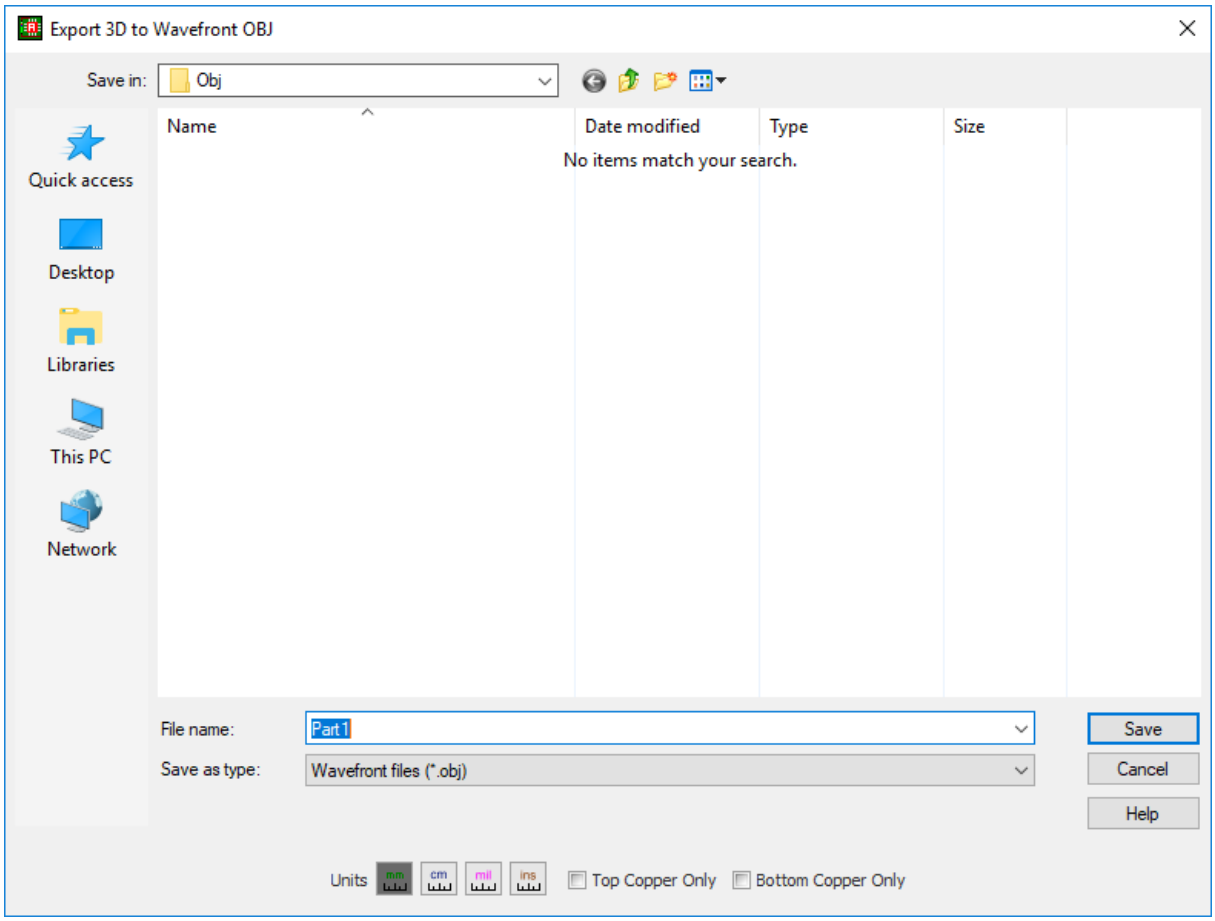
OBJ (or .OBJ) is a geometry definition file format first developed by Wavefront Technologies for its Advanced Visualizer animation package. The file format is open and has been adopted by other 3D graphics application vendors.

The OBJ file format is a simple data-format that represents 3D geometry alone — namely, the position of each vertex, the UV position of each texture coordinate vertex, vertex normals, and the faces that make each polygon defined as a list of vertices, and texture vertices. Vertices are stored in a counter-clockwise order by default, making explicit declaration of face normals unnecessary. OBJ coordinates have no units, but OBJ files can contain scale information in a human readable comment line. You are

An OBJ file may contain vertex data, free-form curve/surface attributes, elements, free-form curve/surface body statements, connectivity between free-form surfaces, grouping and display/render attribute information. The most common elements are geometric vertices, texture coordinates, vertex normals and polygonal faces.

Find Out More...

The Export to OBJ Dialog



Set the units by click on the appropriate unit button 

.

Check 

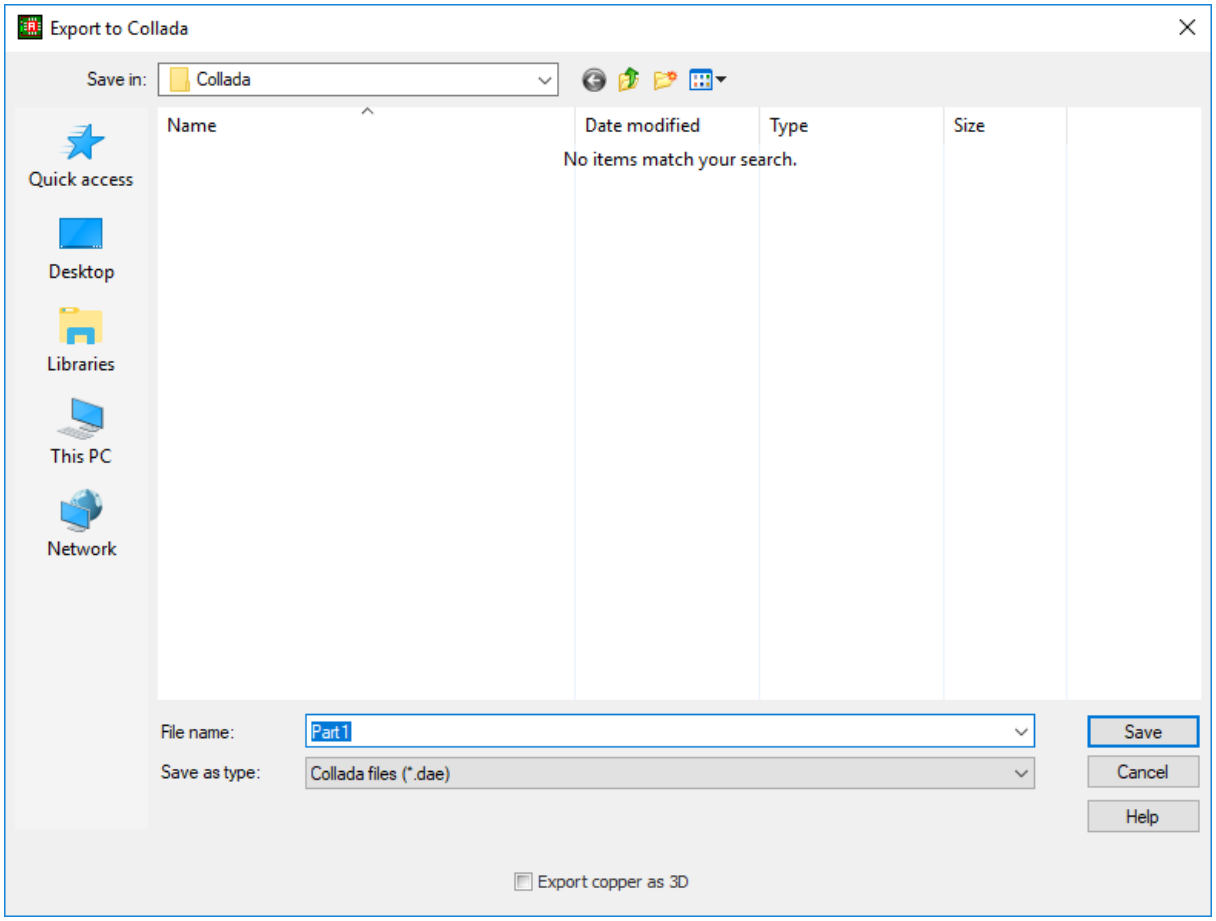
to export only the top copper.

Check Bottom Copper Only to export only the bottom copper.

COLLADA (Collaborative Design Activity) is an interchange file format for interactive 3D applications. It is managed by the nonprofit technology consortium, the Khronos Group, and has been adopted by ISO as a publicly available specification.

Find Out More...

The Export to Collada Dialog

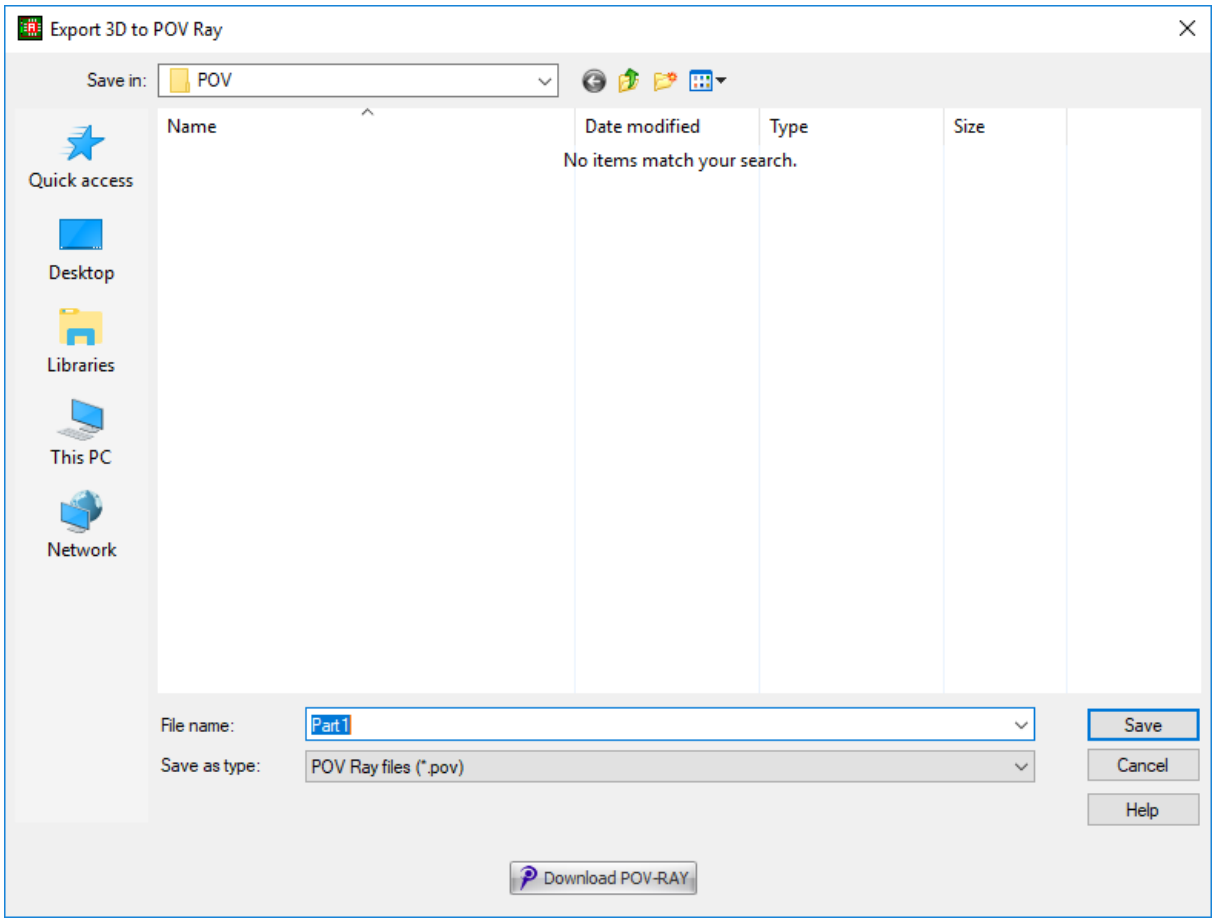


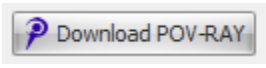
Check the Export copper as 3D to export all copper as 3D. This will make the files considerably larger in size and slower to render.

The Persistence of Vision Ray Tracer, or POV-Ray, is a ray tracing program which generates images from a text-based scene description, and is available for a variety of computer platforms.

Find Out More...

The Export to POVRAY dialog



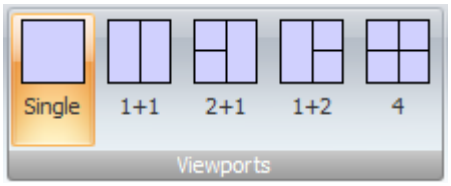
Click the 

to download the POV ray tracer.

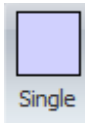
If there is more than 1 view visible then the active view has a red border.

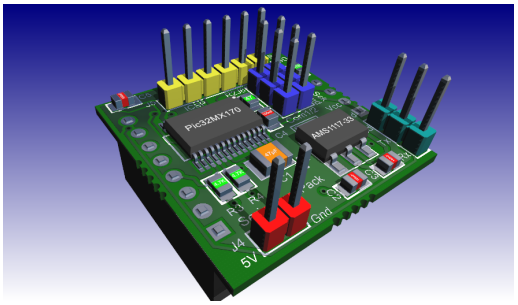
View can have separators that you can drag to re-size.

You have 5 preset view configurations.

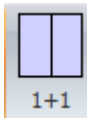


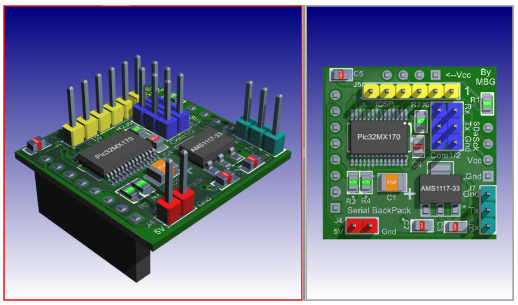
Single



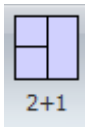


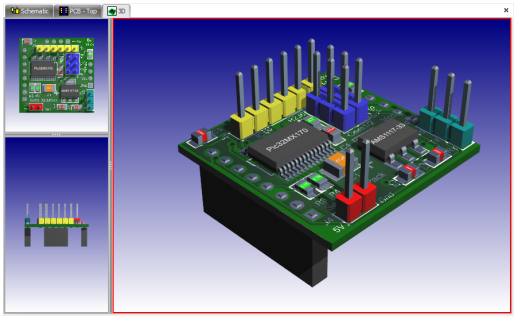
1+1



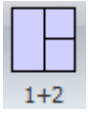


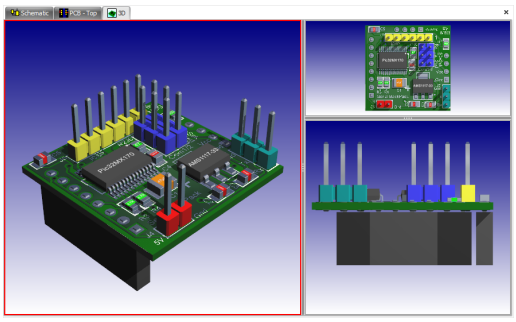
2+1



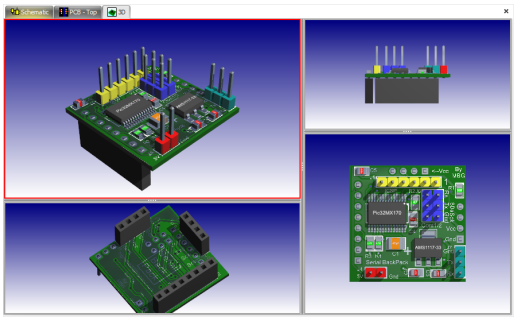


1+2





4

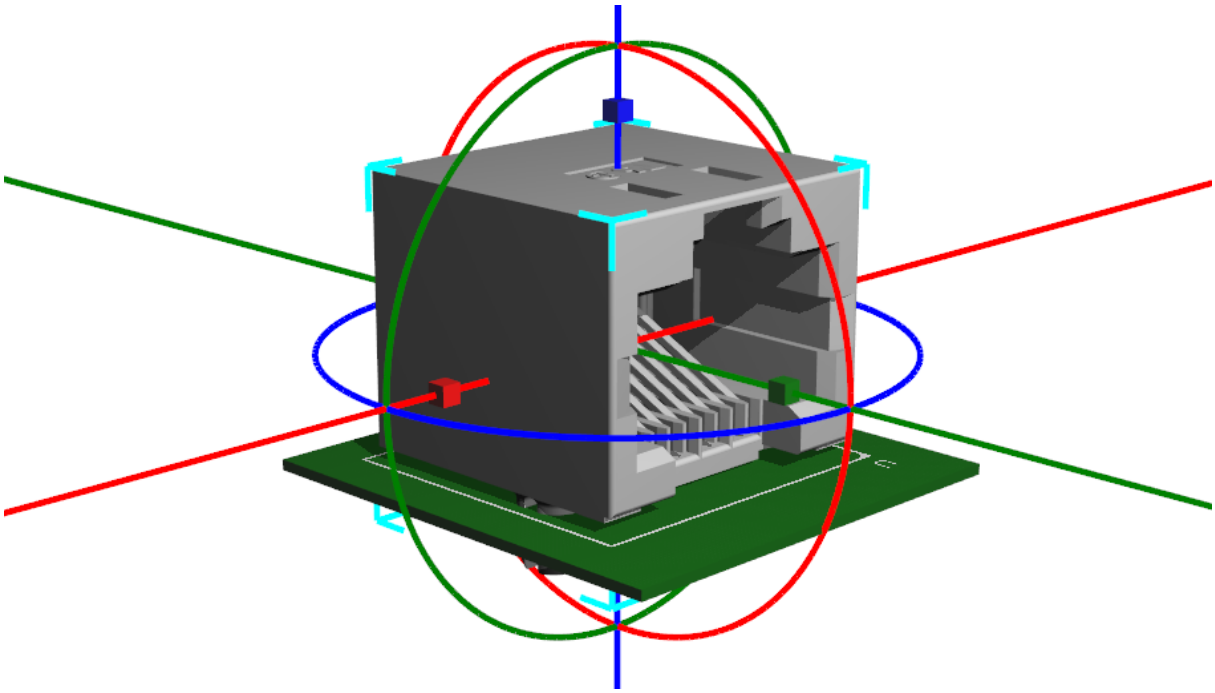


1.2.6.11.38.5 Moving, Rotating and Scaling 3D Objects

To move, rotate or scale the 3-D object first select it by clicking on it. You must be in pick mode rather than view mode.

You can quickly switch between pick and view mode by pressing the space bar.

When selected a collection of transform manipulators appears as shown below.



A selected Object with Manipulators

The X axis is coloured red.

The Y axis is coloured green.

The Z axis is coloured blue.

Moving the selected object

To move the object along an axis, hold the mouse cursor over either the red, green or blue straight lines and drag the mouse. The selected object will follow.

Rotating the selected object

To rotate the object around one of the axes click the left mouse button over either the red, green or blue circles and drag the mouse. The selected object will rotate about that axis.

Scaling the select object

To scale the object along an axis hold down the mouse over either the small red, blue or green boxes and then drag the mouse. The selected object will scale in that direction.

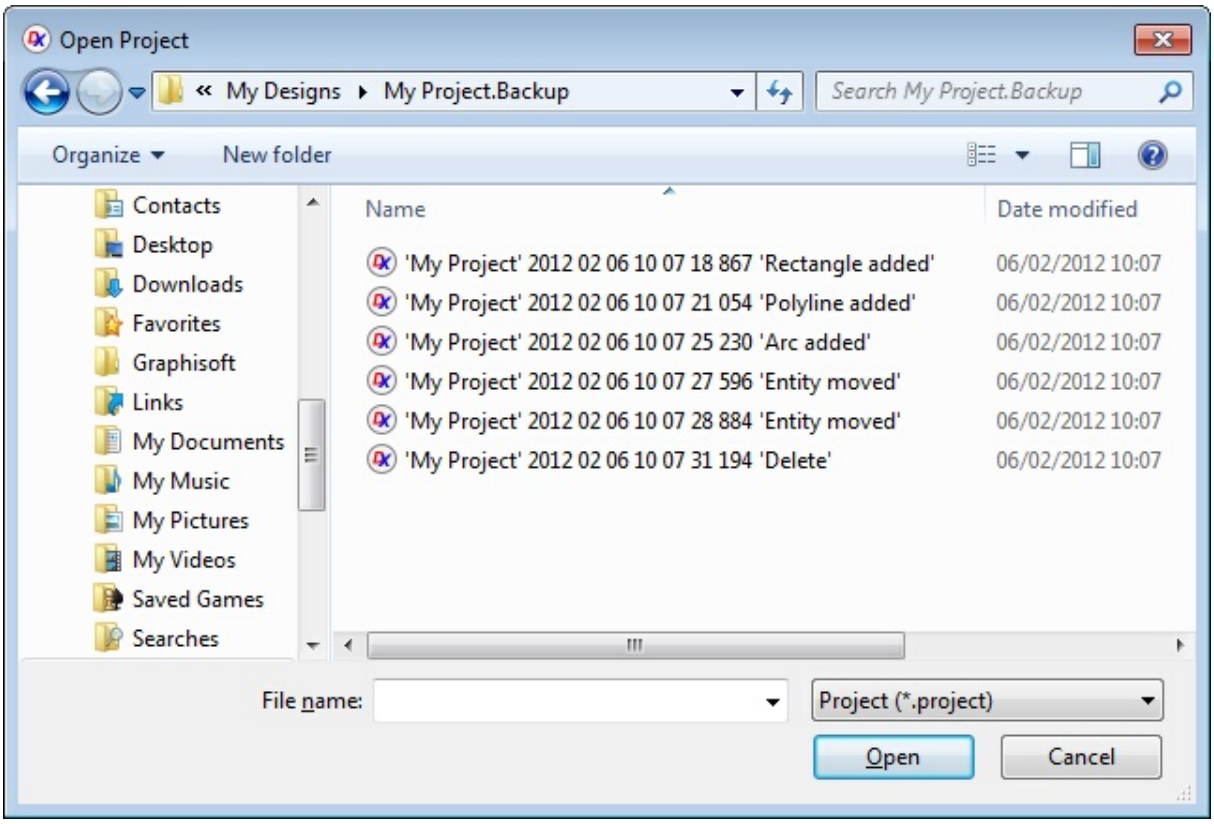
1.2.6.12 Automatic Backups

AutoTRAX DEX automatically archives your work in a directory with the same name as your project but with the '.Backup' extension added. For example if you project is called 'My Project', backups are saved to 'My Project. Backup' as shown below.

The backups have the save name with the date appended and a short description of the action that caused a backup to be made.

It is a good idea to use some 3rd party automatic backup software to backup this directory, giving you full confidence that none of your work will be lost.

AutoTRAX DEX will never delete any of these files even if you undo changes you make.

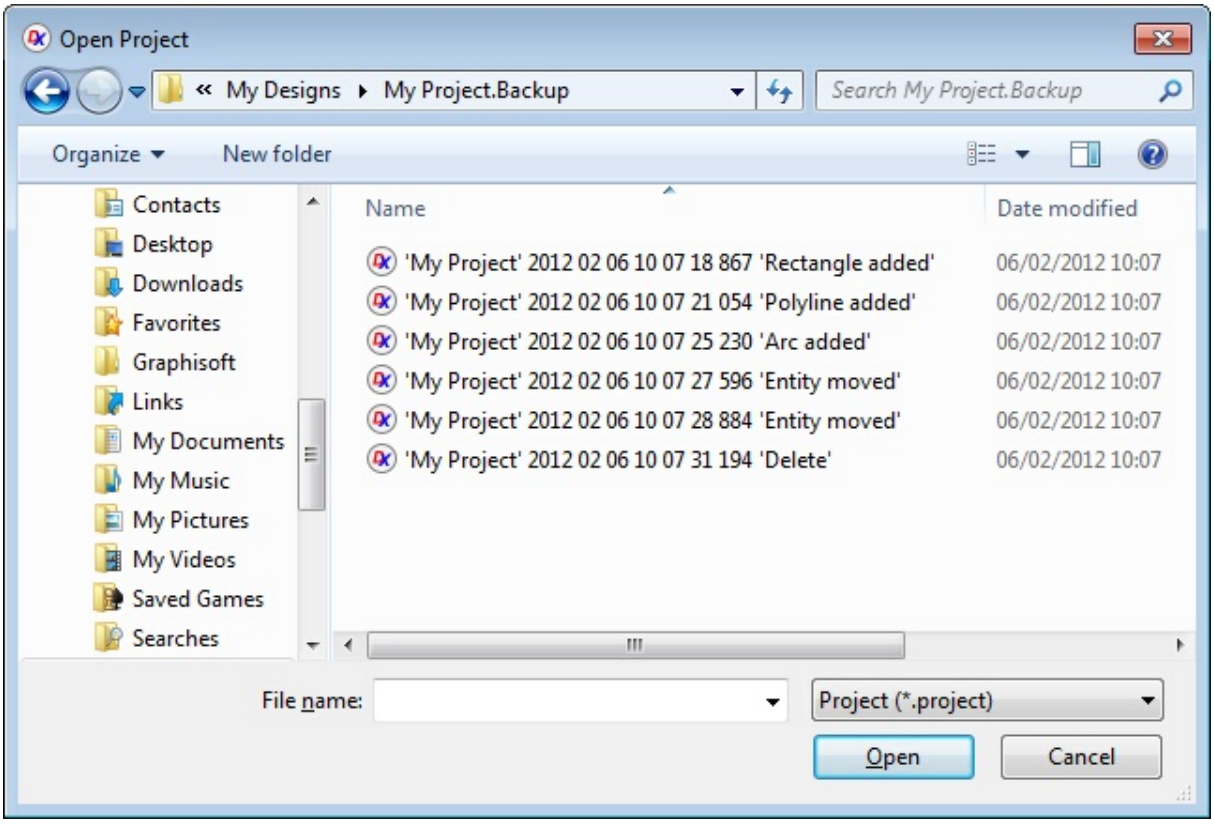


List of Automatically archived copies of the current project.

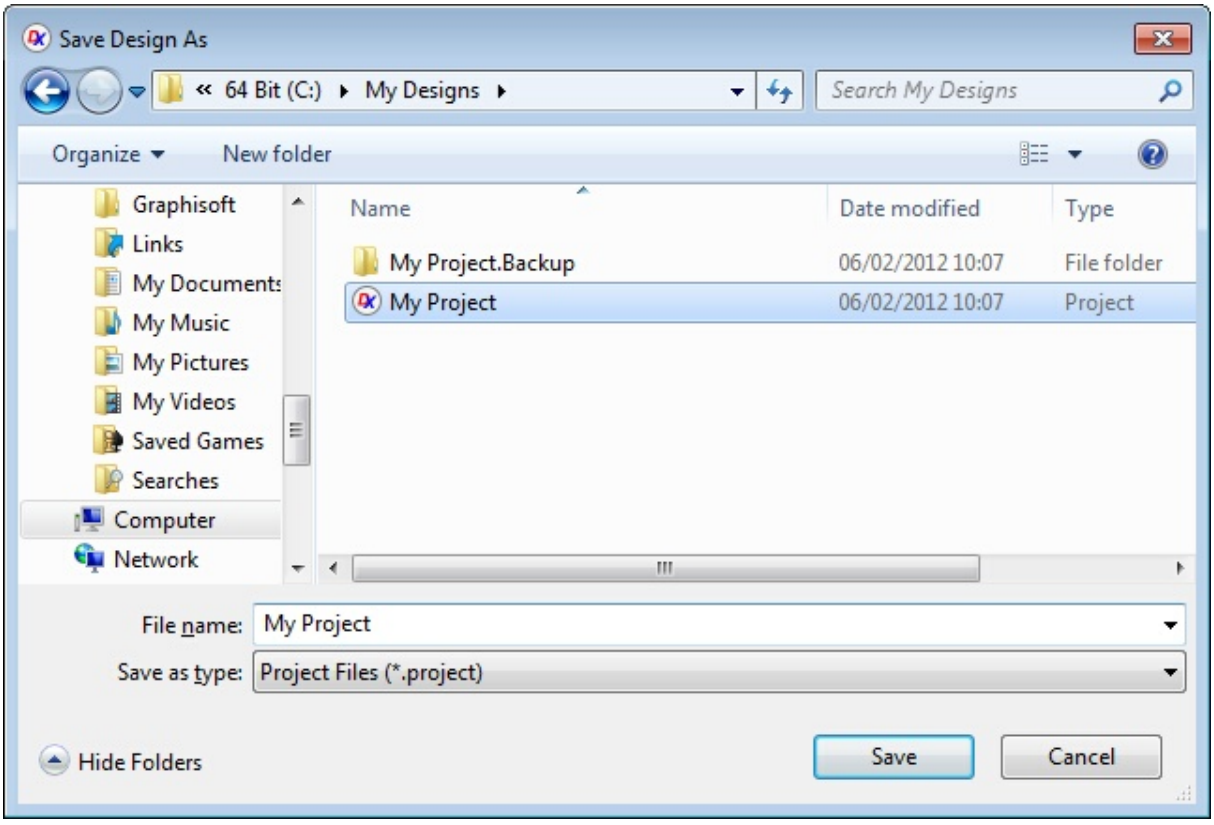
1.2.6.13 Restoring Backups

To restore a backup file open it and then save it. **!!To maintain the original backup, save the project to a different name using the Save-As button from the Home Tab of the Ribbon menu!!**

Opening a backup...



Saving the backup as your original...



1.2.6.14 Sheets

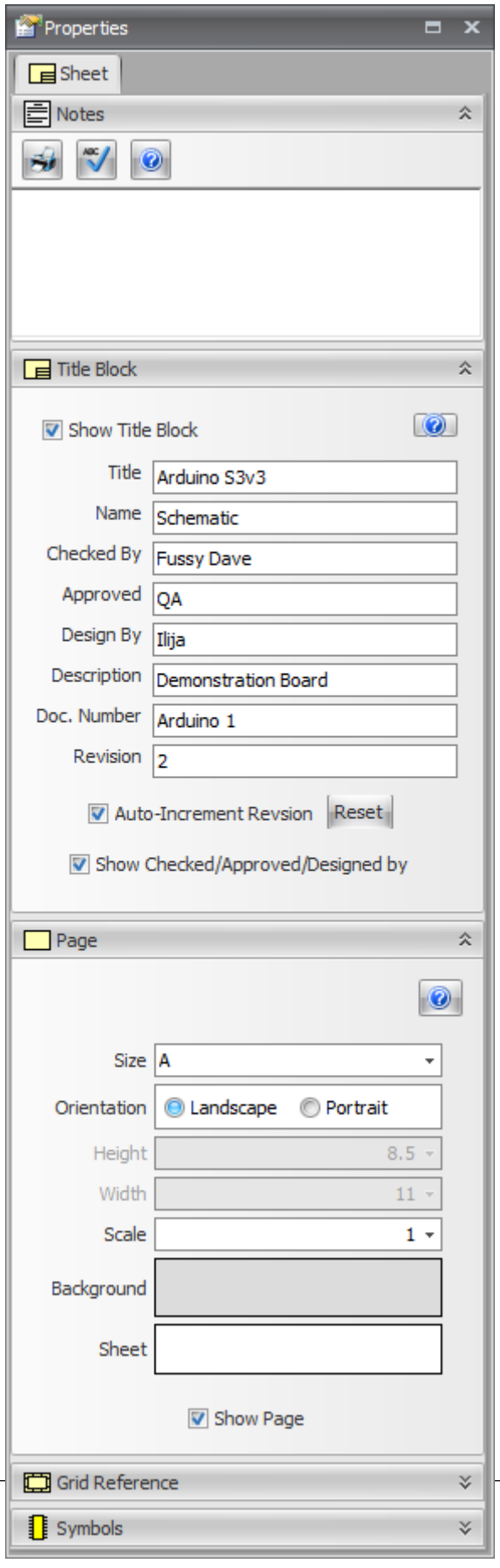
AutoTRAX DEX can contain the following types of sheets in addition to the PCB.

- Schematic Sheets
- Text Sheets
- PDF Sheets

1.2.6.14.1 Setting Sheet Sizes

To set the sheet size use the Sheet properties dialog in the Properties panel.

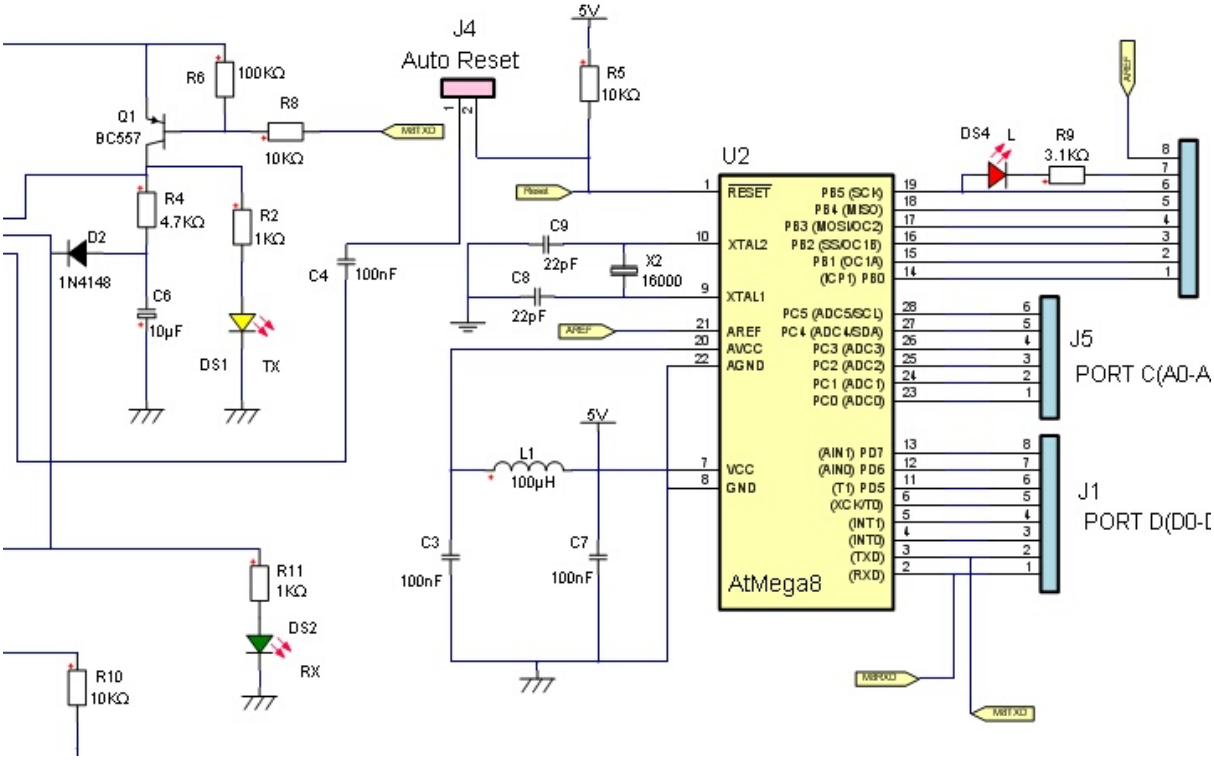
The Sheet properties dialog is visible when nothing is selected.



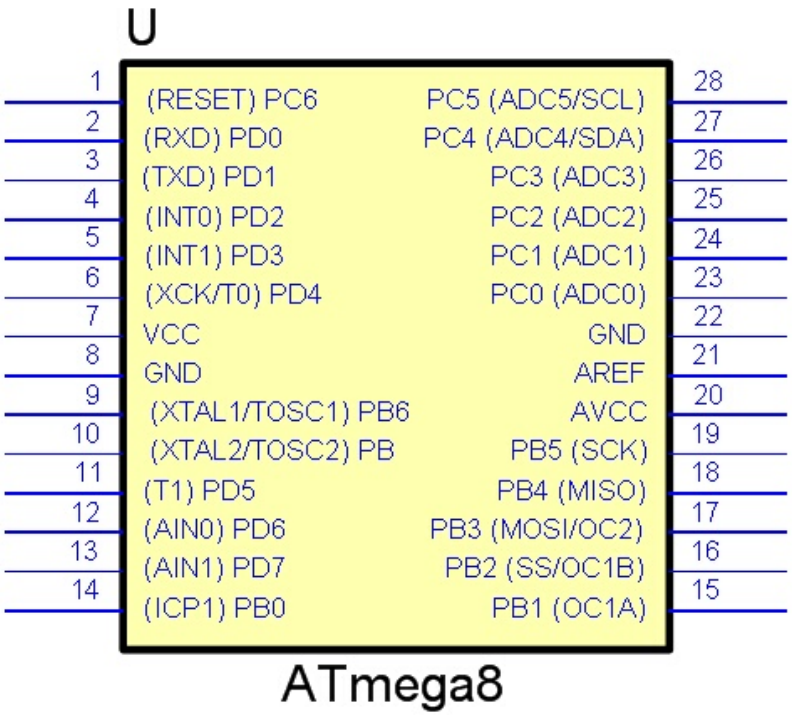
1.2.6.14.2 Schematic Sheets

A schematic sheet contains a symbolic representation of the design of your project or of a part. See the chapter on schematics for more details.

A typical schematic for a project



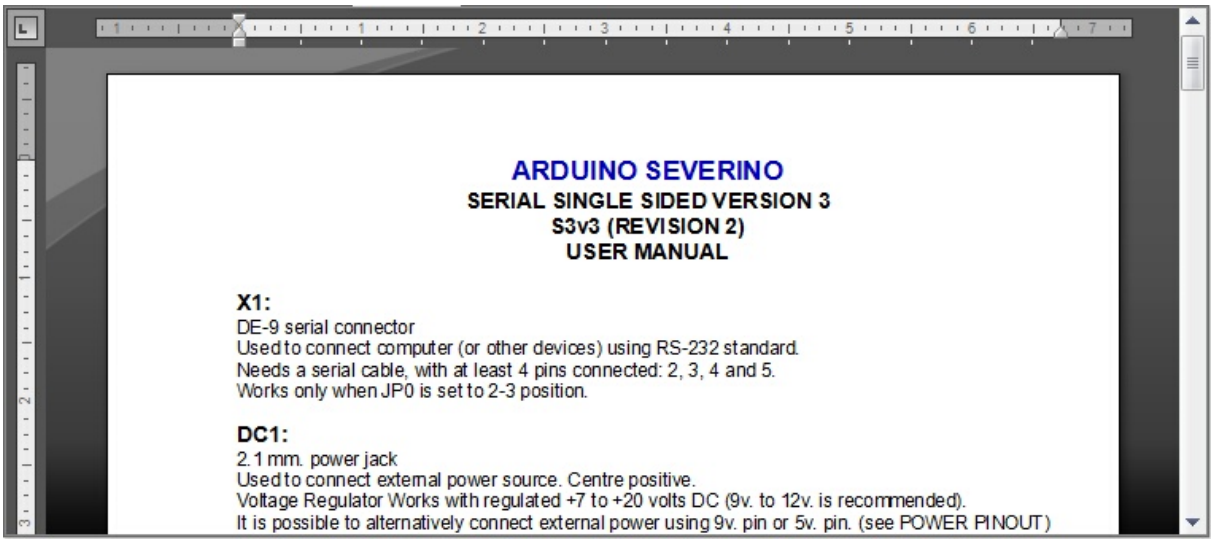
A typical schematic sheet for a part. (It's symbol)



1.2.6.14.3 Text Sheets

You can add text sheets to your design or parts. These text sheets can be used to add documentation or anything else you like. Full text formatting features are available. See the text sheets chapter for more details.

A Typical Text Sheet (Document).



1.2.6.14.4 Hierarchical Layout

Schematic sheets and text sheets are arranged in an hierarchy.

Each schematic sheet or text document can have zero or more child schematic sheets and text documents respectively.

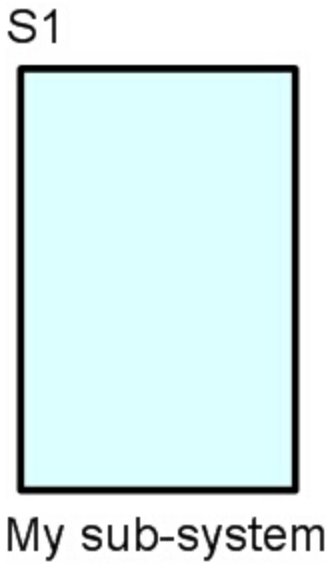
As many schematics sheets and text documents can be added as desired or necessary. These can be added using either the Project Panel or the Add Ribbon menu.

1.2.6.14.5 Sub-Systems

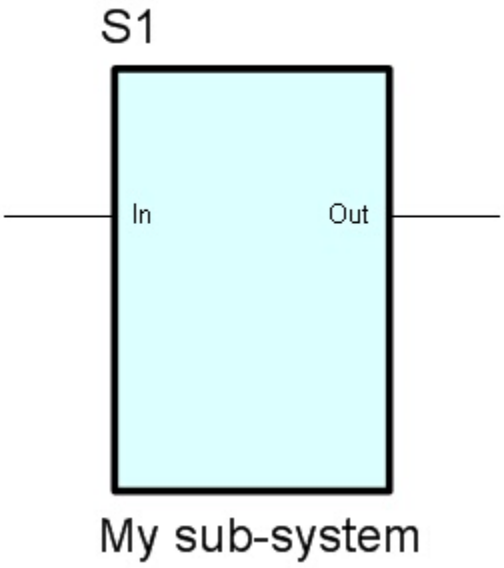
A sub-system in AutoTRAX DEX is a schematic sheet in a project that can be referenced from another schematic sheet using a sub-system reference symbol. Below is a sub-system placed on a schematic. It is labeled S1 and refers to schematic 'My sub-system'.

A sub-system can, optionally, have ports (terminals) on their boundaries which connect to ports in the sub-system schematic.

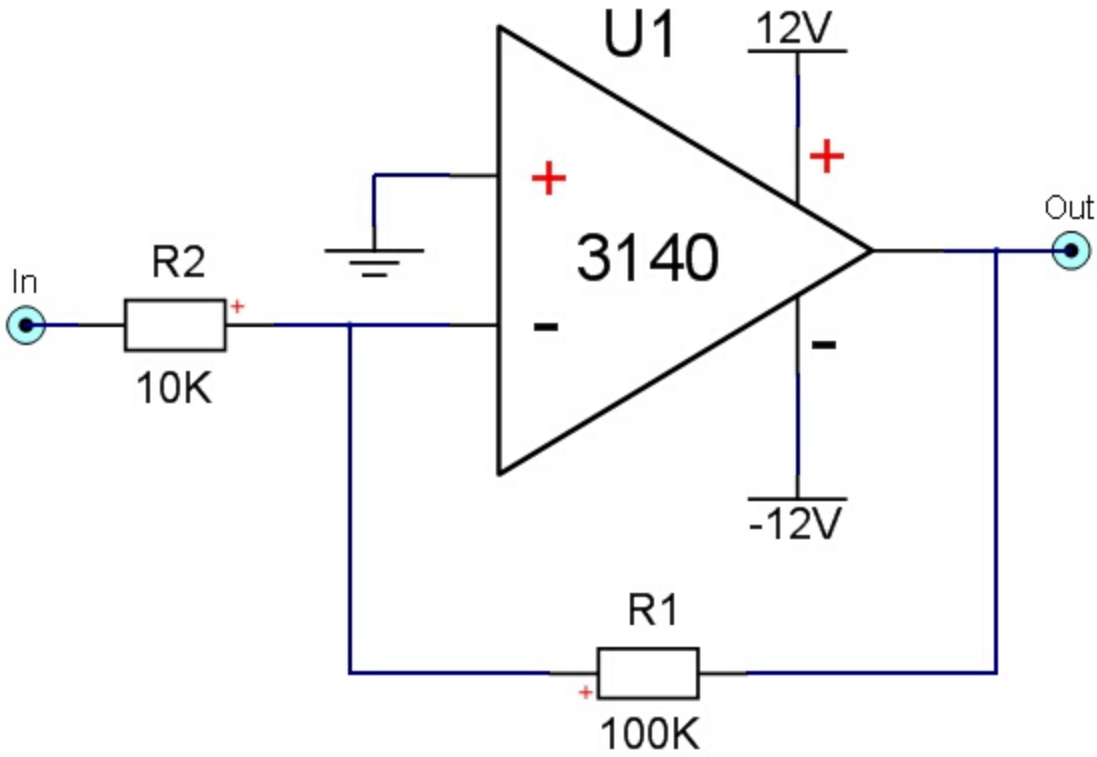
The example on the right, S1 has 2 terminals, In and Out. In the 'My sub-system' schematic, the 2 ports are connected to the input and output of the op-amp. So S1 symbolizes an amplifier while the 'My sub-system' schematic implements it. This allows you to break down a design into more logical parts.



Sub-system reference symbol without terminals



Sub-system reference symbol with terminals that connect to ports on the 'My sub-system' schematic



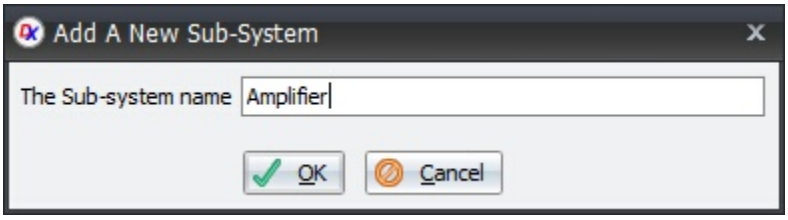
Contents of the 'My sub-system' schematic.

1.2.6.14.5.1 Adding New Sub-Systems

To add a sub-system to a schematic click the Add→Sheets→ 

button.

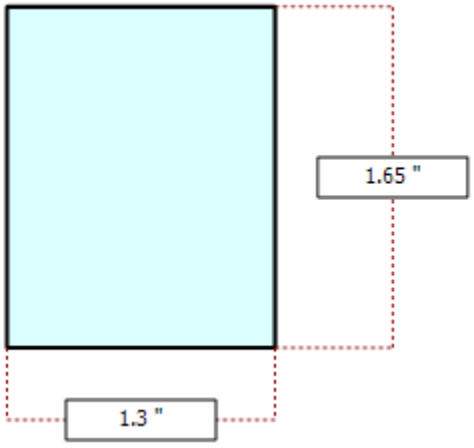
You will then see the following dialog box.



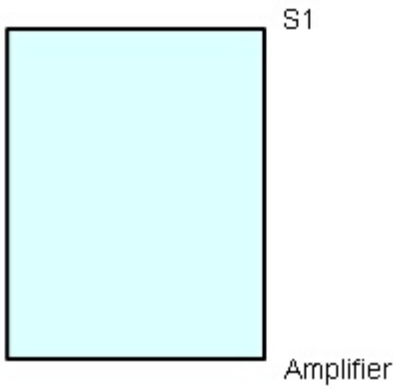
Enter the name of the new sub-system. In this case Amplifier and click the **OK** button.

Move the mouse inside the graphical sheet's viewport. You will see the point cross follow the mouse. ***Left-click*** when the point cross is where you want to start a rectangle or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to exactly place the starting point.

Now drag the mouse and you will see the border change in size. ***Left-click*** when the point cross is where you want to end the **Sub-System** or press the ***Enter*** key followed by the X value, ***Enter*** key, the Y value, and then ***Enter*** to exactly place the end point.



The **Sub-System** reference is shown below.



Double-clicking inside the blue rectangle for the sub-system reference will open the sub-system's schematic sheet.

1.2.6.14.5.2 Opening Sub-systems

To open a **Sub-System**, ***double-click*** on its sub-system reference symbol in the schematic or ***double-click*** on it in the project panel.

1.2.6.14.5.3 Adding Terminals to a Sub-system Reference

There are 2 different ways to add terminals to a sub-system reference.

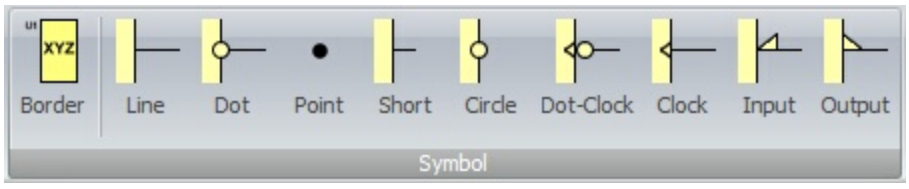
- Add terminals directly to the sub-system reference.
- Open the sub-system reference's schematic by ***doubling-clicking*** on or in the sub-system reference's border and adding ports.

Adding terminals directly to the sub-system reference

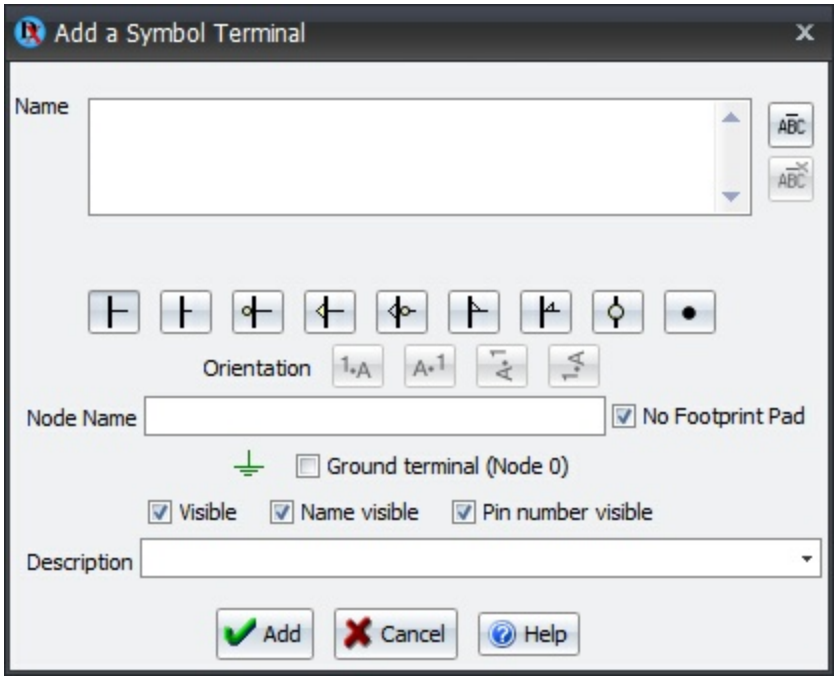
To add terminals directly to the sub-system reference:

1. Select the sub-system reference by clicking on or in it's border. (If there is only 1 sub-system reference or part in the schematic then you do not need to do this as AutoTRAX DEX knows what to add the terminal to.)

2. Click on one of the symbol terminals in the Symbols group in the Add ribbon menu.



The symbol terminal dialog box will appear.



3. Enter the Name and click the Add button.

4. Now move the mouse around the sub-system reference. You will see the terminal move around and try to follow the mouse. ***Left-click*** to finally place the terminal.

Repeat 1-4 to add more terminals.

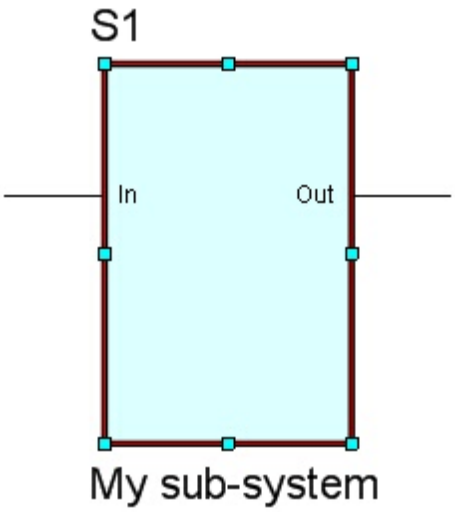
When you add terminals to a sub-system reference, ports are automatically added to the sub-system reference's schematic.

For more about symbol terminals click here.

1.2.6.14.5.4 Editing Sub-Systems

You can edit a sub-subsystem's reference by:

1. Dragging the sub-subsystem reference's value or ID.
2. ***Double-click*** on the sub-subsystem reference's value or ID to edit its value.
3. Drag any of the terminals. As the mouse moves, the terminal will follow the mouse but will always remain attached to the sub-subsystems reference's border.
4. Re size the sub-subsystem reference's border by selecting it using sub-pick from the menu or pressing down the ***CTRL*** key while selecting the sub-subsystem reference's border by clicking on it. You can then re-size the border by dragging the border's manipulator points.



Drag any of the blue rectangles to re-size the border. The terminals will move to remain on the border.

If you delete a terminal from a sub-subsystem reference then all equivalent terminals is all sub-subsystems references that reference the same schematic will be deleted. The port in the schematic will also be deleted.

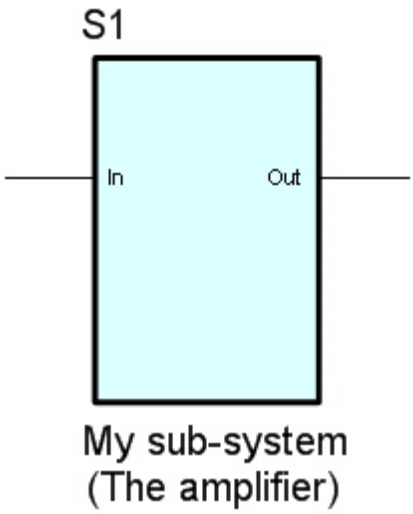
If you add a terminal to a sub-subsystems reference then all sub-subsystems references that reference the same schematic will have a terminal added. A corresponding port will be added to the sub-subsystem reference's schematic.

If you delete a port in a schematic then the corresponding terminal will be removed from all sub-subsystem references that reference that sheet.

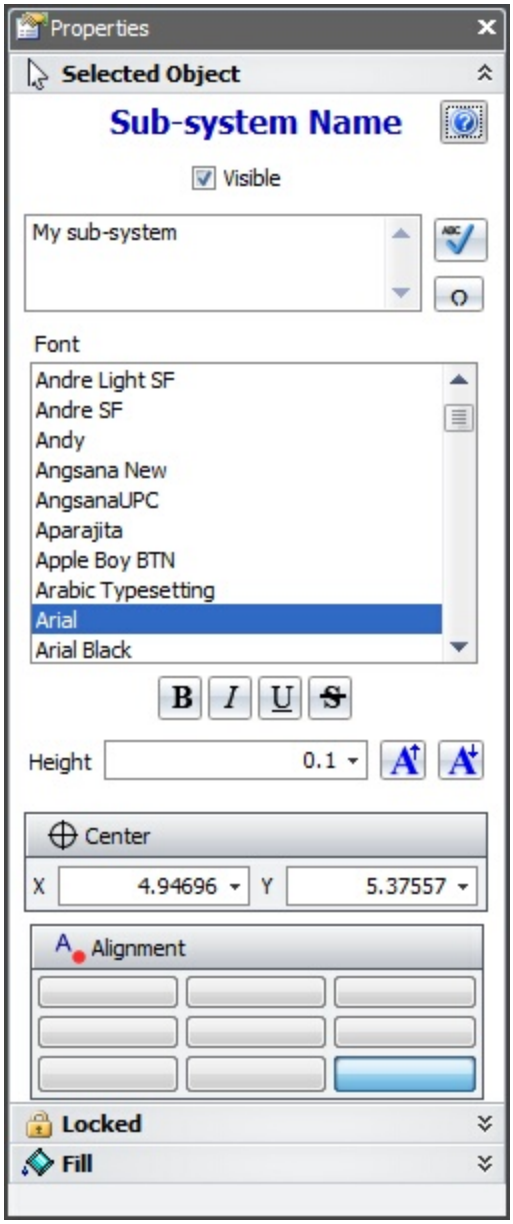
1.2.6.14.5.5 Sub-Systems Names

The sub-system name is linked to the name of the sub-system schematic and displays it's name. If you change the name of the sub-system schematic then the text displayed by the sub-system's name changes.

The name can be multi-lined.



A multi-lined sub-system name



1.2.6.14.5.6 Using Sub-Systems

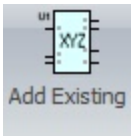
Sub-systems are a great way to break down a complex design into management check. You can use a top down method, a bottom up method or a mixture of both.

Top Down

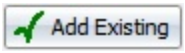
To partition a design, on a schematic add 1 or more Sub-Systems.

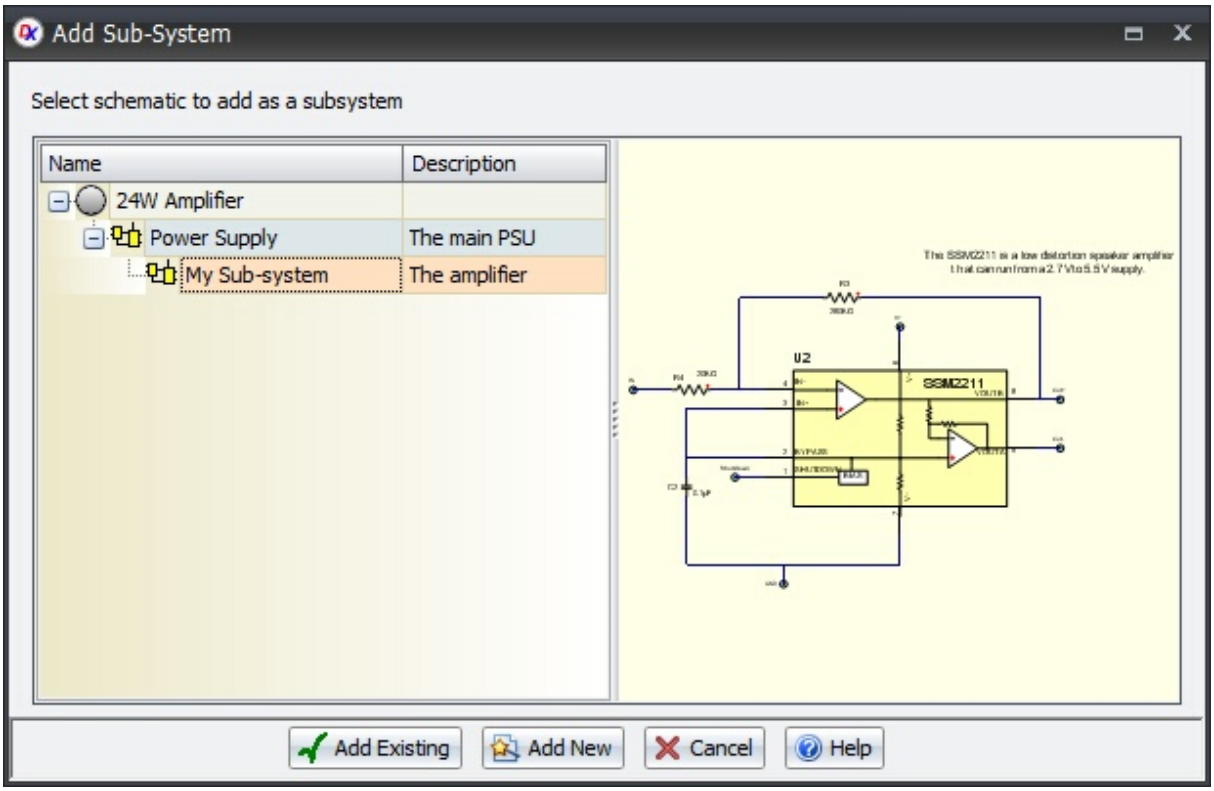
Bottom Up

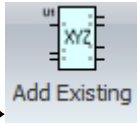
Add a new schematic and create your bottom level design.

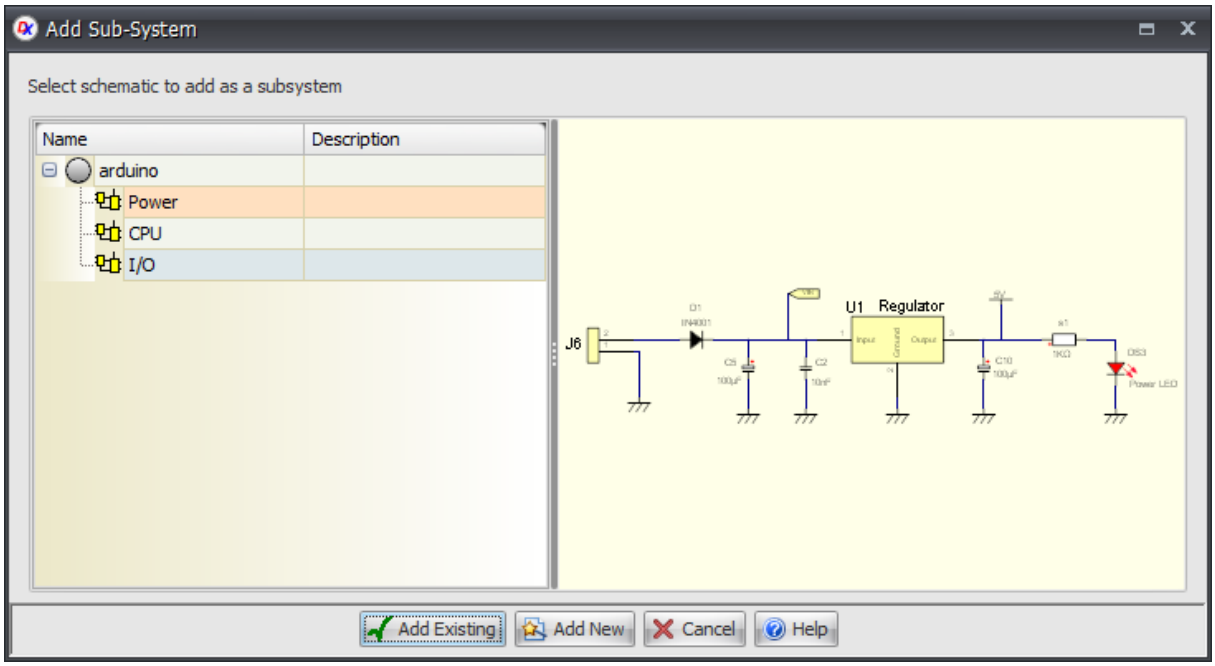
In another schematic from which you want to reference your new bottom level design, add a sub-system from the **Add | Add Existing** menu. 

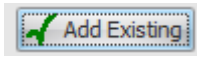
The **Add Sub-System** dialog box shown below will appear.

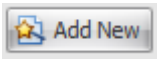
Select the sub-system and click on the  button.

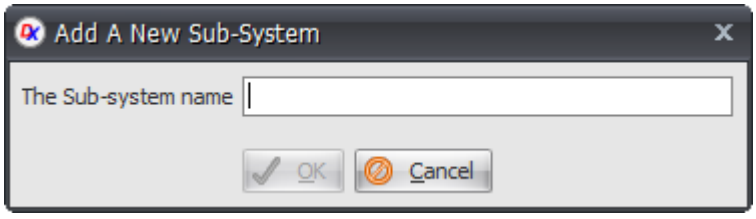


To add a sub-system reference the Add→Sub→  ribbon button group.



Select the sub-system and click the  button to add it.

Click the  button to create a new sub-system and add a reference to it. Enter its name.



1.2.6.14.6 Graphical Sheets

Schematic sheets, symbol sheets and PCB sheets are all graphical sheets.

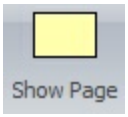
Graphical sheets can optionally consist of:

- A Page Border
- A Grid Reference
- A Title Block

1.2.6.14.6.1 Page Borders

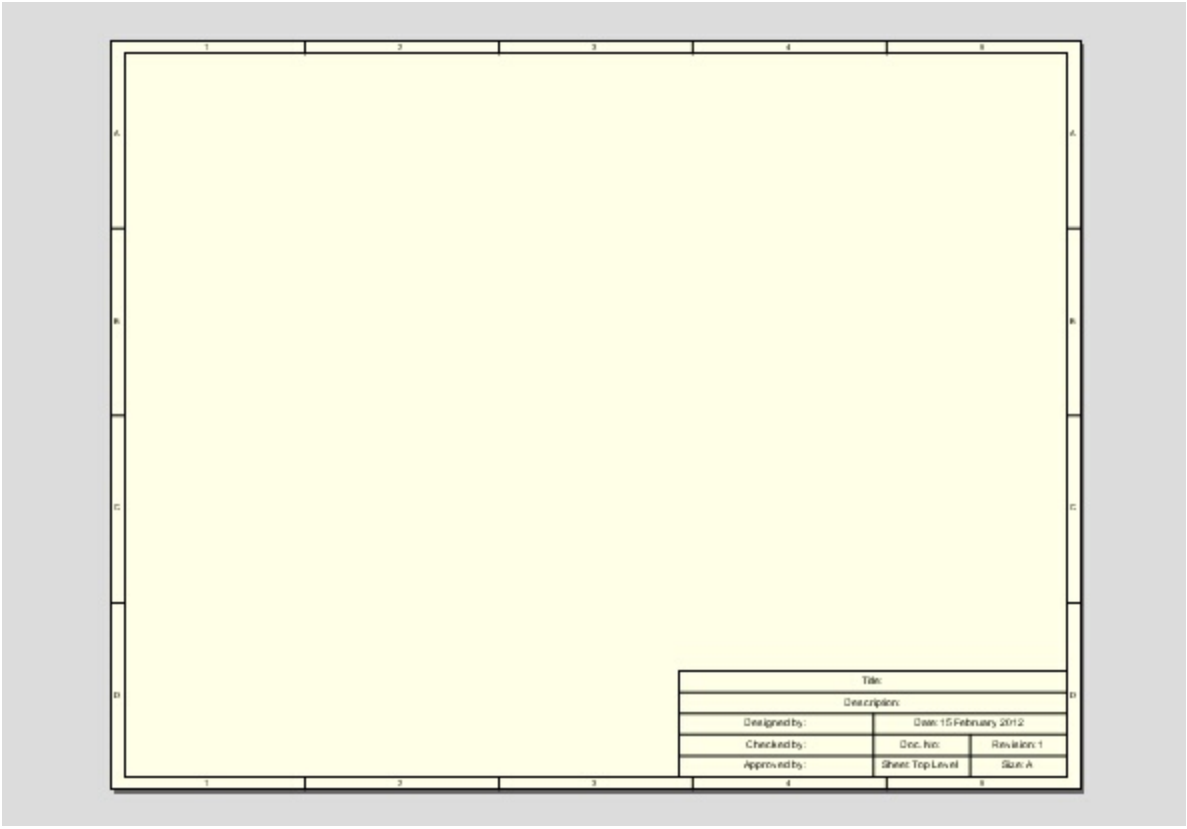
You can have a page border appear on a schematic or a PCB sheet. Page borders can be optionally decorated with a Grid Reference and/or a Title Block.

To view/hide a page border either:

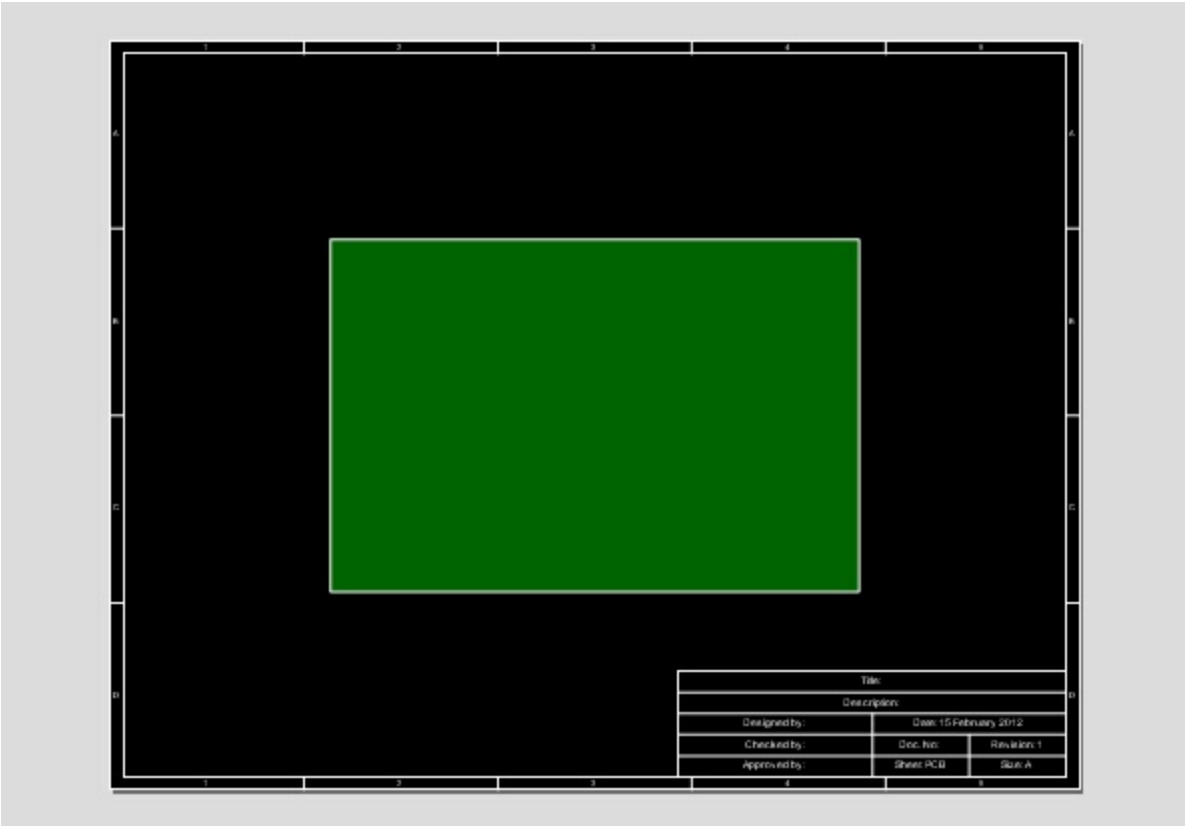
- ***Right-click*** and select **Show Page** from the context menu or
- Click on the 

button in the **View→Page** menu or
- Click on the Show Page checkbox in the Page Properties Panel.
- Use the Sheet Settings Pop-up.
- Use the Properties Panel (with nothing selected).

You can edit the size and other attributes of the page using the Sheet Settings Pop-up or the Properties Panel.



A schematic with a page border, grid reference and a title block.

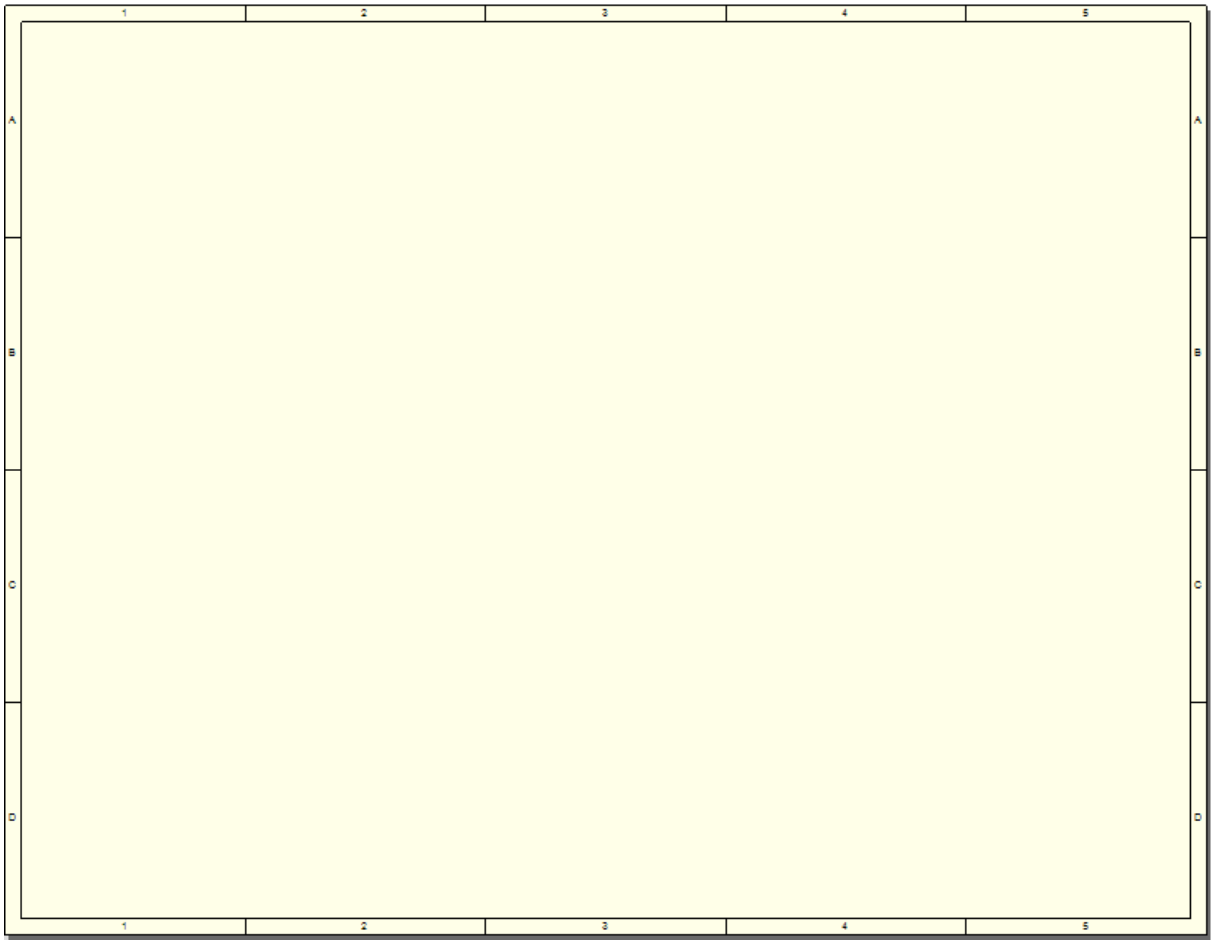


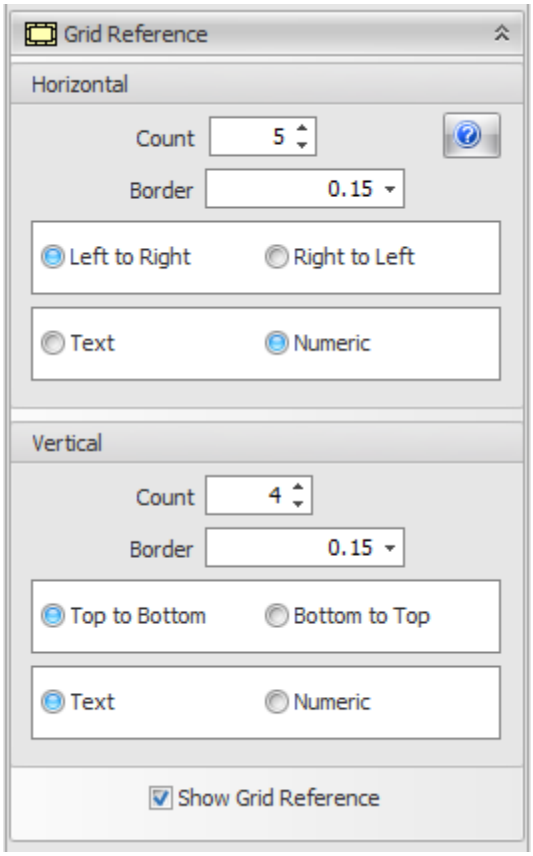
A PCB sheet with a page border, grid reference and a title block.

1.2.6.14.6.2 The Grid Reference

The grid reference is a set of reference boxes placed around the edges of the page as shown below.

You can toggle the visibility of the grid reference using the View tab.





Click  to displays this help topic.

Horizontal

This group sets the parameters for the horizontal grid cells.

Vertical

This group sets the parameters for the vertical grid cells.

Count

This sets the number of cells.

Border

The size of the grid cell from the edge towards the center of the sheet.

Text

If checked then the cell value is displayed as a character from 'A' onwards.

Numeric

If checked then the cell value is displayed as a number from '1' upwards.

Left to Right

If checked then the cell values increases from the left to the right of the sheet.

Right to Left

If checked then the cell values increases from the right to the left of the sheet.

Top To Bottom

If checked then the cell values increases from the top to the bottom of the sheet.

Bottom to Top

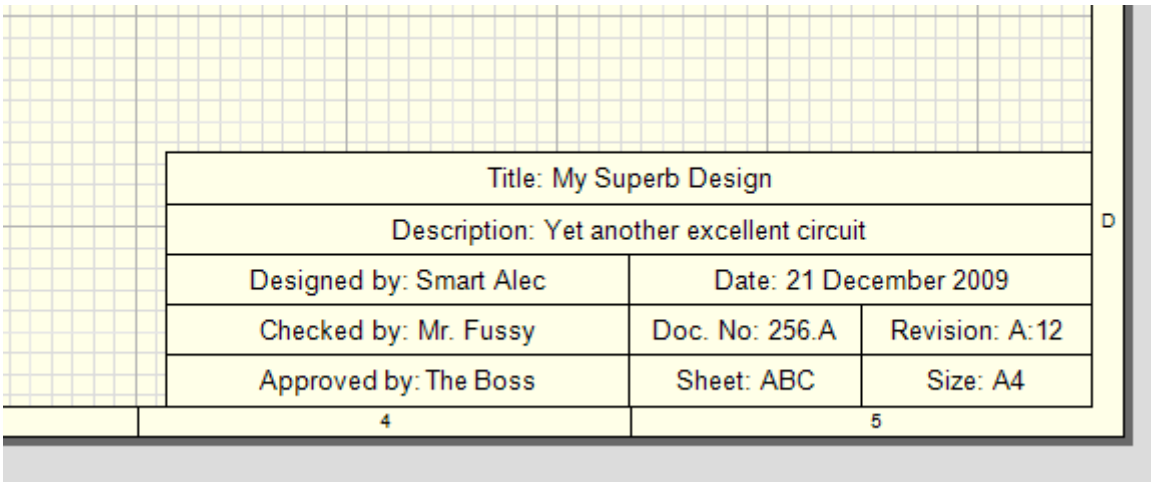
If checked then the cell values increases from the bottom to the top of the sheet.

Show Grid Reference

This will hide/show the grid reference.

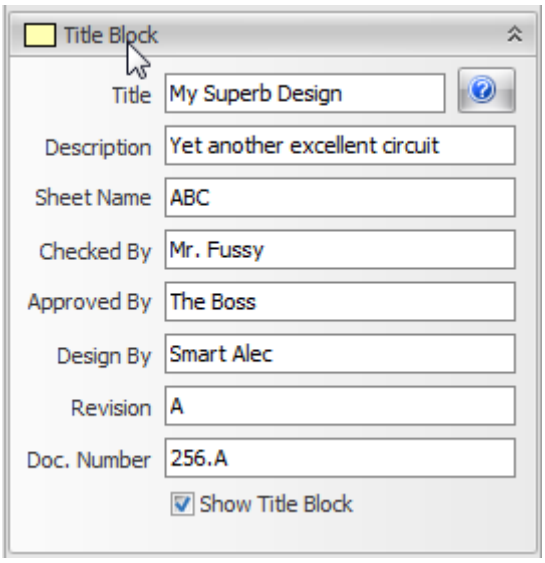
1.2.6.14.6.3 The Title Block

The title block is a rectangular region displayed at the bottom right of the sheet as shown below.



The date is the date of the last save in your local time. If the same sheet is displayed in a different time zone then the date displayed is adjusted.

You can edit the parameters for the title block by displaying the Properties Panel. If nothing is selected on the sheet you will see the Title Block dialog.



 Clicking this displays this help topic.

Title

This is the title for the sheet.

Description

This is a more detailed description of the contents of the sheet.

Sheet Name

The name of the sheet. It is also the name displayed in the Projects Panel and in the viewport tab title.

Checked By

The name of the person who checked the design.

Approved By

The name of the person who approved the design.

Design By

The name of the person who designed the sheet.

Revision

The revision prefix. This is displayed along with the save count. The same revision text is displayed on all sheets.

Doc. Number

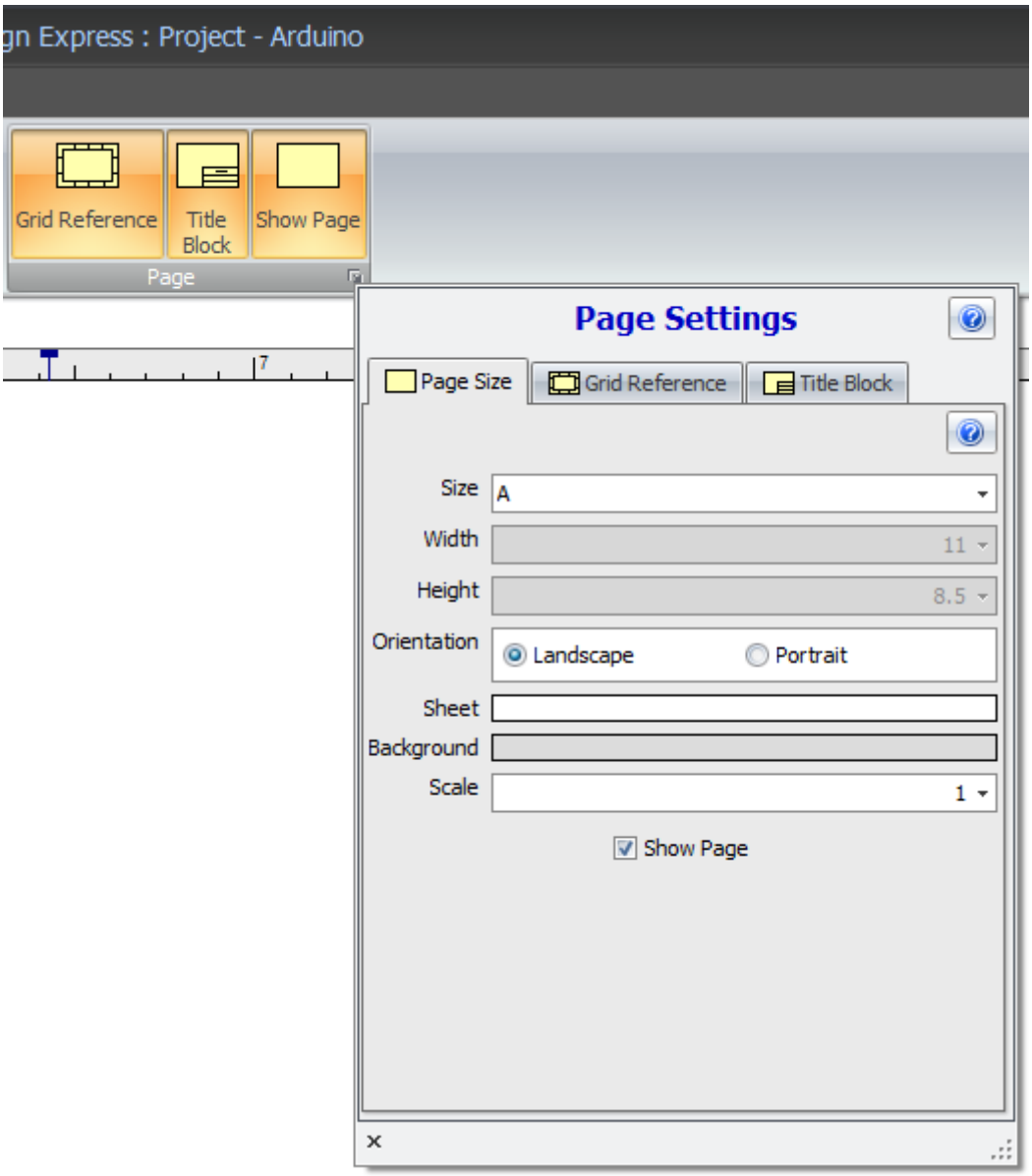
The document number. The same number is displayed on all sheets.

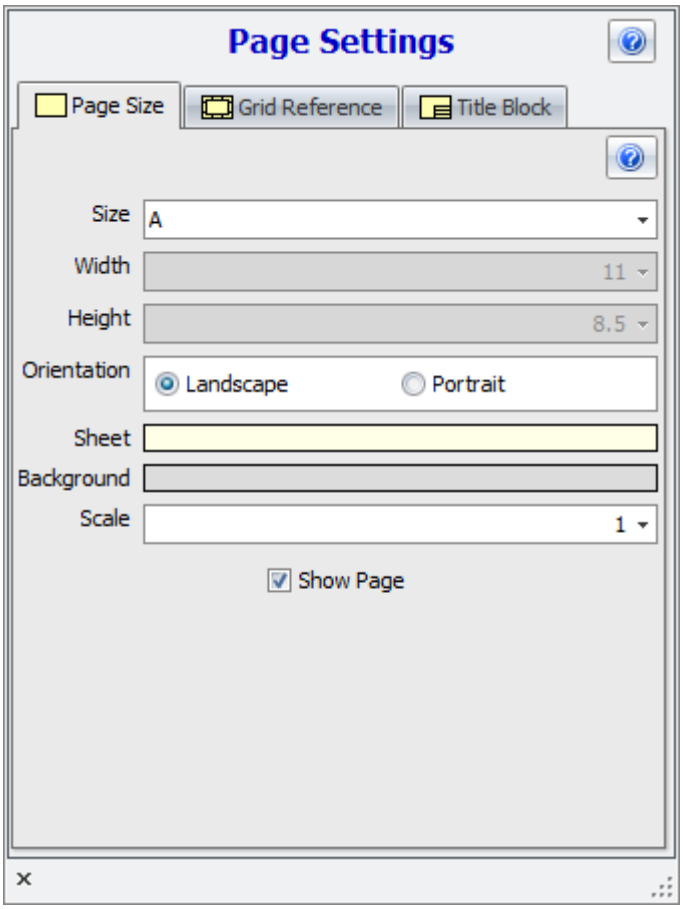
Show Title Block

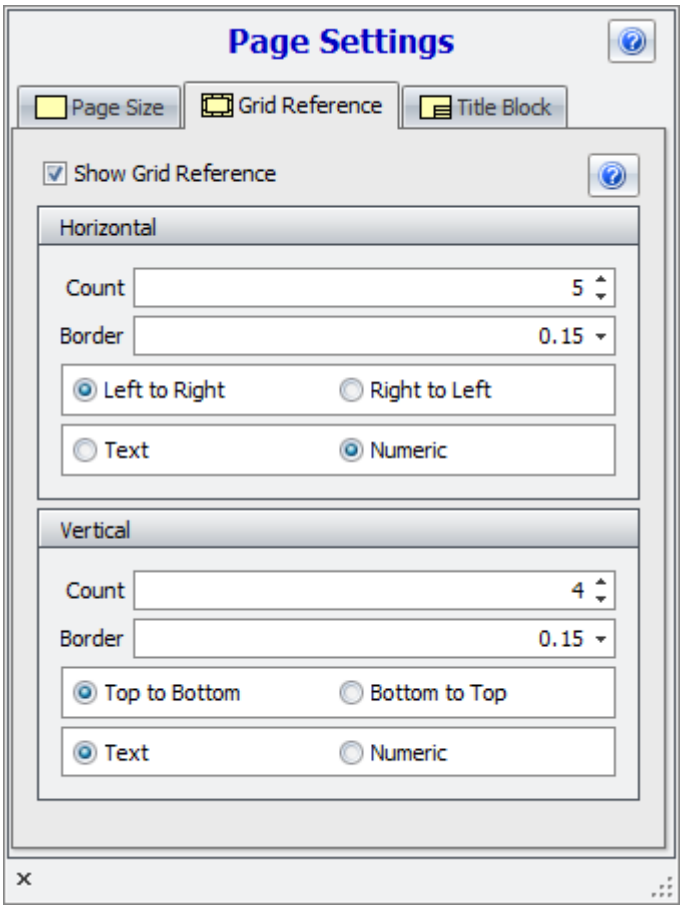
Check to display the title block on the sheet, otherwise it is hidden.

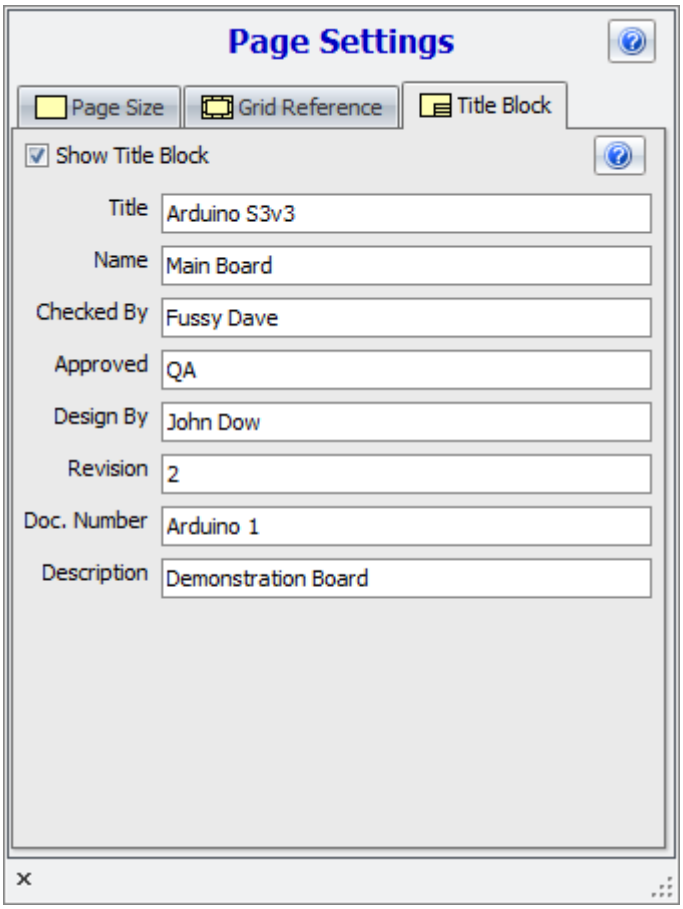
1.2.6.14.6.4 Sheet Settings

To set the sheet/page settings click the small button at the bottom right of **View-Snap→Page** ribbon button group.





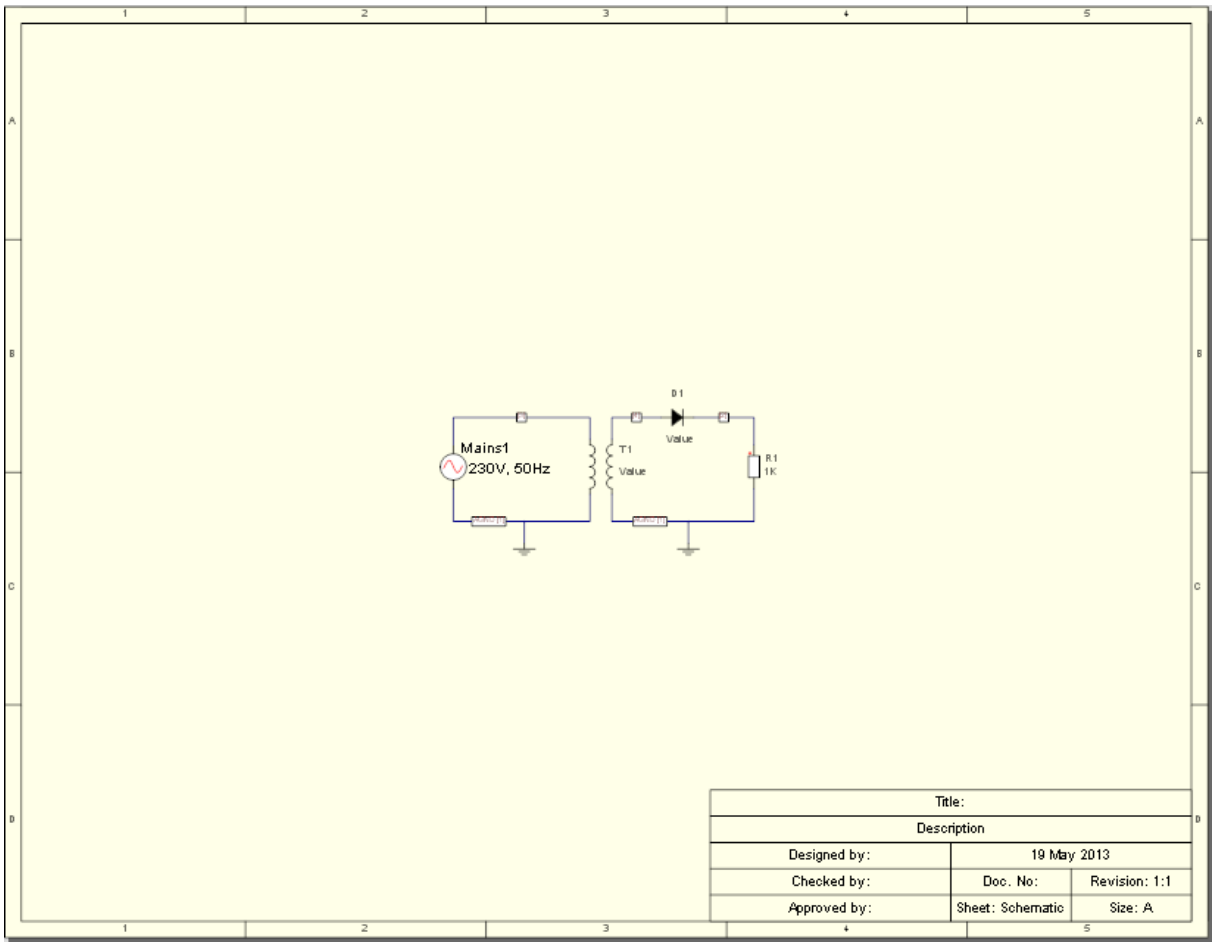




1.2.6.14.6.5 Page Scale

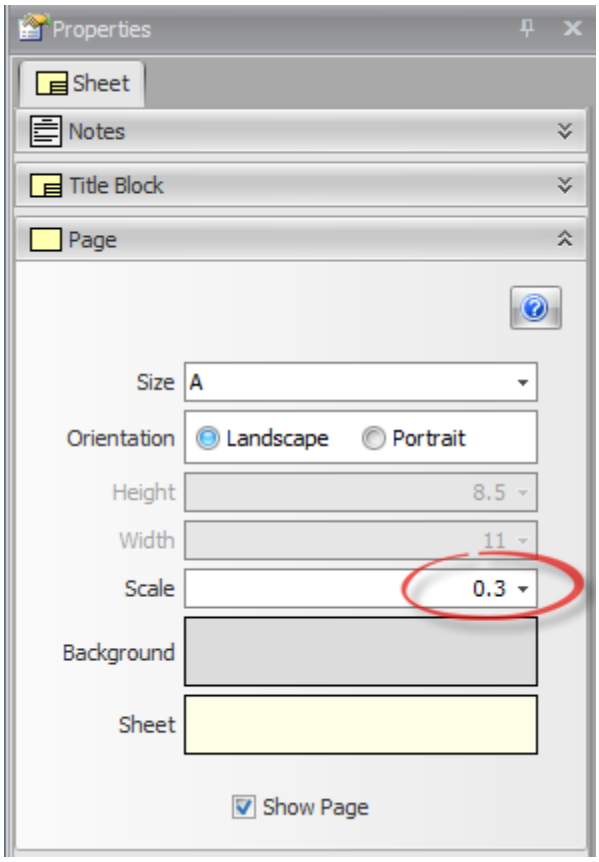
You can set the page scale so your drawing fills the sheet. The circuit below only occupies a small area of the sheet.

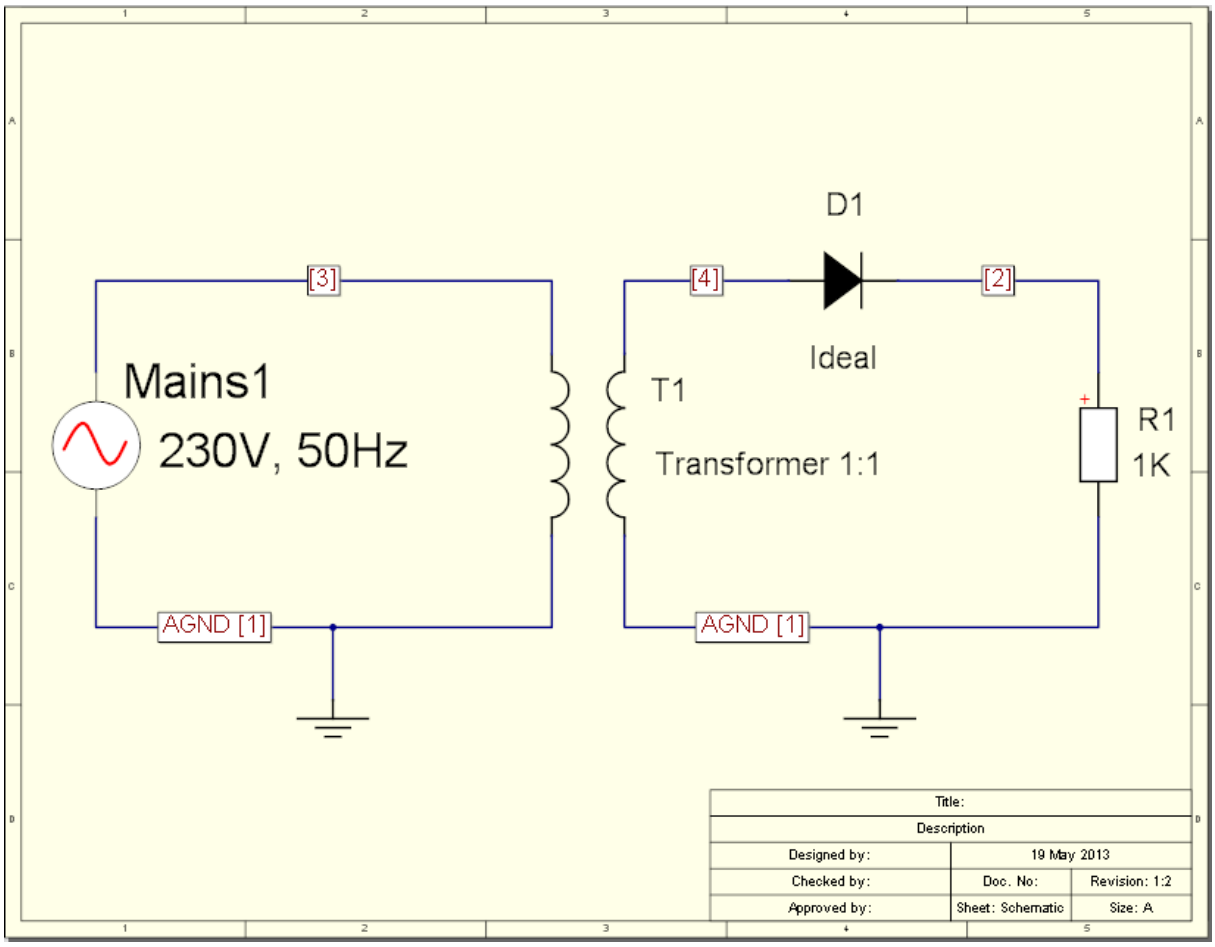
NOTE: This does not affect the units and so the page rulers etc. will adjust automatically.



Page scale = 1

Below the page scale has been set to 0.3 so the small circuit fills the sheet. Setting the scale to more than 1 will shrink the design down so more will find onto a sheet.



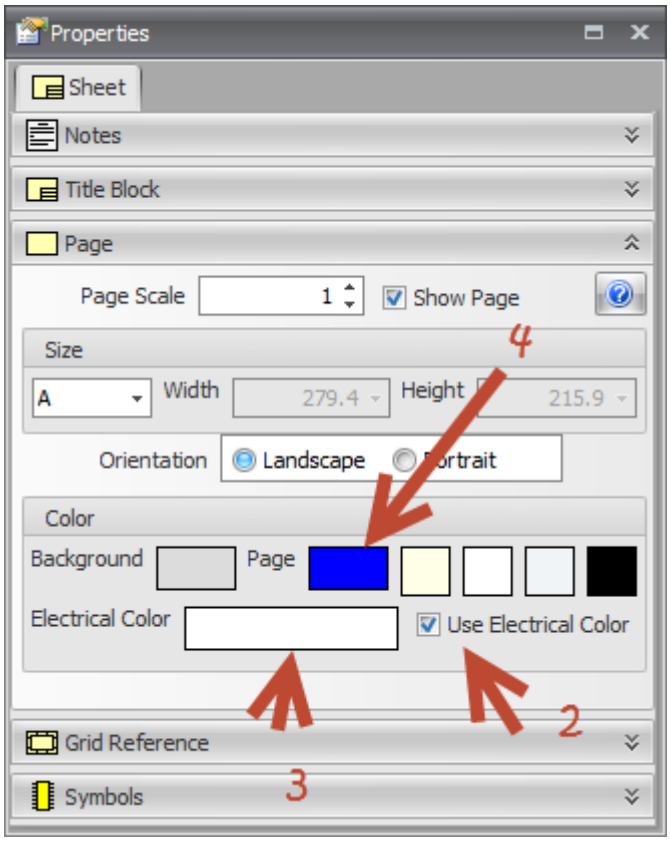


Page scale = 0.3

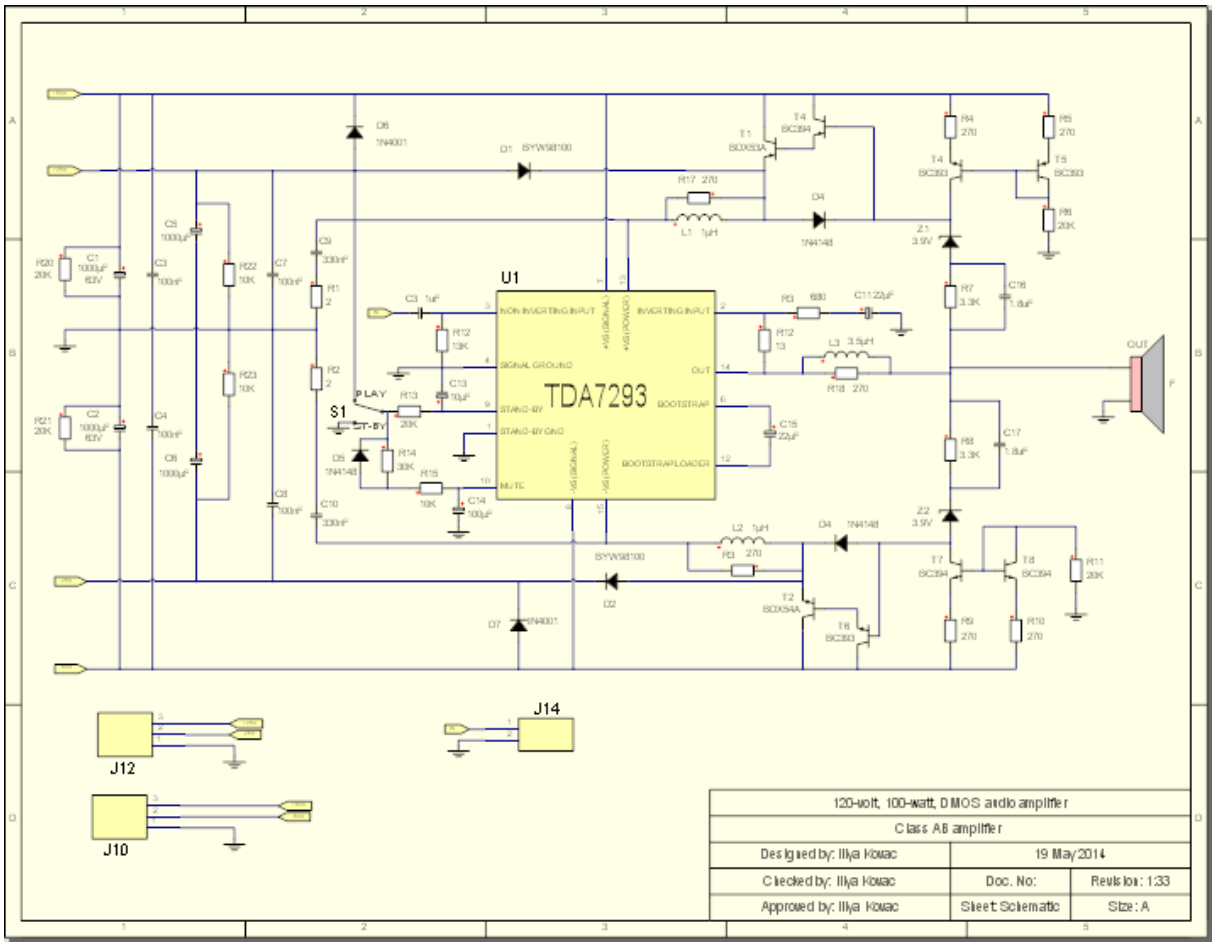
1.2.6.14.6.6 Customizing Sheet Colors

You can change the style of the schematic sheets from the standard multi-color style to a blueprint where all electrical items are in one color in the sheet is in a different color. See below.

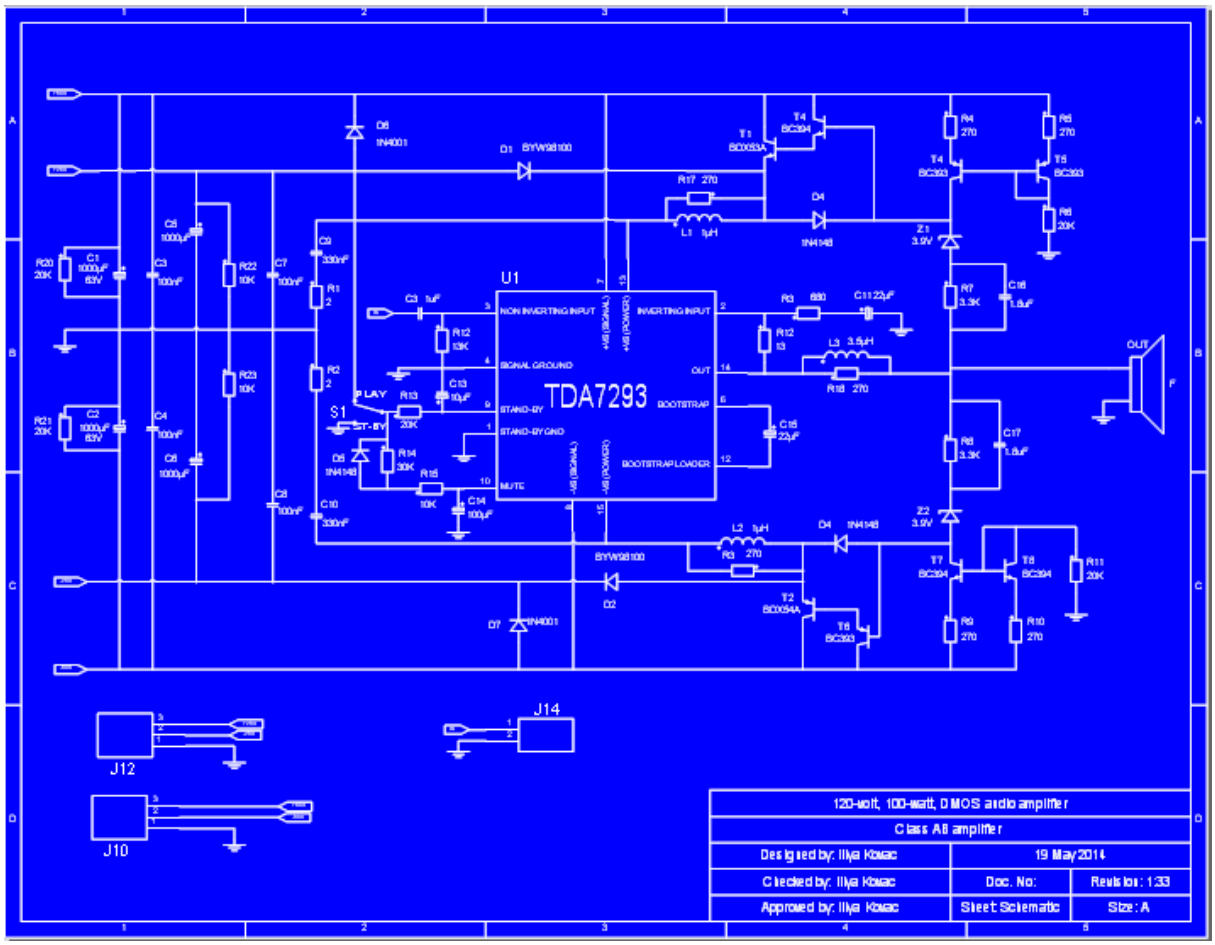
1. To change the colors first make sure the properties panel is open and you have nothing selected.
2. Check the **use electrical color** checkbox
3. now select the color you wish to have for the electrical wiring by double-clicking on the color button next to the use electrical color checkbox.
4. Finally you need to set the color of the page. You can do this by clicking on the page color button.



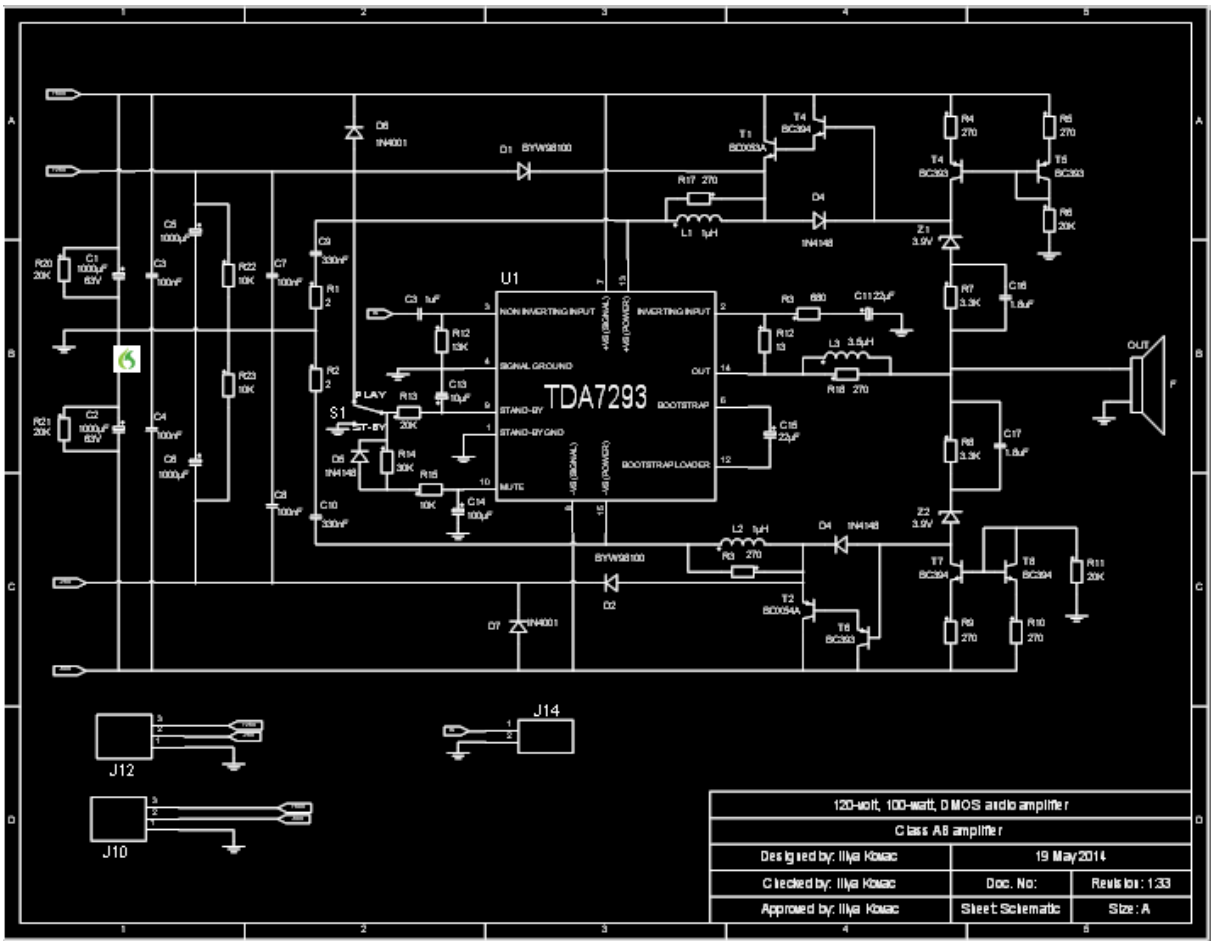
The page properties panel



The standard multicolor display



A blueprint type display



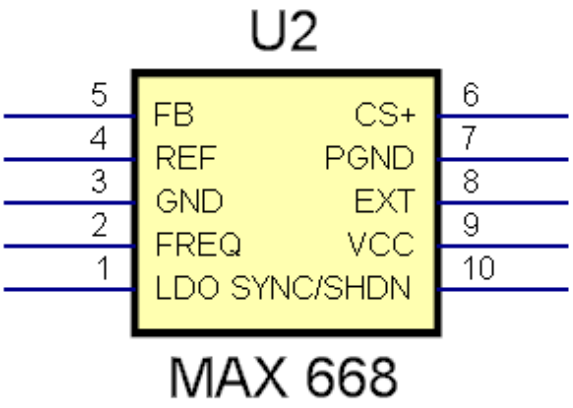
High contrast white on black

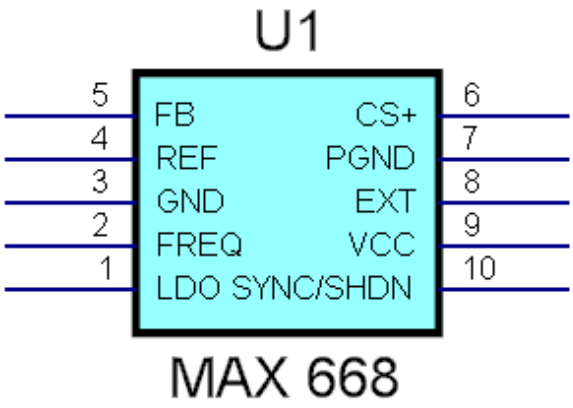
1.2.6.14.6.7 Customizing Symbols

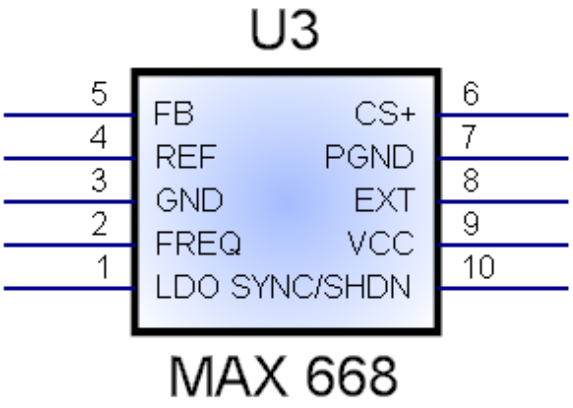
You can fully customize your symbols.

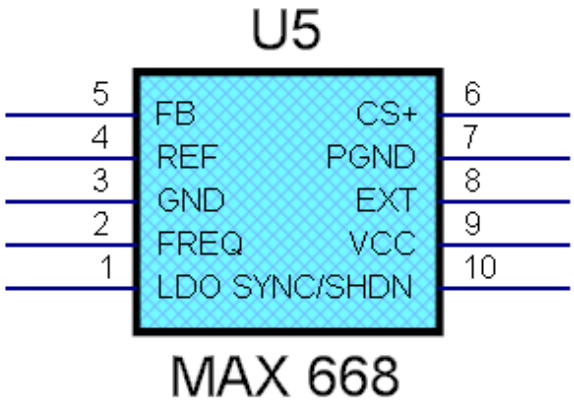
You can set the fill color, fill pattern and border color, border thickness and border line style as well as the border corner style.

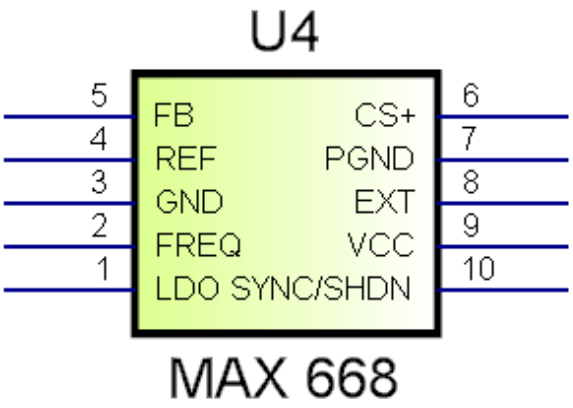
Below you can see several examples of a terminal magnet that has been filled and colored. The terminal magnet line style has also been changed.

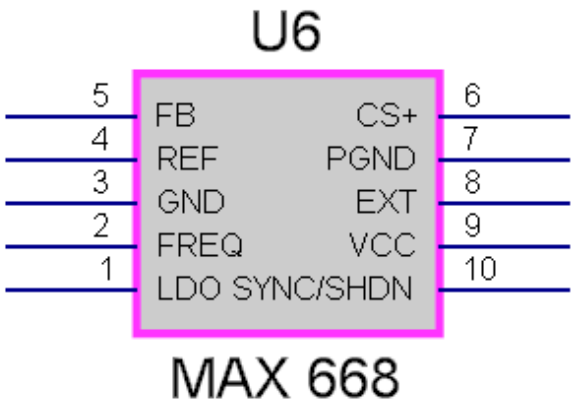

U2
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668

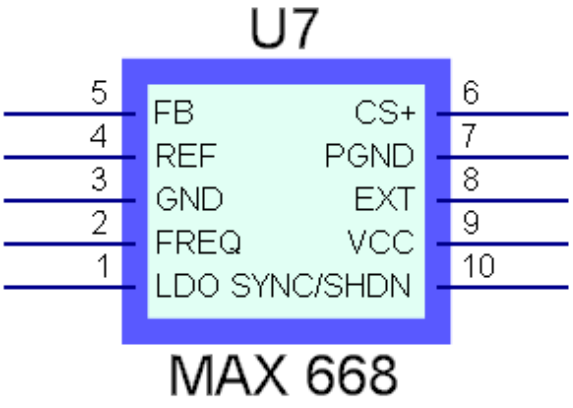

U1
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668

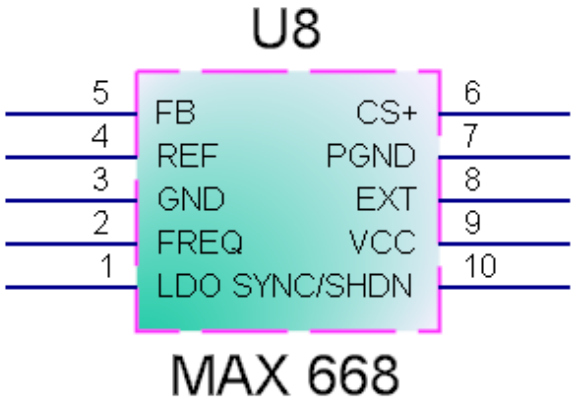

U3
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668


U5
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668


U4
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668


U6
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668


U7
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668


U8
5
4
3
2
1
FB
REF
GND
FREQ
LDO SYNC/SHDN
CS+
PGND
EXT
VCC
6
7
8
9
10
MAX 668

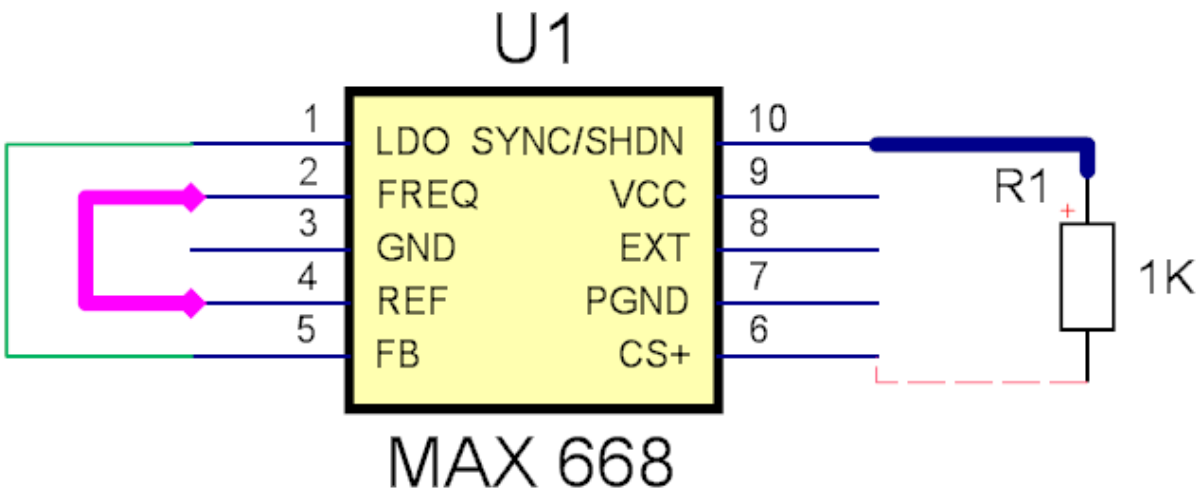
Examples of symbols that have been modified

1.2.6.14.6.8 Customizing Schematic Wires

You can set the line styles for all nodes to be different, just select the node and adjust the properties in the node properties panel.

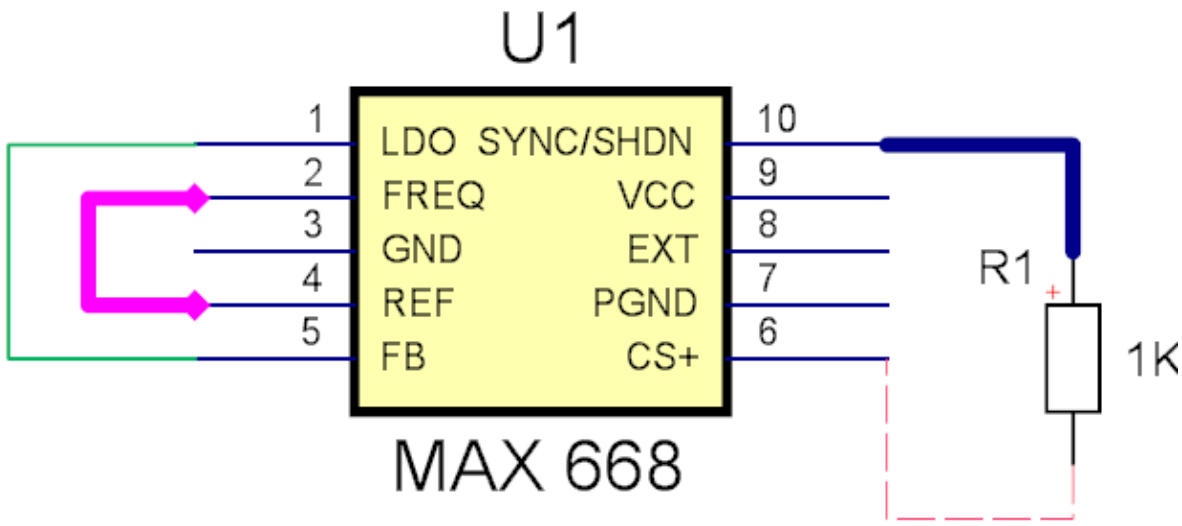
Below you see several different nodes each one with a different color and a different line style.

Note: even the wire that is shown as a red dashed line is actually an electrical wire and it does make An electrical connection.

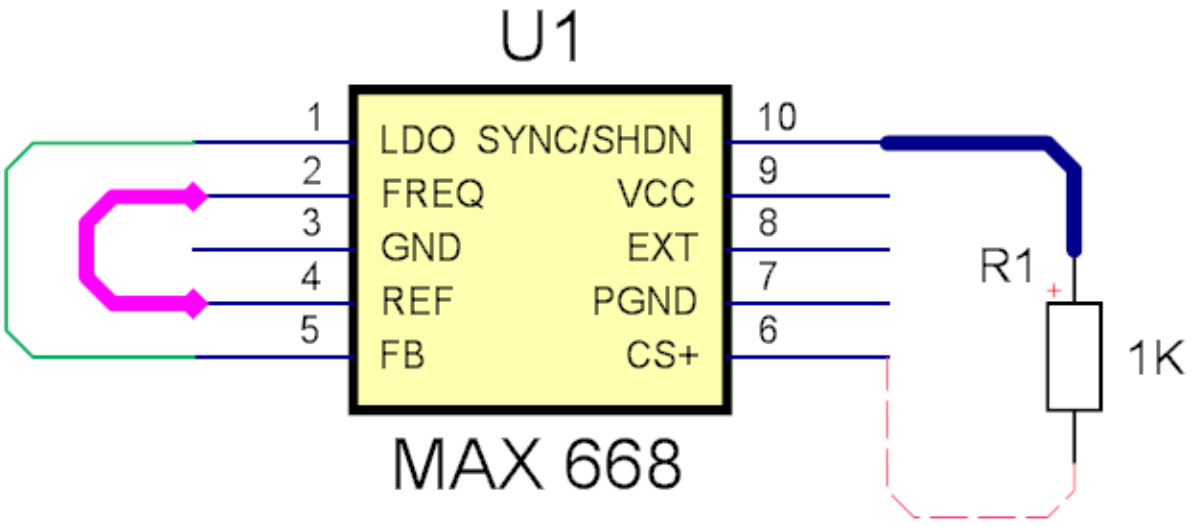


Electrical node (wire) styles

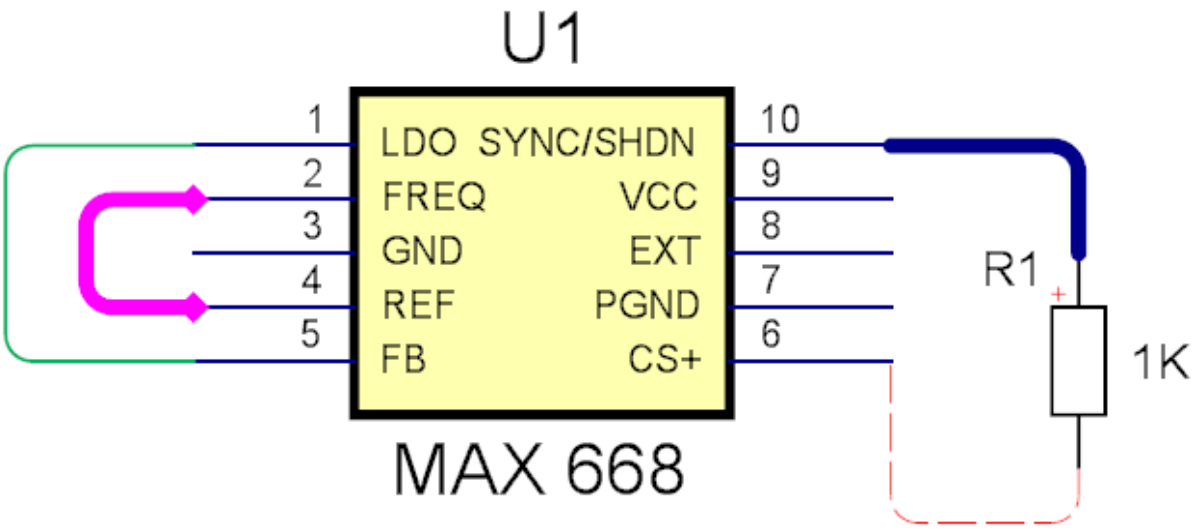
You can set the corner style for electrical wires to be orthogonal, beveled or rounded as shown below.



Orthogonal



Beveled

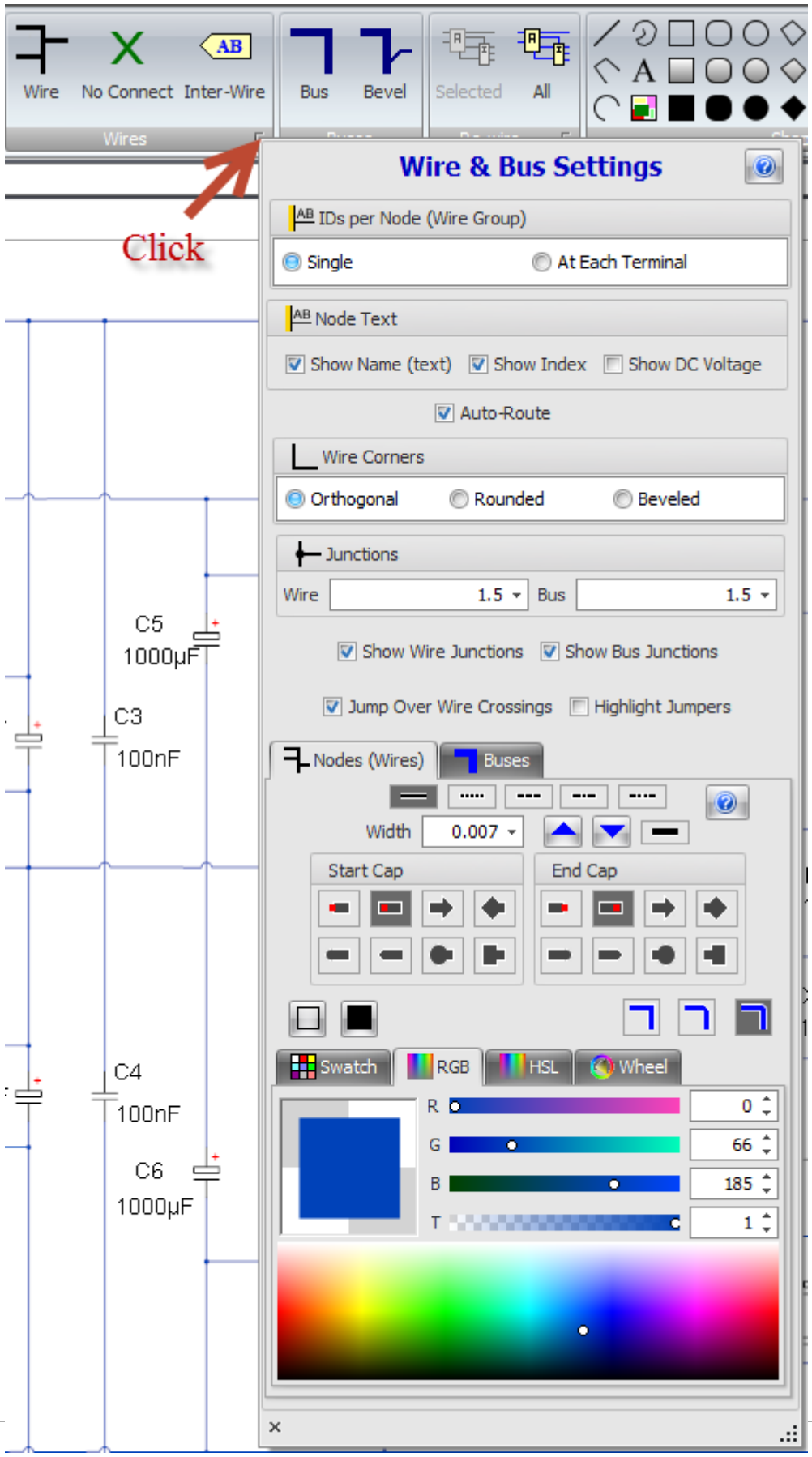


Rounded

1.2.6.14.6.9 The Wire and Bus Settings Popup

The settings for the schematic wires and buses are shown below.

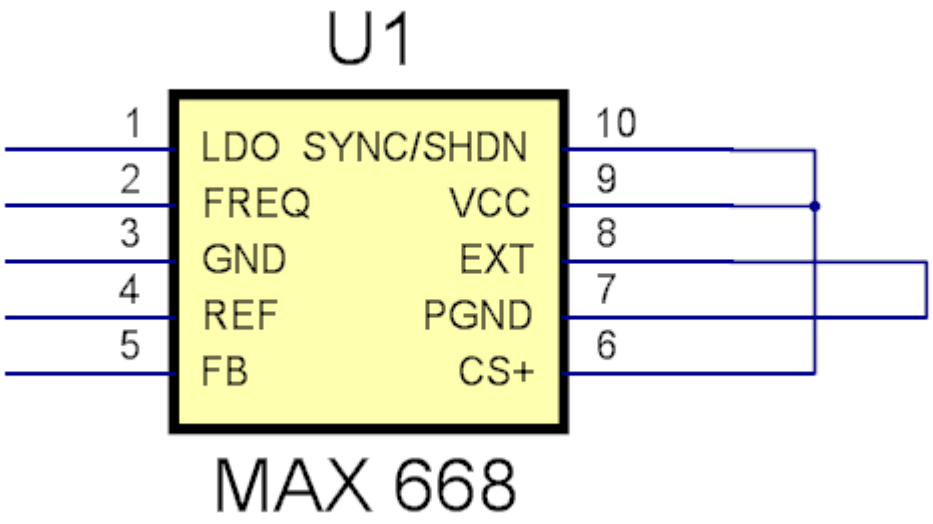
To display the settings pop-up click on the small button at the bottom right of the schematics wires group.



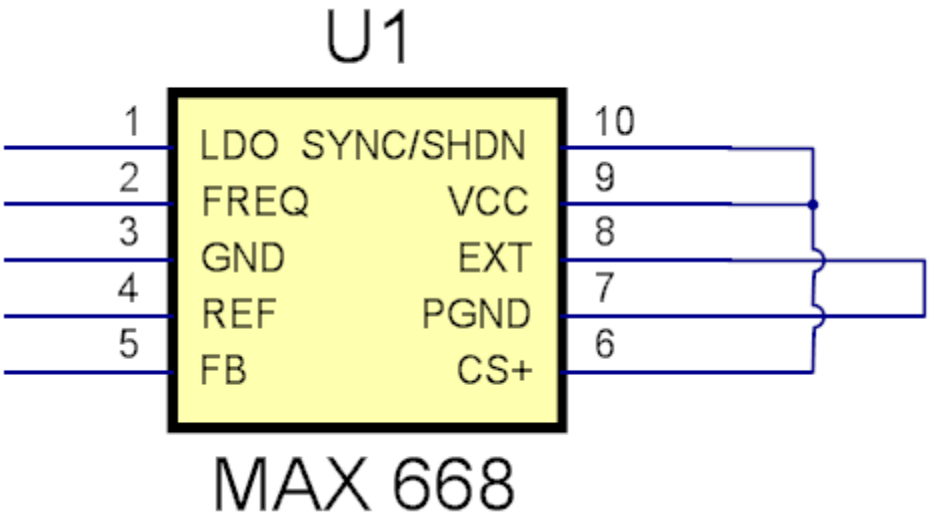
Wire and bus settings

1.2.6.14.6.10 Customizing Schematic Joins and Cross-overs

You can optionally have a small jump as wire crosses another one for you can add a wire to simply cross. see The Wire and Bus Settings Popup

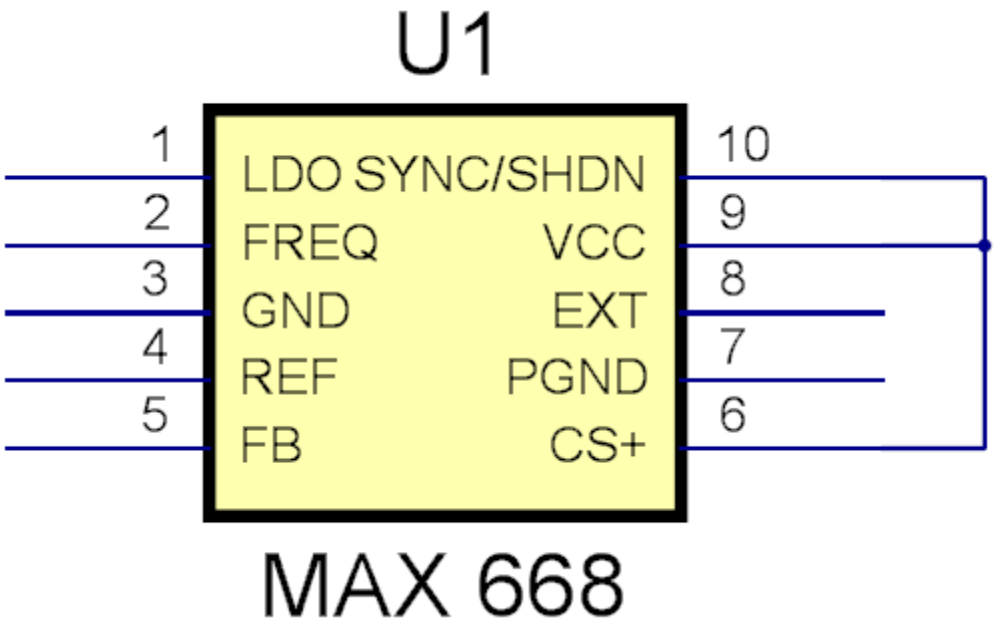


No jump

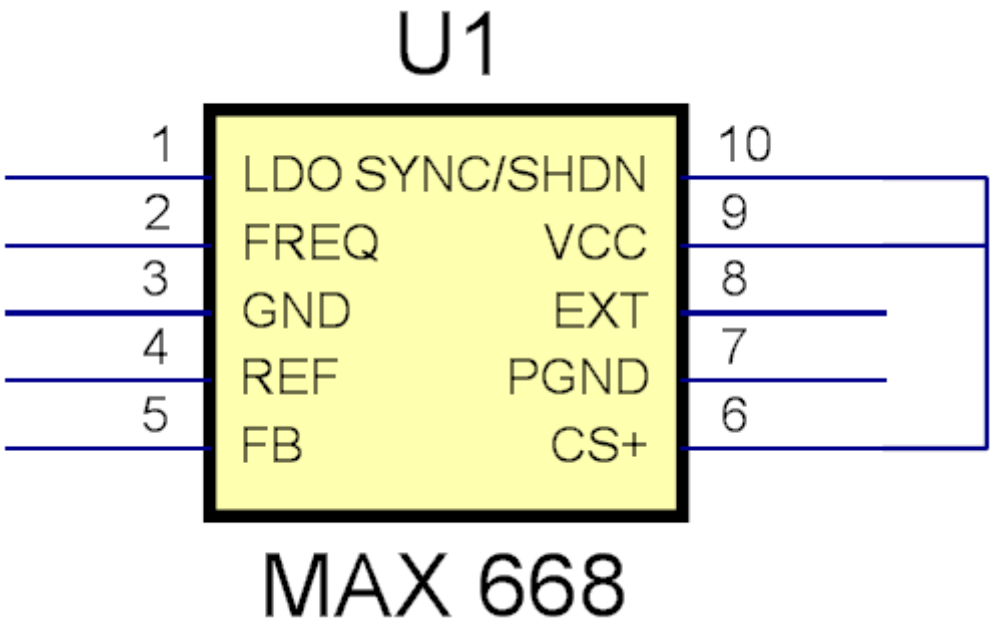


With jumper

You can optionally set the junction with one wire and another to be a point or a junction to have no point at all as shown below.



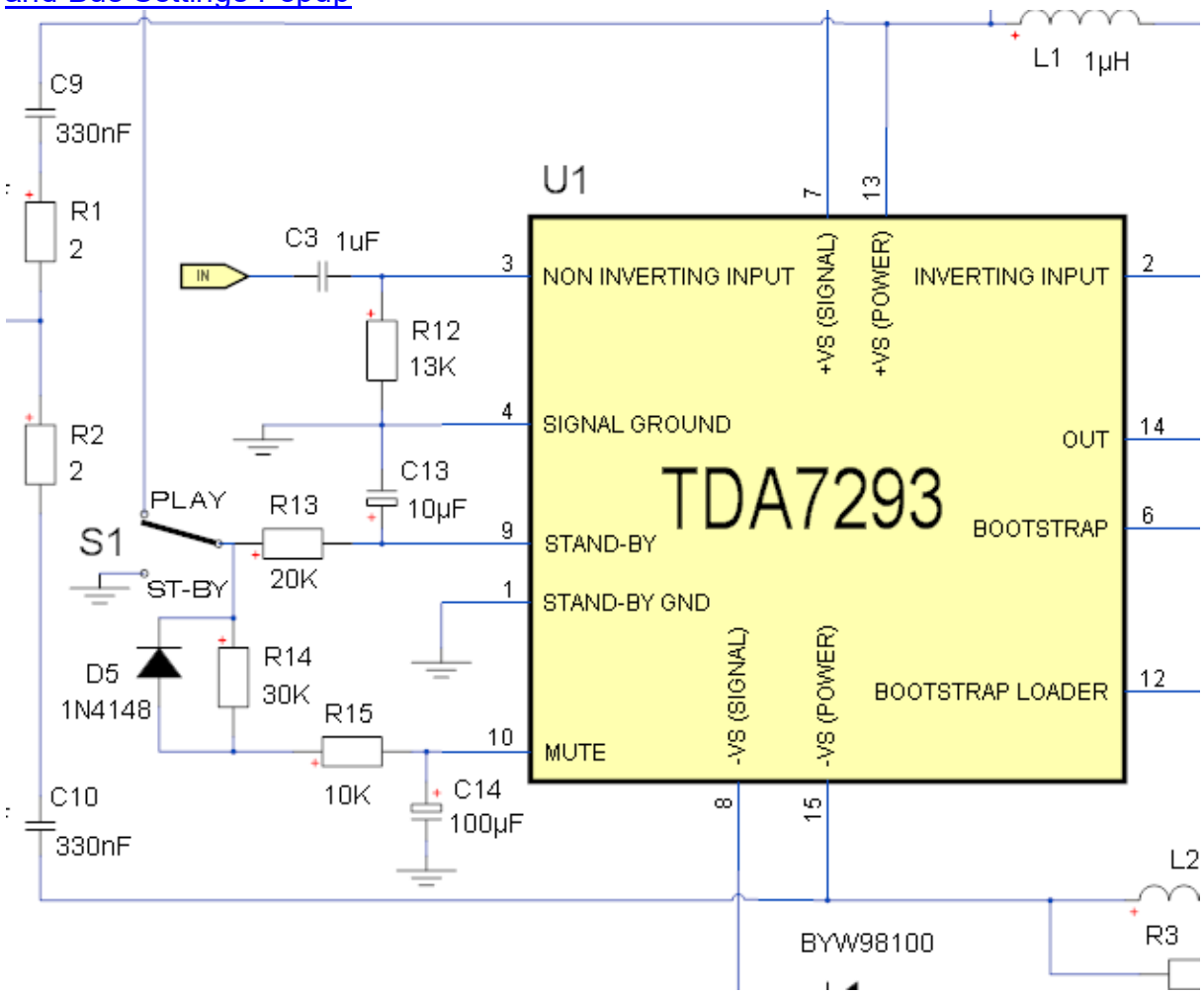
With junction point

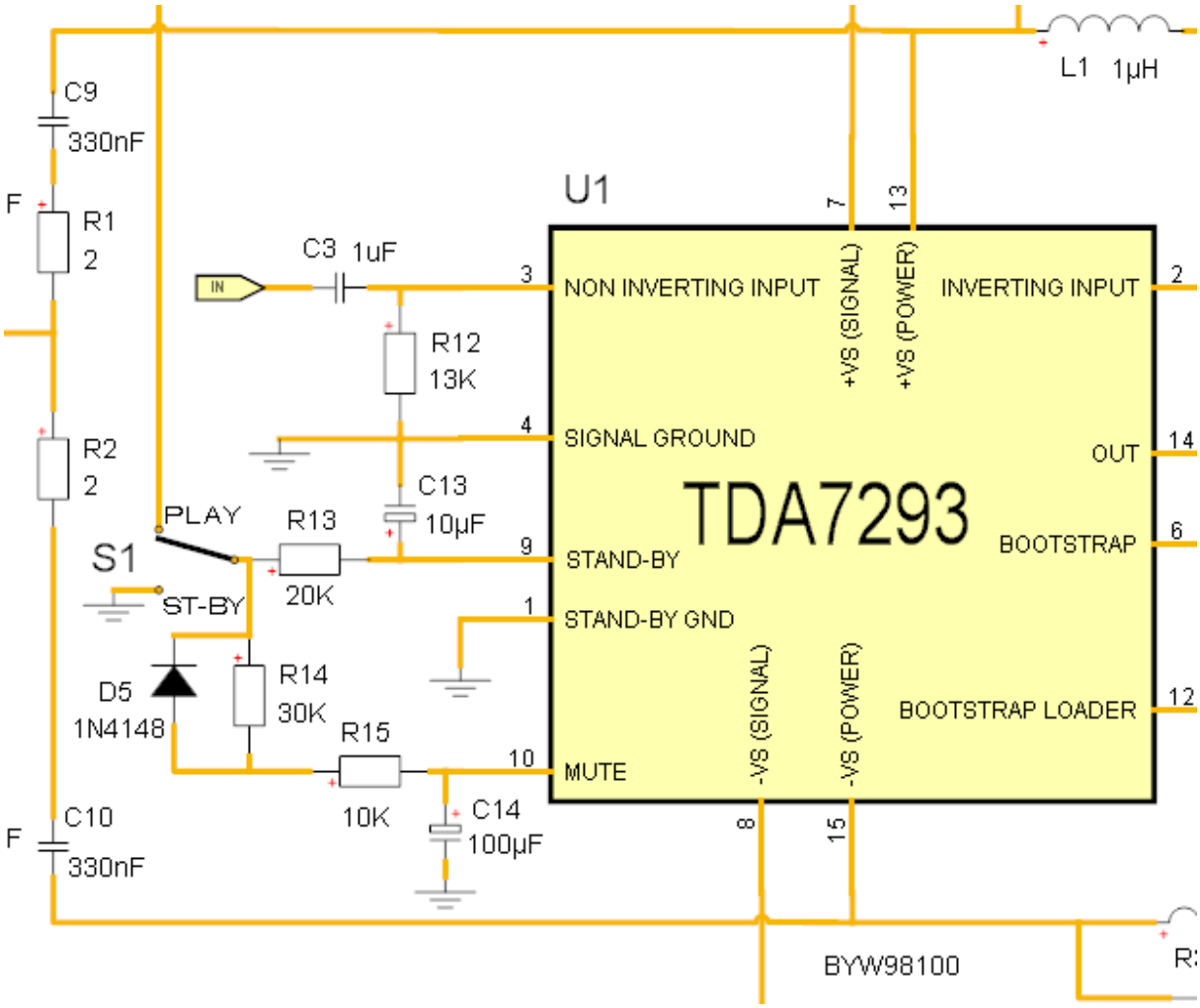


No junction point

1.2.6.14.6.11 Customizing All Schematic Wires

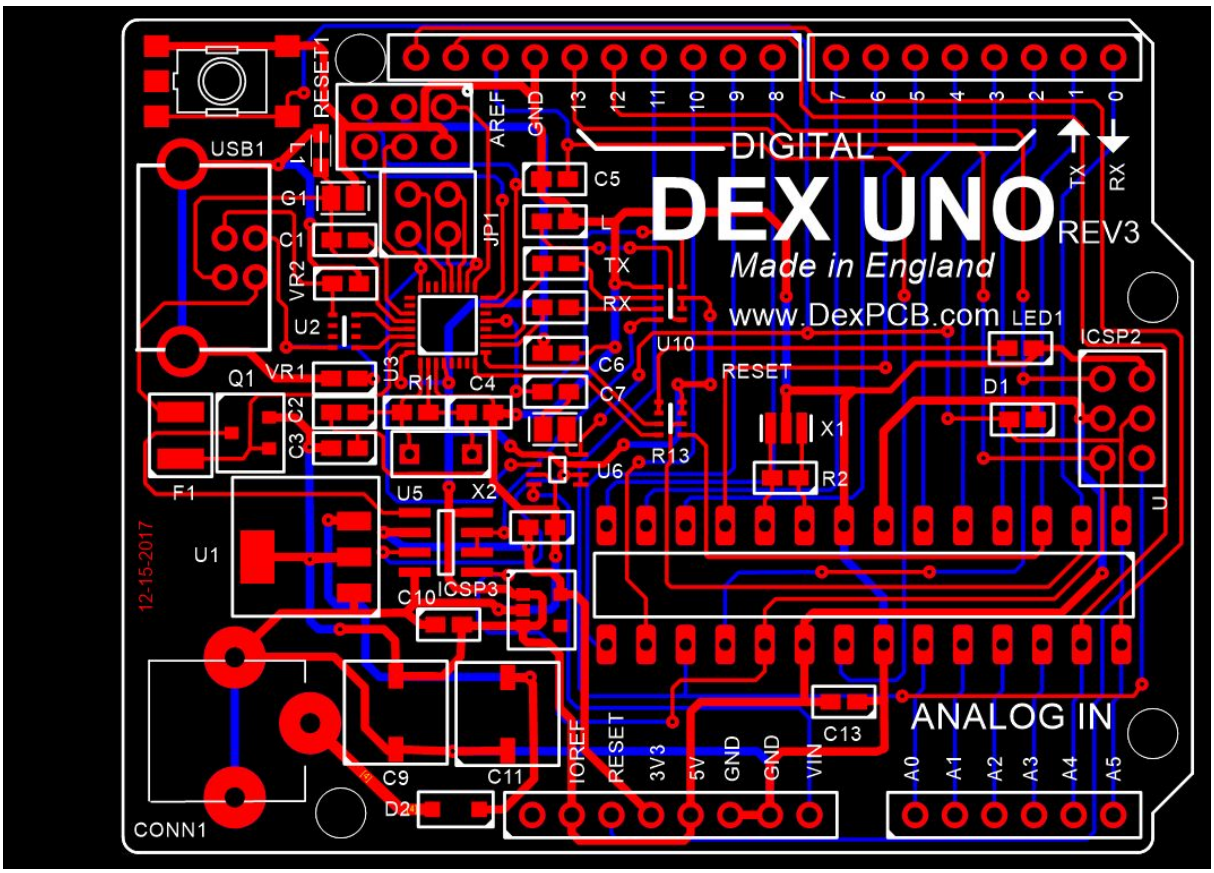
You can set the line style, colors etc for all wires on the schematics. See The Wire and Bus Settings Popup



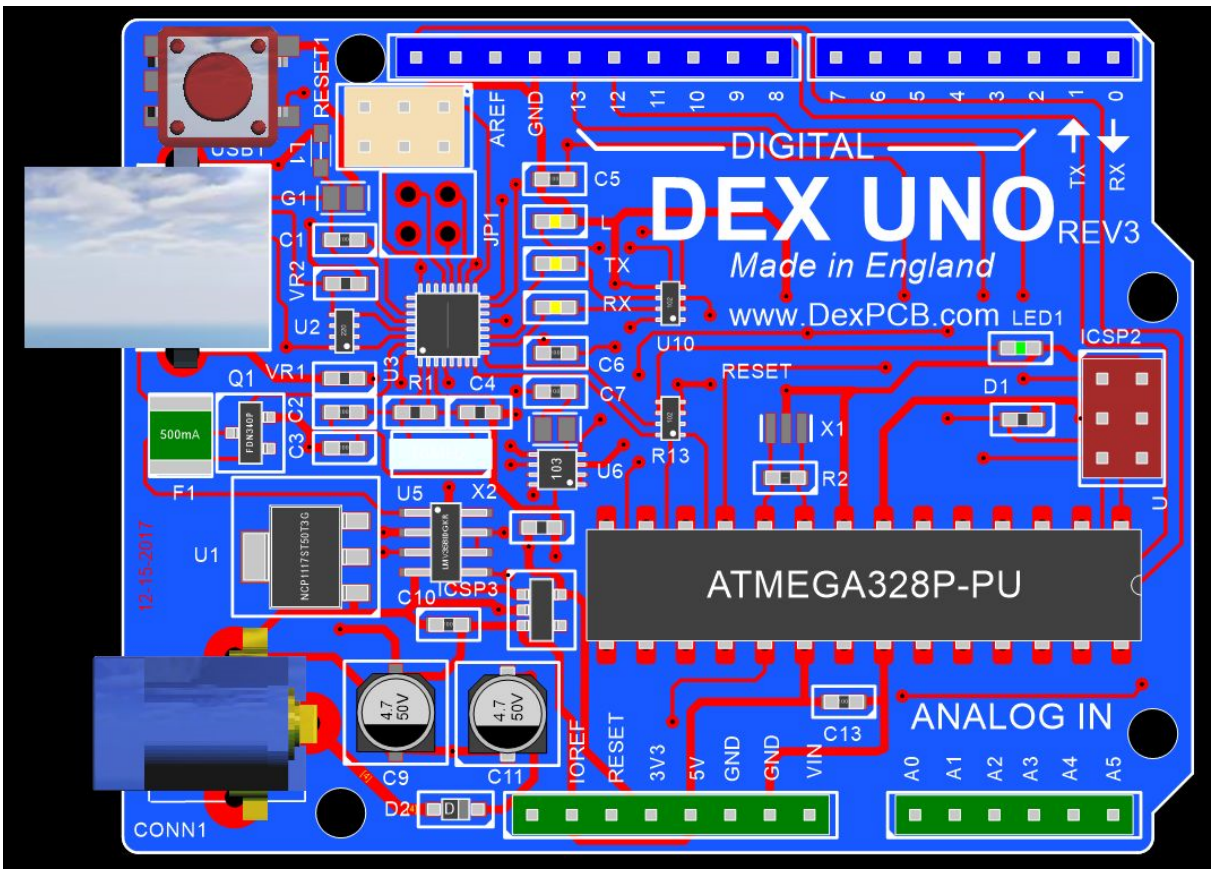


1.2.6.15 The PCB

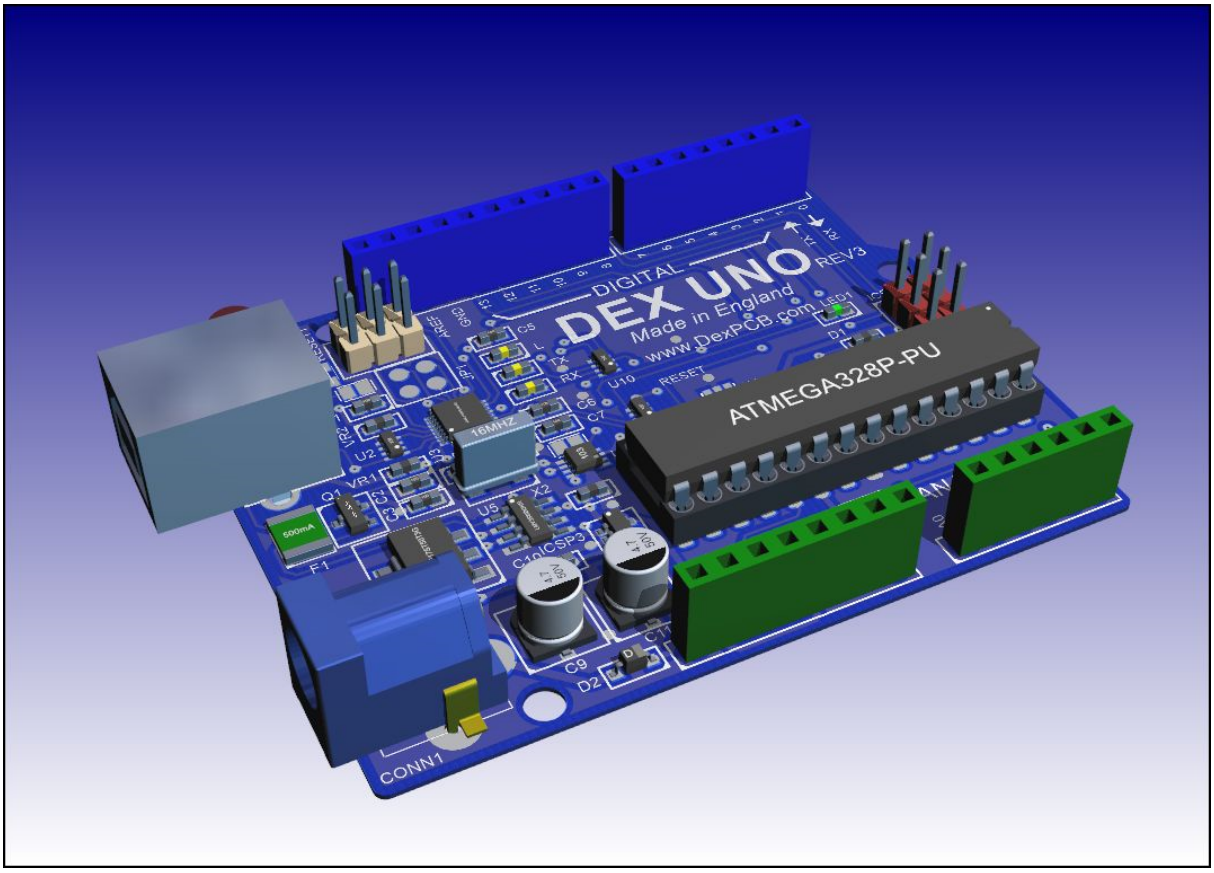
The PCB is the physical printed circuit board.



A typical PCB



The PCB showing the PCB filled and physical parts on the top of the PCB



The PCB viewed in 3-D

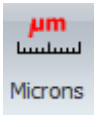
1.2.6.16 Units

You can set the units for all Graphical Sheets.

The possible units are:

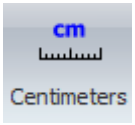
- 

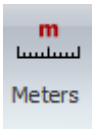
Mil/Thou.
- 

Inches
- 

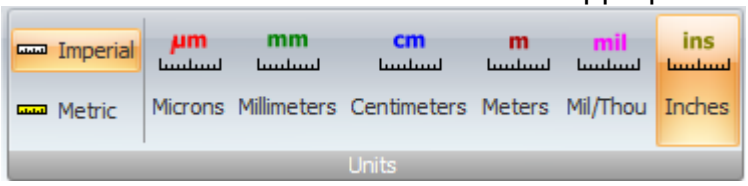
Microns

- 

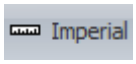
MM
- 

CM
- 

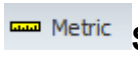
Meters

To set the units for a sheet click on the appropriate button in the 

Tools ribbon button group.



Set Design Settings to Imperial (Inches). This will set snap and grid spacing to 0.1" for all schematics and the PCB. Units will be set to inches for all schematics and the PCB. This will apply to all new projects. Existing schematic symbols and wires will be snapped to the new grid**.**



Set Design Settings to Metric. This will set snap and grid spacing to 2 mm for all schematics and to 1 mm for the PCB. Units will be set to cm for all schematics and to mm for the PCB. This will apply to all new projects.

1.2.6.16.1 The Origin

To change the origin ***right-click*** on the origin or the vertical or horizontal ruler and select the appropriate menu command.

1.2.6.16.2 Manufacturing Your PCB

AutoTRAX DEX gives you a complete set of tools to help make your PCBs.

- Plotting Your PCBs
- Previewing Your Gerber Files
- Generating Gerber Files

- Generating Drill Files
- Creating a Pick and Place File
- Your Bill of Materials (BOM)
- Putting It All Together in a Zip File

1.2.6.16.2.1 Plotting Your PCBs

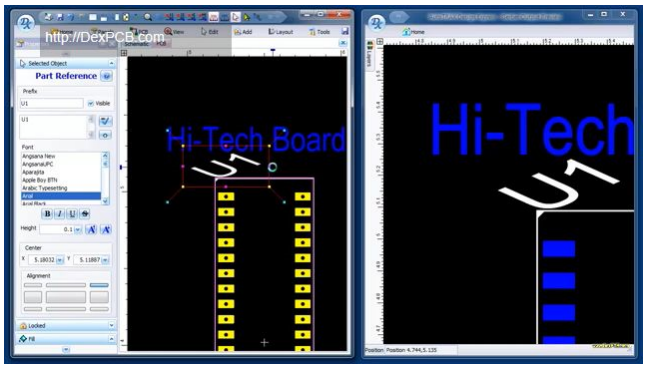
To plot your PCB click the **Tools→CAM→** 

button.

See Plottings for more on plotting.

1.2.6.16.2.2 True-type fonts

With AutoTRAX DEX you can choose between classic simple 'stick' text or professional looking True-type fonts for your PCB silkscreens. This video shows you the difference and how to make your choice.



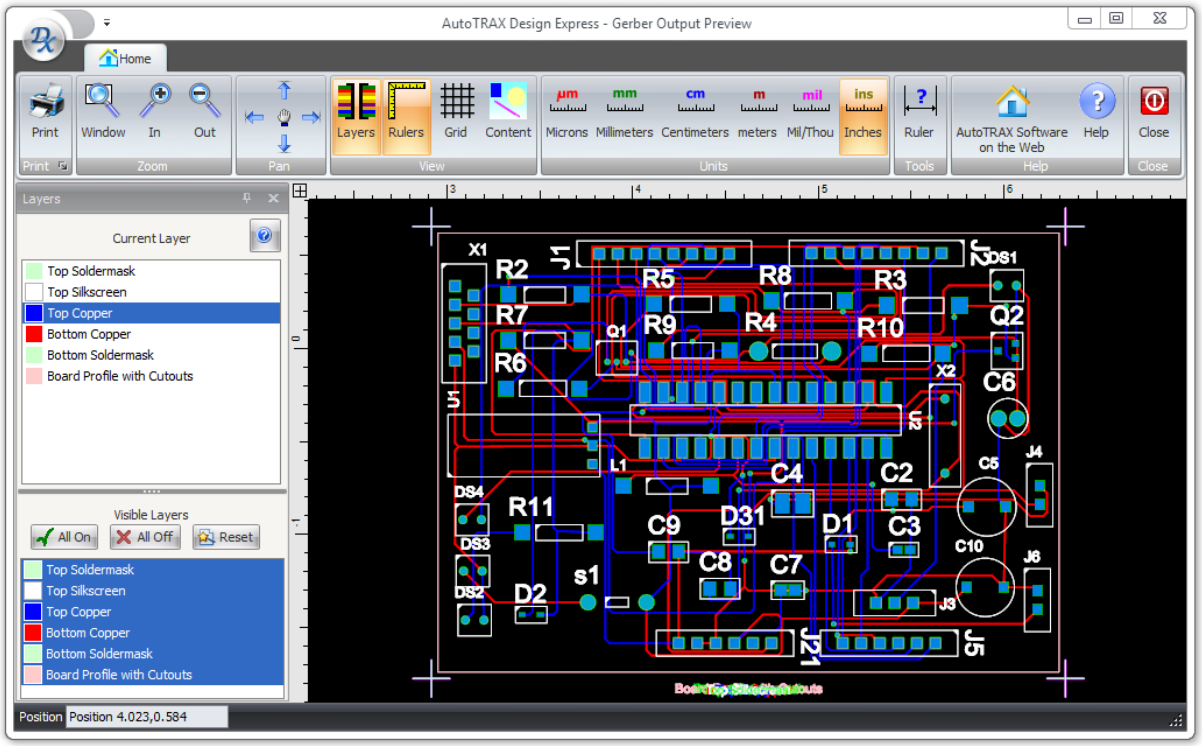
True-type fonts for PCB production

1.2.6.16.2.3 Previewing Your Gerber Files

You can preview your Gerber files using AutoTRAX DEX's own Gerber viewer.

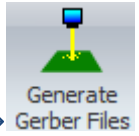
To display the Gerber viewer click the Tools→CAM→ 

button.



The Gerber Viewer

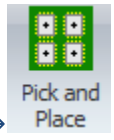
1.2.6.16.2.4 Generating Gerber Files

To generate a set of Gerber files click the Tools→CAM→  button.

1.2.6.16.2.5 Generating Drill Files

To generate a NC drill file click the Tools→CAM→  button.

1.2.6.16.2.6 Creating a Pick and Place File

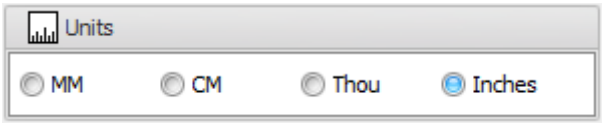
To generate a NC drill file click the Tools→CAM→  button.

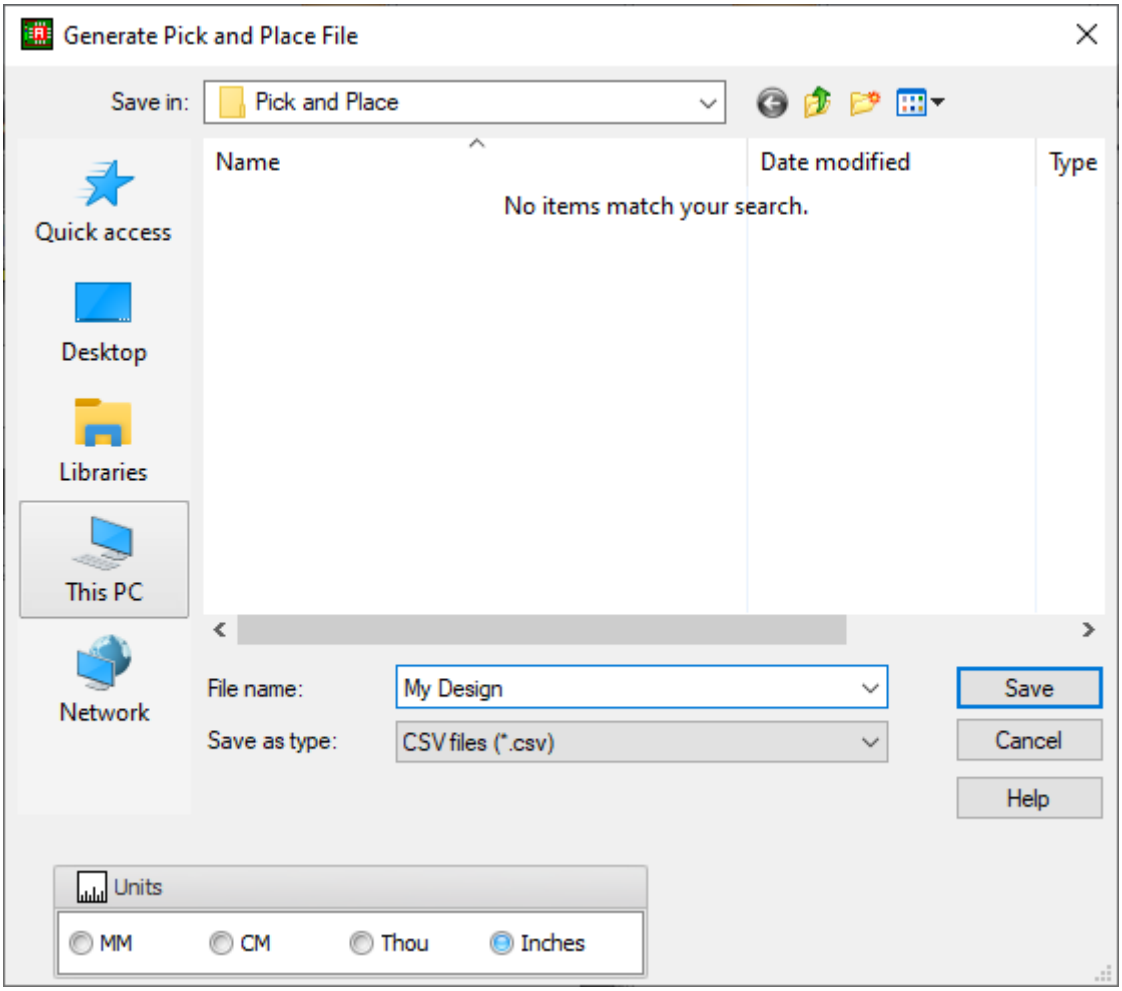
The pick and place file is a comma separated file (CSV) compatible with Microsoft Exel and the spreadsheet programs.

Pick and Place Origin

If you add automatic fiducial to your panel then the origin for the Pick and Place file coordinates is set to the lower left fiducial else it is set to the bottom left corner of the panel.

Check the appropriate units in the radio check box.





1.2.6.16.2.7 Your Bill of Materials

To generate a Bill of Materials (BOM) file click the Tools→CAM→ 

button.

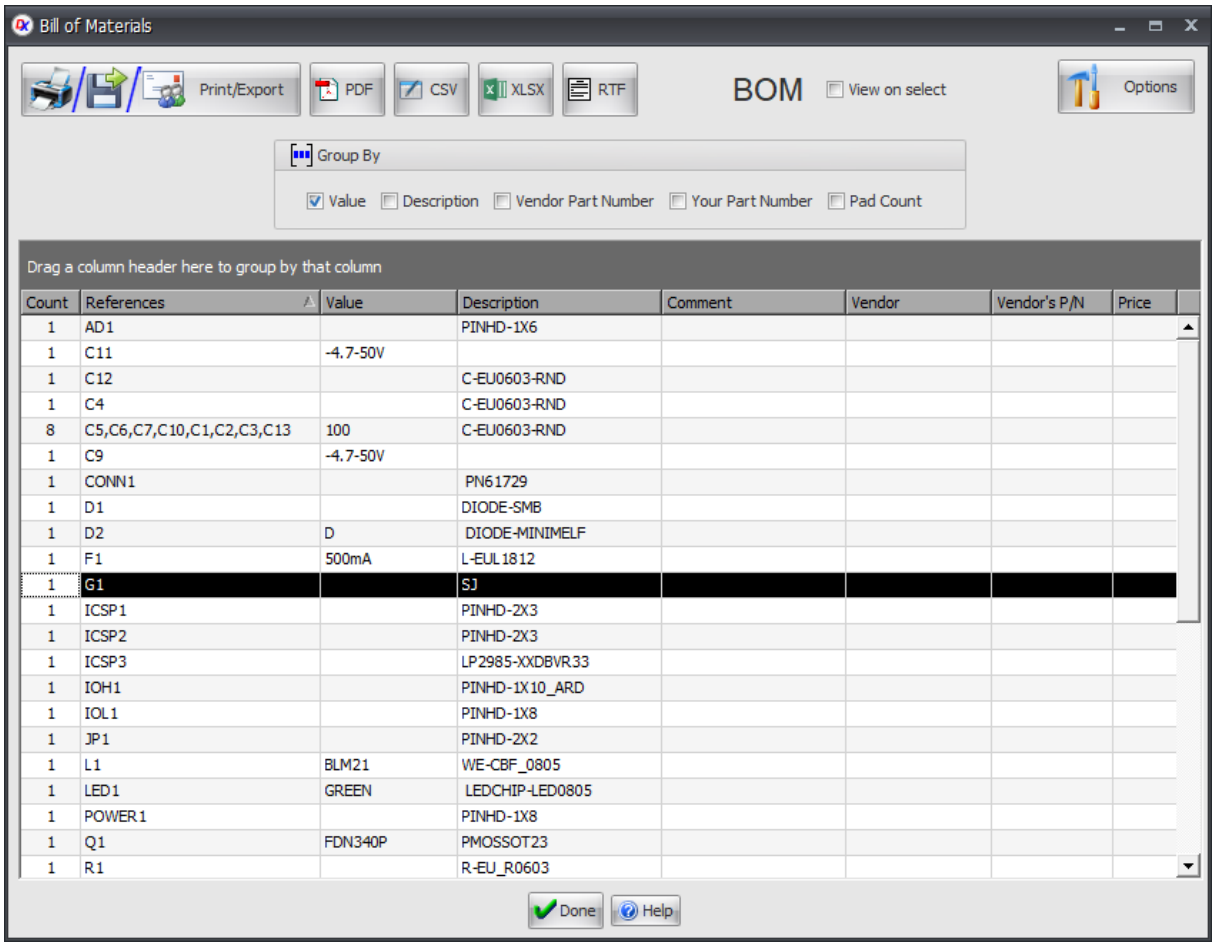
A bill of materials or product structure (sometimes bill of material, BOM or associated list) is a list of the raw materials, sub-assemblies, intermediate assemblies, sub-

components, parts and the quantities of each needed to manufacture an end product. A BOM may be used for communication between manufacturing partners, or confined to a single manufacturing plant. A bill of materials is often tied to a production order whose issuance may generate reservations for components in the bill of materials that are in stock and requisitions for components that are not in stock.

A BOM can define products as they are designed (engineering bill of materials), as they are ordered (sales bill of materials), as they are built (manufacturing bill of materials), or as they are maintained (service bill of materials or pseudo bill of material). The different types of BOMs depend on the business need and use for which they are intended. In process industries, the BOM is also known as the formula, recipe, or ingredients list. The phrase "bill of material" (or BOM) is frequently used by engineers as an adjective to refer not to the literal bill, but to the current production configuration of a product, to distinguish it from modified or improved versions under study or in test.

In electronics, the BOM represents the list of components used on the printed wiring board or printed circuit board. Once the design of the circuit is completed, the BOM list is passed on to the PCB layout engineer as well as component engineer who will procure the components required for the design.

Items in a Bill of Materials



Count	References	Value	Description	Comment	Vendor	Vendor's P/N	Price
1	AD1		PINHD-1X6				
1	C11	-4.7-50V					
1	C12		C-EU0603-RND				
1	C4		C-EU0603-RND				
8	C5,C6,C7,C10,C1,C2,C3,C13	100	C-EU0603-RND				
1	C9	-4.7-50V					
1	CONN1		PN61729				
1	D1		DIODE-SMB				
1	D2	D	DIODE-MINIMELF				
1	F1	500mA	L-EUL1812				
1	G1		SJ				
1	ICSP1		PINHD-2X3				
1	ICSP2		PINHD-2X3				
1	ICSP3		LP2985-XXDBVR33				
1	IOH1		PINHD-1X10_ARD				
1	IOL1		PINHD-1X8				
1	JP1		PINHD-2X2				
1	L1	BLM21	WE-CBF_0805				
1	LED1	GREEN	LEDCHIP-LED0805				
1	POWER1		PINHD-1X8				
1	Q1	FDN340P	PMOSSOT23				
1	R1		R-EU_R0603				

You can include the follow items in a bill of materials:

Reference

The reference is the footprint reference that you will see in the PCB. This identifies the part both on your PCB and in your schematics e.g. R1 for a resistor.

Value

The value is the type of the part for instance it could be a capacitor where the value might be 100pF or it could be a device type such as 7400. You will most likely see the value in the schematic and sometimes it will be displayed in the PCB as a marker on the 3-D component.

Package

The package is the type of physical packaging for the part. For instance, it might be a DIP 40.

Pad Count

The pad count is a number of pads that make up the footprint for the part. For instance a DIP 40 would have 40 pads

Placement

The placement is the placement point for the part on the PCB and is defined by the placement point in the footprint.

Rotation

The rotation is the rotation of the footprint on the PCB and is given in degrees

Side

The side details which side of the PCB the parties mounted. It can be either the top side on the bottom side.

SMT/TPH

This despair is whether the part is a Surface mount technology part (SMT) or a Through-plated-hole (TPH) part.

Description

The description is a short textual description of your part.

Datasheet

The datasheet is the web address of the datasheet for the part. This is often a link to a PDF file

Comment

The comment is what it says it is: a comment!

Part Number

This is the part number that you want to use to specify the part in your own organization.

Manufacturer

This is the manufacturer of the part

Manufacturer's Website

This is the website for the manufacturer.

Manufacturer's Part Number

This is the part number for the part as specified by the manufacturer.

Vendor

This is the vendor formal you will actually purchase the part. For instance it could be Stock, Mouser, Farnell or Digikey.

Vendor's Website

This is the website for the vendor.

Vendor's Part Number

This is the part number for the part as specified by the vendor.

Price

This is the price for a part.

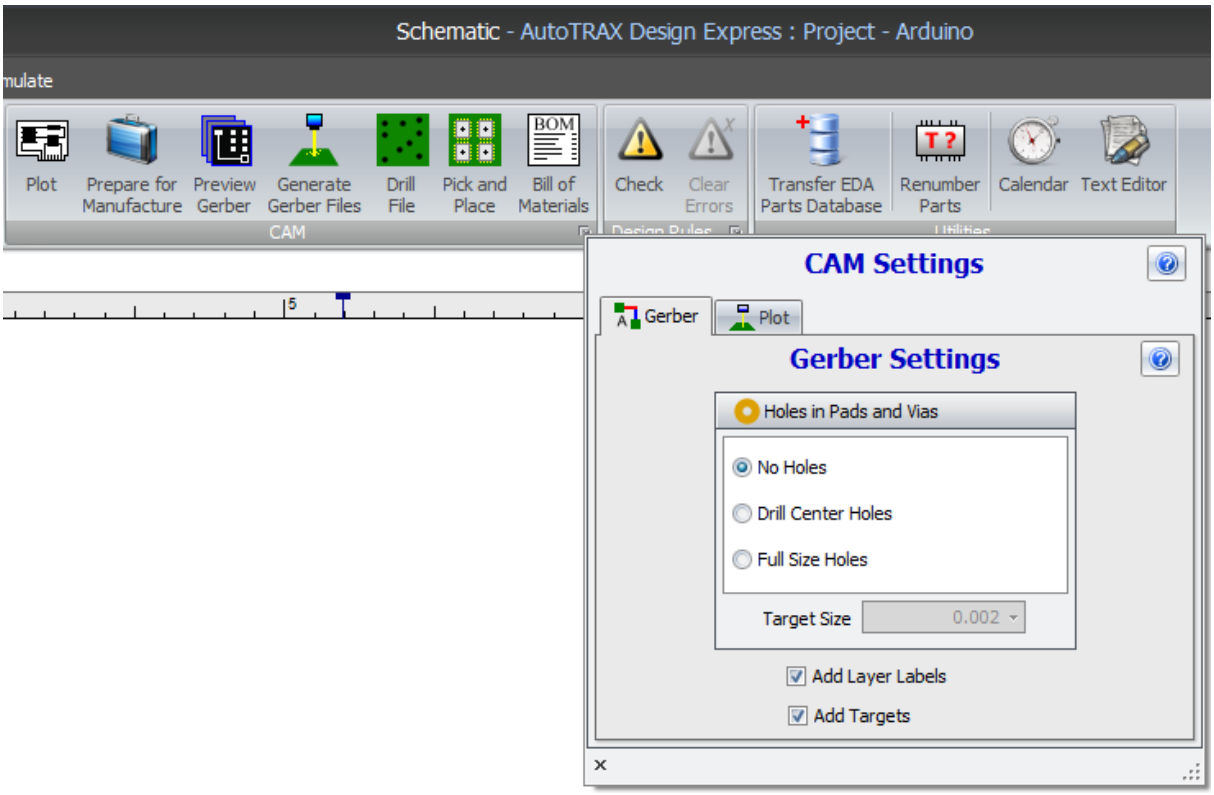
1.2.6.16.2.8 Putting It All Together in a Zip File

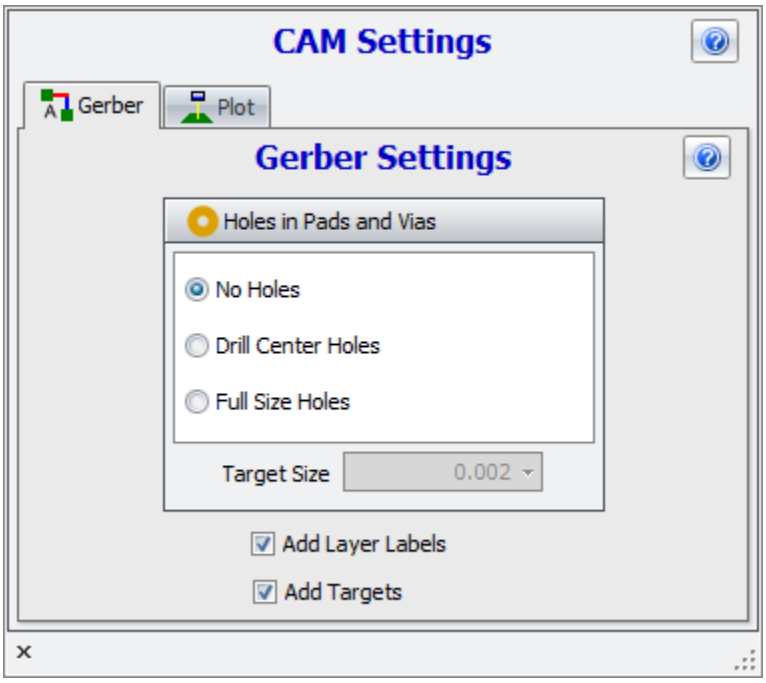
To create a complete set of files for making your PCB, click the Tools→CAM→ 

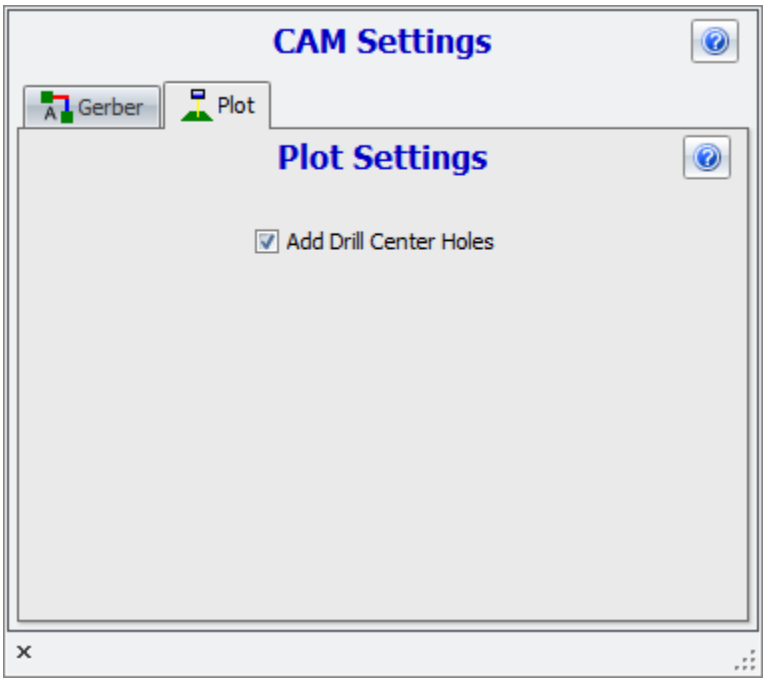
button.

1.2.6.16.2.9 Manufacturing Settings

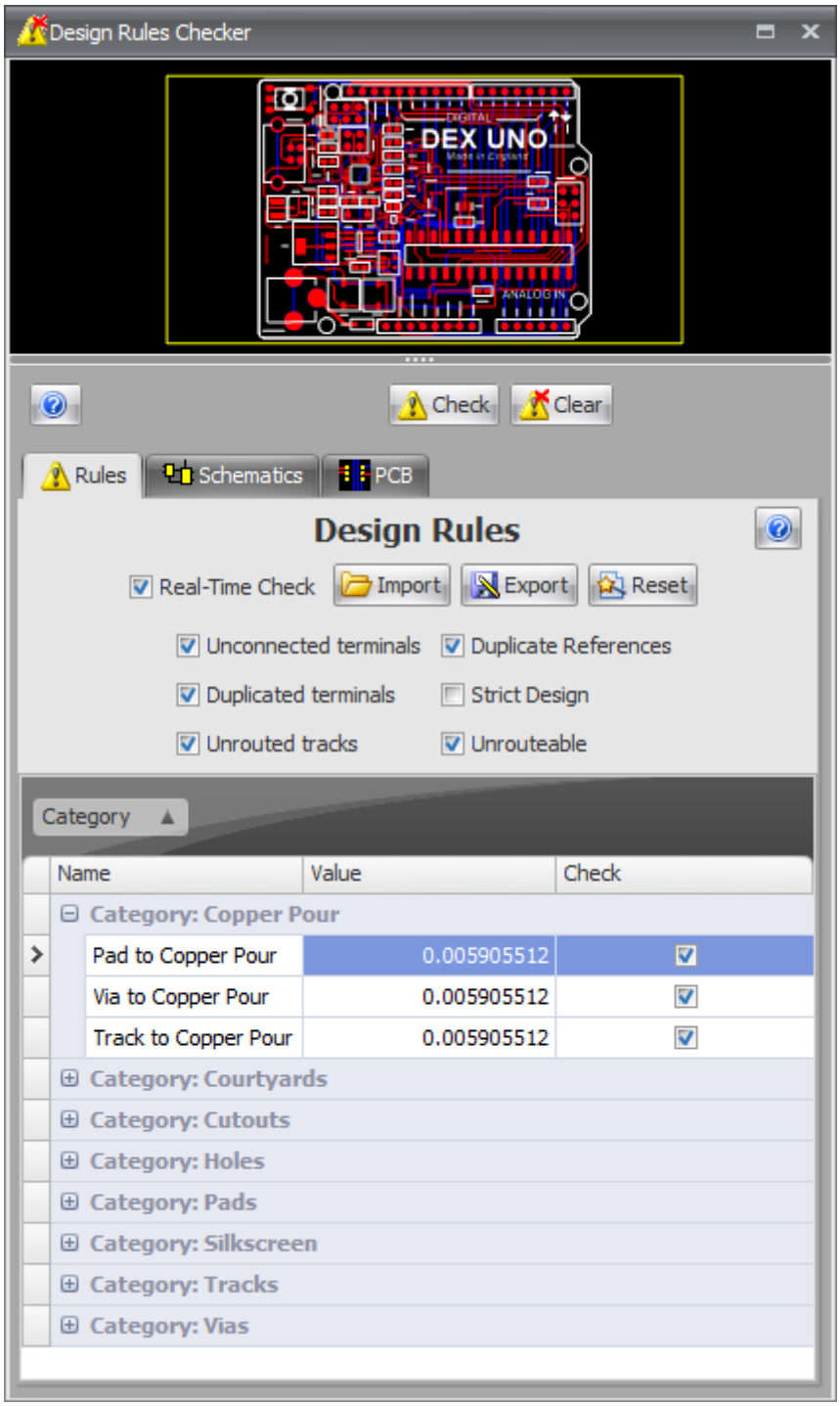
To set the Gerber and plotting settings click on the small button at the bottom right of the **Tools→Cam Rules** ribbon button group.







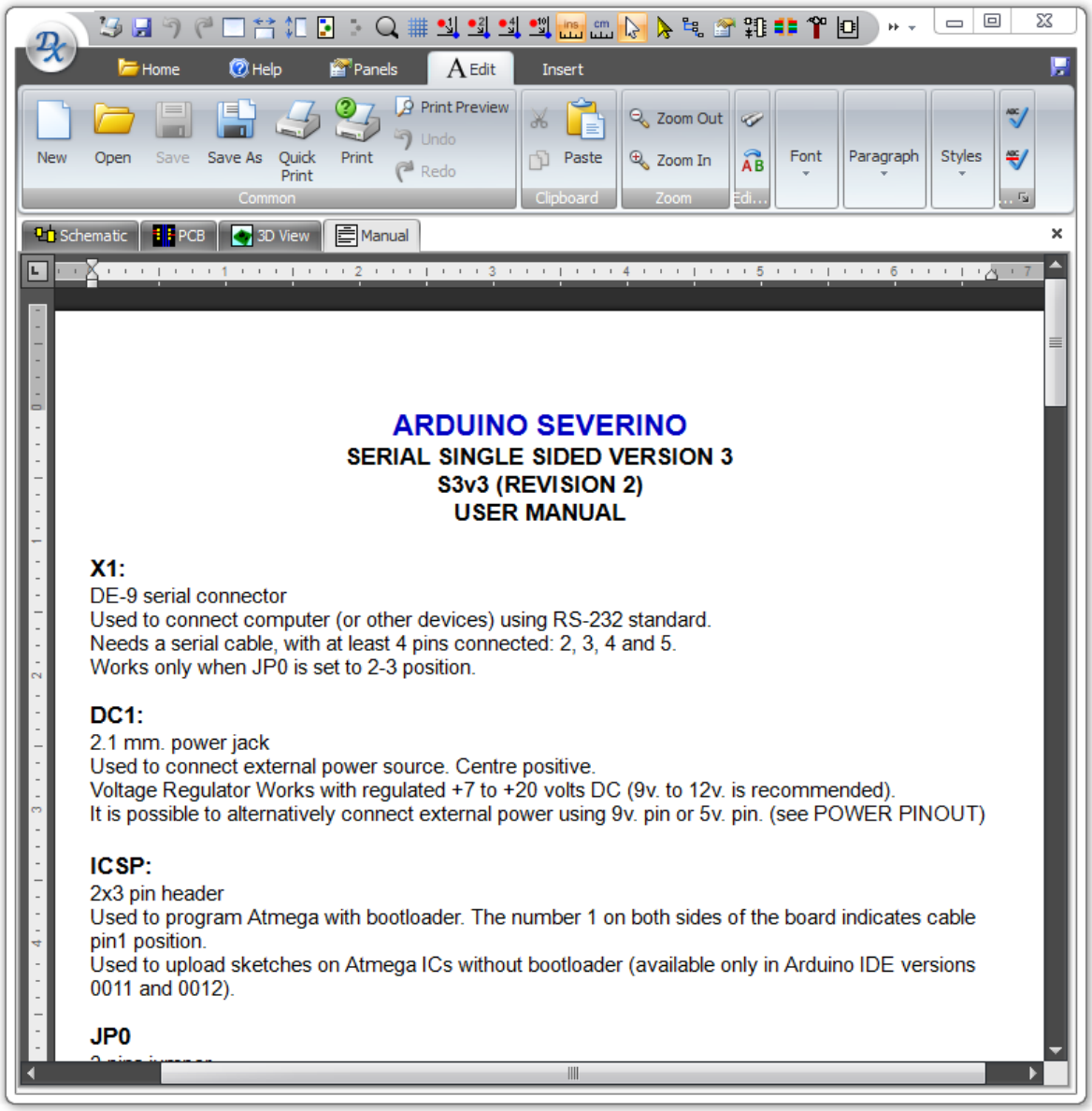
To set Design Rules, click on the small button at the bottom right of the **Tools→Design Rules** ribbon button group.



1.2.7 Text Document Sheets

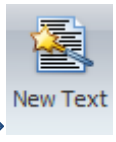
You can add text documents to your design.

Adding Text Document Sheets



A Text Document

1.2.7.1 Adding Text Document Sheets

To add a text documents to your design click the Home→Sheet→ 

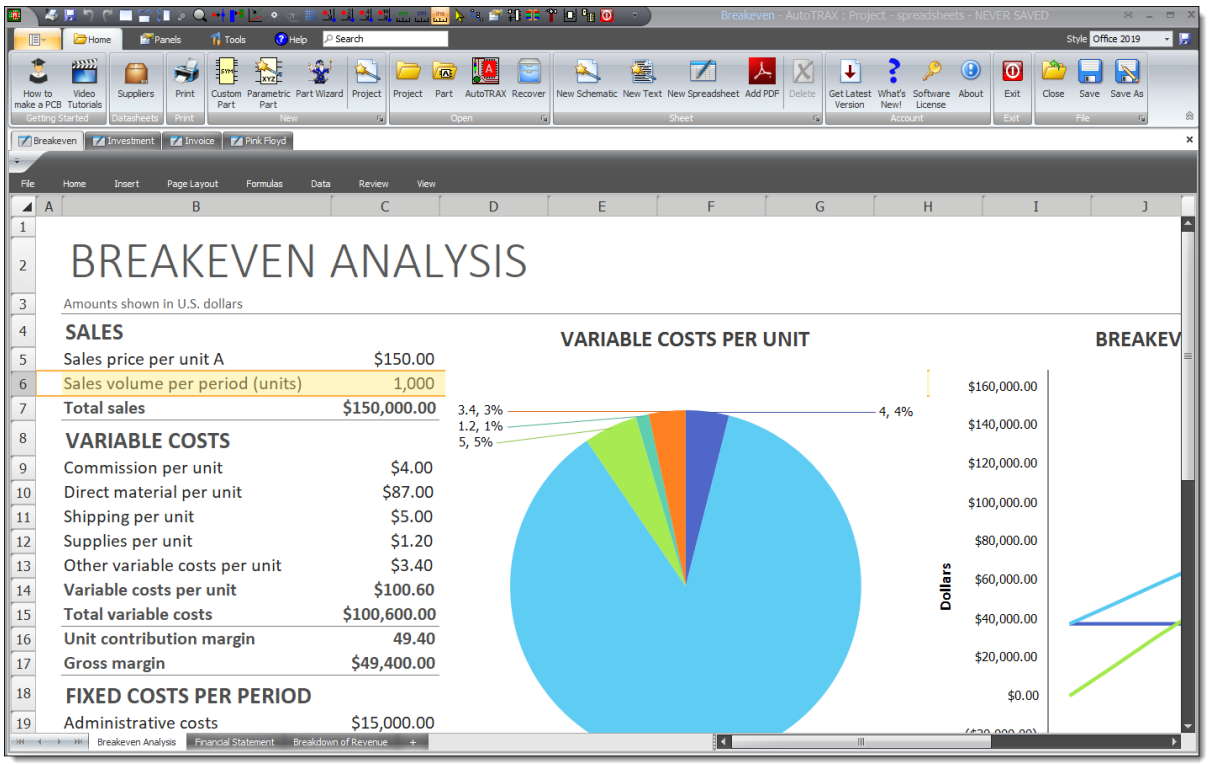
button.

1.2.8 PDF Sheets

You can add PDF sheets to your design.

1.2.9 Spreadsheets

Spreadsheets are computer applications or programs that let you store, organize, and manipulate data. They are set up in a grid of rows and columns, allowing for the arrangement of data to be easy to browse, and each cell in the grid can contain a number, text, or a formula. Formulas can perform calculations on other cells, allowing for complex mathematical modeling.



Sample Spreadsheet

Here are some features of spreadsheets:

Data Entry: They allow for the manual input of data or importation of data from various formats.

Data Organization: They help in arranging data in a structured manner, such as in rows and columns, which makes it easier to understand.

Calculations: They can perform calculations using data in the cells. This includes simple operations such as addition or multiplication, as well as more complex mathematical and statistical functions.

Data Analysis: They offer numerous tools for analyzing data, such as pivot tables, charting tools, and conditional formatting.

Automation and Scripting: Advanced users can automate tasks or complex operations using scripting languages like VBA (for Microsoft Excel) or Google Apps Script (for Google Sheets).

Collaboration: Many modern spreadsheet tools allow for real-time collaboration, with multiple users able to view and edit a spreadsheet at the same time.

The most commonly used spreadsheet programs include Microsoft Excel, Google Sheets, and Apple's Numbers. Each of these has their own strengths and weaknesses, but they all provide the core spreadsheet functionality of organizing and manipulating data in a grid format.

1.2.10 Printing and Plotting

AutoTRAX DEX can print any view and produce a series of plots for your PCB.

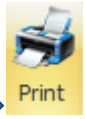
You can optionally add watermarks to your printed pages.

Printing

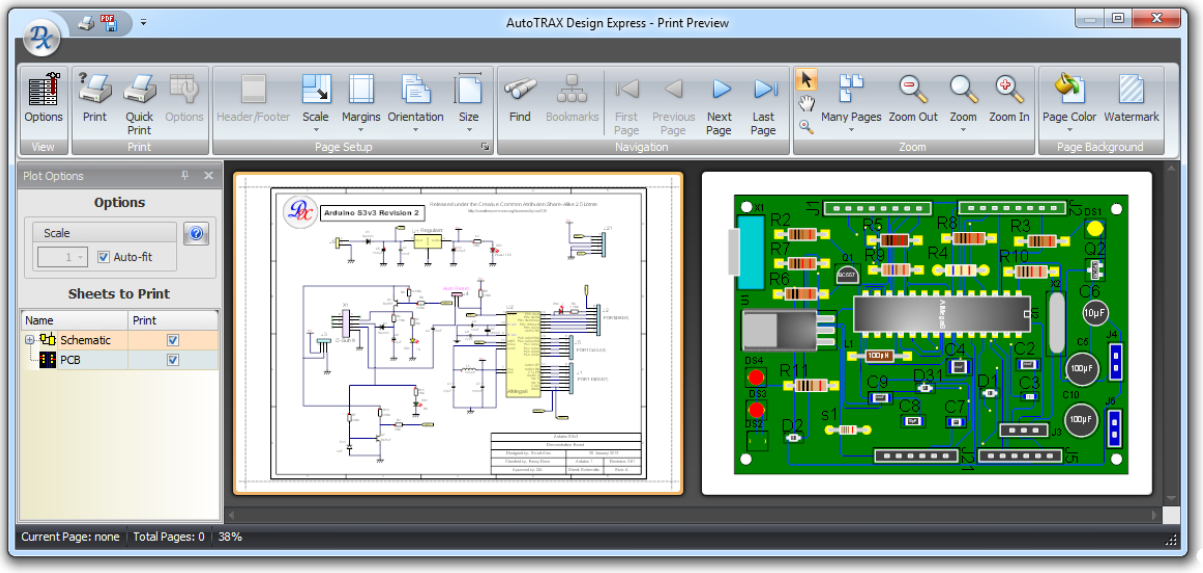
Plottings

Watermarks

1.2.10.1 Printing

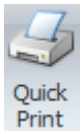
To print any view click the Home→Print→ 

button.



Print Preview

The print preview shown above consists of a ribbon menu, a options panel on the left and a preview window to the right of the options panel and below the ribbon menu.

click 

to print the selected pages.

Ribbon Menu

The ribbon menu can be split into 2 parts, the View button group on the left and the common ribbon button groups on the right.

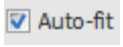
The 

button shows/hides the option panel shown below.

Click to read about the Common Print Menu

The Option Panel

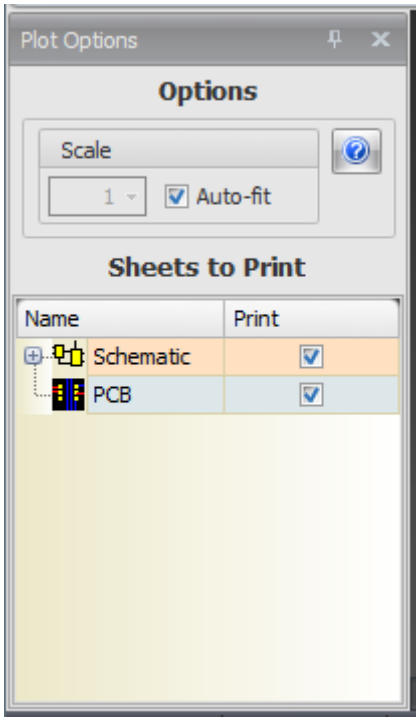
The options panel lets you scale the printing or have the sheets automatically fill the printed page,

Check 

to have the sheets automatically fill the printed page, If you uncheck it, you can set the print scale. Note, it is possible to set the scale such that the sheet will not all fit on the printed page.

Check/uncheck the check boxes in the Print column to include/exclude the sheet in the print run.

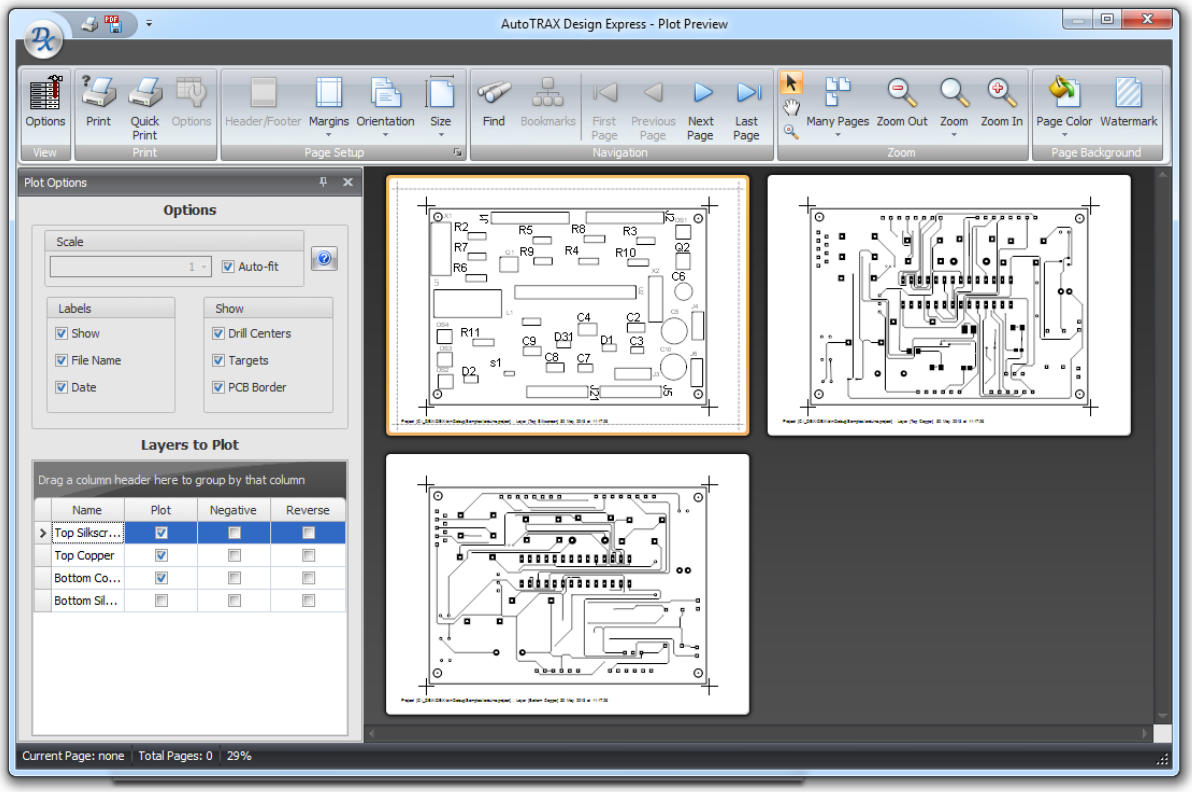
Clicking  displays this help topic.



Options Dialog

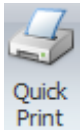
1.2.10.2 Plottings

To plot your PCB click the Tools→CAM→  button.



Plot Preview

The ploy preview shown above consists of a ribbon menu, a options panel on the left and a preview window to the right of the options panel and below the ribbon menu.

click 

to print the selected pages.

Ribbon Menu

The ribbon menu can be split into 2 parts, the View button group on the left and the common ribbon button groups on the right.

The 

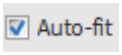
button shows/hides the option panel shown below.

Click to read about the Common Print Menu

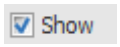
The Option Panel

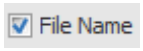
The options panel lets you scale the printing or have the sheets automatically fill the printed page.

Scale

Check  to have the sheets automatically fill the printed page, If you uncheck it, you can set the print scale. Note, it is possible to set the scale such that the sheet will not all fit on the printed page.

Labels

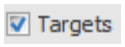
 Check to print plot layer labels

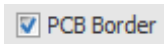
 Check to add a the file name to each plot.

 Check to add a date to each plot.

Show

 Check add small drill target/centers to each pad/via that has a hole.

 Check to add targets to the printed output.

 Check to include the PCB border in the printed output.

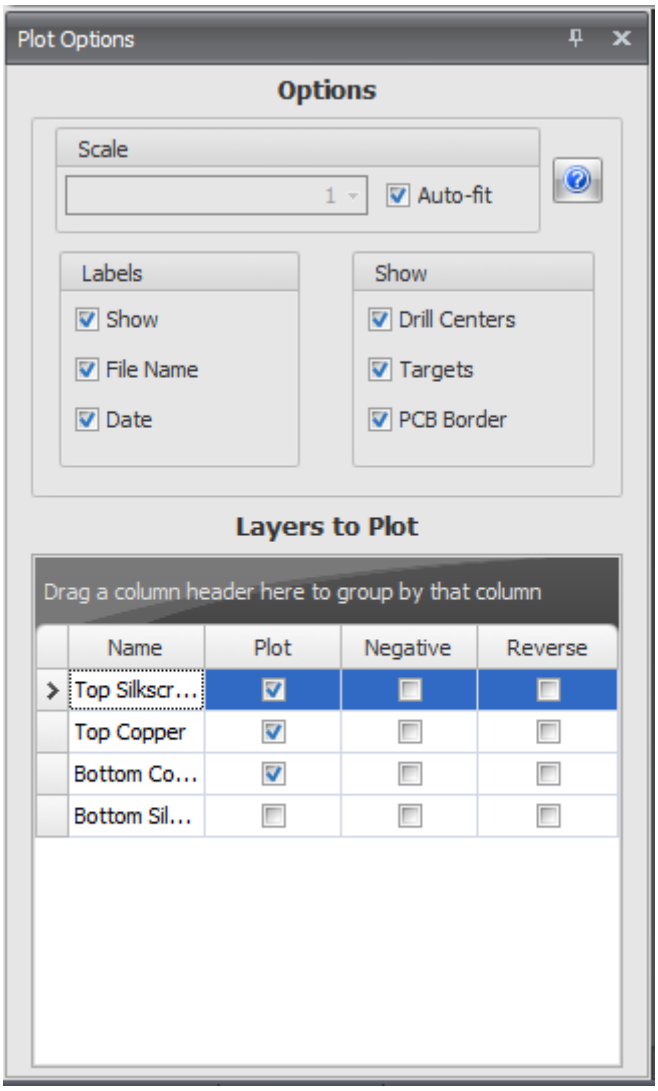
Layers to Plot

Check/uncheck the check boxes in the Plot column to include/exclude the layer in the print run.

Check/uncheck the check boxes in the Negative column to print that sheet in negative.

Check/uncheck the check boxes in the Reverse column to print that sheet reversed in the horizontal direction.

Clicking  displays this help topic.



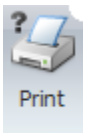
Options Dialog

1.2.10.3 Print Menu

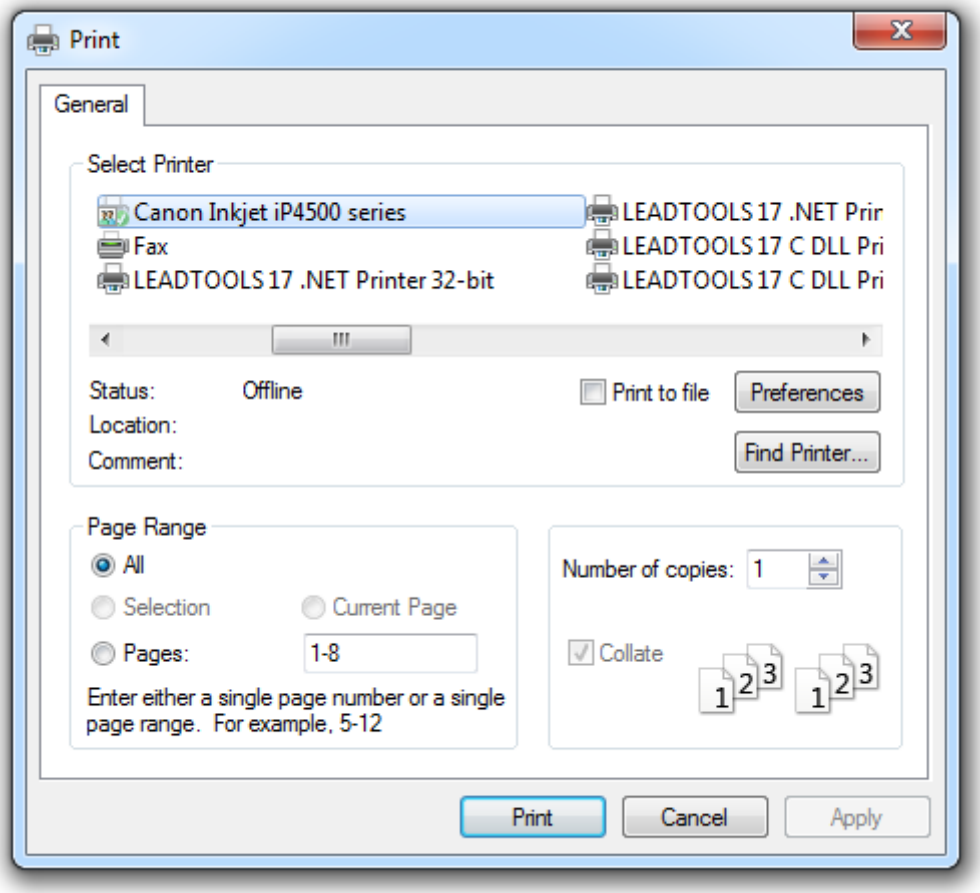
The common print menu is shown below.

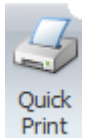
Print common menu items

Print

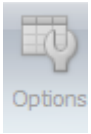


Select a printer, number of copies, and other printing options before printing.



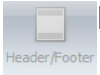


Send the document directly to the default printer without making changes.

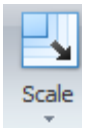


Open the Print Options dialog, in which you can change printing options.

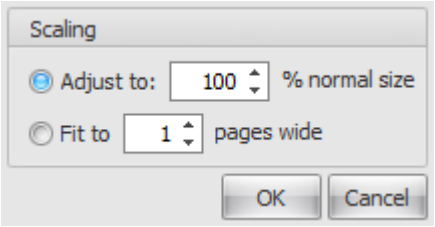
Page Setup



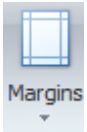
Edit the header and footer of the document.



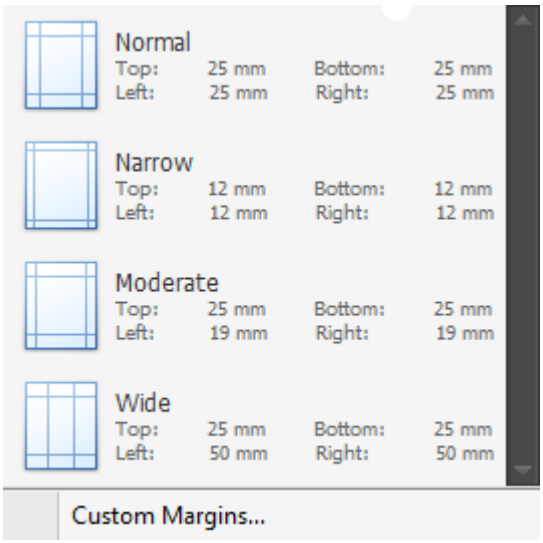
Stretch or shrink the printed output to a percentage of its actual size.



The scaling dialog



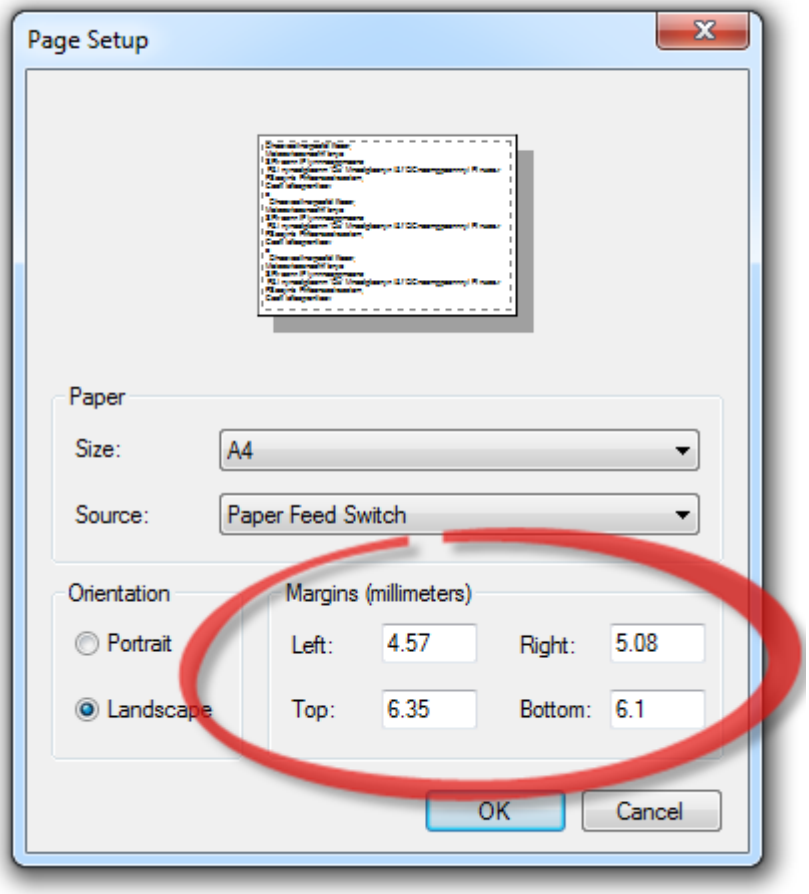
Select the margin sizes for the entire document. To apply specific margin sizes to the document, click Custom Margins.



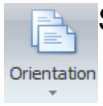
The margins dialog

Custom Margins

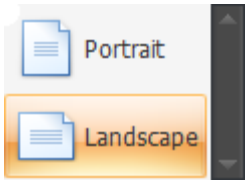
Click this to display the custom margins dialog.



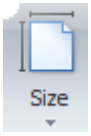
The custom margins dialog



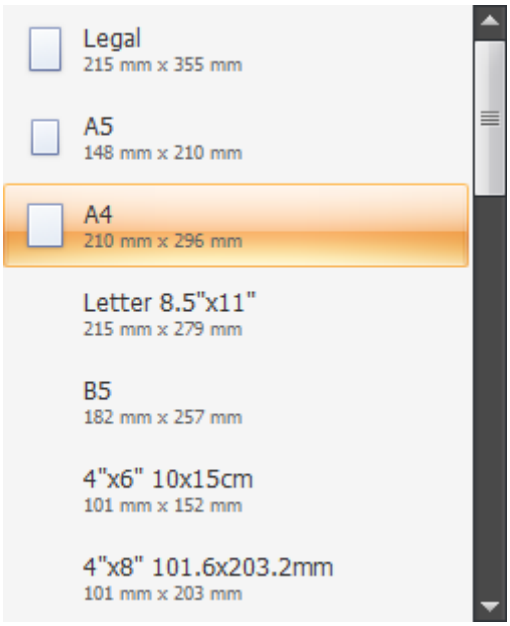
Switch the pages between portrait and landscape layouts.



The orientation selector

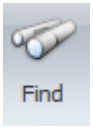


Choose a paper size for the current section.

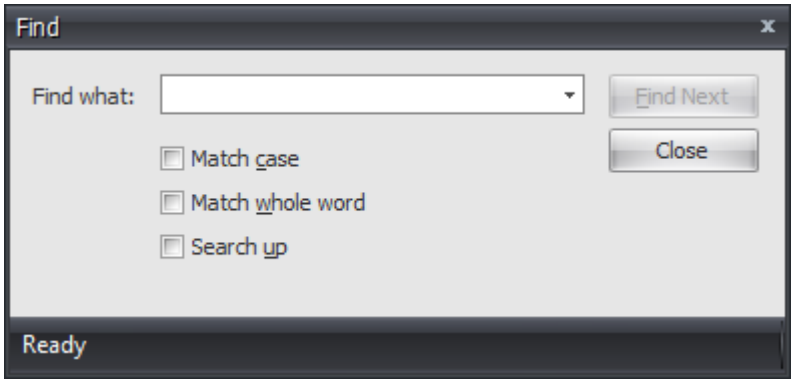


The page size selector

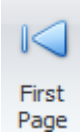
Navigation



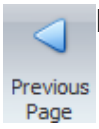
Display the find dialog.



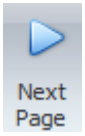
Open the document map which allows you to which allows you to view a structural view of the design.



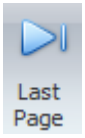
Navigate to the first page of the design.



Navigate to the previous page of the design.



Navigate to the next page of the design.



Navigate to the last page of the design.

Zoom

Mouse Pointer

Show the mouse pointer.

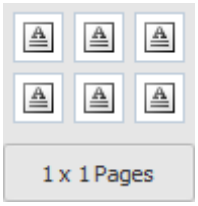
Hand Tool

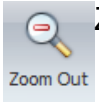
Invoke the Hand tool to manually scroll through pages.

Magnifier

Invoke the Magnifier tool. Clicking once on a document zooms it so that a single page becomes entirely visible, while clicking another time zooms it to 100% of the normal size.

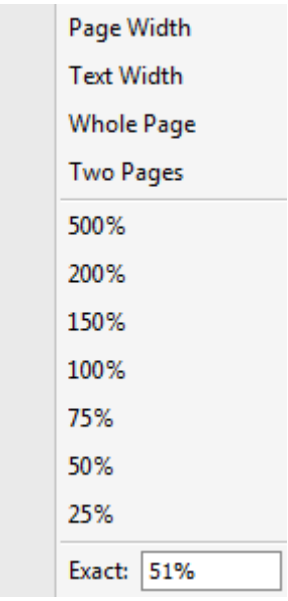
Choose the page layout to arrange the document pages in preview.

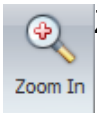




Zoom out to see more of the page at a reduced size.

Change the zoom level of the document preview.

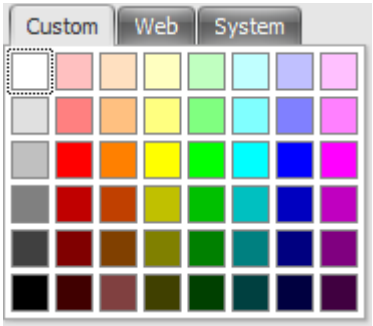




Zoom in to get a close-up view of the document.

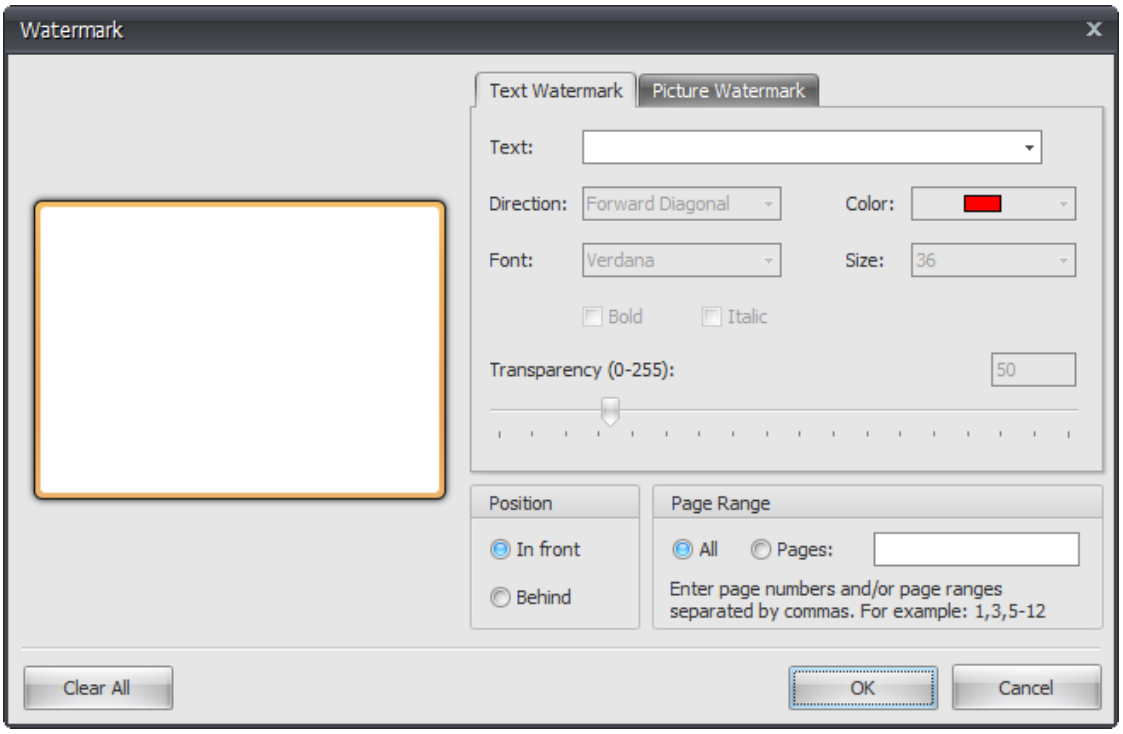
Page Background

Choose a color for the background of the document pages.



The background color dialog

Insert ghosted text or image behind the content of a page. This is often used to indicate that a document is to be treated specially see Watermarks



The watermark dialog

1.2.10.4 Watermarks

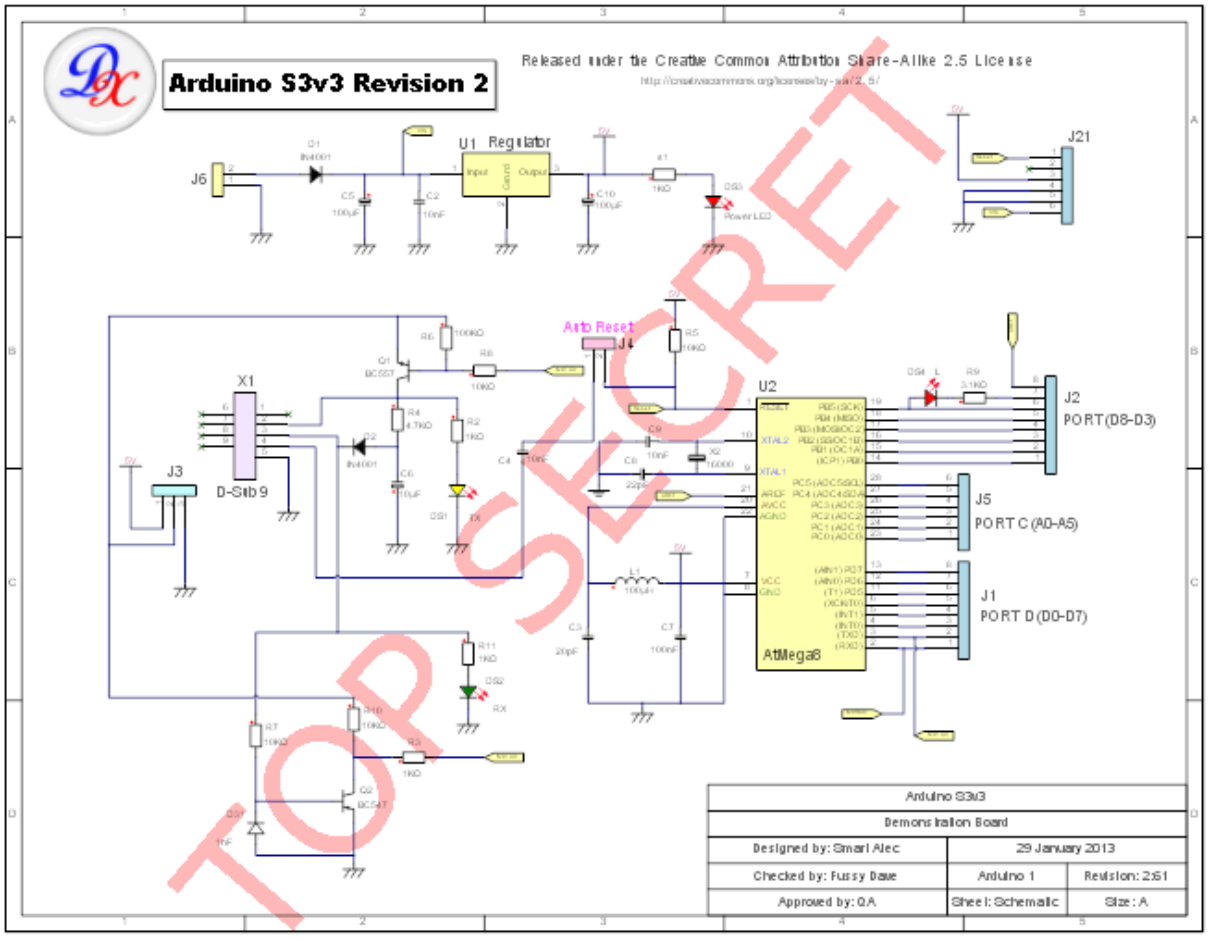
AutoTRAX DEX can add watermarks to your printed output. There are 2 basic types of watermark.

1. Text Watermarks
2. Picture Watermarks

To set the watermark click the 

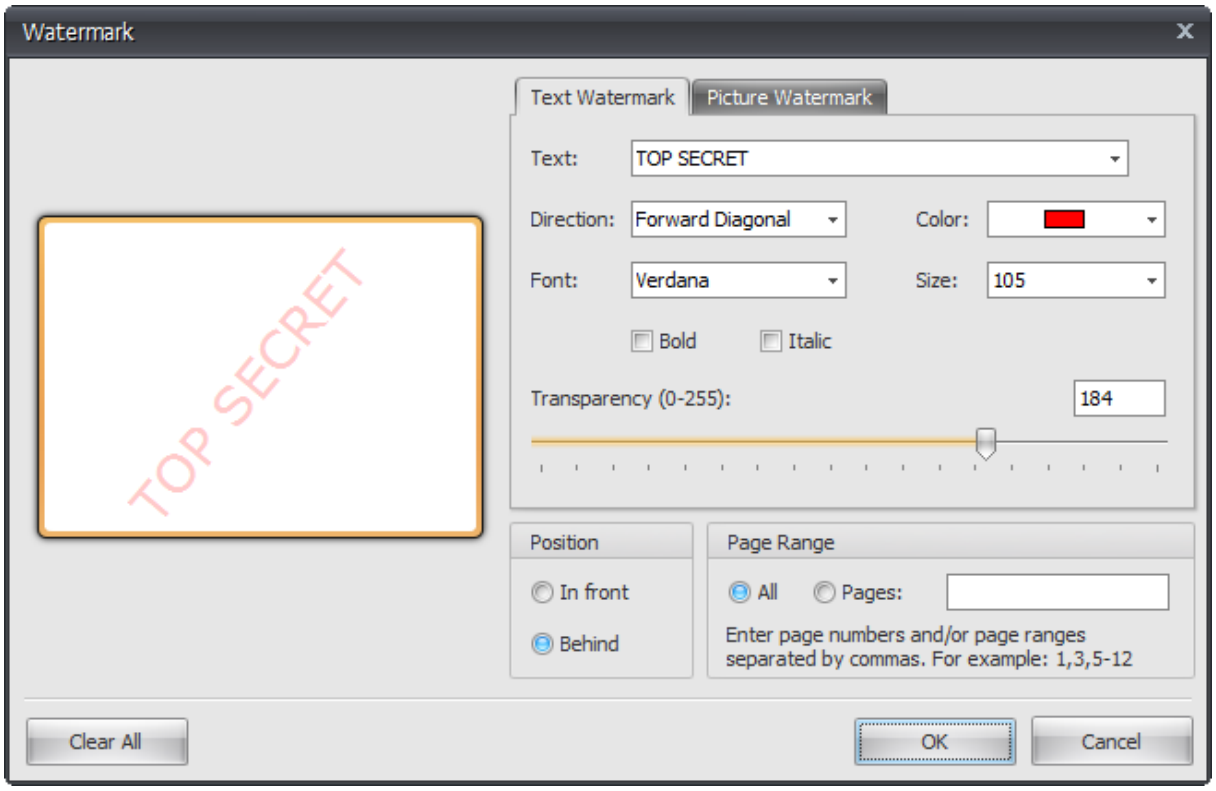
button in the print preview menu.

1.2.10.4.1 Text Watermarks



A Text Watermark

Select the Text Watermark in the watermark dialog to add a text watermark and define its properties.



The text watermark dialog

Preview

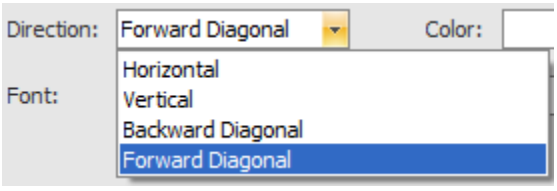
You can see a preview of the watermark on the left of the dialog.

Text

Enter the text for the watermark.

Direction

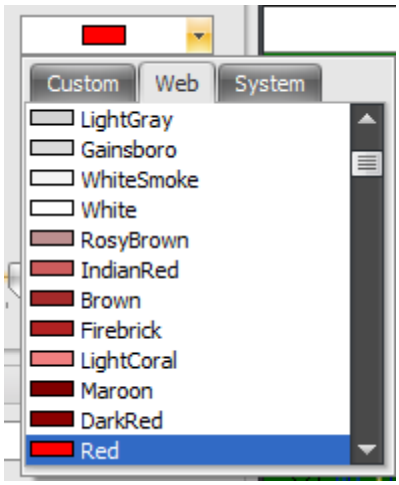
Select the direction for the text from one of four options.



Set the watermark's direction

Color

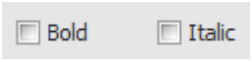
Select the color for the watermark text.



Set the watermark color

Font

Select the font for the watermark text. You can also select a bold and/or italic



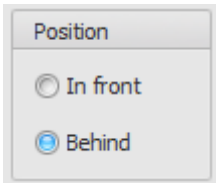
font.

Size

Select the height for the watermark text.

Position

You can set the position of the watermark to be in front of the page or behind all graphics on the page.



Set the watermark's position

Transparency

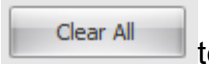
You can set the level of transparency for the text so you can 'see through' the watermark.

Set the watermark's transparency

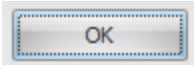
Page Range

Enter the page numbers and/or page ranges separated by commas. For example: 1,3,5-12 to add watermarks to pages 1,3 and 5 to 12.

Finally

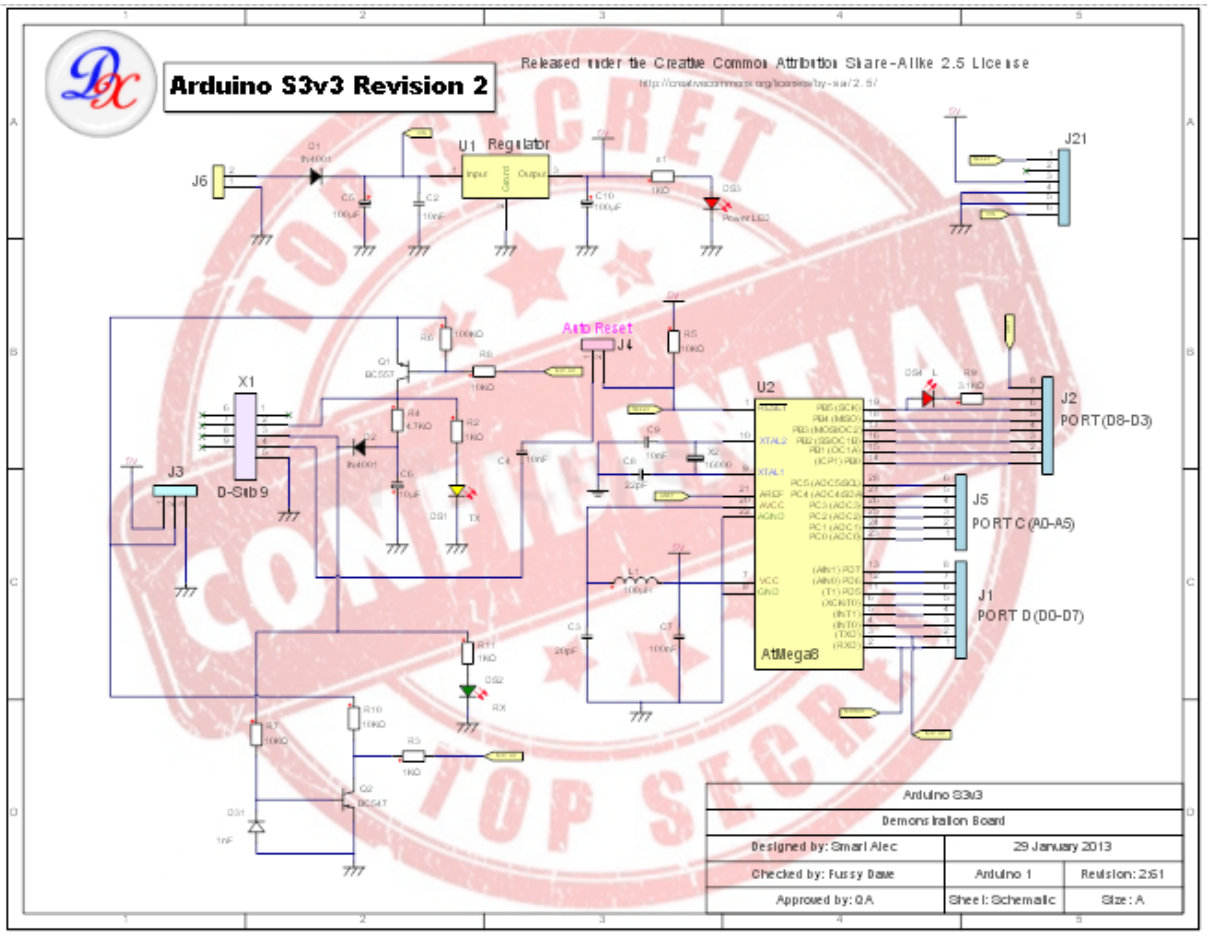
Click 

to reset the watermark.

Click the 

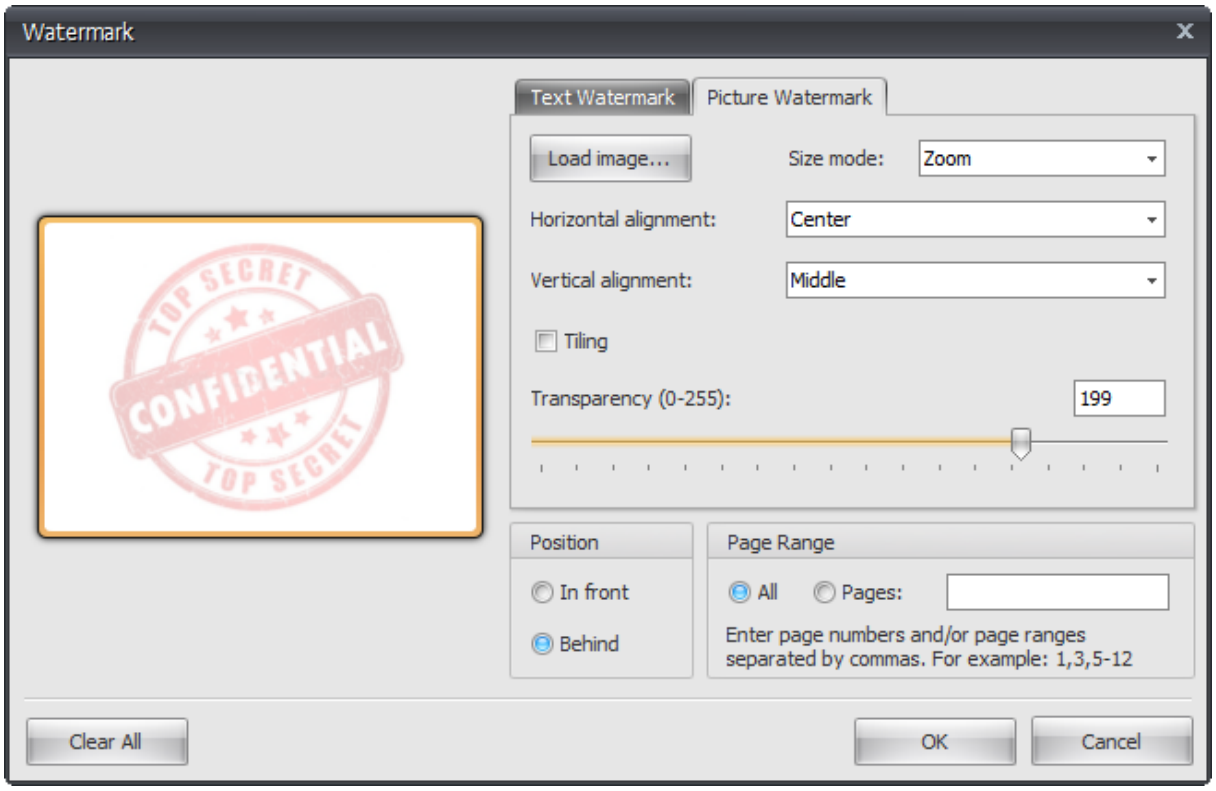
button to accept your changes.

1.2.10.4.2 Picture Watermarks



A Picture Watermark

Select the Picture Watermark to add an image watermark add define its properties.

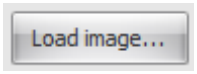


The picture watermark dialog

Preview

You can see a preview of the watermark on the left of the dialog.

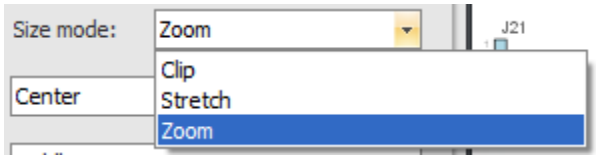
Load Image

Click 

to display the load image file dialog.

Size Mode

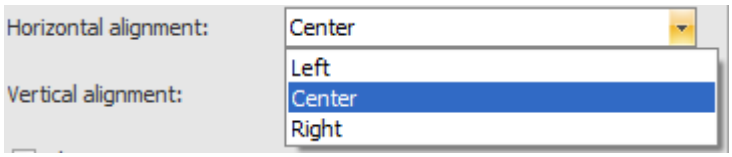
Select the sizing mode for the picture.



The picture's sizing mode

Horizontal alignment

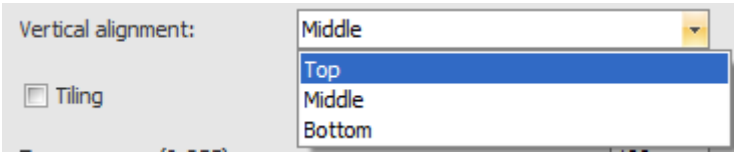
Select the horizontal alignment.



The picture's horizontal alignment

Vertical alignment

Select the vertical alignment.



The picture's vertical alignment

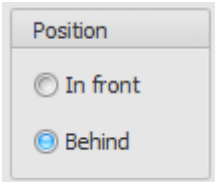
Tiling

Check 

to tile the image.

Position

You can set the position of the watermark to be in front of the page or behind all graphics on the page.



Set the watermark's position

Transparency

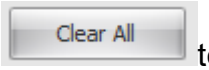
You can set the level of transparency for the text so you can 'see through' the watermark.

Set the watermark's transparency

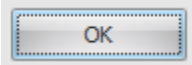
Page Range

Enter the page numbers and/or page ranges separated by commas. For example: 1,3,5-12 to add watermarks to pages 1,3 and 5 to 12.

Finally

Click 

to reset the watermark.

Click the 

button to accept your changes.

1.2.11 Circuit Simulation

AutoTRAX DEX uses the SPICE 3f5 circuit simulator.

SPICE is an analog-circuit simulator that is used to calculate and display the circuit behavior. It was originally developed at the Electronics Research Laboratory of the University of California, Berkeley (1975). The source code was made freely available and has been used as the basis of many commercial simulators, and academic projects.

SPICE is a useful tool, not a magic wand, and its results should always be examined critically; in some cases it may not give results at all, for others the results may be misleading.

Sample Projects

Simulation Samples

Adding Circuit Elements

Adding Circuit Elements

Adding Voltage and Current Sources

Always Add a Ground

Instruments

Adding Instruments

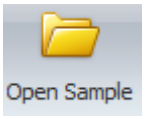
Analysis Result

The Analysis Probe

The Spice Reference Manual

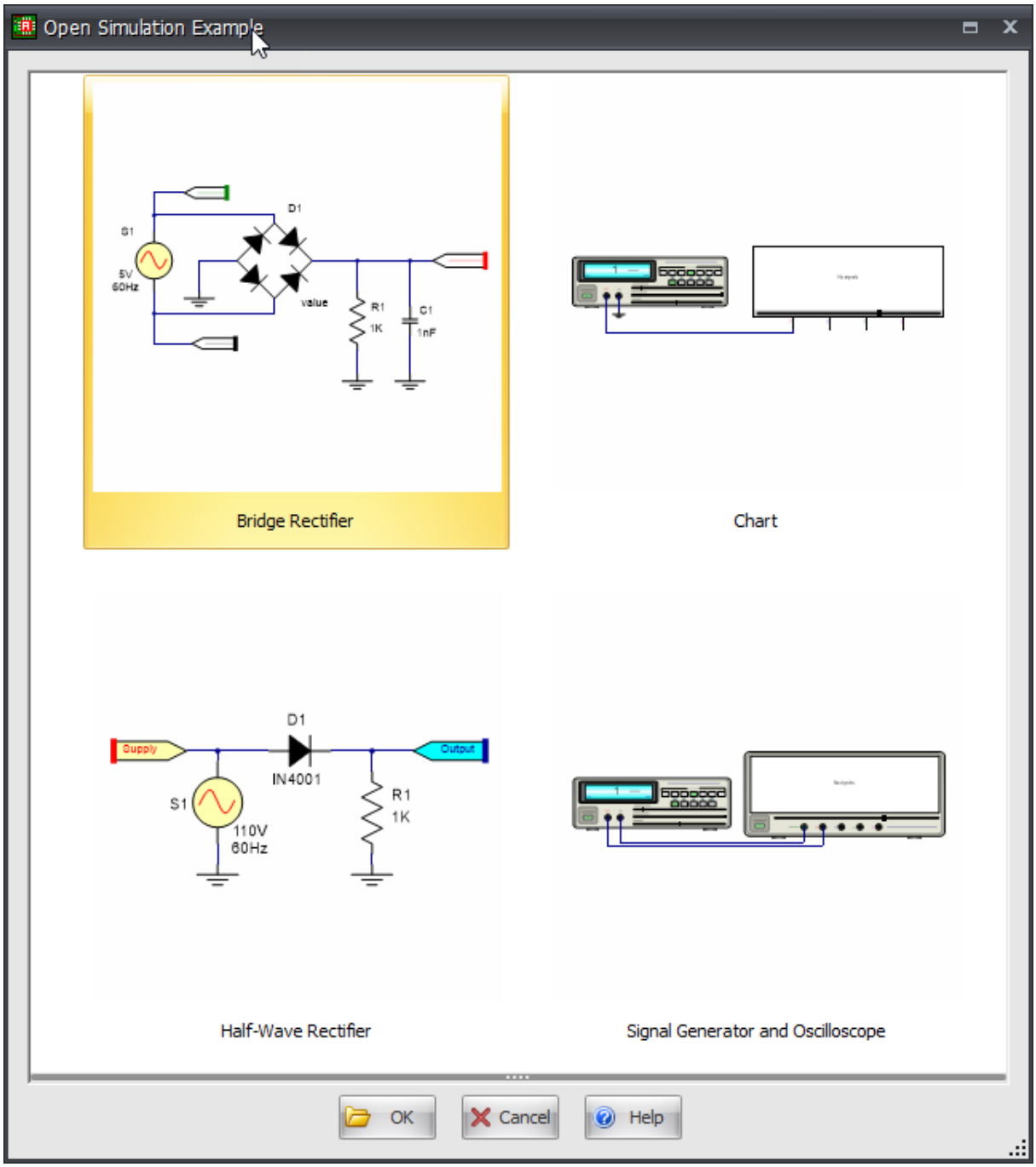
1.2.11.1 Simulation Samples

AutoTRAX DEX comes with many sample projects that demonstrate circuit simulation.

To open a sample clink on the 

button in the Simulation ribbon page.

You will then see a selection of circuit samples you can open. Double click on any circuit to open it.



Here are several samples:

Bridge Rectifier

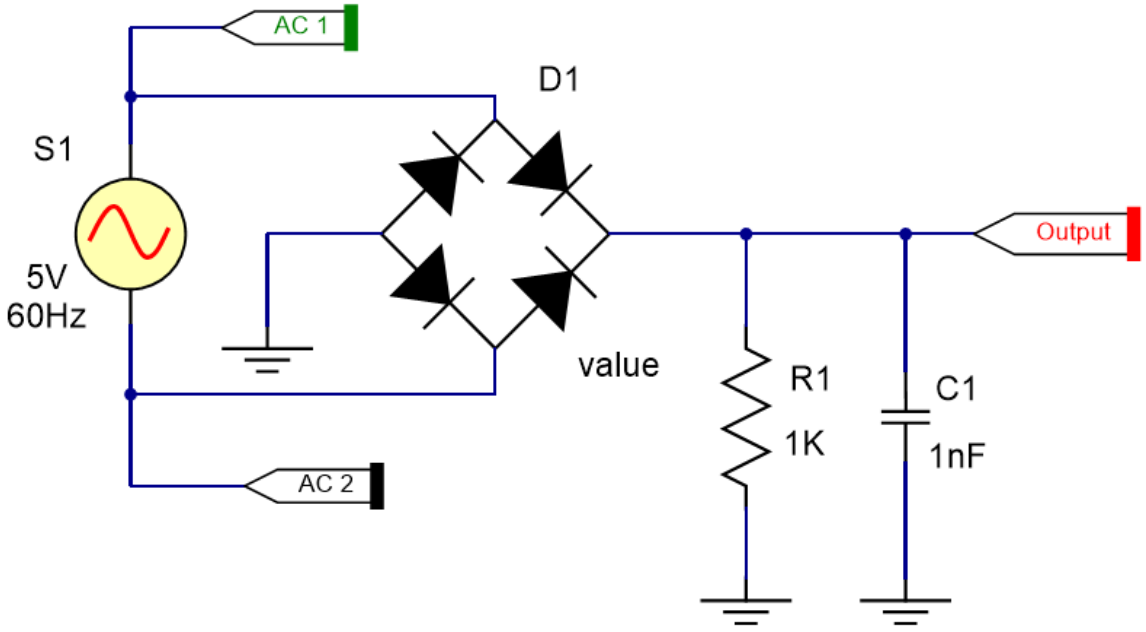
1.2.11.1.1 Bridge Rectifier

The bridge rectifier uses four diodes connected as shown below. When the input cycle is positive as in part (a), diodes D1 and D2 are forward-biased and conduct current in the direction shown. A voltage is developed across R1 that looks like the positive half ofthe input cycle. During this time, diodes D3 and D4 are reverse-biased.

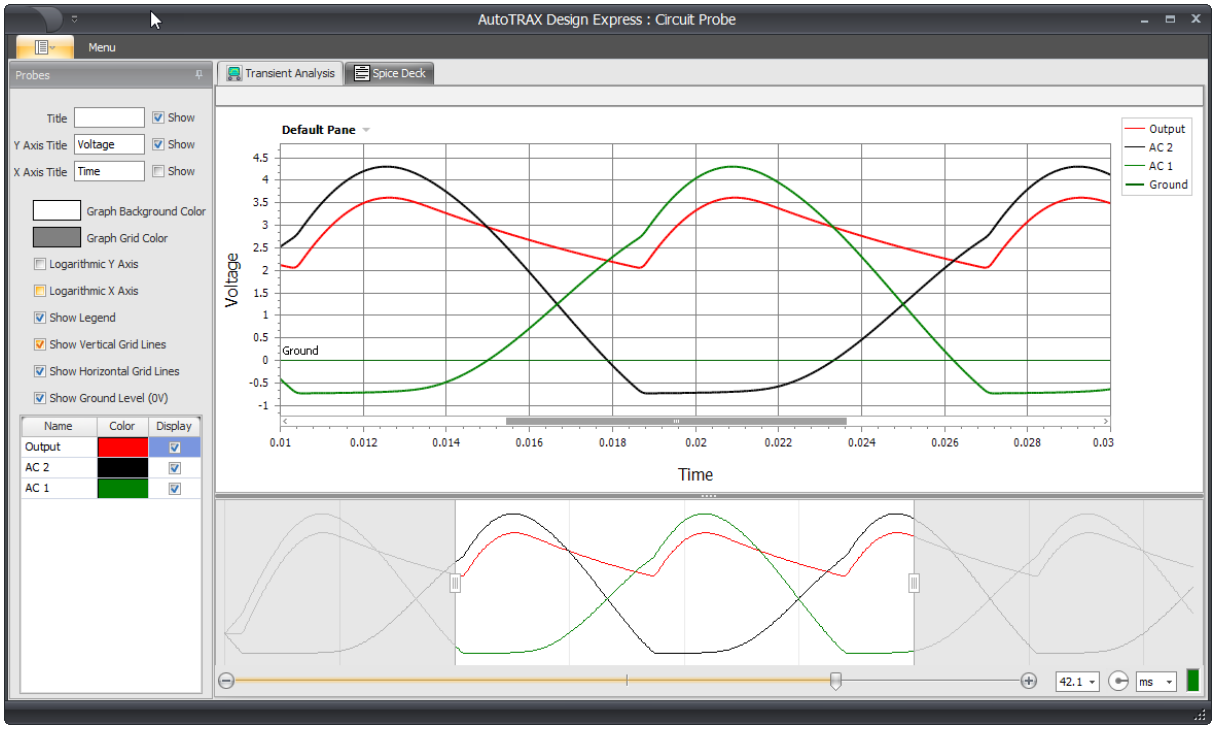
When the input cycle is negative as in Figure 2–38(b), diodes D3 and D4 are forwardbiased and conduct current in the same direction through RL as during the positive half-cycle. During the negative half-cycle, D1 and D2 are reverse-biased. A full-wave rectified output voltage appears across RL as a result of this action.

Bridge Output Voltage

A bridge rectifier with a transformer-coupled input is shown in Figure 2–39(a). During the positive half-cycle of the total secondary voltage, diodes D1 and D2 are forward-biased. Neglecting the diode drops, the secondary voltage appears across the load resistor. The same is true when D3 and D4 are forward-biased during the negative half-cycle.



The Spice Probe shows the electrical signals.



1.2.11.2 Adding Circuit Elements

There are many different circuit elements you can add to your schematic.

Passive Devices

Adding a Resistor

Adding a Capacitor

Adding an Inductor

Adding a Transformer

Active Devices

Adding a BJT Transistor

Adding a JFET Transistor

Adding a MOSFET Transistor

Adding a MESFET Transistor

1.2.11.2.1 Adding a Resistor

Coming soon...

1.2.11.2.2 Adding a Capacitor

Coming soon...

1.2.11.2.3 Adding an Inductor

Coming soon...

1.2.11.2.4 Adding a Transformer

Coming soon...

1.2.11.2.5 Adding a Diode

Coming soon...

1.2.11.2.6 Adding Transmission Lines

There are 3 types of transmission lines.

The Lossless Transmission Line

The Lossy Transmission Line 1

The Lossy Transmission Line 2

1.2.11.2.6.1 The Lossless Transmission Line

Coming soon...

1.2.11.2.6.2 The Lossy Transmission Line 1

Coming soon...

1.2.11.2.6.3 The Lossy Transmission Line 2

Coming soon...

1.2.11.2.7 Adding Transistors

You can add 4 types of transistor to your design.

- **BJT Transistor**
- **JFET Transistor**
- **MOSFET Transistor**
- **MESFET Transistor**

1.2.11.2.7.1 Adding a BJT Transistor

Coming soon...

1.2.11.2.7.2 Adding a JFET Transistor

Coming soon...

1.2.11.2.7.3 Adding a MOSFET Transistor

Coming soon...

1.2.11.2.7.4 Adding a MESFET Transistor

Coming soon...

1.2.11.2.7.5 Adding a Switch

Coming soon...

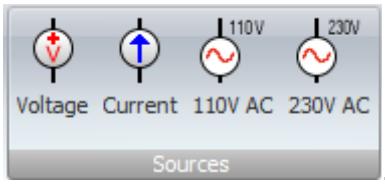
1.2.11.2.7.6 Adding a Custom Device

Coming soon...

1.2.11.3 Adding Voltage and Current Sources

Independent Sources

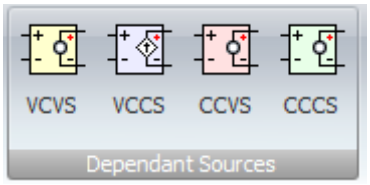
To add an independent source click on one of the buttons in the **Simulate→Source** ribbon button group



.

Dependent Sources

To add an dependent source click on one of the buttons in the **Simulate→Source** ribbon button group



1.2.11.3.1 Adding Independent Sources

Coming soon...

1.2.11.4 Adding Instruments

You can add virtual electrical instruments to your schematics.

Adding a Power Supply

Adding a Signal Generator

Adding an Oscilloscope

Adding a Chart

1.2.11.4.1 Adding a Power Supply

Coming soon...

1.2.11.4.2 Adding a Signal Generator

Coming soon...

1.2.11.4.3 Adding an Oscilloscope

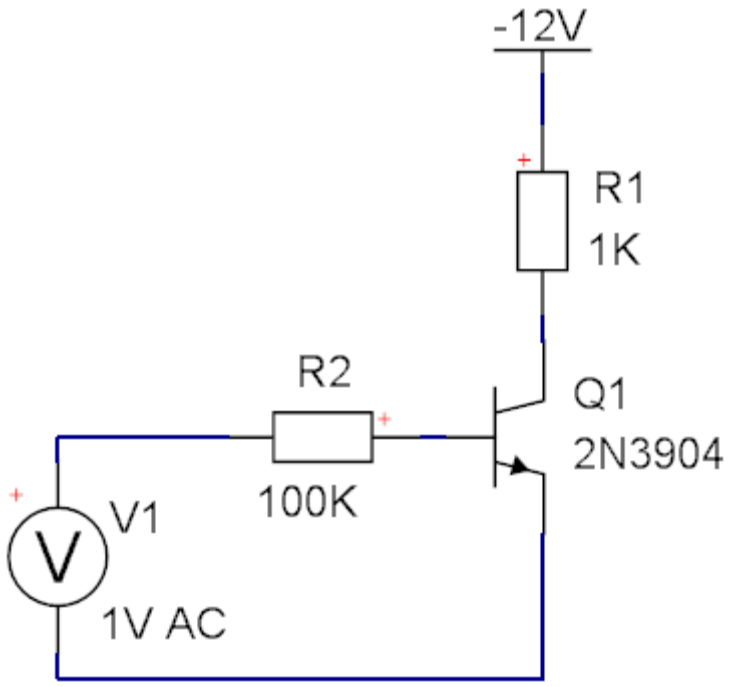
Coming soon...

1.2.11.4.4 Adding a Chart

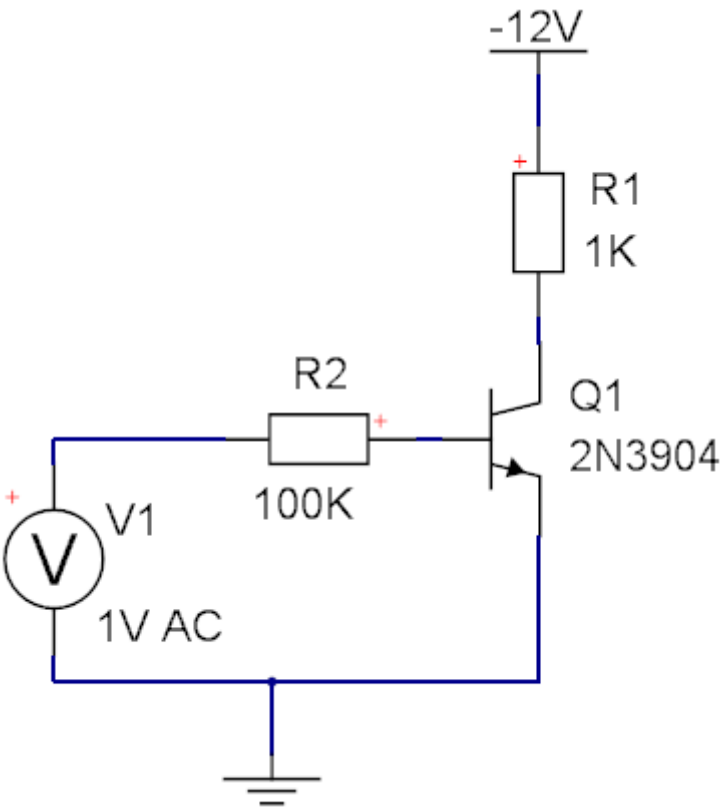
Coming soon...

1.2.11.5 Always Add a Ground

For the simulator to work you must include a ground connection.



No ground, simulation not possible

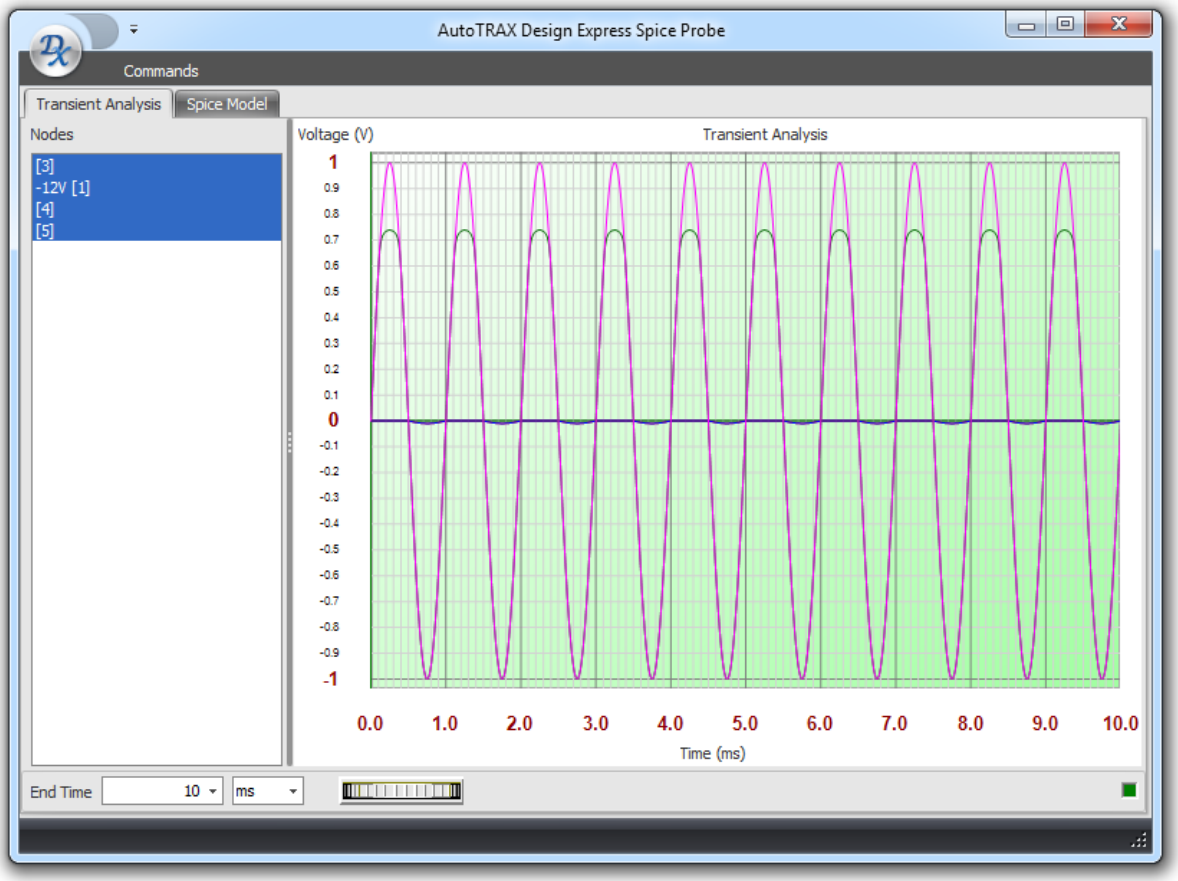


Ground present, simulation now possible

1.2.11.6 The Analysis Probe

To display the Analysis Probe click the Simulate→Analysis→ 

button.



The Analysis Probe

1.2.11.7 The Spice Reference Manual

SPICE (Simulation Program with Integrated Circuit Emphasis) is a general-purpose, open source analog electronic circuit simulator. It is a powerful program that is used in integrated circuit and board-level design to check the integrity of circuit designs and to predict circuit behavior.

1.2.11.7.1 Introduction

SPICE is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analysis.

It was developed at the Electronics Research Laboratory of the University of California, Berkeley by Laurence Nagel with direction from his research adviser, Prof. Donald Pederson.

The version used in AutoTRAX DEX is version 3f5.

SPICE is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analysis. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, loss-less and lossy transmission lines (two separate implementations), switches, uniform distributed RC lines, and the five most common semiconductor devices: diodes, BJTs, JFETs, MESFETs, and MOSFETs.

The SPICE3 version is based directly on SPICE 2G.6. While SPICE3 is being developed to include new features, it continues to support those capabilities and models that remain in extensive use in the SPICE2 program.

SPICE has built-in models for the semiconductor devices, and the user need specify only the pertinent model parameter values. The model for the BJT is based on the integral-charge model of Gummel and Poon; however, if the Gummel- Poon parameters are not specified, the model reduces to the simpler Ebers-Moll model. In either case, charge-storage effects, ohmic resistances, and a current-dependent output conductance may be included. The diode model can be used for either junction diodes or Schottky barrier diodes. The JFET model is based on the FET model of Shichman and Hodges. Six MOSFET models are implemented: MOS1 is described by a square-law I-V characteristic, MOS2 is an analytical model, while MOS3 is a semi-empirical model; MOS6 [2] is a simple analytic model accurate in the short-channel region; MOS4 and MOS5 are the BSIM (Berkeley Short-channel IGFET Model) and BSIM2. MOS2, MOS3, and MOS4 include second-order effects such as channel-length modulation, sub threshold conduction, scattering-limited velocity saturation, small-size effects, and charge-controlled capacitance.

1.2.11.7.1.1 Types of Analysis

DC Analysis

The dc analysis portion of SPICE determines the dc operating point of the circuit with inductors shorted and capacitors opened. The dc analysis options are specified on the .DC, .TF, and .OP control lines. A dc analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an ac small-signal analysis to determine the linearized, small-signal models for nonlinear devices. If requested, the dc small-signal value of a transfer function (ratio of output variable to input source), input resistance, and output resistance is also computed as a part of the dc solution. The dc analysis can also be used to generate dc transfer curves: a specified independent voltage or current source is stepped over a user-specified range and the dc output variables are stored for each sequential source value.

AC Small-Signal Analysis

The ac small-signal portion of SPICE computes the ac output variables as a function of frequency. The program first computes the dc operating point of the circuit and determines

linearized, small-signal models for all of the nonlinear devices in the circuit. The resultant linear circuit is then analyzed over a user-specified range of frequencies. The desired output of an ac small- signal analysis is usually a transfer function (voltage gain, transimpedance, etc). If the circuit has only one ac input, it is convenient to set that input to unity and zero phase, so that output variables have the same value as the transfer function of the output variable with respect to the input.

Transient Analysis

The transient analysis portion of SPICE computes the transient output variables as a function of time over a user-specified time interval. The initial conditions are automatically determined by a dc analysis. All sources that are not time dependent (for example, power supplies) are set to their dc value. The transient time interval is specified on a .TRAN control line.

Pole-Zero Analysis

The pole-zero analysis portion of SPICE computes the poles and/or zeros in the small-signal ac transfer function. The program first computes the dc operating point and then determines the linearized, small-signal models for all the nonlinear devices in the circuit. This circuit is then used to find the poles and zeros of the transfer function.

Two types of transfer functions are allowed : one of the form (output voltage)/(input voltage) and the other of the form (output voltage)/(input current). These two types of transfer functions cover all the cases and one can find the poles/zeros of functions like input/output impedance and voltage gain. The input and output ports are specified as two pairs of nodes.

The pole-zero analysis works with resistors, capacitors, inductors, linear-controlled sources, independent sources, BJTs, MOSFETs, JFETs and diodes. Transmission lines are not supported.

The method used in the analysis is a sub-optimal numerical search. For large circuits it may take a considerable time or fail to find all poles and zeros. For some circuits, the method becomes "lost" and finds an excessive number of poles or zeros.

Small-Signal Distortion Analysis

The distortion analysis portion of SPICE computes steady-state harmonic and intermodulation products for small input signal magnitudes. If signals of a single frequency are specified as the input to the circuit, the complex values of the second and third harmonics are determined at every point in the circuit. If there are signals of two frequencies input to the circuit, the analysis finds out the complex values of the circuit variables at the sum and difference of the input frequencies, and at the difference of the smaller frequency from the second harmonic of the larger frequency.

Distortion analysis is supported for the following nonlinear devices: diodes (DIO), BJT, JFET, MOSFETs (levels 1, 2, 3, 4/BSIM1, 5/BSIM2, and 6) and MESFETS. All linear devices are automatically supported by distortion analysis. If there are switches present in the circuit, the analysis continues to be accurate provided the switches do not change state under the small excitations used for distortion calculations.

Sensitivity Analysis

Spice3 will calculate either the DC operating-point sensitivity or the AC small-signal sensitivity of an output variable with respect to all circuit variables, including model parameters. Spice calculates the difference in an output variable (either a node voltage or a branch current) by perturbing each parameter of each device independently. Since the method is a numerical approximation, the results may demonstrate second order affects in highly sensitive parameters, or may fail to show very low but non-zero sensitivity. Further, since each variable is perturbed by a small fraction of its value, zero-valued parameters are not analyzed (this has the benefit of reducing what is usually a very large amount of data).

Noise Analysis

The noise analysis portion of SPICE does analysis device-generated noise for the given circuit. When provided with an input source and an output port, the analysis calculates the noise contributions of each device (and each noise generator within the device) to the output port voltage. It also calculates the input noise to the circuit, equivalent to the output noise referred to the specified input source. This is done for every frequency point in a specified range - the calculated value of the noise corresponds to the spectral density of the circuit variable viewed as a stationary gaussian stochastic process.

After calculating the spectral densities, noise analysis integrates these values over the specified frequency range to arrive at the total noise voltage/current (over this frequency range). This calculated value corresponds to the variance of the circuit variable viewed as a stationary gaussian process.

1.2.11.7.1.2 Analysis at Different Temperatures

All input data for SPICE is assumed to have been measured at a nominal temperature of 27 C, which can be changed by use of the TNOM parameter on the .OPTION control line. This value can further be overridden for any device which models temperature effects by specifying the TNOM parameter on the model itself. The circuit simulation is performed at a temperature of 27 C, unless overridden by a TEMP parameter on the .OPTION control line. Individual instances may further override the circuit temperature through the specification of a TEMP parameter on the instance.

Temperature dependent support is provided for resistors, diodes, JFETs, BJTs, and level 1, 2, and 3 MOSFETs. BSIM (levels 4 and 5) MOSFETs have an alternate temperature dependency scheme which adjusts all of the model parameters before input to SPICE. For details of the BSIM temperature adjustment, see [6] and [7].

Temperature appears explicitly in the exponential terms of the BJT and diode model equations. In addition, saturation currents have a built-in temperature dependence. The temperature dependence of the saturation current in the BJT models is determined by:

where k is Boltzmann's constant, q is the electronic charge, e.g. is the energy gap which is a model parameter, and XTI is the saturation current temperature exponent (also a model parameter, and usually equal to 3).

The temperature dependence of forward and reverse beta is according to the formula:

where T1 and T0 are in degrees Kelvin, and XTB is a user-supplied model parameter. Temperature effects on beta are carried out by appropriate adjustment to the values of F, ISE, R, and ISC (spice model parameters BF, ISE, BR, and ISC, respectively).

Temperature dependence of the saturation current in the junction diode model is determined by:

where N is the emission coefficient, which is a model parameter, and the other symbols have the same meaning as above. Note that for Schottky barrier diodes, the value of the saturation current temperature exponent, XTI, is usually 2.

Temperature appears explicitly in the value of junction potential, (in spice PHI), for all the device models. The temperature dependence is determined by:

where k is Boltzmann's constant, q is the electronic charge, Na is the acceptor impurity density, Nd is the donor impurity density, Ni is the intrinsic carrier concentration, and Eg is the energy gap.

Temperature appears explicitly in the value of surface mobility, 0 (or UO), for the MOSFET model. The temperature dependence is determined by:

The effects of temperature on resistors is modeled by the formula:

where T is the circuit temperature, T0 is the nominal temperature, and TC1 and TC2 are the first- and second-order temperature coefficients.

1.2.11.7.1.3 Convergence

Both dc and transient solutions are obtained by an iterative process which is terminated when both of the following conditions hold:

1. The nonlinear branch currents converge to within a tolerance of 0.1% or 1 picoamp (1.0e-12 Amp), whichever is larger.
2. The node voltages converge to within a tolerance of 0.1% or 1 microvolt (1.0e-6Volt),whichever is larger.

Although the algorithm used in SPICE has been found to be very reliable, in some cases it fails to converge to a solution. When this failure occurs, the program terminates the job.

Failure to converge in dc analysis is usually due to an error in specifying circuit connections, element values, or model parameter values. Regenerative switching circuits or circuits with positive feedback probably will not converge in the dc analysis unless the OFF option is used

for some of the devices in the feedback path, or the .NODESET control line is used to force the circuit to converge to the desired state.

1.2.11.7.2 Circuit Description

The circuit to be analyzed is described to SPICE by a set of element lines, which define the circuit topology and element values, and a set of control lines, which define the model parameters and the run controls.

1.2.11.7.2.1 Circuit Elements and Models

The circuit element include:

- Transistors and Diodes
- Transmission Lines
- Voltage and Current Sources

Spice includes models for transistors:

- Bipolar Junction Transistors (BJTs)
- BJT Models (NPN/PNP)
- JFET Models (NJF/PJF)[****]
- Junction Field-Effect Transistors (JFETs)
- MESFET Models (NMF/PMF)
- MOSFETs
- MOSFET Models (NMOS/PMOS)

And the following diode models:

- Diode Model
- Junction Diodes

The area factor used on the diode, BJT, JFET, and MESFET devices determines the number of equivalent parallel devices of a specified model. The affected parameters are marked with an asterisk under the heading 'area' in the model descriptions below. Several geometric factors associated with the channel and the drain and source diffusions can be specified on the MOSFET device line.

Two different forms of initial conditions may be specified for some devices. The first form is included to improve the dc convergence for circuits that contain more than one stable state. If a device is specified OFF, the dc operating point is determined with the terminal voltages for that device set to zero. After convergence is obtained, the program continues to iterate to obtain the exact value for the terminal voltages. If a circuit has more than one dc stable state, the OFF option can be used to force the solution to correspond to a desired state. If a device is specified OFF when in reality the device is conducting, the program still obtains the correct solution (assuming the solutions converge) but more iterations are required since the program must independently converge to two separate solutions. The .NODESET control line serves a similar purpose as the OFF option. The .NODESET option is easier to apply and is the preferred means to aid convergence.

The second form of initial conditions are specified for use with the transient analysis. These are true 'initial conditions' as opposed to the convergence aids above. See the description of the .IC control line and the .TRAN control line for a detailed explanation of initial conditions.

General form:

QXXXXXXX NC NB NE <NS> MNAME <AREA> <OFF> <IC=VBE, VCE>

<TEMP=T>

Examples:

Q23 10 24 13 QMOD IC=0.6, 5.0 Q50A 11 26 4 20 MOD1

NC, NB, and NE are the collector, base, and emitter nodes, respectively. NS is the (optional) substrate node. If unspecified, ground is used. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for the dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification using IC=VBE, VCE is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line description for a better way to set transient initial conditions. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

The bipolar junction transistor model in SPICE is an adaptation of the integral charge control model of Gummel and Poon. This modified Gummel-Poon model extends the original model to include several effects at high bias levels. The model automatically simplifies to the simpler Ebers-Moll model when certain parameters are not specified. The parameter names used in the modified Gummel-Poon model have been chosen to be more easily understood by the program user, and to reflect better both physical and circuit design thinking.

The dc model is defined by the parameters IS, BF, NF, ISE, IKF, and NE which determine the forward current gain characteristics, IS, BR, NR, ISC, IKR, and NC which determine the reverse current gain characteristics, and VAF and VAR which determine the output

conductance for forward and reverse regions. Three ohmic resistances RB, RC, and RE are included, where RB can be high current dependent. Base charge storage is modeled by forward and reverse transit times, TF and TR, the forward transit time TF being bias dependent if desired, and nonlinear depletion layer capacitances which are determined by CJE, VJE, and MJE for the B-E junction , CJC, VJC, and MJC for the B-C junction and CJS, VJS, and MJS for the C-S (Collector-Substrate) junction. The temperature dependence of the saturation current, IS, is determined by the energy-gap, EG, and the saturation current temperature exponent, XTI. Additionally base current temperature dependence is modeled by the beta temperature exponent XTB in the new model. The values specified are assumed to have been measured at the temperature TNOM, which can be specified on the .OPTIONS control line or overridden by a specification on the .MODEL line.

The BJT parameters used in the modified Gummel-Poon model are listed below. The parameter names used in earlier versions of SPICE2 are still accepted.

name	parameter	units	default	example	area
IS	transport saturation current	A	1.0e-16	1.0e-15	*
BF	ideal maximum forward beta	-	100	100	
NF	forward current emission coefficient	-	1.0	1	
VAF	forward Early voltage	V	infinite	1	
IKF	corner for forward beta high current roll-off	A	infinite	200	*
ISE	B-E leakage saturation current	A	0	0.01	*
NE	B-E leakage emission coefficient	-	1.5	1.0e-13	
BR	ideal maximum reverse beta	-	1	2	
NR	reverse current emission coefficient	-	1	0.11	
VAR	reverse Early voltage	V	infinite	1200	
IKR	corner for reverse beta high current roll-off	A	infinite	2000.01	*
ISC	leakage saturation current	A	0	0.01.0e-13	*
NC	leakage emission coefficient	-	2	1.5	
RB	zero bias base resistance	Ohm	0	100	*
IRB	current where base resistance falls halfway to its min value	A	infinite	0.1	*

RBM	minimum base resistance at high currents	Ohm	RB	10	*
RE	emitter resistance	Ohm	0	1	*
RC	collector resistance	Ohm	0	10	*
CJE	B-E zero-bias depletion capacitance	F	0	2pF	*
VJE	B-E built-in potential	V	0.75	0.6	
MJE	B-E junction exponential factor	-	0.33	0.33	
TF	ideal forward transit time	sec	0	0.1ns	
XTF	coefficient for bias dependence of TF	-	0		
VTF	voltage describing VBC dependence of TF	V	infinite		
ITF	high-current parameter for effect on TF	A	0		*
PTF	excess phase at freq=1.0/(TF*2PI) Hz	deg	0		
CJC	B-C zero-bias depletion capacitance	F	0	2pF	*
VJC	B-C built-in potential	V	0.75	0.5	
MJC	B-C junction exponential factor	-	0.33	0.5	
XCJC	fraction of B-C depletion capacitance connected to internal base node	-	1		
TR	ideal reverse transit time	sec	0	10ns	
CJS	zero-bias collector-substrate capacitance	F	0	2pF	*
VJS	ubstrate junction built-in potential	V	0.75		
MJS	substrate junction exponential factor	-	0	0.5	
XTB	forward and reverse beta temperature exponent	-	0		
EG	energy gap for temperature	eV	1.11		
XTI	temperature exponent for effect on IS	-	3		
KF	flicker-noise coefficient	-	0		
AF	flicker-noise exponent	-	1		

FC	coefficient for forward-bias depletion capacitance formula	-	-.5		
TNOM	Parameter measurement temperature	ºC	27	50	

The dc characteristics of the diode are determined by the parameters IS and N. An ohmic resistance, RS, is included. Charge storage effects are modeled by a transit time, TT, and a nonlinear depletion layer capacitance which is determined by the parameters CJO, VJ, and M. The temperature dependence of the saturation current is defined by the parameters EG, the energy and XTI, the saturation current temperature exponent. The nominal temperature at which these parameters were measured is TNOM, which defaults to the circuit-wide value specified on the .OPTIONS control line. Reverse breakdown is modeled by an exponential increase in the reverse diode current and is determined by the parameters BV and IBV (both of which are positive numbers).

The JFET model is derived from the FET model of Shichman and Hodges. The dc characteristics are defined by the parameters VTO and BETA, which determine the variation of drain current with gate voltage, LAMBDA, which determines the output conductance, and IS, the saturation current of the two gate junctions. Two ohmic resistances, RD and RS, are included. Charge storage is modeled by nonlinear depletion layer capacitances for both gate junctions which vary as the -1/2 power of junction voltage and are defined by the parameters CGS, CGD, and PB.

Note that in Spice3f and later, a fitting parameter B has been added.

name	parameter	units	default	example	area
VTO	threshold voltage (VT0)	V	-2.0	-2.0	
BETA	transconductance parameter (Beta)	A/V^2	1.0e-4	1.0e-3	*
LAMBDA	channel-length modulation parameter ( )	1/V	0	1.0e-4	
RD	drain ohmic resistance	Ohm	0	100	*
RS	source ohmic resistance	Ohm	0	100	*
CGS	zero-bias G-S junction capacitance (Cgs)	F	0	5pF	*
CGD	zero-bias G-D junction capacitance (Cgs)	F	0	1pF	*

PB	gate junction potential	V	1	0.6	
IS	gate junction saturation current (IS)	A	1.0e-14	1.0E-14	*
B	doping tail parameter	-	1		
KF	flicker noise coefficient	-	0		
AF	flicker noise exponent	-	1		
FC	coefficient for forward-bias	-	0.5		
TNOM	parameter measurement temperature	ºC		50	

General form:

DXXXXXXX N+ N- MNAME <AREA>> <OFF> <IC=VD> <TEMP>

Examples:

DBRIDGE 2 10 DIODE1 DCLMP 3 7 DMOD 3.0 IC=0.2

N+ and N- are the positive and negative nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) starting condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification using IC=VD is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

General form:

JXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS, VGS> <TEMP>

Examples:

J1 7 2 3 JM1 OFF

ND, NG, and NS are the drain, gate, and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification, using IC=VDS, VGS is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line for a better way to set initial conditions. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

The MESFET model is derived from the GaAs FET model of Statz et al. as described in [11]. The dc characteristics are defined by the parameters VTO, B, and BETA, which determine the variation of drain current with gate voltage, ALPHA, which determines saturation voltage, and LAMBDA, which determines the output conductance. The formula are given by:

Two ohmic resistances, RD and RS, are included. Charge storage is modeled by total gate charge as a function of gate-drain and gate-source voltages and is defined by the parameters CGS, CGD, and PB.

name	parameter	units	default	example	area
VTO	pinch-off voltage	V	-2.0	-2.0	
BETA	transconductance parameter	A/V^2	1.0e-4	1.0e-3	*
B	doping tail extending parameter	1/V	0.3	0.3	*
ALPHA	saturation voltage parameter	1/V	2	2	*
LAMBDA	channel-length modulation parameter	1/V	0	1.0e-4	
RD	drain ohmic resistance	Ohms	0	100	*
RS	source ohmic resistance	Ohms	0	100	*
CGS	zero-bias G-S junction capacitance	F	0	5pF	*
CGD	zero-bias G-D junction capacitance	F	0	1pF	*
PB	gate junction potential	V	1	0.6	
KF	flicker noise coefficient	-	0		
AF	flicker noise exponent	-	1		
FC	coefficient for forward-bias depletion capacitance formula	-	0.5		

.

General form:

ZXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS, VGS>

Examples:

Z1 7 2 3 ZM1 OFF

SPICE provides four MOSFET device models, which differ in the formulation of the I-V characteristic. The variable LEVEL specifies the model to be used:

LEVEL=1 -> Shichman-Hodges

LEVEL=2 -> MOS2 (as described in [1])

LEVEL=3 -> MOS3, a semi-empirical model(see [1])

LEVEL=4 -> BSIM (as described in [3])

LEVEL=5 -> new BSIM (BSIM2; as described in [5])

LEVEL=6 -> MOS6 (as described in [2])The dc characteristics of the level 1

through level 3 MOSFETs are defined by the device parameters VTO, KP, LAMBDA, PHI and GAMMA. These parameters are computed by SPICE if process parameters (NSUB, TOX, ...) are given, but user-specified values always override. VTO is positive (negative) for enhancement mode and negative (positive) for depletion mode N-channel (P-channel) devices. Charge storage is modeled by three constant capacitors, CGSO, CGDO, and CGBO which represent overlap capacitances, by the nonlinear thin-oxide capacitance which is distributed among the gate, source, drain, and bulk regions, and by the nonlinear depletion-layer capacitances for both substrate junctions divided into bottom and periphery, which vary as the MJ and MJSW power of junction voltage respectively, and are determined by the parameters CBD, CBS, CJ, CJSW, MJ, MJSW and PB. Charge storage effects are modeled by the piecewise linear voltages-dependent capacitance model proposed by Meyer. The thin-oxide charge-storage effects are treated slightly different for the LEVEL=1 model. These voltage-dependent capacitances are included only if TOX is specified in the input description and they are represented using Meyer's formulation.

There is some overlap among the parameters describing the junctions, e.g. the reverse current can be input either as IS (in A) or as JS (in A/m2). Whereas the first is an absolute value the second is multiplied by AD and AS to give the reverse current of the drain and source junctions respectively. This methodology has been chosen since there is no sense in relating always junction characteristics with AD and AS entered on the device line; the areas can be defaulted. The same idea applies also to the zero-bias junction capacitances CBD and CBS (in F) on one hand, and CJ (in F/m2) on the other. The parasitic drain and source series resistance can be expressed as either RD and RS (in ohms) or RSH (in ohms/sq.), the latter being multiplied by the number of squares NRD and NRS input on the device line.

A discontinuity in the MOS level 3 model with respect to the KAPPA parameter has been detected (see [10]). The supplied fix has been implemented in Spice3f2 and later. Since this fix may affect parameter fitting, the option "BADMOS3" may be set to use the old implementation (see the section on simulation variables and the ".OPTIONS" line).

SPICE level 1, 2, 3 and 6 parameters:

name	parameter	units	default	example
LEVEL	model	-	1	
VTO	zero-bias threshold voltage (VT0)	V	0	1
KP	transconductance parameter	A/V^2	2.0e-5	3.1e-5
GAMMA	bulk threshold parameter ( )	$V^{1/2}$	0	0.37
PHI	surface potential ( )	V	0.6	0.65
LAMBDA	channel-length modulation (MOS1 and MOS2 only) ( )	1/V	0	0.02
RD	drain ohmic resistance	Ohm	0	1
RS	source ohmic resistance	Ohm	0	1
CBD	zero-bias B-D junction capacitance	F	0	20fF
CBS	zero-bias B-S junction capacitance	F	0	20fF
IS	bulk junction saturation current (IS)	A	1.oe-14	1.0e-15
PB	bulk junction potential	V	0.8	0.87
CGSO	gate-source overlap capacitance per meter channel width	F/m	0	4.0e-11
CGDO	gate-drain overlap capacitance per meter channel width	F/m	0	4.0e-11
CGBO	gate-bulk overlap capacitance per meter channel length	F/m	0	2e-1-
RSH	drain and source diffusion sheet resistance	Ohm/square	0	10
CJ	zero-bias bulk junction bottom cap. per sq-meter of junction area	F/m^2	0	2.0e-4
MJ	bulk junction bottom grading coeff.	-	0.5	0.5
CJSW	zero-bias bulk junction sidewall cap. per meter of junction perimeter	F/m	10	1.0e-9

MJSWE	bulk junction sidewall grading coeff.	-	0.5 (level1) 0.33(level 2,3)	
FS	bulk junction saturation current per sq-meter of junction area	A/m^2		1.0e-8
TOX	oxide thickness	meter	1.0e-7	1.0e-7
NSUB	substrate doping	$1/cm^3$	0	4.0e15
NSS	surface state density	$1/cm^2$	0	1.0e10
NFS	fast surface state density	$1/cm^2$	0	1.0e10
TPG	type of gate material: +1 opp. to substrate -1 same as substrate 0 Al gate	-	1.0	
XJ	metallurgical junction depth	meter	0	1 micron
LD	lateral diffusion	meter	0	0.8 micron
UO	surface mobility	cm2/Vs	600	700
UCRIT	critical field for mobility degradation (MOS2 only	V/cm	600	1.0e-4
UEXP	critical field exponent in mobility degradation (MOS2 only)	-	1.0e04	0.1
UTRA	transverse field coeff. (mobility) (deleted for MOS2)	-	0	0.3
VMAX	maximum drift velocity of carriers	m/s	0	5.0e40.3
NEFF	total channel-charge (fixed and mobile) coefficient (MOS2 only	-	0	5.0
KF	flicker noise exponent	-	0	1.0e-26
AF	flicker noise exponent	-	1	1.2
FC	coefficient for forward-bias depletion capacitance formula	-	0.5	

DELTA	width effect on threshold voltage (MOS2 and MOS3)	-	0	1.0
THETA	mobility modulation (MOS3 only	1/V	0	0.1
ETA	static feedback (MOS3 only)	-	0	1.0
KAPPA	saturation field factor (MOS3 only)	-	0.2	0.5
TNOM	parameter measurement temperature	ºC	25	50

The level 4 and level 5 (BSIM1 and BSIM2) parameters are all values obtained from process characterization, and can be generated automatically. J. Pierret [4] describes a means of generating a 'process' file, and the program Proc2Mod provided with SPICE3 converts this file into a sequence of BSIM1 ".MODEL" lines suitable for inclusion in a SPICE input file. Parameters marked below with an * in the l/w column also have corresponding parameters with a length and width dependency. For example, VFB is the basic parameter with units of Volts, and LVFB and WVFB also exist and have units of Volt- meter The formula

is used to evaluate the parameter for the actual device specified with

and

Note that unlike the other models in SPICE, the BSIM model is designed for use with a process characterization system that provides all the parameters, thus there are no defaults for the parameters, and leaving one out is considered an error. For an example set of parameters and the format of a process file, see the SPICE2 implementation notes[3].

For more information on BSIM2, see reference [5].

SPICE BSIM (level 4) parameters:

name	parameter	units	1/w
VFB	flat-band voltage	V	*
PHI	surface inversion potential	V	*
K1	body effect coefficient	$V^{1/2}$	*
K2	drain/source depletion charge-sharing coefficient	-	*
ETA	zero-bias drain-induced barrier-lowering coefficient	-	*
MUZ	zero-bias mobility	$cm^2/V\text{-}s$	
DL	shortening of channel	micron	

DW	narrowing of channel	micron	
U0	zero-bias transverse-field mobility degradation coefficient	V^{-1}	*
U1	zero-bias velocity saturation coefficient	micron/V	*
X2MZ	sens. of mobility to substrate bias at vds=0	$cm^2/V\text{-}s$	*
X2E	sens. of drain-induced barrier lowering effect to substrate bias	V^{-1}	*
X3E	sens. of drain-induced barrier lowering effect to drain bias at Vds=Vdd	V^{-1}	*
X2U0	sens. of transverse field mobility degradation effect to substrate bias	V^{-2}	*
X2U1	sens. of velocity saturation effect to substrate bias	micron/V 2	*
MUS	mobility at zero substrate bias and at Vds=Vdd	$cm^2/V\text{-}s$	
X2MS	sens. of mobility to substrate bias at Vds=Vdd	$cm^2/V\text{-}s$	*
X3MS	sens. of mobility to drain bias at Vds=Vdd	$cm^2/V\text{-}s$	*
X3U1	sens. of velocity saturation effect on drain bias at Vds=Vdd	$cm^2/V\text{-}s$	*
TOX	TOX gate oxide thickness	micron	
TEMP	temperature at which parameters were measured	ºC	
VDD	measurement bias range	V	
CGD0	gate-drain overlap capacitance per meter channel width	F/m	
CGSO	gate-source overlap capacitance per meter channel width	F/m	
CGBO	gate-bulk overlap capacitance per meter channel length	F/m	
XPART	gate-oxide capacitance-charge model flag	-	*
NO	zero-bias subthreshold slope coefficient	-	*
NB	sens. of subthreshold slope to substrate bias	-	*

ND	sens. of subthreshold slope to drain bias	-	
RSH	drain and source diffusion sheet resistance	Ohm/square	
JS	source drain junction current density	A/m^2	
PB	built in potential of source drain junction	V	
MJ	Grading coefficient of source drain junction	-	
PBSW	built in potential of source, drain junction sidewall	V	
MJSW	grading coefficient of source drain junction sidewall	-	
CJ	Source drain junction capacitance per unit area	F/m^2	
CJSW	source drain junction sidewall capacitance per unit length	F/m	
WDF	source drain junction default width	m	
DELL	Source drain junction length reduction	m	

XPART = 0 selects a 40/60 drain/source charge partition in saturation, while XPART=1 selects a 0/100 drain/source charge partition.

ND, NG, and NS are the drain, gate, and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification, using IC=VDS, VGS is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line for a better way to set initial conditions.

General form:

MXXXXXXX ND NG NS NB MNAME <L=VAL> <W=VAL> <AD=VAL> <AS=VAL>

+ <PD=VAL> <PS=VAL> <NRD=VAL> <NRS=VAL> <OFF>

+ <IC=VDS, VGS, VBS> <TEMP=T>

Examples:

M1 24 2 0 20 TYPE1 M31 2 17 6 10 MODM L=5U W=2U M1 2 9 3 0 MOD1 L=10U W=5U AD=100P AS=100P PD=40U PS=40U

ND, NG, NS, and NB are the drain, gate, source, and bulk (substrate) nodes, respectively. MNAME is the model name. L and W are the channel length and width, in meters. AD and AS are the areas of the drain and source diffusions, in m2. Note that the suffix U specifies microns (1e-6 m) and P sq-microns (1e-12 m2). If any of L,

W, AD, or AS are not specified, default values are used. The use of defaults simplifies input file preparation, as well as the editing required if device geometries are to be changed. PD and PS are the perimeters of the drain and source junctions, in meters. NRD and NRS designate the equivalent number of squares of the drain and source diffusions; these values multiply the sheet resistance RSH specified on the .MODEL control line for an accurate representation of the parasitic series drain and source resistance of each transistor. PD and PS default to 0.0 while NRD and NRS to 1.0. OFF indicates an (optional) initial condition on the device for dc analysis. The (optional) initial condition specification using IC=VDS, VGS, VBS is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line for a better and more convenient way to specify transient initial conditions. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line. The temperature specification is ONLY valid for level 1, 2, 3, and 6 MOSFETs, not for level 4 or 5 (BSIM) devices.

There are several different transmission line models:

Lossless Transmission Lines

Lossy Transmission Line Model (LTRA)

Lossy Transmission Lines

Uniform Distributed RC Lines (Lossy)

Uniform Distributed RC Model (URC)

General form:

TXXXXXXX N1 N2 N3 N4 Z0=VALUE <TD=VALUE> <F=FREQ

<NL=NRMLEN>>

+ <IC;=V1, I1, V2, I2>

Examples:

T1 1 0 2 0 Z0=50 TD=10NS

N1 and N2 are the nodes at port 1; N3 and N4 are the nodes at port 2. Z0 is the characteristic impedance. The length of the line may be expressed in either of two forms. The transmission delay, TD, may be specified directly (as TD=10ns, for example). Alternatively, a frequency F may be given, together with NL, the normalized electrical length of the transmission line with respect to the wavelength in the line at the frequency F. If a frequency is specified but NL is omitted, 0.25 is assumed (that is, the frequency is assumed to be the quarter-wave frequency). Note that although both forms for expressing the line length are indicated as optional, one of the two must be specified.

Note that this element models only one propagating mode. If all four nodes are distinct in the actual circuit, then two modes may be excited. To simulate such a situation, two transmission-line elements are required. (see the example in *(AA for further clarification.)

The (optional) initial condition specification consists of the voltage and current at each of the transmission line ports. Note that the initial conditions (if any) apply 'only' if the UIC option is specified on the .TRAN control line.

Note that a lossy transmission line (see below) with zero loss may be more accurate than than the lossless transmission line due to implementation details.

The uniform RLC/RC/LC/RG transmission line model (referred to as the LTRA model henceforth) models a uniform constant-parameter distributed transmission line. The RC and LC cases may also be modeled using the URC and TRA models; however, the newer LTRA model is usually faster and more accurate than the others. The operation of the LTRA model is based on the convolution of the transmission line's impulse responses with its inputs (see [8]).

The LTRA model takes a number of parameters, some of which must be given and some of which are optional.

name	parameter	unirts/type	default	example
R	resistance/length	ohm/unit	0	0.2
L	nductance/length	henrys/unit	0	9.13e-9
G	conductance/length	mhos/unit	0	0
C	capacitance/length	farads/unit	0	3.65e-12
LEN	lenght of line		no default	1
REL	breakpoint control	arbitary unit	1	0.5
ABS	don't limit timestep to less than line delay		1	5
NOSTEPLIMIT		flag	not set	set
NOCONTROL	use lineiaairinterpolation	flag	not set	set
LININTERP	use lineair when quadratic seems bad	flag	not set	set

MIXEDINTERP	special reltol for history compaction	flag	not set	set
COMPACTREL	special abstol for history compaction	flag	RELTOL	1.0e-3
COMPACTABS	use Newton-Raphson method for timestep control		ABSTOL	1.0e-9
TRUNCNR	use Newton-Raphson method for timestep control	flag	not set	set
TRUNCDONTCUT	don't limit timestep to keep impulse-response errors low	flag	not set	set

The following types of lines have been implemented so far: RLC (uniform transmission line with series loss only), RC (uniform RC line), LC (lossless transmission line), and RG (distributed series resistance and parallel conductance only). Any other combination will yield erroneous results and should not be tried. The length LEN of the line must be specified.

NOSTEPLIMIT is a flag that will remove the default restriction of limiting time-steps to less than the line delay in the RLC case. NOCONTROL is a flag that prevents the default limiting of the time-step based on convolution error criteria in the RLC and RC cases. This speeds up simulation but may in some cases reduce the accuracy of results. LININTERP is a flag that, when specified, will use linear interpolation instead of the default quadratic interpolation for calculating delayed signals. MIXEDINTERP is a flag that, when specified, uses a metric for judging whether quadratic interpolation is not applicable and if so uses linear interpolation; otherwise it uses the default quadratic interpolation. TRUNCDONTCUT is a flag that removes the default cutting of the time-step to limit errors in the actual calculation of impulse-response related quantities. COMPACTREL and COMPACTABS are quantities that control the compaction of the past history of values stored for convolution. Larger values of these lower accuracy but usually increase simulation speed. These are to be used with the TRYTOCOMPACT option, described in the .OPTIONS section. TRUNCNR is a flag that turns on the use of Newton-Raphson iterations to determine an appropriate timestep in the timestep control routines. The default is a trial and error procedure by cutting the previous timestep in half. REL and ABS are quantities that control the setting of breakpoints.

The option most worth experimenting with for increasing the speed of simulation is REL. The default value of 1 is usually safe from the point of view of accuracy but occasionally increases computation time. A value greater than 2 eliminates all breakpoints and may be worth trying depending on the nature of the rest of the circuit, keeping in mind that it might not be safe from the viewpoint of accuracy. Breakpoints may usually be entirely eliminated if it is expected the circuit will not display sharp discontinuities. Values between 0 and 1 are usually not required but may be used for setting many breakpoints.

COMPACTREL may also be experimented with when the option TRYTOCOMPACT is specified in a .OPTIONS card. The legal range is between 0 and 1. Larger values usually decrease the accuracy of the simulation but in some cases improve speed. If TRYTOCOMPACT is not specified on a .OPTIONS card, history compaction is not attempted and accuracy is high. NOCONTROL, TRUNCDONTCUT and NOSTEPLIMIT also tend to increase speed at the expense of accuracy.

General form:

OXXXXXXX N1 N2 N3 N4 MNAME

Examples:

O23 1 0 2 0 LOSSYMOD OCONNECT 10 5 20 5 INTERCONNECT

This is a two-port convolution model for single-conductor lossy transmission lines. N1 and N2 are the nodes at port 1; N3 and N4 are the nodes at port 2. Note that a lossy transmission line with zero loss may be more accurate than than the lossless transmission line due to implementation details.

General form:

UXXXXXXX N1 N2 N3 MNAME L=LEN <N=LUMPS>

Examples:

U1 1 2 0 URCMOD L=50U URC2 1 12 2 UMODL I=1MIL N=6

N1 and N2 are the two element nodes the RC line connects, while N3 is the node to which the capacitances are connected. MNAME is the model name, LEN is the length of the RC line in meters. LUMPS, if specified, is the number of lumped segments to use in modeling the RC line (see the model description for the action taken if this parameter is omitted).

The URC model is derived from a model proposed by L. Gertzberrg in 1974. The model is accomplished by a subcircuit type expansion of the URC line into a network of lumped RC segments with internally generated nodes. The RC segments are in a geometric progression, increasing toward the middle of the URC line, with K as a proportionality constant. The number of lumped segments used, if not specified for the URC line device, is determined by the following formula:

The URC line is made up strictly of resistor and capacitor segments unless the ISPERL parameter is given a non-zero value, in which case the capacitors are replaced with reverse biased diodes with a zero-bias junction capacitance equivalent to the capacitance replaced, and with a saturation current of ISPERL amps per meter of transmission line and an optional series resistance equivalent to RSPERL ohms per meter.

name	parameter	units	default	example	area

K	Propagation Constant	-	2	1.2	-
FMAX	Maximum Frequency of interest	Hz	1.0G	6.5Meg	-
RPERL	Resistance per unit length	ohm/m	o1000	10	-
CPERL	Capacitance per unit length	F/m	1.0e015	1pF	-
ISPERL	Saturation Current per unit length	A/m	0	-	-
RSPERL	Diode Resistance per unit length	ohm/m	0	-	-

Spice has the following voltage and current sources:

- Independent Sources
- Linear Dependent Sources

Spice has the following independant sources:

- Exponential
- Independent Sources
- Piece-Wise Linear
- Pulse
- Single Frequency FM
- Sinusoidal

General Form:

EXP(V1 V2 TD1 TAU1 TD2 TAU2)

Examples:

VIN 3 0 EXP(-4 -1 2NS 30NS 60NS 40NS)

parameter	default value	units
V1 (initial value)	-	Volts or Amps
V2 (pulsed value)	-	Volts or Amps
TD1 (rise delay time)	0	seconds

TAU1 (rise time constant)	TSTEP	seconds
TD2 (fall delay time)	TD1+TSTEP	seconds
TAU2 (fall time)	TSTEP	seconds

The shape of the waveform is described by the following table:

time	value
0 to TD1	V1
TD1 to TD2	
TD2 to TSTOP	

General form:

VXXXXXXX N+ N- <DC<> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>

+ <DISTOF1 <F1MAG <F1PHASE>>> <DISTOF2 <F2MAG <F2PHASE>>>

IYYYYYYY N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>

+ <DISTOF1 <F1MAG <F1PHASE>>> <DISTOF2 <F2MAG <F2PHASE>>>

Examples:

VCC 10 0 DC 6 VIN 13 2 0.001 AC 1 SIN(0 1 1MEG) ISRC 23 21 AC 0.333 45.0 SFFM(0 1 10K 5 1K) VMEAS 12 9 VCARRIER 1 0 DISTOF1 0.1 -90.0 VMODULATOR 2 0 DISTOF2 0.01 IIN1 1 5 AC 1 DISTOF1 DISTOF2 0.001

N+ and N- are the positive and negative nodes, respectively. Note that voltage sources need not be grounded. Positive current is assumed to flow from the positive node, through the source, to the negative node. A current source of positive value forces current to flow out of the N+ node, through the source, and into the N- node. Voltage sources, in addition to being used for circuit excitation, are the 'ammeters' for SPICE, that is, zero valued voltage sources may be inserted into the circuit for the purpose of measuring current. They of course have no effect on circuit operation since they represent short-circuits.

DC/TRAN is the dc and transient analysis value of the source. If the source value is zero both for dc and transient analyses, this value may be omitted. If the source value is time-invariant (e.g., a power supply), then the value may optionally be preceded by the letters DC.

ACMAG is the ac magnitude and ACPHASE is the ac phase. The source is set to this value in the ac analysis. If ACMAG is omitted following the keyword AC, a value

of unity is assumed. If ACPHASE is omitted, a value of zero is assumed. If the source is not an ac small-signal input, the keyword AC and the ac values are omitted.

DISTOF1 and DISTOF2 are the keywords that specify that the independent source has distortion inputs at the frequencies F1 and F2 respectively (see the description of the .DISTO control line). The keywords may be followed by an optional magnitude and phase. The default values of the magnitude and phase are 1.0 and 0.0 respectively.

Any independent source can be assigned a time-dependent value for transient analysis. If a source is assigned a time-dependent value, the time-zero value is used for dc analysis. There are five independent source functions: pulse, exponential, sinusoidal, piece-wise linear, and single-frequency FM. If parameters other than source values are omitted or set to zero, the default values shown are assumed. (TSTEP is the printing increment and TSTOP is the final time (see the .TRAN control line for explanation)).

General Form:

PWL(T1 V1 <T2; V2 T3 V3 T4 V4 ...>)

Examples:

VCLOCK 7 5 PWL(0 -7 10NS -7 11NS -3 17NS -3 18NS -7 50NS -7)

Each pair of values (Ti, Vi) specifies that the value of the source is Vi (in Volts or Amps) at time=Ti. The value of the source at intermediate values of time is determined by using linear interpolation on the input values.

General form:

PULSE(V1 V2 TD TR TF PW PER)

Examples:

VIN 3 0 PULSE(-1 1 2NS 2NS 2NS 50NS 100NS)

parameter	default value	units
V1 (initial value)	-	Volts orAmps
V2 (pulsed value)	-	Volts orAmps
V2 (pulsed value)	0	seconds
TR (rise time)	TSTEP	seconds
TF (fall time)	TSTEP	seconds
PW (pulse width)	TSTOP	seconds

PER(period)	TSTOP	seconds

A single pulse so specified is described by the following table:

time	value
0	V1
TD	V1
TD+TR	V2
TD+TR+PW	V2
TD+TR+PW V2	V1
TSTOP	V1

Intermediate points are determined by linear interpolation..

General Form:

SFFM(VO VA FC MDI FS)

Examples:

V1 12 0 SFFM(0 1M 20K 5 1K)

parameter	default value	units
VO (offset)	-	Volts or Amps
VA (amplitude)	-	Volts or Amps
FC (carrier frequency)	1/TSTOP	Hz
MDI (modulation index)	-	-
FS (signal frequency)	1/TSTOP	Hz

The shape of the waveform is described by the following equation:

General form:

SIN(VO VA FREQ TD THETA)

Examples:

VIN 3 0 SIN(0 1 100MEG 1NS 1E10)

parameter	default value	units
VO (offset)	-	Volts or Amps
VA (amplitude)	-	Volts or Amps
FREQ (frequency)	1/TSTOP	Hz
TD (delay)	0	seconds
THETA (damping factor)	0	1/seconds

The shape of the waveform is described by the following table:

time	value
0 to TD	V0
TD to TSTOP	

Spice has the following dependant sources:

- Current-Controlled Current Sources
- Current-Controlled Voltage Sources
- Voltage-Controlled Current Sources
- Voltage-Controlled Voltage Sources

SPICE allows circuits to contain linear dependent sources characterized by any of the four equations

$$i = g\,v \qquad v = e\,v \qquad i = f\,i \qquad = h\,i$$

where g, e, f, and h are constants representing transconductance, voltage gain, current gain, and transresistance, respectively.

General form:

FXXXXXXX N+ N- VNAM VALUE

Examples:

F1 13 5 VSENS 5

N+ and N- are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the current gain.

General form:

HXXXXXXX N+ N- VNAM VALUE

Examples:

HX 5 17 VZ 0.5K

N+ and N- are the positive and negative nodes, respectively. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the transresistance (in ohms).

General form:

GXXXXXXX N+ N- NC+ NC- VALUE

Examples:

G1 2 0 5 0 0.1MMHO

N+ and N- are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. NC+ and NC- are the positive and negative controlling nodes, respectively. VALUE is the transconductance (in mhos).

General form:

EXXXXXXX N+ N- NC+ NC- VALUE

Examples:

E1 2 3 14 1 2.0

N+ is the positive node, and N- is the negative node. NC+ and NC- are the positive and negative controlling nodes, respectively. VALUE is the voltage gain..

General form:

BXXXXXXX N+ N- <I=EXPR> <V=EXPR>

Examples:

B1 0 1 I=cos(v(1))+sin(v(2)) B1 0 1 V=ln(cos(log(v(1,2)^2)))-v(3)^4+v(2)^v(1) B1 3 4 I=17 B1 3 4 V=exp(pi^i(vdd))

N+ is the positive node, and N- is the negative node. The values of the V and I parameters determine the voltages and currents across and through the device, respectively. If I is given then the device is a current source, and if V is given the device is a voltage source. One and only one of these parameters must be given.

The small-signal AC behavior of the nonlinear source is a linear dependent source (or sources) with a proportionality constant equal to the derivative (or derivatives) of the source at the DC operating point.

The expressions given for V and I may be any function of voltages and currents through voltage sources in the system. The following functions of real variables are defined:

abs	asinh	cosh	sin
acos	atan	exp	sinh
acosh	atanh	ln	sqrt
asin	cos	log	tan

The function "u" is the unit step function, with a value of one for arguments greater than one and a value of zero for arguments less than zero. The function "uramp" is the integral of the unit step: for an input x, the value is zero if x is less than zero, or if x is greater than zero the value is x. These two functions are useful in sythesizing piece-wise non-linear functions, though convergence may be adversely affected.

The following standard operators are defined:

+ - * / ^ unary -

If the argument of log, ln, or sqrt becomes less than zero, the absolute value of the argument is used. If a divisor becomes zero or the argument of log or ln becomes zero, an error will result. Other problems may occur when the argument for a function in a partial derivative enters a region where that function is undefined.

To get time into the expression you can integrate the current from a constant current source with a capacitor and use the resulting voltage (don't forget to set the initial voltage across the capacitor). Non-linear resistors, capacitors, and inductors may be synthesized with the nonlinear dependent source. Non-linear resistors are obvious. Non-linear capacitors and inductors are implemented with their linear counterparts by a change of variables implemented with the nonlinear dependent source. The following subcircuit will implement a nonlinear capacitor:

```
.Subckt nlcap pos neg

* Bx: calculate f(input voltage)

Bx 1 0 v = f(v(pos,neg))

* Cx: linear capacitance
```

```
Cx 2 0 1

* Vx: Ammeter to measure current into the capacitor

Vx 2 1 DC 0Volts

* Drive the current through Cx back into the circuit

Fx pos neg Vx 1

.ends
```

Non-linear inductors are similar.

General form:

CXXXXXXX N+ N- VALUE <IC=INCOND>

Examples:

CBYP 13 0 1UF COSC 17 23 10U IC=3V

N+ and N- are the positive and negative element nodes, respectively. VALUE is the capacitance in Farads.

The (optional) initial condition is the initial (time-zero) value of capacitor voltage (in Volts). Note that the initial conditions (if any) apply 'only' if the UIC option is specified on the .TRAN control line.

General form:

KXXXXXXX LYYYYYYY LZZZZZZZ VALUE

Examples:

K43 LAA LBB 0.999 KXFRMR L1 L2 0.87

LYYYYYYY and LZZZZZZZ are the names of the two coupled inductors, and VALUE is the coefficient of coupling, K, which must be greater than 0 and less than or equal to 1. Using the 'dot' convention, place a 'dot' on the first node of each inductor.

General form:

LYYYYYYY N+ N- VALUE <IC=INCOND>

Examples:

LLINK 42 69 1UH LSHUNT 23 51 10U IC=15.7MA

N+ and N- are the positive and negative element nodes, respectively. VALUE is the inductance in Henries.

The (optional) initial condition is the initial (time-zero) value of inductor current (in Amps) that flows from N+, through the inductor, to N-. Note that the initial conditions (if any) apply only if the UIC option is specified on the .TRAN analysis line.

General form:

RXXXXXXX N1 N2 VALUE

Examples:

R1 1 2 100 RC1 12 17 1K

N1 and N2 are the two element nodes. VALUE is the resistance (in ohms) and may be positive or negative but not zero.

The capacitor model contains process information that may be used to compute the capacitance from strictly geometric information.

name	parameter	units	default	example
CJ	junction bottom capacitance	F/meter/meter	-	5e-5
CJSW	junction sidewall capacitance	F/meter	-	2e-11
DEFW	default device width	meters	1e-6	2e-6
NARROW	narrowing due to side etching	meters	0	1e-7

The capacitor has a capacitance computed as

General form:

CXXXXXXX N1 N2 <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <IC=VAL>

Examples:

CLOAD 2 10 10P CMOD 3 7 CMODEL L=10u W=1u

This is the more general form of the Capacitor presented in section 6.2, and allows for the calculation of the actual capacitance value from strictly geometric information and the specifications of the process. If VALUE is specified, it defines the capacitance. If MNAME is specified, then the capacitance is calculated from the process information in the model MNAME and the given LENGTH and WIDTH. If VALUE is not specified, then MNAME and LENGTH must be specified. If WIDTH is not specified, then it is taken from the default width given in the model. Either VALUE or MNAME, LENGTH, and WIDTH may be specified, but not both sets.

The resistor model consists of process-related device data that allow the resistance to be calculated from geometric information and to be corrected for temperature. The parameters available are:

name	parameter	units	default	example
TC1	first order temperature coeff.	Ohm/ºC	0	-
TC2	second order temperature coeff.	Ohm/ºC/ºC	0	-
RSH	sheet resistance	Ohm/q	-	50
DEFW	default width	meters	1e-6	2e-6
NARROW	narrowing due to side etching	meters	0	1e-7
TNOM	parameter measurement temperature	ºC	27	50

The sheet resistance is used with the narrowing parameter and L and W from the resistor device to determine the nominal resistance by the formula

DEFW is used to supply a default value for W if one is not specified for the device. If either RSH or L is not specified, then the standard default resistance value of 1k is used. TNOM is used to override the circuit-wide value given on the .OPTIONS control line where the parameters of this model have been measured at a different temperature. After the nominal resistance is calculated, it is adjusted for temperature by the formula:

General form:

RXXXXXXX N1 N2 <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <TEMP=T>

Examples:

RLOAD 2 10 10K RMOD 3 7 RMODEL L=10u W=1u

This is the more general form of the resistor presented in section 6.1, and allows the modeling of temperature effects and for the calculation of the actual resistance value from strictly geometric information and the specifications of the process. If VALUE is specified, it overrides the geometric information and defines the resistance. If MNAME is specified, then the resistance may be calculated from the process information in the model MNAME and the given LENGTH and WIDTH. If VALUE is not specified, then MNAME and LENGTH must be specified. If WIDTH is not specified, then it is taken from the default width given in the model. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

The switch model allows an almost ideal switch to be described in SPICE. The switch is not quite ideal, in that the resistance can not change from 0 to infinity, but must always have a finite positive value. By proper selection of the on and off resistances, they can be effectively zero and infinity in comparison to other circuit elements. The parameters available are:

name	parameter	units	default	switch
VT	threshold voltage	Volts	0	S
IT	threshold current	Amps	0	W
VH	hysteresis voltage	Volts	0	S
IH	hysteresis current	Amps	0	W
RON	on resistance	Ohms	1	both
ROFF	off resistance	Ohms	1/GMIN*	both

*(See the .OPTIONS control line for a description of GMIN, its default value results in an off-resistance of 1.0e+12 ohms.)

The use of an ideal element that is highly nonlinear such as a switch can cause large discontinuities to occur in the circuit node voltages. A rapid change such as that associated with a switch changing state can cause numerical roundoff or tolerance problems leading to erroneous results or timestep difficulties. The user of switches can improve the situation by taking the following steps:

First, it is wise to set ideal switch impedances just high or low enough to be negligible with respect to other circuit elements. Using switch impedances that are close to "ideal" in all cases aggravates the problem of discontinuities mentioned above. Of course, when modeling real devices such as MOSFETS, the on resistance should be adjusted to a realistic level depending on the size of the device being modeled.

If a wide range of ON to OFF resistance must be used in the switches (ROFF/RON >1e;+12), then the tolerance on errors allowed during transient analysis should be decreased by using the .OPTIONS control line and specifying TRTOL to be less than the default value of 7.0. When switches are placed around capacitors, then the option CHGTOL should also be reduced. Suggested values for these two options are 1.0 and 1e-16 respectively. These changes inform SPICE3 to be more careful around the switch points so that no errors are made due to the rapid change in the circuit.

General form:

SXXXXXXX N+ N- NC+ NC- MODEL <ON><OFF>

WYYYYYYY N+ N- VNAM MODEL <ON><OFF>

Examples:

s1 1 2 3 4 switch1 ONs2 5 6 3 0 sm2 off Switch1 1 2 10 0 smodel1 w1 1 2 vclock switchmod1 W2 3 0 vramp sm1 ON wreset 5 6 vclck lossyswitch OFF

Nodes 1 and 2 are the nodes between which the switch terminals are connected. The model name is mandatory while the initial conditions are optional. For the voltage controlled switch, nodes 3 and 4 are the positive and negative controlling nodes respectively. For the current controlled switch, the controlling current is that through the specified voltage source. The direction of positive controlling current flow is from the positive node, through the source, to the negative node.

1.2.11.7.2.2 Combining Files. .include lines

General form:

.INCLUDE filename

Examples:

.INCLUDE /users/spice Simulation Program for Integrated Circuit Emulation./common/wattmeter.cir

Frequently, portions of circuit descriptions will be reused in several input files, particularly with common models and sub-circuits. In any spice input file, the ".include" line may be used to copy some other file as if that second file appeared in place of the ".include" line in the original file. There is no restriction on the file name imposed by spice beyond those imposed by the local operating system.

1.2.11.7.2.3 Device Models

General form:

.MODEL MNAME TYPE(PNAME1=PVAL1 PNAME2=PVAL2 ...)

Examples:

.MODEL MOD1 NPN (BF=50 IS=1E-13 VBF=50)

Most simple circuit elements typically require only a few parameter values. However, some devices (semiconductor devices in particular) that are included in SPICE Simulation Program for Integrated Circuit Emulation. require many parameter values. Often, many devices in a circuit are defined by the same set of device model parameters. For these reasons, a set of device model parameters is defined on a separate .MODEL line and assigned a unique model name. The device element lines in SPICE then refer to the model name.

For these more complex device types, each device element line contains the device name, the nodes to which the device is connected, and the device model name. In addition, other optional parameters may be specified for some devices: geometric factors and an initial condition (see the following section on Transistors and Diodes for more details).

MNAME in the above is the model name, and type is one of the following fifteen types:

R Semiconductor resistor model

C Semiconductor capacitor model

SW Voltage controlled switch

CSW Current controlled switch

URC Uniform distributed RC model

LTRA Lossy transmission line model

D Diode model

NPN NPN BJT model

PNP PNP BJT model

NJF N-channel JFET model

PJF P-channel JFET model

NMOS N-channel MOSFET model

PMOS P-channel MOSFET model

NMF N-channel MESFET model

PMF P-channel MESFET model

Parameter values are defined by appending the parameter name followed by an equal sign and the parameter value. Model parameters that are not given a value are assigned the default values given below for each model type. Models, model parameters, and default values are listed in the next section along with the description of device element lines.

1.2.11.7.2.4 GENERAL STRUCTURE AND CONVENTIONS

The circuit to be analyzed is described to SPICE by a set of element lines, which define the circuit topology and element values, and a set of control lines, which define the model parameters and the run controls. The first line in the input file must be the title, and the last line must be ".END". The order of the remaining lines is arbitrary (except, of course, that continuation lines must immediately follow the line being continued).

Each element in the circuit is specified by an element line that contains the element name, the circuit nodes to which the element is connected, and the values of the parameters that determine the electrical characteristics of the element. The first letter of the element name specifies the element type. The format for the SPICE element types is given in what follows. The strings XXXXXXX, YYYYYYY, and ZZZZZZZ denote arbitrary alphanumeric strings.

For example, a resistor name must begin with the letter R and can contain one or more characters. Hence, R, R1, RSE, ROUT, and R3AC2ZY are valid resistor names. Details of each type of device are supplied in a following section.

Fields on a line are separated by one or more blanks, a comma, an equal ('=') sign, or a left or right parenthesis; extra spaces are ignored. A line may be continued by entering a '+' (plus) in column 1 of the following line; SPICE continues reading beginning with column 2.

A name field must begin with a letter (A through Z) and cannot contain any delimiters.

A number field may be an integer field (12, -44), a floating point field (3.14159), either an integer or floating point number followed by an integer exponent (1e-14, 2.65e3), or either an integer or a floating point number followed by one of the following scale factors:

T = 1012 G = 109 Meg = 106 K = 103 mil = 25.4 10-6

m = 10-3 u = 10-6 n = 10-9 p = 10-12 f = 10-15

Letters immediately following a number that are not scale factors are ignored, and letters immediately following a scale factor are ignored. Hence, 10, 10V, 10Volts, and 10Hz all represent the same number, and M, MA, MSec, and MMhos all represent the same scale factor. Note that 1000, 1000.0, 1000Hz, 1e3, 1.0e3, 1KHz, and 1K all represent the same number.

Nodes names may be arbitrary character strings. The datum (ground) node must be named '0'. Note the difference in SPICE3 where the nodes are treated as character strings and not evaluated as numbers, thus '0' and '00' are distinct nodes in SPICE3 but not in SPICE2. The circuit cannot contain a loop of voltage sources and/or inductors and cannot contain a cut-set of current sources and/or capacitors. Each node in the circuit must have a dc path to ground. Every node must have at least two connections except for transmission line nodes (to permit unterminated transmission lines) and MOSFET substrate nodes (which have two internal connections anyway).

1.2.11.7.2.5 SubCircuits

A subcircuit that consists of SPICE Simulation Program for Integrated Circuit Emulation. elements can be defined and referenced in a fashion similar to device models. The subcircuit is defined in the input file by a grouping of element lines; the program then automatically inserts the group of elements wherever the subcircuit is referenced. There is no limit on the size or complexity of subcircuits, and subcircuits may contain other subcircuits. An example of subcircuit usage is given in *(AA.

.SUBCKT

General form:

.SUBCKT subnam N1 &ltN2; N3 ...>

Examples:

.SUBCKT OPAMP 1 2 3 4

A circuit definition is begun with a .SUBCKT line. SUBNAM is the subcircuit name, and N1, N2, ... are the external nodes, which cannot be zero. The group of element lines which immediately follow the .SUBCKT line define the subcircuit. The last line in a subcircuit definition is the .ENDS line (see below). Control lines may not appear within a subcircuit definition; however, subcircuit definitions may contain anything else, including other subcircuit definitions, device models, and subcircuit calls (see below). Note that any device models or subcircuit definitions included as part of a subcircuit definition are strictly local (i.e., such models and definitions are not known outside the subcircuit definition). Also, any element nodes not included on the .SUBCKT line are strictly local, with the exception of 0 (ground) which is always global.

.ENDS

General form:

.ENDS &ltSUBNAM;>

Examples:

.ENDS OPAMP

The "Ends" line must be the last one for any subcircuit definition. The subcircuit name, if included, indicates which subcircuit definition is being terminated; if omitted, all subcircuits being defined are terminated. The name is needed only when nested subcircuit definitions are being made.

Subcircuit Calls

General form:

XYYYYYYY N1 &ltN2; N3 ...> SUBNA

Examples:

X1 2 4 17 3 1 MULTI

Subcircuits are used in SPICE by specifying pseudo-elements beginning with the letter X, followed by the circuit nodes to be used in expanding the subcircuit.

1.2.11.7.2.6 Title Line, Comment Lines and .end line

Title Line

Examples:

POWER AMPLIFIER CIRCUIT

TEST OF CAM CELL

The title line must be the first in the input file. Its contents are printed verbatim as the heading for each section of output.

End line

Examples:

.END

The "End" line must always be the last in the input file. Note that the period is an integral part of the name.

Comments

General Form:

* &ltany; comment>

Examples:

* RF=1K Gain should be 100 * Check open-loop gain and phase margin

The asterisk in the first column indicates that this line is a comment line. Comment lines may be placed anywhere in the circuit description. Note that SPICE3 also considers any line with leading white space to be a comment.

1.2.11.7.3 Analysis and output Control

The following analyzes are available:

- DC or Small-Signal AC Sensitivity Analysis
- DC Transfer Function
- Distortion Analysis
- Fourier Analysis
- Initial Conditions
- Noise Analysis
- Operating Point Analysis
- Pole-Zero Analysis
- Small-Signal AC Analysis
- Transfer Function Analysis
- Transient Analysis

1.2.11.7.3.1 DC or Small-Signal AC Sensitivity Analysis

General form:

.SENS OUTVAR

.SENS OUTVAR AC DEC ND FSTART FSTOP

.SENS OUTVAR AC OCT NO FSTART FSTOP

.SENS OUTVAR AC LIN NP FSTART FSTOP

Examples:

.SENS V(1,OUT) .SENS V(OUT) AC DEC 10 100 100k .SENS I(VTEST)

The sensitivity of OUTVAR to all non-zero device parameters is calculated when the SENS analysis is specified. OUTVAR is a circuit variable (node voltage or voltage-source branch current). The first form calculates sensitivity of the DC operating-point value of OUTVAR. The second form calculates sensitivity of the AC values of OUTVAR. The parameters listed for AC sensitivity are the same as in an AC analysis (see ".AC" above). The output values are in dimensions of change in output per unit change of input (as opposed to percent change in output or per percent change of input).

1.2.11.7.3.2 DC Transfer Function

General form:

.DC SRCNAM VSTART VSTOP VINCR [SRC2 START2 STOP2 INCR2]

Examples:

.DC VIN 0.25 5.0 0.25 .DC VDS 0 10 .5 VGS 0 5 1 .DC VCE 0 10 .25 IB 0 10U 1U

The DC line defines the dc transfer curve source and sweep limits (again with capacitors open and inductors shorted). SRCNAM is the name of an independent voltage or current source. VSTART, VSTOP, and VINCR are the starting, final, and incrementing values respectively. The first example causes the value of the voltage source VIN to be swept from 0.25 Volts to 5.0 Volts in increments of 0.25 Volts. A second source (SRC2) may optionally be specified with associated sweep parameters. In this case, the first source is swept over its range for each value of the second source. This option can be useful for obtaining semiconductor device output characteristics. See the second example circuit description in Appendix A.

1.2.11.7.3.3 Distortion Analysis

General form:

.DISTO DEC ND FSTART FSTOP &ltF2OVERF1;>

.DISTO OCT NO FSTART FSTOP &ltF2OVERF1;>

.DISTO LIN NP FSTART FSTOP &ltF2OVERF1;>

Examples:

.DISTO DEC 10 1kHz 100Mhz .DISTO DEC 10 1kHz 100Mhz 0.9

The Disto line does a small-signal distortion analysis of the circuit. A multi-dimensional Volterra series analysis is done using multi-dimensional Taylor series to represent the nonlinearities at the operating point. Terms of up to third order are used in the series expansions.

If the optional parameter F2OVERF1 is not specified, .DISTO does a harmonic analysis - i.e., it analyses distortion in the circuit using only a single input frequency F1, which is swept as specified by arguments of the .DISTO command exactly as in the .AC command. Inputs at this frequency may be present at more than one input source, and their magnitudes and phases are specified by the arguments of the DISTOF1 keyword in the input file lines for the input sources (see the description for independent sources). (The arguments of the DISTOF2 keyword are not relevant in this case). The analysis produces information about the A.C. values of all node voltages and branch currents at the harmonic frequencies 2 F1 and 3 F1, vs. the input frequency F1 as it is swept. (A value of 1 (as a complex distortion output) signifies cos(2 (2 F1) t) at 2 F1 and cos (2 (3 F1) t) at 3 F1, using the convention that 1 at the input fundamental frequency is equivalent to cos(2 F1 t).) The distortion component desired (2 F1 or 3 F1) can be selected using commands in nutmeg, and then printed or plotted. (Normally, one is interested primarily in the magnitude of the harmonic components, so the magnitude of the AC distortion value is looked at). It should be noted that these are the A.C. values of the actual harmonic components, and are not equal to HD2 and HD3. To obtain HD2 and HD3, one must divide by the corresponding A.C. values at F1, obtained from an .AC line. This division can be done using nutmeg commands.

If the optional F2OVERF1 parameter is specified, it should be a real number between (and not equal to) 0.0 and 1.0; in this case, .DISTO does a spectral analysis. It considers the circuit with sinusoidal inputs at two different frequencies F1 and F2. F1 is swept according to the .DISTO control line options exactly as in the .AC control line. F2 is kept fixed at a single frequency as F1 sweeps - the value at which it is kept fixed is equal to F2OVERF1 times FSTART. Each independent source in the circuit may potentially have two (superimposed) sinusoidal inputs for distortion, at the frequencies F1 and F2. The magnitude and phase of the F1 component are specified by the arguments of the DISTOF1 keyword in the source's input line (see the description of independent sources); the magnitude and phase of the F2 component are specified by the arguments of the DISTOF2 keyword. The analysis produces plots of all node voltages/branch currents at the intermodulation product frequencies F1 + F2, F1 - F2, and (2 F1) - F2, vs the swept frequency F1. The IM product of interest may be selected using the setplot command, and displayed with the print and plot commands. It is to be noted as in the harmonic analysis case, the results are the actual AC voltages and currents at the intermodulation frequencies, and need to be normalized with respect to .AC values to obtain the IM parameters.

If the DISTOF1 or DISTOF2 keywords are missing from the description of an independent source, then that source is assumed to have no input at the corresponding frequency. The

default values of the magnitude and phase are 1.0 and 0.0 respectively. The phase should be specified in degrees.

It should be carefully noted that the number F2OVERF1 should ideally be an irrational number, and that since this is not possible in practice, efforts should be made to keep the denominator in its fractional representation as large as possible, certainly above 3, for accurate results (i.e., if F2OVERF1 is represented as a fraction A/B, where A and B are integers with no common factors, B should be as large as possible; note that A < B because F2OVERF1 is constrained to be < 1). To illustrate why, consider the cases where F2OVERF1 is 49/100 and 1/2. In a spectral analysis, the outputs produced are at F1 + F2, F1 - F2 and 2 F1 - F2. In the latter case, F1 - F2 = F2, so the result at the F1-F2 component is erroneous because there is the strong fundamental F2 component at the same frequency. Also, F1 + F2 = 2 F1 - F2 in the latter case, and each result is erroneous individually. This problem is not there in the case where F2OVERF1 = 49/100, because F1-F2 = 51/100 F1 < > 49/100 F1 = F2. In this case, there are two very closely spaced frequency components at F2 and F1 - F2. One of the advantages of the Volterra series technique is that it computes distortions at mix frequencies expressed symbolically (i.e. n F1 m F2), therefore one is able to obtain the strengths of distortion components accurately even if the separation between them is very small, as opposed to transient analysis for example. The disadvantage is of course that if two of the mix frequencies coincide, the results are not merged together and presented (though this could presumably be done as a postprocessing step). Currently, the interested user should keep track of the mix frequencies himself or herself and add the distortions at coinciding mix frequencies together should it be necessary.

1.2.11.7.3.4 Fourier Analysis

General form:

.FOUR FREQ OV1 <OV2 OV3 ...>

Examples:

.FOUR 100K V(5)

The Four (or Fourier) line controls whether SPICE performs a Fourier analysis as a part of the transient analysis. FREQ is the fundamental frequency, and OV1, ..., are the output variables for which the analysis is desired. The Fourier analysis is performed over the interval &ltTSTOP-period;, TSTOP>, where TSTOP is the final time specified for the transient analysis, and period is one period of the fundamental frequency. The dc component and the first nine harmonics are determined. For maximum accuracy, TMAX (see the .TRAN line) should be set to period/100.0 (or less for very high-Q circuits).

1.2.11.7.3.5 Initial Conditions

.NODESET: Specify Initial Node Voltage Guesses

General form:

.NODESET V(NODNUM)=VAL V(NODNUM)=VAL ...

Examples:

.NODESET V(12)=4.5 V(4)=2.23

The Nodeset line helps the program find the dc or initial transient solution by making a preliminary pass with the specified nodes held to the given voltages. The restriction is then released and the iteration continues to the true solution. The .NODESET line may be necessary for convergence on bistable or a-stable circuits. In general, this line should not be necessary.

.IC: Set Initial Conditions

General form: .IC V(NODNUM)=VAL V(NODNUM)=VAL ... Examples: .IC V(11)=5 V(4)=-5 V(2)=2.2

The IC line is for setting transient initial conditions. It has two different interpretations, depending on whether the UIC parameter is specified on the .TRAN control line. Also, one should not confuse this line with the .NODESET line. The .NODESET line is only to help dc convergence, and does not affect final bias solution (except for multi-stable circuits). The two interpretations of this line are as follows:

1. When the UIC parameter is specified on the .TRAN line, then the node voltages specified on the .IC control line are used to compute the capacitor, diode, BJT, JFET, and MOSFET initial conditions. This is equivalent to specifying the IC=... parameter on each device line, but is much more convenient. The IC=... parameter can still be specified and takes precedence over the .IC values. Since no dc bias (initial transient) solution is computed before the transient analysis, one should take care to specify all dc source voltages on the .IC control line if they are to be used to compute device initial conditions.

2. When the UIC parameter is not specified on the .TRAN control line, the dc bias (initial transient) solution is computed before the transient analysis. In this case, the node voltages specified on the .IC control line is forced to the desired initial values during the bias solution. During transient analysis, the constraint on these node voltages is removed. This is the preferred method since it allows SPICE to compute a consistent dc solution.

1.2.11.7.3.6 Noise Analysis

General form:

.NOISE V(OUTPUT <,REF>) SRC (DEC | LIN | OCT) PTS FSTART FSTOP <PTS_PER_SUMMARY>

Examples:

.NOISE V(5) VIN DEC 10 1kHZ 100Mhz .NOISE V(5,3) V1 OCT 8 1.0 1.0e6 1

The Noise line does a noise analysis of the circuit. OUTPUT is the node at which the total output noise is desired; if REF is specified, then the noise voltage V(OUTPUT) - V(REF) is

calculated. By default, REF is assumed to be ground. SRC is the name of an independent source to which input noise is referred. PTS, FSTART and FSTOP are .AC type parameters that specify the frequency range over which plots are desired. PTS_PER_SUMMARY is an optional integer; if specified, the noise contributions of each noise generator is produced every PTS_PER_SUMMARY frequency points.

The .NOISE control line produces two plots - one for the Noise Spectral Density curves and one for the total Integrated Noise over the specified frequency range. All noise voltages/currents are in squared units V2/Hz and A2/Hz for spectral density, V2 and A2 for integrated noise).

1.2.11.7.3.7 Operating Point Analysis

General form:

.OP

The inclusion of this line in an input file directs SPICE to determine the dc operating point of the circuit with inductors shorted and capacitors opened. Note: a DC analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an AC small-signal, Noise, and Pole-Zero analysis to determine the linearized, small-signal models for nonlinear devices (see the KEEPOPINFO variable above).

1.2.11.7.3.8 PLOT Lines

General form:

.PLOT PLTYPE OV1 <(PLO1, PHI1)> <OV2 <(PLO2, PHI2)> ... OV8>

Examples:

.PLOT DC V(4) V(5) V(1) .PLOT TRAN V(17, 5) (2, 5) I(VIN) V(17) (1, 9) .PLOT AC VM(5) VM(31, 24) VDB(5) VP(5) .PLOT DISTO HD2 HD3(R) SIM2 .PLOT TRAN V(5, 3) V(4) (0, 5) V(7) (0, 10)

The Plot line defines the contents of one plot of from one to eight output variables. PLTYPE is the type of analysis (DC, AC, TRAN, NOISE, or DISTO) for which the specified outputs are desired. The syntax for the OVI is identical to that for the .PRINT line and for the plot command in the interactive mode.

The overlap of two or more traces on any plot is indicated by the letter X.

When more than one output variable appears on the same plot, the first variable specified is printed as well as plotted. If a printout of all variables is desired, then a companion .PRINT line should be included.

There is no limit on the number of .PLOT lines specified for each type of analysis.

1.2.11.7.3.9 Pole-Zero Analysis

General form:

.PZ NODE1 NODE2 NODE3 NODE4 CUR POL

.PZ NODE1 NODE2 NODE3 NODE4 CUR ZER

.PZ NODE1 NODE2 NODE3 NODE4 CUR PZ

.PZ NODE1 NODE2 NODE3 NODE4 VOL POL

.PZ NODE1 NODE2 NODE3 NODE4 VOL ZER

.PZ NODE1 NODE2 NODE3 NODE4 VOL PZ

Examples:

.PZ 1 0 3 0 CUR POL .PZ 2 3 5 0 VOL ZER .PZ 4 1 4 1 CUR PZ

CUR stands for a transfer function of the type (output voltage)/(input current) while VOL stands for a transfer function of the type (output voltage)/(input voltage). POL stands for pole analysis only, ZER for zero analysis only and PZ for both. This feature is provided mainly because if there is a nonconvergence in finding poles or zeros, then, at least the other can be found. Finally, NODE1 and NODE2 are the two input nodes and NODE3 and NODE4 are the two output nodes. Thus, there is complete freedom regarding the output and input ports and the type of transfer function.

In interactive mode, the command syntax is the same except that the first field is PZ instead of .PZ. To print the results, one should use the command 'print all'.

1.2.11.7.3.10 PRINT Lines

General form:

.PRINT PRTYPE OV1 <OV2 ... OV8>

Examples:

.PRINT TRAN V(4) I(VIN) .PRINT DC V(2) I(VSRC) V(23, 17) .PRINT AC VM(4, 2) VR(7) VP(8, 3)

The Print line defines the contents of a tabular listing of one to eight output variables. PRTYPE is the type of the analysis (DC, AC, TRAN, NOISE, or DISTO) for which the specified outputs are desired. The form for voltage or current output variables is the same as given in the previous section for the print command; Spice2 restricts the output variable to the following forms (though this restriction is not enforced by Spice3):

V(N1<,N2>)

specifies the voltage difference between nodes N1 and N2. If N2 (and the preceding comma) is omitted, ground (0) is assumed. See the print command in the previous section for more details. For compatibility with spice2, the following five additional values can be accessed for the ac analysis by replacing the "V" in V(N1,N2) with:

VR - real part

VI - imaginary part

VM - magnitude

VP - phase

VDB - 20 log10(magnitude)

I(VXXXXXXX) specifies the current flowing in the independent voltage source named VXXXXXXX. Positive current flows from the positive node, through the source, to the negative node. For the ac analysis, the corresponding replacements for the letter I may be made in the same way as described for voltage outputs.Output variables for the noise and distortion analyses have a different general form from that of the other analyses.

There is no limit on the number of .PRINT lines for each type of analysis.

1.2.11.7.3.11 SAVE Lines

General form:

.SAVE vector vector vector ...

Examples:

.SAVE i(vin) input output .SAVE @m1[id]

The vectors listed on the .SAVE line are recorded in the rawfile for use later with spice3 or nutmeg (nutmeg is just the data-analysis half of spice3, without the ability to simulate). The standard vector names are accepted. If no .SAVE line is given, then the default set of vectors are saved (node voltages and voltage source branch currents). If .SAVE lines are given, only those vectors specified are saved. For more discussion on internal device data, see Appendix B. See also the section on the interactive command interpretor for information on how to use the rawfile.

1.2.11.7.3.12 Analysis Options

The Analysis Options dialog box is shown below.

Select the option to change by clicking the left mouse button over the option name.

For full details on the options see Simulator variables.

Reset to default. Click to set the options back to the SPICE defaults.

Use default value. Check to use the default value. Uncheck to change the value.

OK. Click to accept your changes.

Cancel. Cancel your changes.

Help. Display this help topic.

1.2.11.7.3.13 Simulator Variables

SIMULATOR VARIABLES (.OPTIONS)

Various parameters of the simulations available in Spice3 can be altered to control the accuracy, speed, or default values for some devices. These parameters may be changed via the "set" command (described later in the section on the interactive front-end) or via the ".OPTIONS" line:

General form:

.OPTIONS OPT1 OPT2 ... (or OPT=OPTVAL ...)

Examples:

.OPTIONS RELTOL=.005 TRTOL=8

The options line allows the user to reset program control and user options for specific simulation purposes. Additional options for Nutmeg may be specified as well and take effect when Nutmeg reads the input file. Options specified to Nutmeg via the 'set' command are also passed on to SPICE3 as if specified on a .OPTIONS line. See the following section on the interactive command interpreter for the parameters which may be set with a .OPTIONS line and the format of the 'set' 'command. Any combination of the following options may be included, in any order. 'x' (below) represents some positive number.

Option	Effect
ABSTOL=x	resets the absolute current error tolerance of the program. The default value is 1 picoamp.
BADMOS3	Use the older version of the MOS3 model with the "kappa" discontinuity.
CHGTOL=x	resets the charge tolerance of the program. The default value is 1.0e-14.
DEFAD=x	resets the charge tolerance of the program. The default value is 1.0e-14.
DEFAS=x	resets the value for MOS source diffusion area; the default is 0.0.
DEFL=x	resets the value for MOS source diffusion area; the default is 0.0.

DEFW=x	resets the value for MOS channel width; the default is 100.0 micrometer
GMIN=x	resets the value of GMIN, the minimum conductance allowed by the program. The default value is 1.0e-12.
ITL1=x	resets the dc iteration limit. The default is 100.
ITL2=x	resets the dc transfer curve iteration limit. The default is 50.
ITL3=x	resets the lower transient analysis iteration limit. the default value is 4. (Note: not implemented in Spice3).
ITL4=x	resets the transient analysis timepoint iteration limit. the default is 10.
ITL5=x	resets the transient analysis total iteration limit. the default is 5000. Set ITL5=0 to omit this test. (Note: not implemented in Spice3).
KEEPOPINFO	Retain the operating point information when either an AC, Distortion, or Pole-Zero analysis is run. This is particularly useful if the circuit is large and you do not want to run a (redundant) ".OP" analysis.
METHOD=name	resets the numerical integration method used by SPICE. Possible names are "Gear" or "trapezoidal" (or just "trap"). The default is trapezoidal.
PIVREL=x	resets the relative ratio between the largest column entry and an acceptable pivot value. The default value is 1.0e-3. In the numerical pivoting algorithm the allowed minimum pivot value is determined by EPSREL=AMAX1(PIVREL*MAXVAL, PIVTOL) where MAXVAL is the maximum element in the column where a pivot is sought (partial pivoting).
PIVTOL=x	resets the absolute minimum value for a matrix entry to be accepted as a pivot. The default value is 1.0e-13.
RELTOL=x	resets the relative error tolerance of the program. The default value is 0.001 (0.1%).
TEMP=x	Resets the operating temperature of the circuit. The default value is 27 deg C (300 deg K). TEMP can be overridden by a temperature specification on any temperature dependent instance.
TNOM=x	resets the nominal temperature at which device parameters are measured. The default value is 27 deg C (300 deg K). TNOM can be

	overridden by a specification on any temperature dependent device model.
TRTOL=x	resets the transient error tolerance. The default value is 7.0. This parameter is an estimate of the factor by which SPICE overestimates the actual truncation error.
TRYTOCOMPACT	Applicable only to the LTRA model. When specified, the simulator tries to condense LTRA transmission lines' past history of input voltages and currents.
VNTOL=x	resets the absolute voltage error tolerance of the program. The default value is 1 microvolt.

In addition, the following options have the listed effect when operating in spice2 emulation mode:

Option	Effect
ACCT	causes accounting and run time statistics to be printed
LIST	causes the summary listing of the input data to be printed
NOMOD	suppresses the printout of the model parameters
NOPAGE	suppresses page ejects
NODE	causes the printing of the node table.
OPTS	causes the option values to be printed.

1.2.11.7.3.14 Small-Signal AC Analysis

General form:

.AC DEC ND FSTART FSTOP

.AC OCT NO FSTART FSTOP

.AC LIN NP FSTART FSTOP

Examples:

.AC DEC 10 1 10K .AC DEC 10 1K 100MEG .AC LIN 100 1 100HZ

DEC stands for decade variation, and ND is the number of points per decade. OCT stands for octave variation, and NO is the number of points per octave. LIN stands for linear variation, and NP is the number of points. FSTART is the starting frequency, and FSTOP is the final frequency. If this line is included in the input file, SPICE Simulation Program for Integrated Circuit Emulation. performs an AC analysis of the circuit over the specified frequency range. Note that in order for this analysis to be meaningful, at least one independent source must have been specified with an ac value.

1.2.11.7.3.15 Transfer Function Analysis

General form:

.TF OUTVAR INSRC

Examples:

.TF V(5, 3) VIN .TF I(VLOAD) VIN

The TF line defines the small-signal output and input for the dc small-signal analysis. OUTVAR is the small-signal output variable and INSRC is the small-signal input source. If this line is included, SPICE computes the dc small-signal value of the transfer function (output/input), input resistance, and output resistance. For the first example, SPICE would compute the ratio of V(5, 3) to VIN, the small

1.2.11.7.3.16 Transient Analysis

General form:

.TRAN TSTEP TSTOP &ltTSTART; &ltTMAX;>>

Examples:

.TRAN 1NS 100NS .TRAN 1NS 1000NS 500NS .TRAN 10NS 1US

TSTEP is the printing or plotting increment for line-printer output. For use with the post-processor, TSTEP is the suggested computing increment. TSTOP is the final time, and TSTART is the initial time. If TSTART is omitted, it is assumed to be zero. The transient analysis always begins at time zero. In the interval &ltzero;, TSTART>, the circuit is analyzed (to reach a steady state), but no outputs are stored. In the interval &ltTSTART;, TSTOP>, the circuit is analyzed and outputs are stored. TMAX is the maximum stepsize that SPICE uses; for default, the program chooses either TSTEP or (TSTOP-TSTART)/50.0, whichever is smaller. TMAX is useful when one wishes to guarantee a computing interval which is smaller than the printer increment, TSTEP.

UIC (use initial conditions) is an optional keyword which indicates that the user does not want SPICE to solve for the quiescent operating point before beginning the transient analysis. If this keyword is specified, SPICE uses the values specified using IC=... on the various elements as

the initial transient condition and proceeds with the analysis. If the .IC control line has been specified, then the node voltages on the .IC line are used to compute the initial conditions for the devices. Look at the description on the .IC control line for its interpretation when UIC is not specified.

1.2.11.8 Independent Sources for Voltage or Current

Coming soon...

1.2.11.8.1 Exponential Sources

Coming soon...

1.2.12 Importing Files

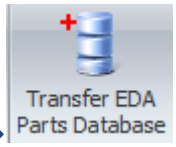
You can import the following file types into AutoTRAX DEX:

- AutoTRAX DEX EDA
- AutoTRAX DEX EDA Library
- AutoCAD DXF Files
- Eagle Libraries and Files

1.2.12.1 Importing Your AutoTRAX EDA Designs

To read AutoTRAX DEX EDA files into AutoTRAX DEX open either the schematic file or the PCB file. AutoTRAX DEX will read both and combine them into a single design.

1.2.12.2 Importing Your AutoTRAX EDA Library

To import your AutoTRAX DEX EDA library click the Tools→Utilities→ 

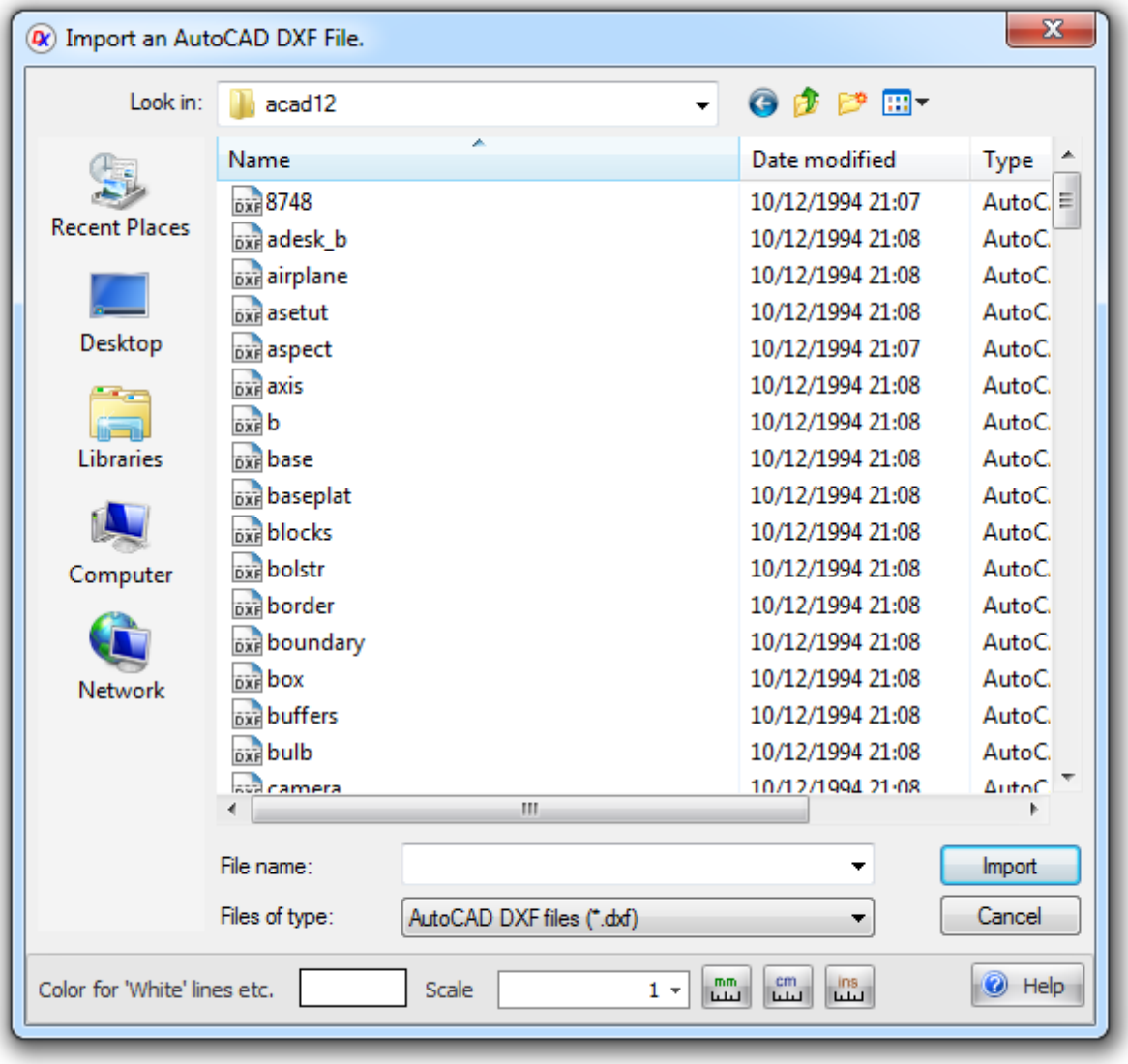
button.

1.2.12.3 Importing AutoCAD DXF Files

To import an AutoCAD DXF file into AutoTRAX DEX click the Tools→Import → 

button.

The objects will be imported on to the current layer and added to a single group. 3D dimensions will be ignored.



DXF import dialog

Color for 'White' lines etc.

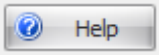
Click the  button to select the color to use for DXF entities.

Scale

Select the scale for importing entities. The larger the number, the larger the entities will be when imported.

DXF Units

Select the units in the DXF file.

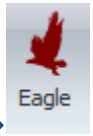


Click to display this help topic.

1.2.12.4 Importing Eagle Projects

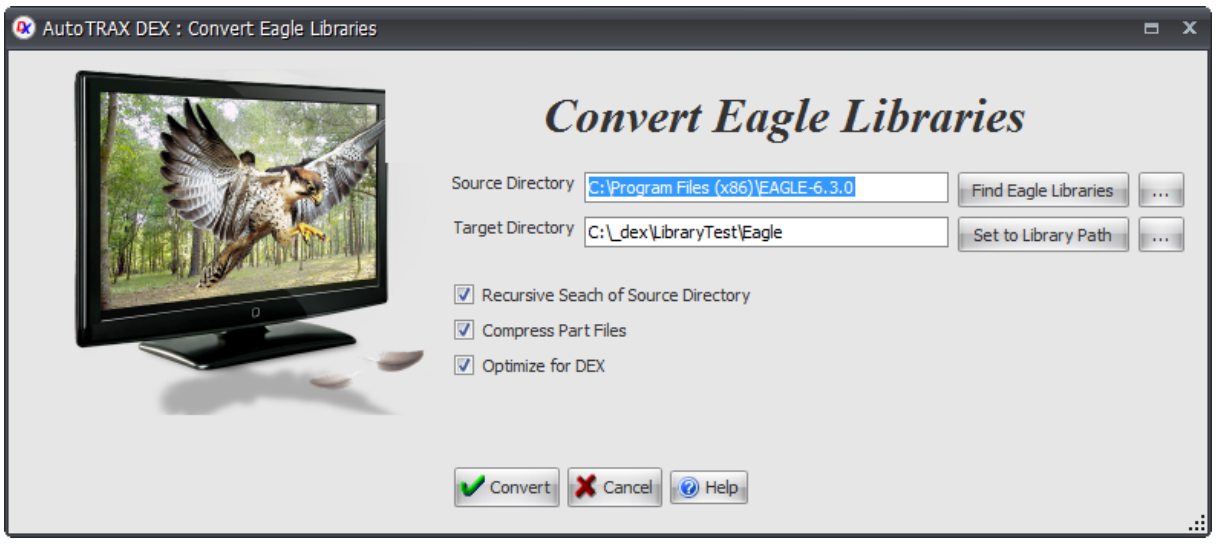
You can import Eagle schematic and board files into AutoTRAX DEX. They will combined into a single project.

1.2.12.5 Importing Eagle Libraries

To import an Eagle library file into AutoTRAX DEX click the Tools→Import→ 

button.

AutoTRAX DEX will combine the part schematic symbols and the footprints (if found) into a single AutoTRAX DEX part.



Eagle import dialog

Source Directory

Enter the directory containing the Eagle libraries.

Target Directory

Enter the target directory for the converted parts.

Recursive Search of Source Directory

Check to search the source directory and all it's child directories for Eagle libraries.

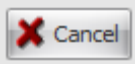
Compress Part Files

Check to automatically compress the resulting AutoTRAX DEX parts. This is highly recommended and will save you over 90% of the disk space for uncompressed files.

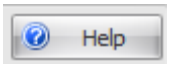
Optimize for AutoTRAX DEX

Check to have terminal magnets and parametric footprints automatically created for each part whenever possible.

Click to convert all libraries found.



Click to cancel this dialog.



Click to show this help topic.

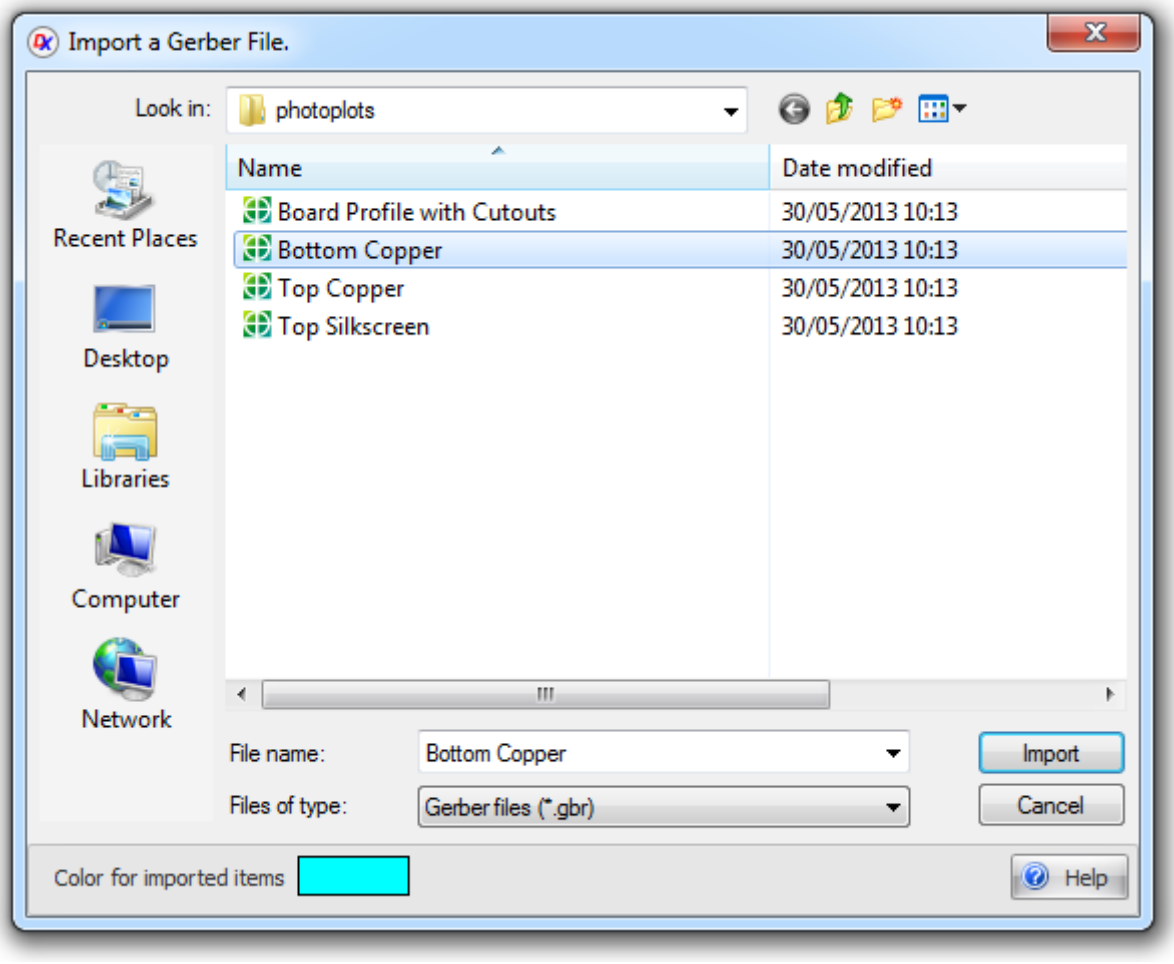
1.2.12.6 Importing Gerber Files

To import a Gerber file into AutoTRAX DEX click the Tools→Import→ 

button.

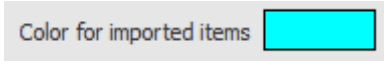
The objects will be imported on to the current layer.

The Gerber will be converted to graphics only and added to a single group. Not nets, tracks, pads or vias will be created.

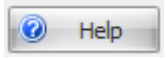


Gerber import dialog

Regards

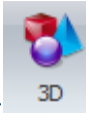


Click to select the color for the graphics when added to non-electrical layers.



Click to display this help topic.

1.2.12.7 Importing 3D

To import a 3D file into AutoTRAX DEX click the Tools→Import→ 

button.

XGL

VRML

WRL

DAE

1.2.12.7.1 XGL

The XGL file format serves the purpose of representing 3D information for visualization purposes. This format encompasses all aspects of 3D data through a tessellated representation. The originators of the XGL format, namely Siemens' Solid Edge and Autodesk's Inventor, were motivated by the need for a versatile graphics language. The XGL format operates in conjunction with the widely-used OpenGL systems, a prevalent graphics rendering library. This symbiotic relationship enables seamless conversion of models crafted within OpenGL into XGL files for utilization across diverse applications.

Compatibility with OpenGL Systems

The XGL file format adopts the XML 1.0 syntax, facilitating its composition and interpretation. The incorporation of tags enhances the referencing of components within XGL files, culminating in more compact and efficient file structures. Moreover, the application of referencing mechanisms simplifies the reading and writing processes associated with XGL files.

XML Syntax and Tag Referencing

An inherent benefit of the XGL format is its user-friendliness. The utilization of tags and attributes, inherent to XML syntax, greatly streamlines the tasks of third-party applications in reading and writing data. This, in turn, facilitates the creation of customized visualization tools capable of seamlessly importing and exporting XGL files.

1.2.12.7.2 VRML

VRML (Virtual Reality Modeling Language) is an open, text-based file format used to describe three-dimensional scenes and objects for interactive 3D graphics applications and virtual reality environments. VRML files define the geometry, appearance, lighting, and interactivity of 3D scenes, allowing them to be rendered and explored in real-time using compatible VRML viewers and browsers.

Key features of VRML include:

Scene Description: VRML files describe the structure of a 3D scene, including objects, their positions, orientations, scales, colors, and textures.

Hierarchy: VRML scenes can have a hierarchical structure with parent-child relationships between objects, allowing for complex scene organization.

Geometry and Appearance: VRML supports basic geometric shapes (cubes, spheres, cones, etc.) and provides options for applying textures, materials, and lighting to objects.

Interactivity: VRML supports interactivity through scripting, allowing users to interact with and manipulate objects within the scene.

Animation: VRML supports animations, including the ability to animate object transformations and properties over time.

Navigation: VRML scenes can include navigation information, enabling users to move through and explore the virtual environment.

File Format: VRML files are typically saved with the .wrl file extension. The format uses a text-based syntax that is human-readable and editable.

Compatibility: VRML was designed to be platform-independent and can be viewed in VRML-enabled browsers, standalone viewers, and 3D modeling software.

Integration: VRML can be integrated with web pages to create interactive 3D content for websites.

VRML was developed in the 1990s and gained popularity as one of the early standards for creating interactive 3D content for the web. However, its adoption diminished over time as newer technologies and formats emerged, such as X3D (an evolution of VRML) and more modern web technologies like WebGL.

X3D is a successor to VRML and extends its capabilities to support advanced features and better integration with modern web standards. WebGL, a JavaScript API, enables hardware-accelerated 3D graphics in web browsers without the need for plugins.

If you're working with 3D content for the web, consider exploring X3D and WebGL as potential alternatives to VRML, as they offer improved performance, better compatibility, and integration with contemporary web technologies.

1.2.12.7.3 WRL

WRL files are files that use the VRML (Virtual Reality Modeling Language) format. VRML, often pronounced as "vermal," stands for Virtual Reality Modeling Language, and it's a text-based file format used for representing three-dimensional scenes and objects for interactive 3D graphics applications and virtual reality environments.

WRL files contain the descriptions of 3D scenes, objects, their appearances, transformations, and more. These files are used to create interactive 3D

environments that can be visualized in VRML viewers, browsers, and compatible 3D modeling software.

Here are some key features of WRL files (VRML format):

Scene Description: WRL files describe the structure and content of a 3D scene. This includes defining objects, their positions, orientations, colors, textures, materials, and more.

Geometry and Appearance: WRL supports various geometric primitives (spheres, cubes, cylinders, etc.) and allows you to define appearance properties like textures, materials, and lighting.

Hierarchical Structure: WRL scenes can have a hierarchical structure, allowing you to organize objects in parent-child relationships.

Interactivity: You can add interactivity to WRL scenes through scripting, allowing users to interact with and manipulate objects in the virtual environment.

Animations: WRL files support animations, enabling you to create dynamic scenes with moving or changing objects.

Navigation: WRL scenes can include navigation information, allowing users to explore the virtual environment.

File Format: WRL files are typically saved with the .wrl file extension. The format uses a text-based syntax that is human-readable and editable.

Integration: WRL files can be embedded in web pages to create interactive 3D content for websites.

It's important to note that while VRML was influential in the early days of 3D graphics on the web, its popularity has waned over time due to the emergence of other formats and technologies like X3D (an evolution of VRML) and WebGL (which enables 3D graphics directly in web browsers).

If you encounter WRL files, it's likely that they are VRML files, and you would need a VRML viewer or compatible software to visualize and interact with the 3D scenes they represent. Additionally, you might consider exploring modern alternatives like X3D and WebGL for more up-to-date and web-friendly 3D graphics solutions.

1.2.12.7.4 DAE

DAE files are 3D models saved in the Collada (COLLAborative Design Activity) file format. Collada is an open standard XML-based file format used to exchange digital

assets between different 3D modeling and computer graphics software applications. DAE files are versatile and can store various aspects of a 3D scene, including geometry, materials, textures, animations, and more.

Here are the key features of DAE files:

- **Geometry:** DAE files contain the geometric information of 3D models, including vertices, normals, texture coordinates, and other properties that define the shape of objects.
- **Materials and Textures:** DAE files support material definitions, texture maps, and shading information to control how objects are visually rendered.
- **Animations:** DAE files can store animations, including skeletal animations, morph targets, and keyframe animations.
- **Scene Hierarchy:** DAE files can represent complex scene hierarchies with parent-child relationships between objects.
- **Cameras and Lights:** DAE files can include camera and light definitions to recreate the lighting and camera perspectives of the original scene.
- **Interactive Elements:** DAE files can support interactivity and user-defined behaviors through scripting and animations.
- **Open Standard:** Collada is an open and widely adopted standard, ensuring compatibility between different 3D software applications.
- **XML-Based:** DAE files are written in XML (eXtensible Markup Language), making them human-readable and editable.

DAE files are commonly used for various purposes:

- **3D Modeling:** They are used for storing and exchanging 3D models created in various modeling software, making it easier to collaborate and share models between different tools.
- **Animation:** DAE files are used to store animations for characters and objects, allowing for consistent playback in different applications.
- **Game Development:** DAE files are used in game development for importing models and animations into game engines.
- **Virtual Reality:** They are used in virtual reality and augmented reality applications to represent 3D scenes and objects.
- **Architectural Visualization:** DAE files are used to exchange architectural models between different design software and visualization tools.

Collada's flexibility and support for multiple aspects of 3D content make DAE files suitable for a wide range of applications across industries. However, like any file format, compatibility and feature support may vary between different software

applications, so it's important to ensure that the software you're using supports the features you need when working with DAE files.

1.2.13 Exporting Data

You can export your design to the following formats:

CNC

PDF

IDF

SVG

Net List

Image

Spice Deck

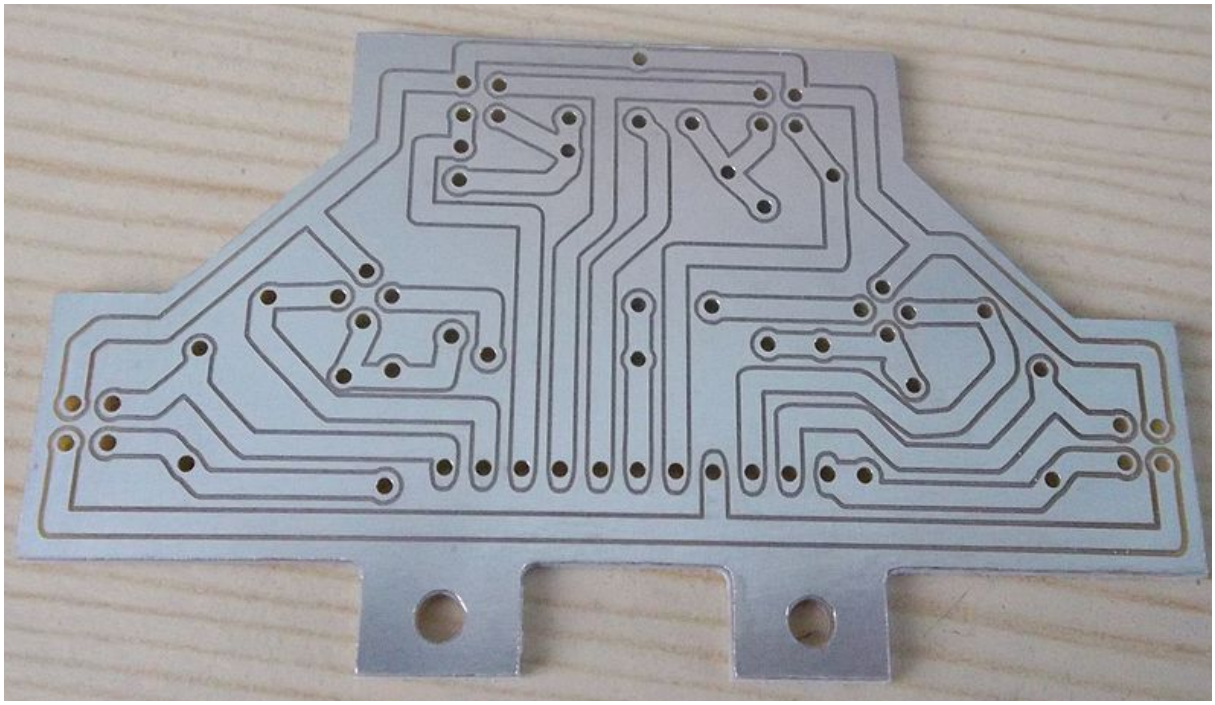
1.2.13.1 CNC

AutoTRAX DEX can directly create the CNC files you need to create a PCB.

Creating a CNC File

Printed circuit board milling (also: isolation milling) is the process of removing areas of copper from a sheet of printed circuit board material to recreate the pads, signal traces and structures according to patterns from a digital circuit board plan known as a layout file. Similar to the more common and well known chemical PCB etch process, the PCB milling process is subtractive: material is removed to create the electrical isolation and ground planes required. However, unlike the chemical etch process, PCB milling is typically a non-chemical process and as such it can be completed in a typical office or lab environment without exposure to hazardous chemicals. High quality circuit boards can be produced using either process. In the case of PCB milling, the quality of a circuit board is chiefly determined by the system's true, or weighted, milling accuracy and control as well as the condition (sharpness, temper) of the milling bits and their respective feed/rotational speeds.

By contrast, in the chemical etch process, the quality of a circuit board depends on the accuracy and/or quality of the photo-masking and the state of the etching chemicals.



Milled Printed Circuit Board

PCB milling has advantages for both prototyping and also for some specialist PCB designs.

Prototyping can provide a fast-turnaround board production process without the need for wet processing. If a CNC mill is already used for drilling, this single machine can carry out both parts of the process.

Some PCBs, requiring extensive drilling or profiling of the board outline are particularly suitable for production by milling. Where the tracks must precisely follow the board shape, using the same CNC mill for both reduces registration and scaling errors. Many boards that are simple for milling would be almost impossible by wet etching and manual drilling afterwards.

In mass production, milling is unlikely to replace etching although the use of CNC is already standard practice for drilling the boards.

PCB milling system is a single machine that can perform all of the required actions to create a prototype board, with the exception of inserting vias and through hole plating. Most of these machines require only a standard AC mains outlet and a shop-type vacuum cleaner for operation.

he mechanics behind a PCB milling machine are fairly straightforward and have their roots in CNC milling technology. A PCB milling system is similar to a miniature and highly accurate NC milling table. For machine control, positioning information and

machine control commands are sent from the controlling software via a serial port or parallel port connection to the milling machine's on-board controller. The controller is then responsible for driving and monitoring the various positioning components which move the milling head and gantry and control the spindle speed. Spindle speeds can range from 30,000 RPM to 100,000 RPM depending on the milling system, with higher spindle speeds equating to better accuracy. Typically this drive system comprises non-monitored stepper motors for the X/Y axis, an on-off non-monitored solenoid or pneumatic piston for the Z-axis, and a DC motor control circuit for spindle speed, none of which provide positional feedback. More advanced systems provide a monitored stepper motor Z-axis drive for greater control during milling and drilling as well as more advanced RF spindle motor control circuits that provide better control over a wider range of speeds.

X and Y axis control

For the X and Y axis drive systems most PCB milling machines use stepper motors that drive a precision lead screw. The lead screw is in turn linked to the gantry or milling head by a special precision machined connection assembly. To maintain correct alignment during milling, the gantry or milling head's direction of travel is guided along using linear or dovetailed bearing(s). Most X/Y drive systems provide user control, via software, of the milling speed, which determines how fast the stepper motors drive their respective axes.

Z axis control

Z axis drive and control are handled in several ways. The first and most common is a simple solenoid that pushes against a spring. When the solenoid is energized it pushes the milling head down against a spring stop which is attached to a pressure foot assembly that limits the milling head's downward travel. The rate of descent as well as the amount of force exerted on the spring stop must be manually set by mechanically adjusting the position of the solenoid's plunger. The second type of Z-axis control is through the use of a pneumatic cylinder - this system functions in the same manner as the solenoid type, pushing against a spring stop/pressure foot assembly. Air for the cylinder is provided by an external compressor with the air flow being controlled by a manually operated regulator and software driven gate valve. Due to the small cylinder size and the amount of air pressure used to drive it there is little range of control between the up and down stops. Both the solenoid and pneumatic system provide no positional feedback while in motion, and are therefore useful for only simple 'up/down' milling tasks. The final type of Z-axis control uses a stepper motor with dynamic positioning feedback. This system allows the milling head to be moved in small accurate steps up or down through its whole range of vertical motion. Further, the speed of these steps can be adjusted to allow tool bits to be eased into the board material rather than hammered into it. The depth (number of steps required) as well as the downward/upward speed is under user control via the controlling software.

Tooling

PCBs may be machined with conventional end-mills, conical d-bit cutters, and spade mills. D-bits and spade mills are cheap and as they have a small point allow the traces to be close together.

Alternatives

A method with similar advantages to mechanical milling is laser etching. Etching PCBs with lasers offers the same advantages as mechanical milling in regards to quick turnaround times, but the nature of the laser etching process is preferable to both milling and chemical etching when it comes to physical variations exerted on the object. Whereas mechanical milling and chemical etching exact physical stress on the board, laser etching offers non-contact surface removal, making it a superior option for PCBs where precision and geometric accuracy are at a premium, such as RF & Microwave designs.

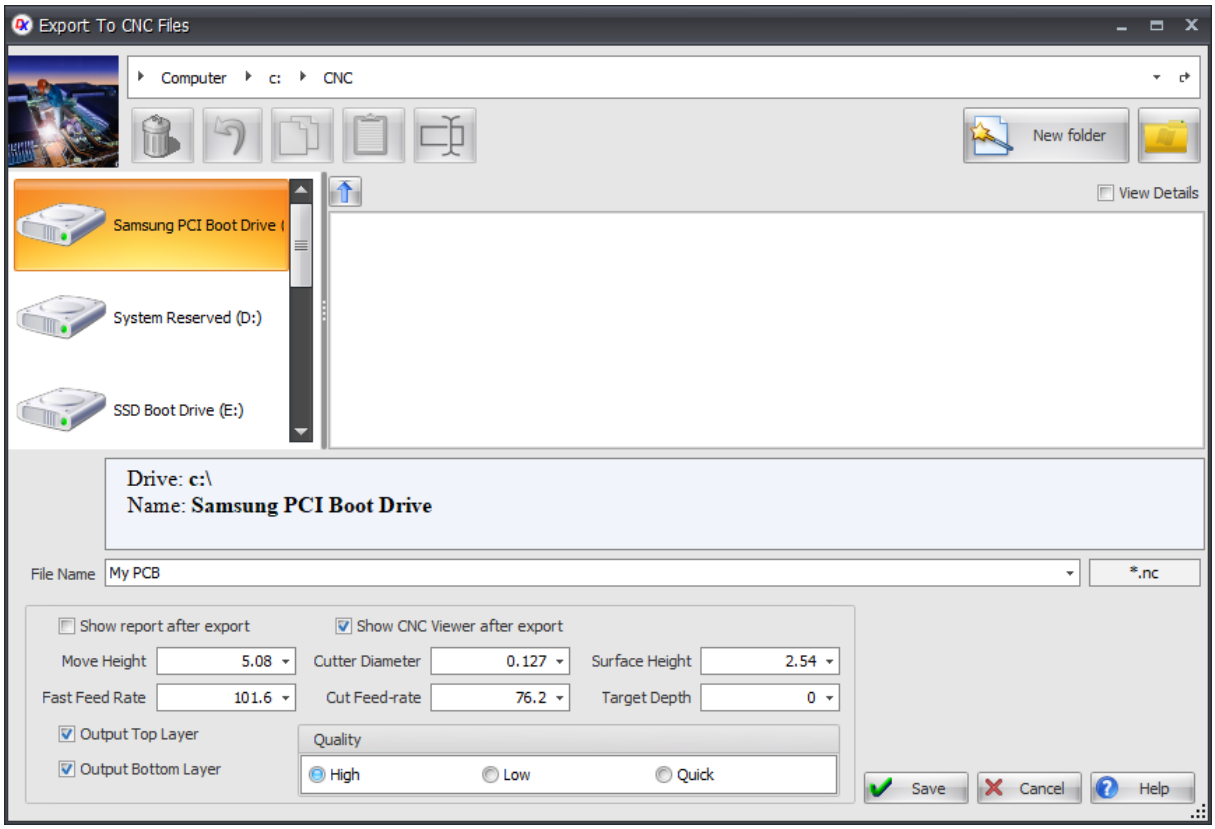
1.2.13.1.1 Creating a CNC File

To create/export to a CNC file click the Tools→Export→ 

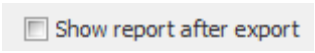
button. The dialog box show below will appear.

AutoTRAX DEX will generate.

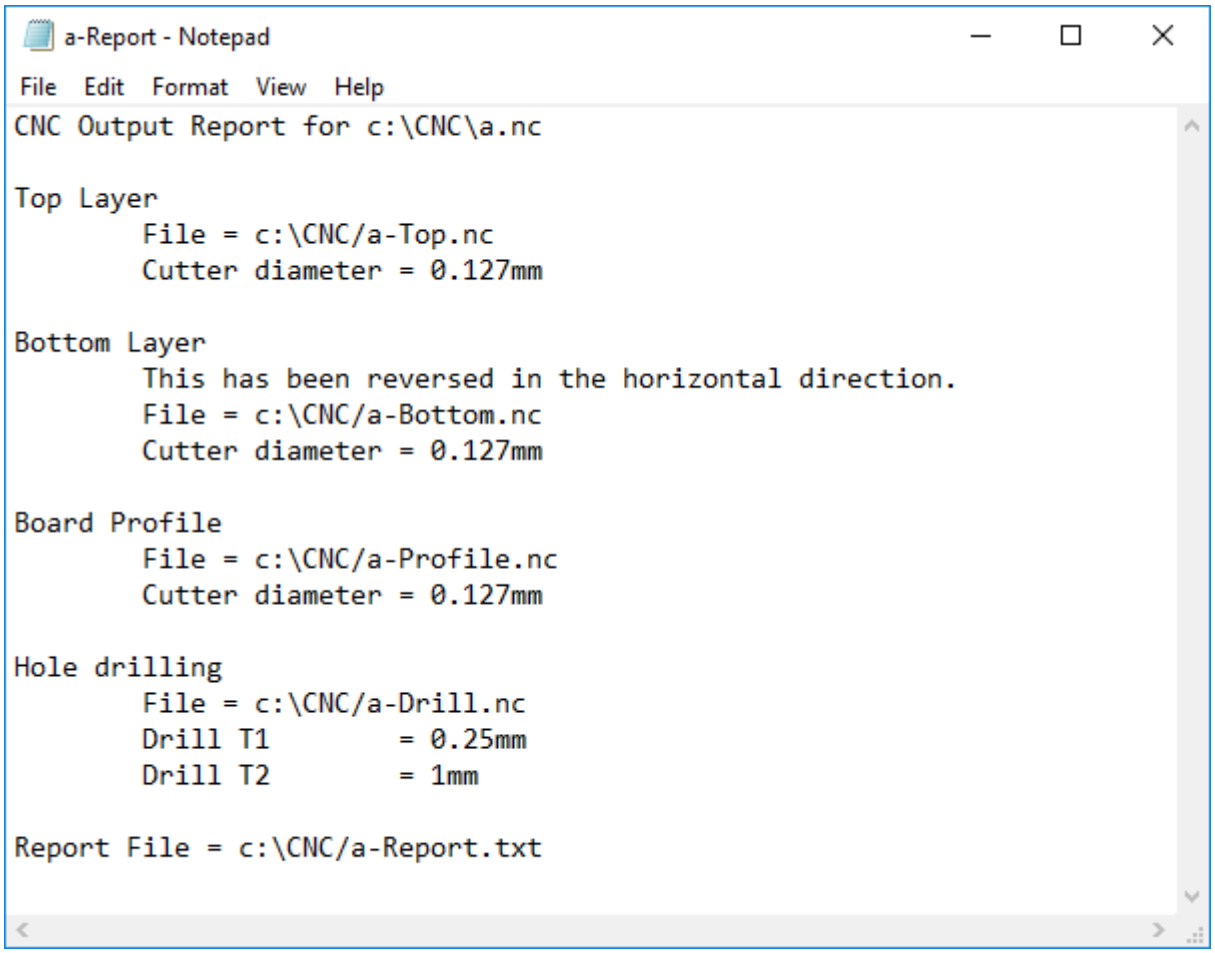
1. A top side CNC file named XXXX-**Top**.nc
2. A bottom side CNC file named XXXX-**Bottom**.nc
3. A Board profile file named XXXX-**Profile**.nc
4. A Drilling file named XXXX-**Drill**.nc
5. Where **XXXX** is the name you select.



Generate CNC files dialog

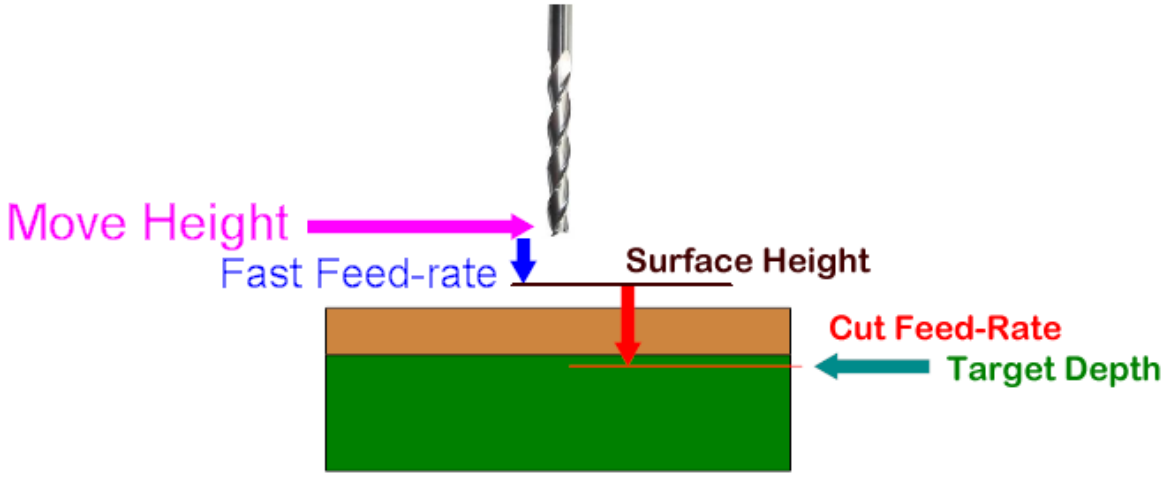


Check to show a report of files generated.



Typical Report File

Check to show a view of the files generated. See CNC Viewer



Move Height

This is the vertical (Z) height for movements of the tool when not cutting.

Fast Feed-Rate

Enter the feed rate when the cutter is moving without cutting at the move height.

Cutter Diameter

Enter the diameter for the cutter.

Cut Feed-Rate

Enter the feed rate when the drill is cutting.

Surface Height

This is the vertical (Z) height for below which the feed-rate is given by the cut feed-rate.

Target Depth

This is the vertical (Z) height for the cutter when cutting copper.

Output Top Layer

Check to output the top layer.

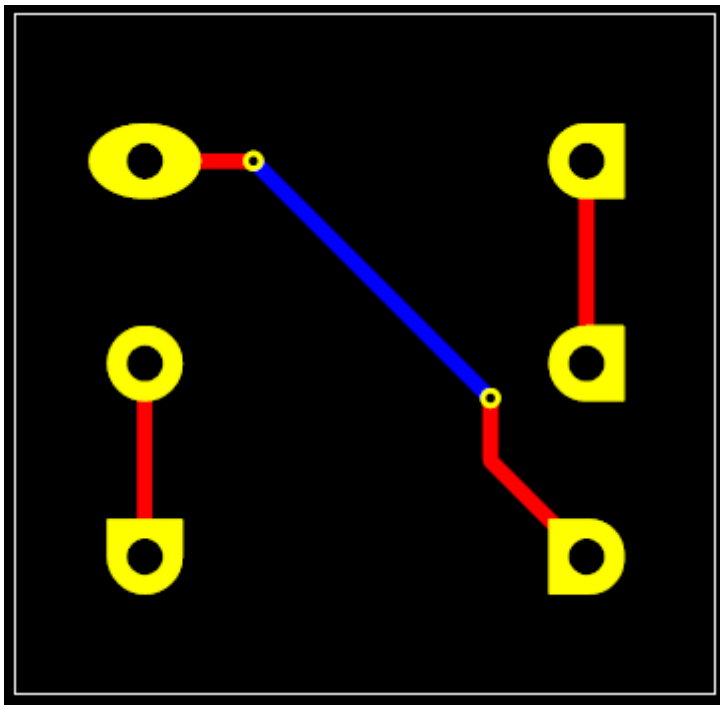
Output Bottom Layer

Check to output the bottom layer.

Quality

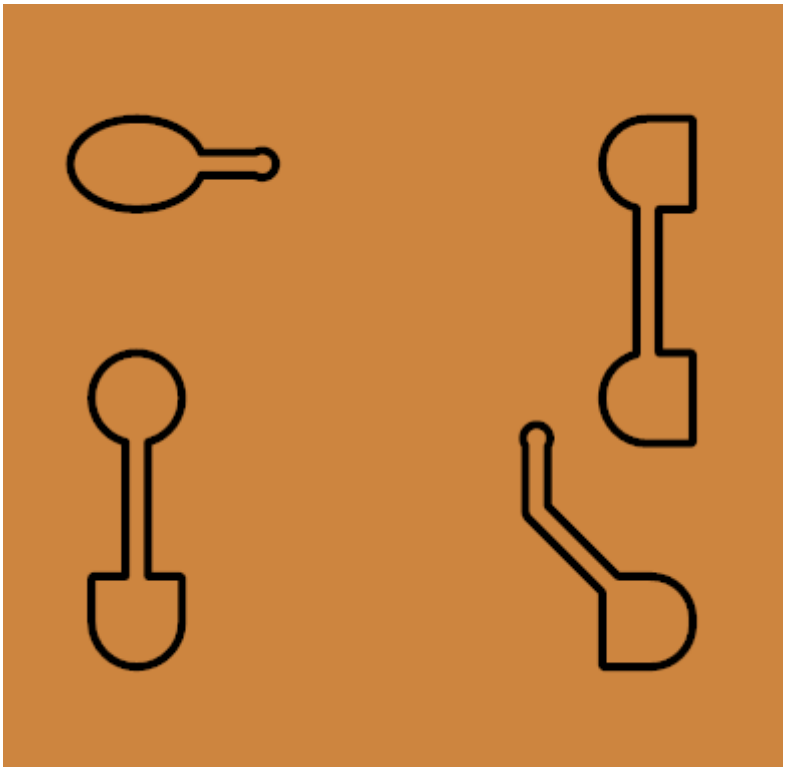
Select quality.

To see the effect of the quality setting consider the following sample PCB...



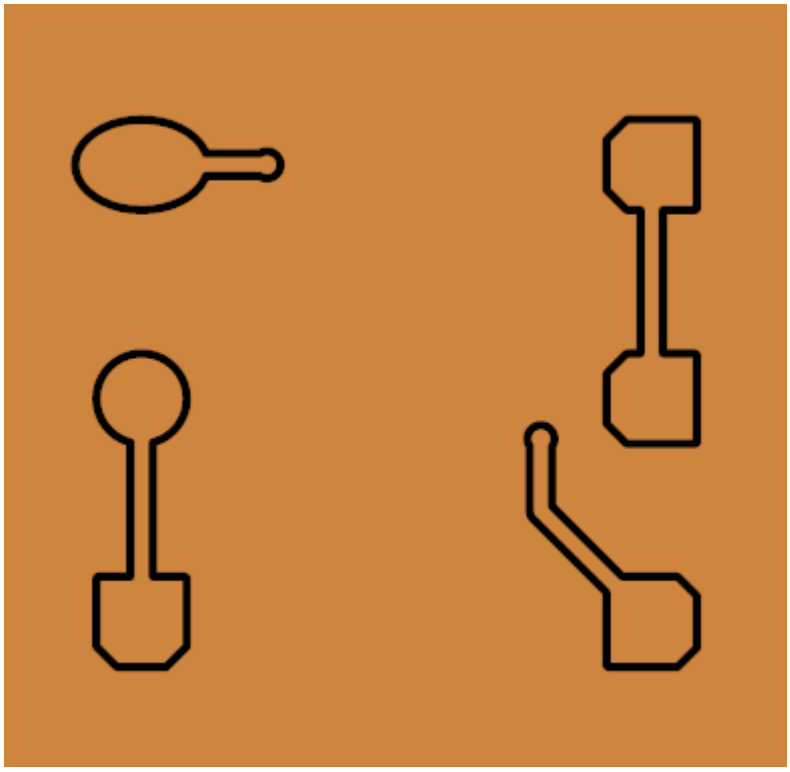
High

The high setting gives you the best quality of CNC cut as shown below.



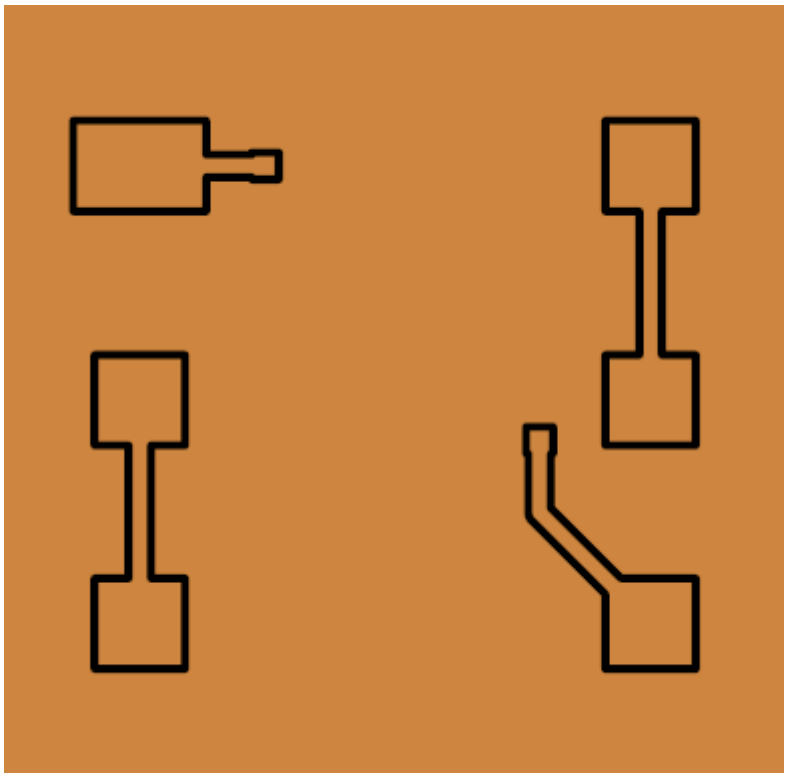
Low

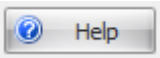
The low setting gives you the best quality of CNC cut as shown below. Note, the rounded corners of the rectangles are replaced by beveled corners. This setting gives you a lot faster production of your CNC files compared to the high setting but is much slower than the Quick setting.



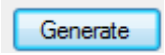
Quick

The quick setting gives you the poorest quality of CNC cut as shown below. Note, elliptical and rounded pad shapes are replaced by rectangular budgets and vias are replaced by rectangles. This will give you the quickest production of your CNC files and is useful for complicated PCBs.

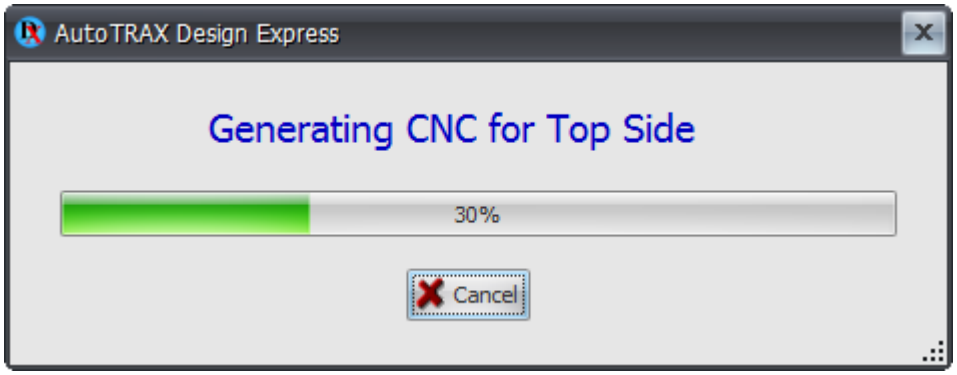




Click to display this help topic.

Click 

to generate the CNC files. You will see 3 progress dialogs like the one shown below; one for the top side and one for the bottom side.

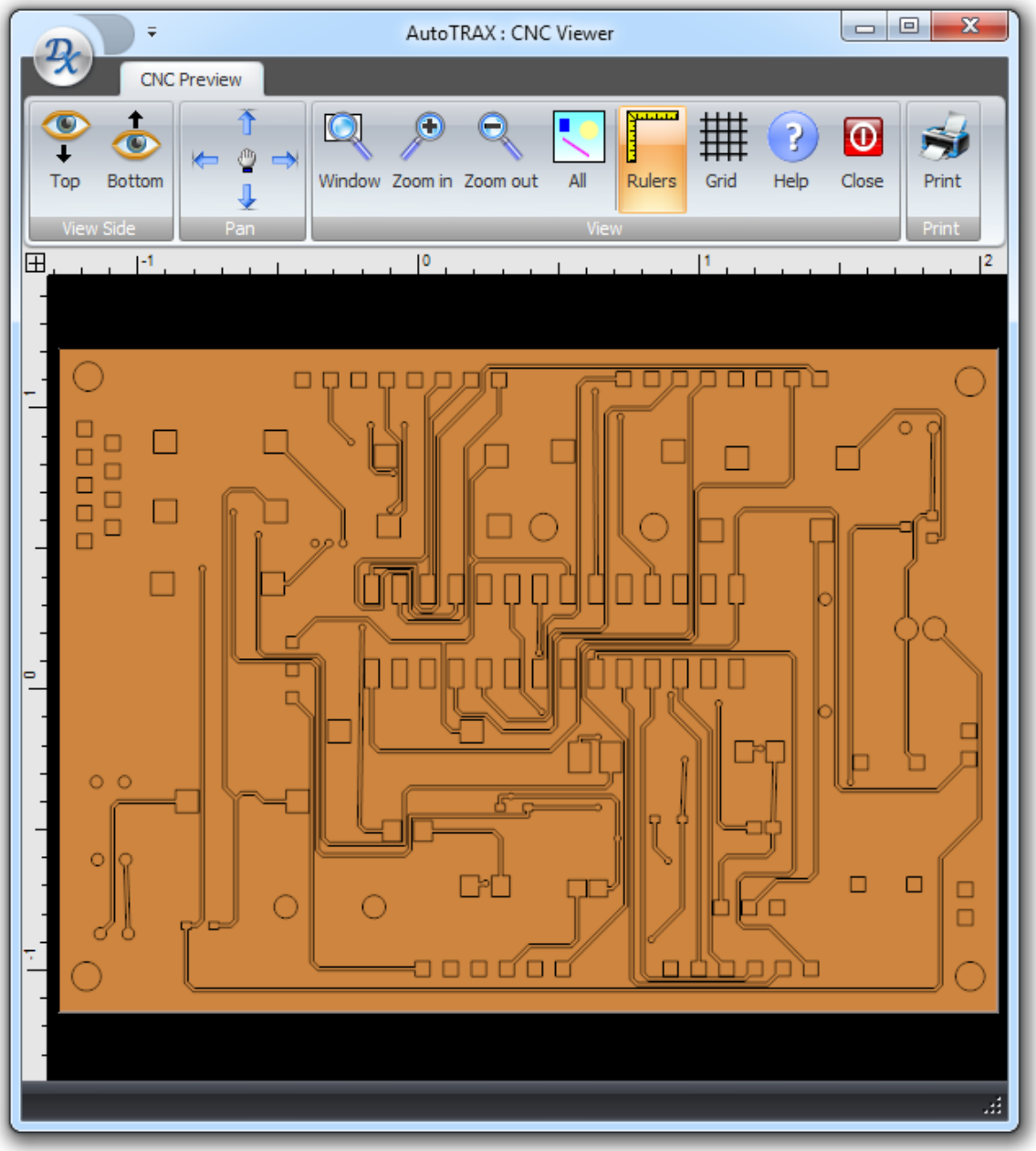


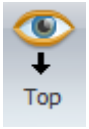
File generation progress dialog.

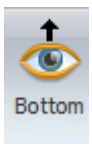
After the files have been generated you will see the report and/or the CNC Viewer

1.2.13.1.2 CNC Viewer

The CNC viewer is shown below.



Click 

to view the top side of the PCB and click 

to view the bottom of the PCB.

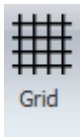
Click this to view an area in the viewport. Then hold down and drag the mouse to define the area to view.

Zoom in to see a magnified view of the PCB.

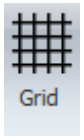
Zoom out to see more of the PCB.

View all the PCB.

Display/hide rulers around the edge of the viewport.



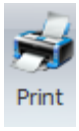
Click to show/hide the grid.



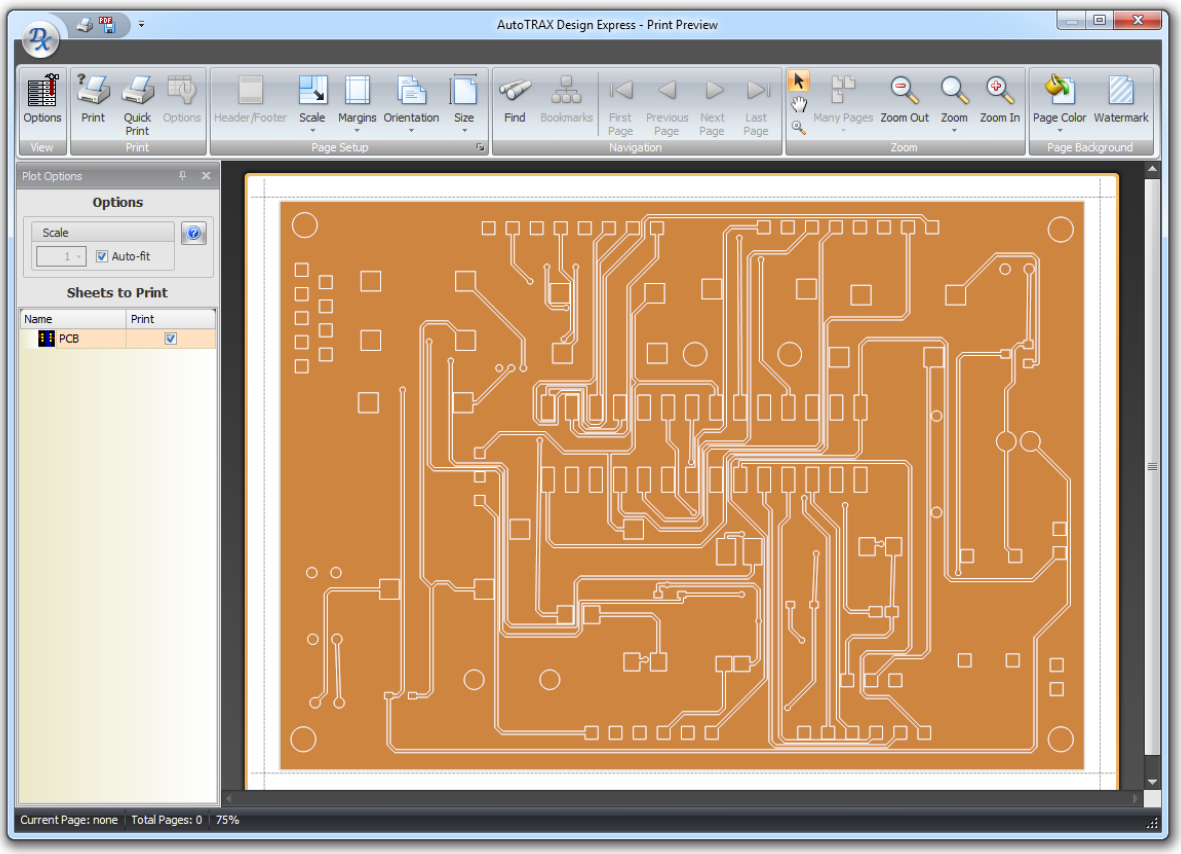
Click to display this help topic.

Click to close the viewer.

CNC Print Preview

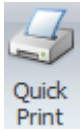


Click this to print a the side being viewed. This will display the Print preview dialog shown below.



Print Preview

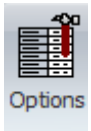
The print preview shown above consists of a ribbon menu, a options panel on the left and a preview window to the right of the options panel and below the ribbon menu.

click 

to print the selected pages.

Ribbon Menu

The ribbon menu can be split into 2 parts, the View button group on the left and the common ribbon button groups on the right.

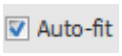
The 

button shows/hides the option panel shown below.

Click to read about the Common Print Menu

The Option Panel

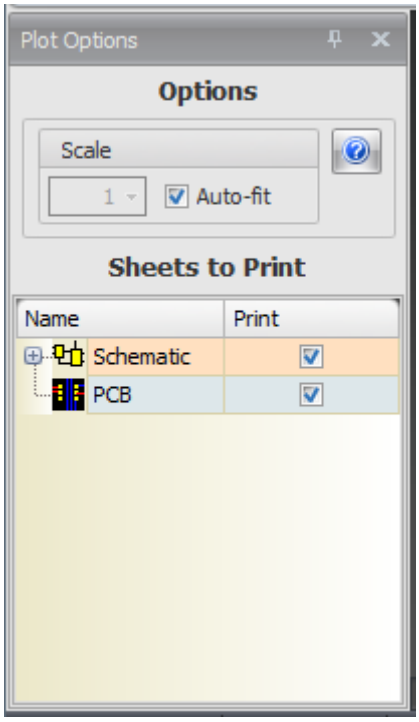
The options panel lets you scale the printing or have the sheets automatically fill the printed page,

Check 

to have the sheets automatically fill the printed page, If you uncheck it, you can set the print scale. Note, it is possible to set the scale such that the sheet will not all fit on the printed page.

Check/uncheck the check boxes in the Print column to include/exclude the sheet in the print run.

Clicking  displays this help topic.

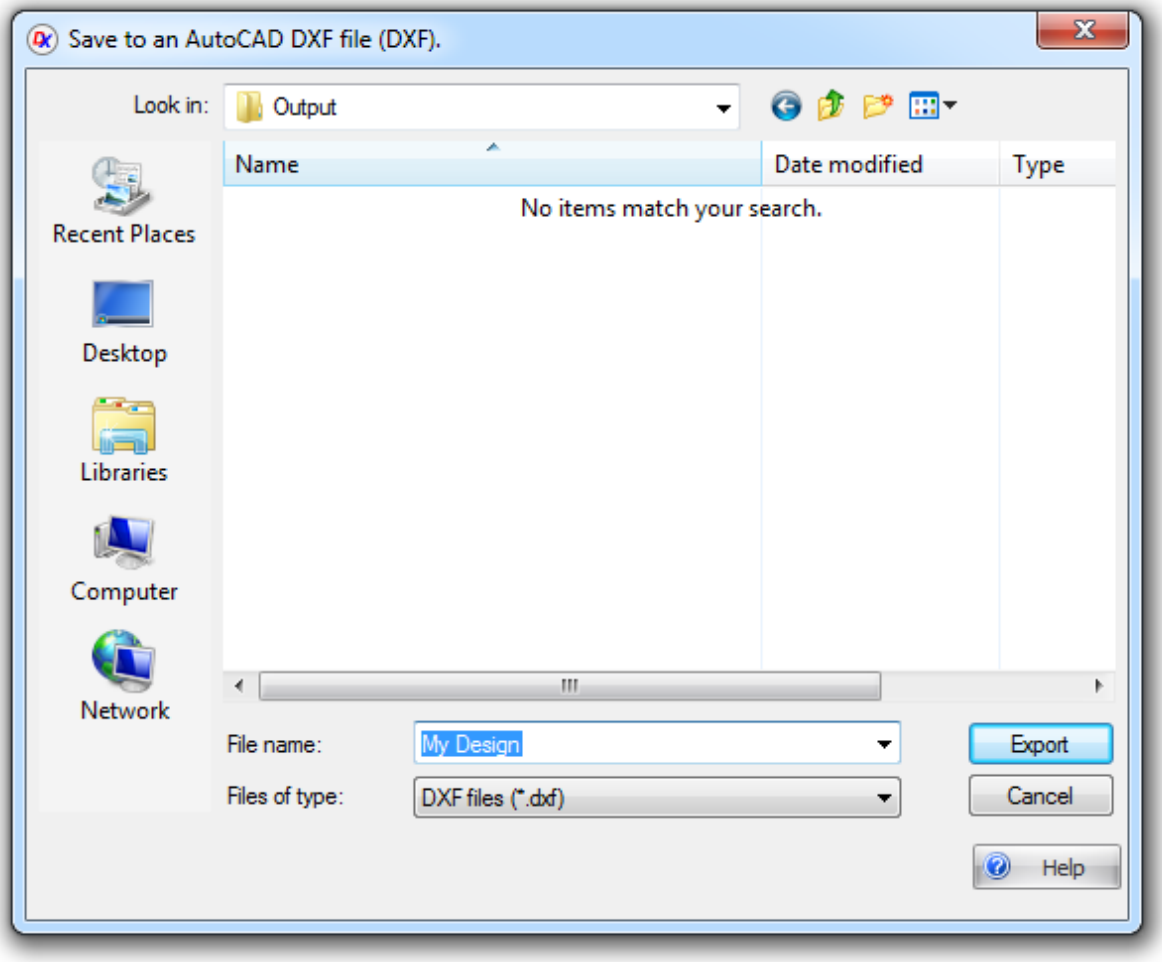


Options Dialog

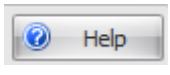
1.2.13.2 Creating a DXF File

To create/export to a DXF file click the Tools→Export→ 

button.



DXF export dialog



Click to show this help topic.

Auto CAD DXF (Drawing Interchange Format, or Drawing Exchange Format) is a CAD data file format developed by Autodesk for enabling data interoperability between AutoCAD and other programs.

DXF was originally introduced in December 1982 as part of AutoCAD 1.0, and was intended to provide an exact representation of the data in the AutoCAD native file format, DWI (Drawing), for which Autodesk for many years did not publish specifications. Because of this, correct imports of DXF files have been difficult. Autodesk now publishes the DXF specifications on its website for versions of DXF dating from AutoCAD Release 13 to AutoCAD 2010.

Versions of AutoCAD from Release 10 (October 1988) and up support both ASCII and binary forms of DXF. Earlier versions support only ASCII.

As AutoCAD has become more powerful, supporting more complex object types, DXF has become less useful. Certain object types, including ACIS solids and regions, are not documented. Other object types, including AutoCAD 2006's dynamic blocks, and all of the objects specific to the vertical market versions of AutoCAD, are partially documented, but not well enough to allow other developers to support them. For these reasons many CAD applications use the DWG format which can be licensed from Autodesk or non-natively from the Open Design Alliance.

The DXF (Drawing Exchange Format) is a widely used file format developed by Autodesk for representing two-dimensional and three-dimensional drawings and models. DXF files are primarily used in computer-aided design (CAD) applications to exchange data between different software programs and to facilitate interoperability among various design and drafting tools.

Key features and characteristics of DXF files include:

- **ASCII and Binary Formats:** DXF files can be stored in both ASCII (text-based) and binary formats. The ASCII format is human-readable and can be edited with a text editor, while the binary format is more compact and efficient for storage.
- **Geometry:** DXF files contain information about geometric entities such as points, lines, arcs, circles, polygons, and 3D objects like solids and surfaces.
- **Layers:** DXF files often utilize layers to organize and categorize different elements of the drawing. Layers can have attributes like color, linetype, and visibility.
- **Attributes:** DXF files can include attribute data associated with different entities, allowing for the storage of additional information.
- **Block Definitions and Insertions:** DXF files support the creation of block definitions, which are reusable groups of entities. These blocks can be inserted multiple times in the drawing.
- **Annotations:** DXF files can include annotations such as text, dimensions, and hatches to provide additional information and context to the drawing.
- **3D Data:** DXF supports 3D entities, allowing for the representation of three-dimensional objects and structures.

- **Compatibility:** DXF is widely supported by various CAD software applications, making it a popular choice for sharing drawings and models between different tools.
- **Industry Standard:** DXF is recognized as an industry-standard format for exchanging CAD data.

DXF files are used for various purposes in the field of design and engineering:

- **CAD Software:** DXF files are commonly used to exchange drawings and models between different CAD software applications, especially when collaborating on projects involving multiple design tools.
- **Engineering and Architecture:** DXF files are used by engineers and architects to share designs, blueprints, and technical drawings.
- **Manufacturing:** DXF files can be used in manufacturing processes to create prototypes, molds, and other components.
- **GIS Applications:** DXF files can also be used in geographic information systems (GIS) to represent map data and spatial information.

DXF files can be opened and edited by various CAD software, including Autodesk's AutoCAD, DraftSight, and other third-party applications. Keep in mind that while DXF is widely supported, there may be version compatibility considerations between different software versions.

1.2.13.3 Creating a PDF File

To create/export to a PDF file click the Tools→Export→ 

button.

PDF export dialog

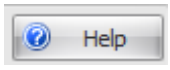
All Schematic - Click to add all schematic diagrams to the PDF file.

PCB - Click to add all schematic diagrams to the PDF file.

Layers - Click to a separate page for each layer to the PDF file.

Top View - Click to add a view of the top of the PCB to the PDF file.

Bottom View - Click to add a view of the bottom of the PCB to the PDF file.



Click to show this help topic.

Portable Document Format (PDF) is a file format used to represent documents in a manner independent of application software, hardware, and operating systems. Each PDF file encapsulates a complete description of a fixed-layout flat document,

including the text, fonts, graphics, and other information needed to display it. In 1991, Adobe Systems co-founder John Warnock outlined a system called "Camelot" that evolved into PDF.

While Adobe Systems made the PDF specification available free of charge in 1993, PDF remained a proprietary format, controlled by Adobe, until it was officially released as an open standard on July 1, 2008, and published by the International Organization for Standardization as ISO 32000-1:2008. In 2008, Adobe published a Public Patent License to ISO 32000-1 granting royalty-free rights for all patents owned by Adobe that are necessary to make, use, sell and distribute PDF compliant implementations.

1.2.13.4 Creating a IDF File

To export to an IDF file click the Tools→Export→ 

button.

The IDF (Intermediate Data Format) is a file format commonly used in the field of electronics and printed circuit board (PCB) design. It's primarily utilized to exchange layout and design information between different PCB design software tools. IDF files facilitate collaboration and interoperability between various design tools, allowing for seamless data exchange without the need for proprietary file formats.

Key features and uses of IDF files include:

PCB Collaboration: IDF files are used to transfer PCB layout and design data from one design tool to another, enabling collaboration between different teams using different software tools.

Synchronization: IDF files ensure that changes made in one software tool can be accurately reflected in another, reducing the need for manual data entry and avoiding errors.

Bill of Materials (BOM) Integration: IDF files often include information about components, their placements, orientations, and connections. This data is crucial for generating accurate BOMs and facilitating assembly processes.

Connector and Component Placement: IDF files allow the precise positioning of connectors and components on the PCB layout. This information ensures that mechanical aspects, such as enclosure fit, can be accurately coordinated.

3D Visualization: Some design tools support the visualization of the 3D representation of components and the PCB itself, helping designers ensure proper clearance, fit, and alignment.

Collaboration with Mechanical Design Tools: IDF files enable collaboration between electronics and mechanical design teams. This is particularly useful when integrating PCBs into larger systems and enclosures.

XML-Based: IDF files are often written in XML format, making them human-readable and allowing for easy integration with software applications.

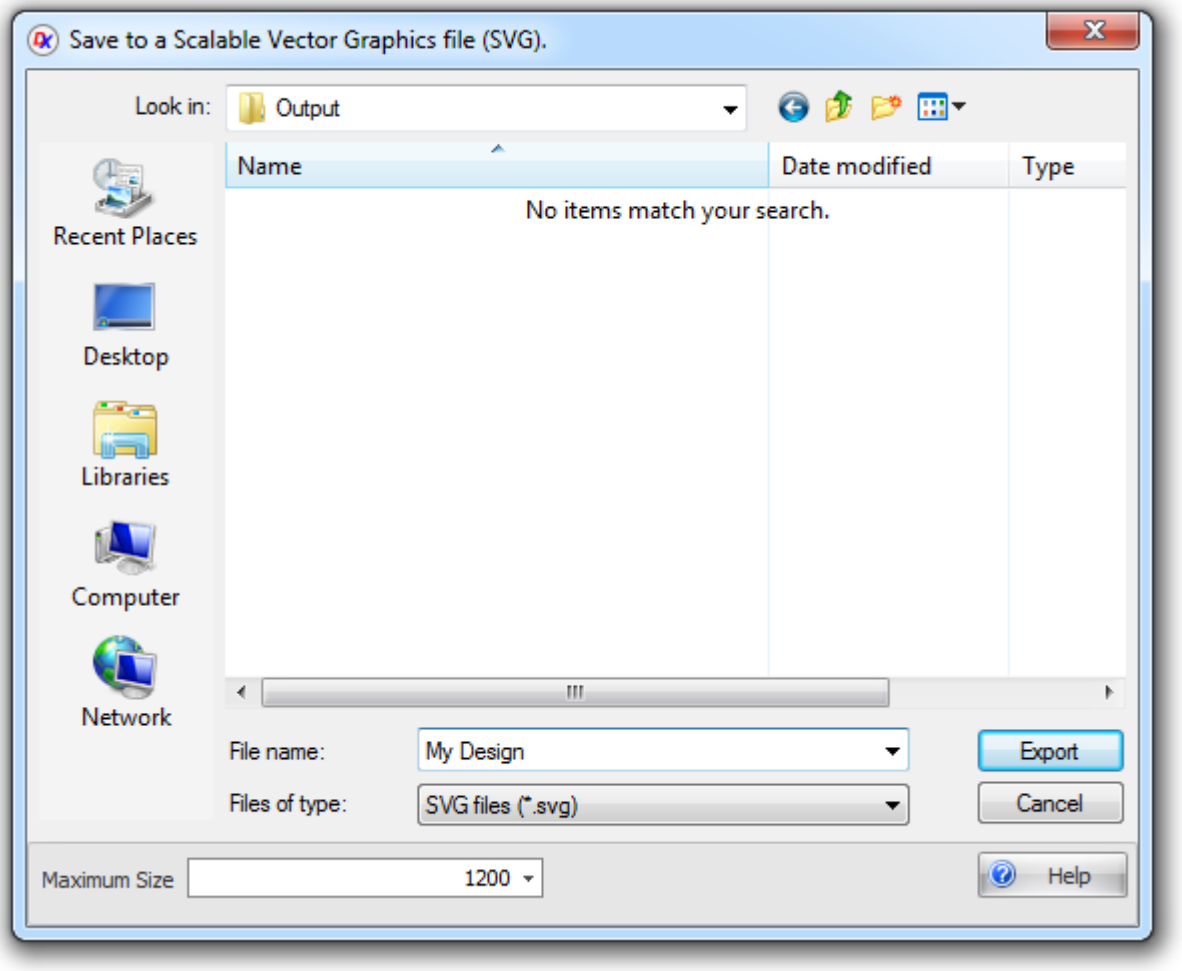
It's important to note that the specific structure and content of IDF files can vary depending on the PCB design software tools being used. Different software vendors may have their own interpretations of the IDF format, and as such, it's recommended to refer to the documentation of the specific tools involved to understand the details of IDF file usage.

IDF files play a crucial role in streamlining the PCB design process, enabling efficient collaboration between different teams, reducing errors, and ensuring accurate representation of designs across various tools and software platforms.

1.2.13.5 Creating a SVG File

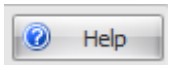
To create/export to a SVG file click the Tools→Export→ 

button.



SVG export dialog

Maximum Size - Enter the maximum dimension of the image in pixels. The other dimension will be set by the aspect ratio of the contents.



Click to show this help topic.

All major modern web browsers—including Mozilla Firefox, Internet Explorer 9 and 10, Google Chrome, Opera, and Safari—have at least some degree of support for SVG and can render the markup directly.

SVG (Scalable Vector Graphics) is a widely used XML-based vector image format that is primarily used for displaying two-dimensional graphics and animations on the web. Unlike raster image formats like JPEG or PNG, SVG images are resolution-independent and can be scaled without losing quality. SVG files describe images using mathematical shapes and text rather than a grid of pixels, making them ideal for logos, icons, illustrations, and other graphics that need to maintain their clarity at various sizes.

Key features of SVG files include:

- **Vector Graphics:** SVG images are defined using geometric shapes such as lines, curves, and polygons, allowing them to be scaled up or down without loss of quality.
- **XML-Based:** SVG files are written in XML format, which means they are human-readable and can be easily edited using a text editor or specialized software.
- **Scalability:** SVG images can be resized without loss of detail. This makes them suitable for responsive web design, where graphics need to adapt to different screen sizes.
- Support for Styling: SVG supports a range of styling options including colors, gradients, transparency, and custom CSS properties.
- **Text Support:** SVG files can include text elements, allowing them to display text along with graphical elements.
- **Interactive and Animatable:** SVG images can include interactivity and animations using CSS and JavaScript.
- **Web Compatibility:** SVG is supported by all modern web browsers, making it a popular choice for web-based graphics.

SVG files are used for various purposes:

- **Web Graphics:** SVG is commonly used for creating graphics on websites, such as logos, icons, buttons, and illustrations.
- **Data Visualization:** SVG is used to create dynamic and interactive charts, graphs, and data visualizations.
- **Icons:** SVG icons are widely used due to their ability to be scaled while maintaining crisp edges.
- **Animations:** SVG animations are used for creating simple animations and transitions on web pages.
- **Print Design:** SVG files can be used in print design, though they might require conversion to other formats like PDF for high-quality printing.
- **Accessible Graphics:** SVG images can be used to create graphics that are accessible to users with disabilities.

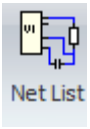
When using SVG files, keep in mind that complex images with numerous intricate details might increase file size, affecting load times on web pages. Additionally, while SVG files are great for most types of graphics, they might not be suitable for images that require high levels of photorealism or complex textures.

1.2.13.6 Creating a Spice Deck File

To create/export to a SPICE deck file click the Tools→Export→ 

button.

1.2.13.7 Creating a Net List File

To create/export to a Netlist file click the Tools→Export→ 

button.

A "netlist" describes the connectivity of your electronic design.

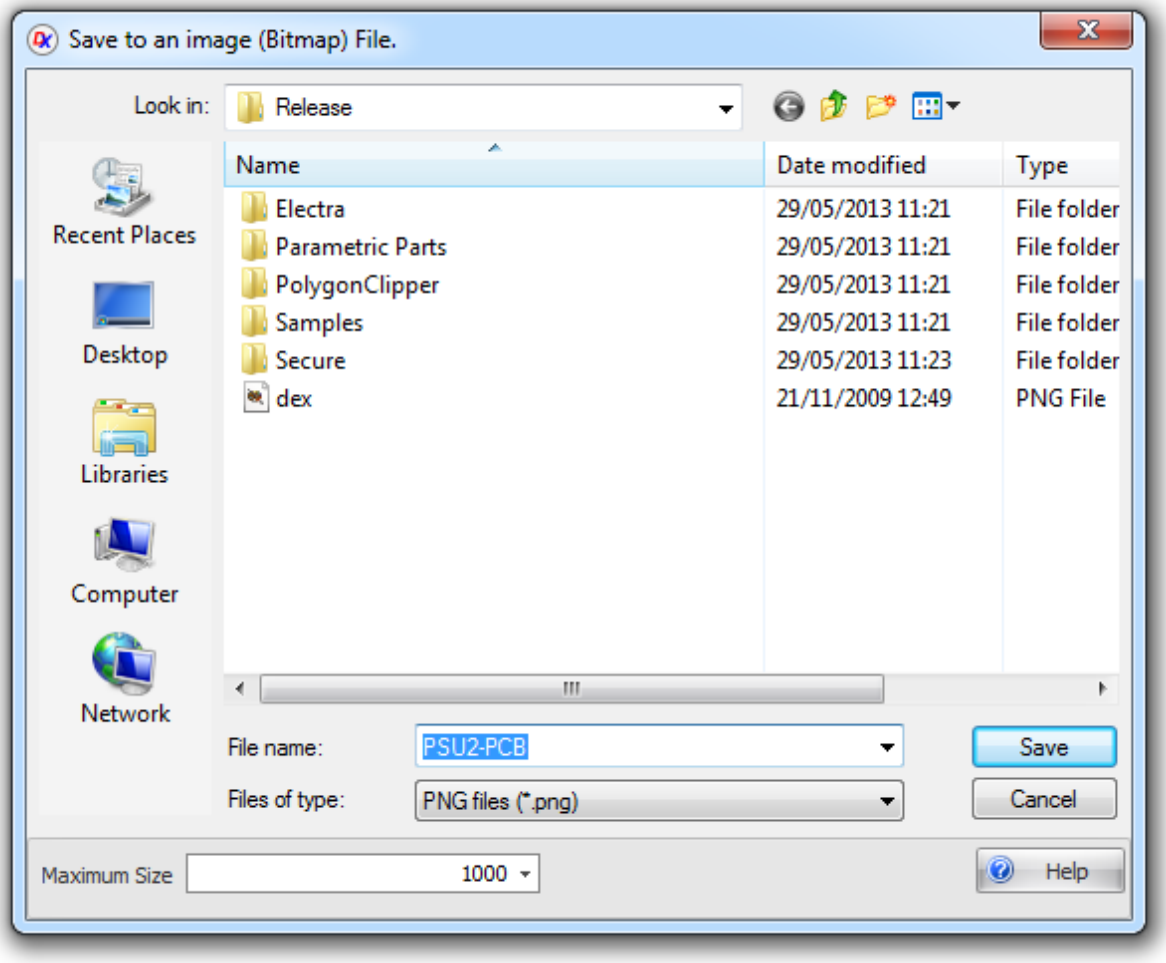
1.2.13.8 Saving to an Image File

To save the current view to an image file click the Tools→Export→ 

button.

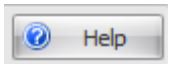
Maximum Size

Enter the maximum pixel/resolution size. The image saved is the same aspect ratio of the current viewport. So, to change the viewport aspect ratio, re-size the viewport.



Save to Image Dialog

Maximum Size - Enter the maximum dimension of the image in pixels. The other dimension will be set by the aspect ratio of the current viewport.



Click to show this help topic.

You can save to the following formats:

BMP

The BMP file format, also known as bitmap image file or device independent bitmap (DIB) file format or simply a bitmap, is a raster graphics image file format used to store bitmap digital images, independently of the display device (such as a graphics adapter), especially on Microsoft Windows and OS/2 operating systems.

EMF

In 1993, the 32-bit version of Win32/GDI introduced the Enhanced Metafile (EMF), a newer version with additional commands. EMF is also used as a graphics language

for printer drivers. Microsoft recommends that "Windows-format" (WMF) functions only "rarely" be used and "enhanced-format" (EMF) functions be used instead.

EXIF

Exchangeable image file format (Exif) is a standard that specifies the formats for images, sound, and ancillary tags used by digital cameras (including smart phones), scanners and other systems handling image and sound files recorded by digital cameras

GIF

The Graphics Interchange Format (GIF) is a bitmap image format that was introduced by CompuServe in 1987 and has since come into widespread usage on the World Wide Web due to its wide support and portability. The format supports up to 8 bits per pixel thus allowing a single image to reference a palette of up to 256 distinct colors. The colors are chosen from the 24-bit RGB color space. It also supports animations and allows a separate palette of 256 colors for each frame. The color limitation makes the GIF format unsuitable for reproducing color photographs and other images with continuous color, but it is well-suited for simpler images such as graphics or logos with solid areas of color.

GIF images are compressed using the Lempel-Ziv-Welch (LZW) lossless data compression technique to reduce the file size without degrading the visual quality. This compression technique was patented in 1985. Controversy over the licensing agreement between the patent holder, Unisys, and CompuServe in 1994 spurred the development of the Portable Network Graphics (PNG) standard. All the relevant patents have now expired.

PNG

Portable Network Graphics (PNG) is a raster graphics file format that supports lossless data compression. PNG was created as an improved, non-patented replacement for Graphics Interchange Format (GIF), and is the most used lossless image compression format on the World Wide Web. The motivation for creating the PNG format was in early 1995, after it became known that the Lempel–Ziv–Welch (LZW) data compression algorithm used in the Graphics Interchange Format (GIF) format was patented by Unisys. There were also other problems with the GIF format that made a replacement desirable, notably its limit of 256 colors at a time when computers able to display far more than 256 colors were growing common. Although GIF allows for animation, it was decided that PNG should be a single-image format.

JPG

JPEG is a commonly used method of lossy compression for digital photography (image). The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality. JPEG typically achieves 10:1 compression with little perceptible loss in image quality.

TIFF

TIFF (originally standing for Tagged Image File Format) is a file format for storing images, popular among graphic artists, the publishing industry,]and both amateur and professional photographers in general. As of 2009, it is under the control of Adobe Systems. Originally created by the company Aldus for use with "desktop publishing", the TIFF format is widely supported by image-manipulation applications, by publishing and page layout applications, by scanning, faxing, word processing, optical character recognition and other applications.[3] Adobe Systems, which acquired Aldus, now holds the copyright to the TIFF specification. TIFF has not had a major update since 1992.

WMF

Windows Metafile (WMF) is an image file format originally designed for Microsoft Windows in the 1990s. Windows Metafiles are intended to be portable between applications and may contain both vector graphics and bitmap components. It acts in a similar manner to SVG files.

1.2.13.9 Exporting 3D to Collada

You can export your PCB as a 3D model to a Collada format file.

Collada files can be imported by Autodesk 3D Studio MAX, Blender and Sketchup.

Collada (COLLAborative Design Activity) is an open standard XML-based file format designed for exchanging digital assets between different 3D modeling and computer graphics software applications. COLLADA files (.dae) are used to represent 3D models, scenes, animations, textures, and other related information. The format is widely used for sharing 3D content across various applications and platforms.

Key features of Collada 3D files include:

Open Standard: COLLADA is an open and vendor-neutral standard developed by the Khronos Group, which promotes open standards in the graphics industry.

XML-Based: COLLADA files are based on XML (eXtensible Markup Language), making them human-readable and easy to parse by software applications.

Content Hierarchy: COLLADA supports a hierarchical structure for scenes, objects, transformations, cameras, lights, materials, animations, and more.

Geometry: COLLADA files can store 3D geometry, including vertices, normals, texture coordinates, and more.

Materials and Textures: The format supports material definitions, textures, shaders, and other rendering properties.

Animations: COLLADA supports skeletal animations, skinning, morph targets, and other animation techniques.

Physics: The format can include physics-related information, such as rigid body dynamics, constraints, and collisions.

Platform and Application Agnostic: COLLADA is designed to work across different 3D applications and platforms, allowing for easier collaboration and interoperability.

Extensions: Extensions can be added to the format to accommodate specific features or optimizations required by certain applications.

Library and Tools: COLLADA files can be generated, imported, and exported using various 3D modeling software and libraries.

The COLLADA format is particularly useful for content creators who work with multiple software applications and platforms. It allows users to create content in one application and then export it to be used in another without losing critical information. COLLADA has gained popularity in the gaming industry, virtual reality, computer-aided design (CAD), and other fields where 3D content needs to be shared across different tools and pipelines.

It's important to note that while COLLADA is a versatile format, compatibility and feature support can vary between different software applications and versions. When working with COLLADA files, ensure that your tools support the features you need and that you're aware of any potential issues with data loss or format inconsistencies when transferring content between applications.

1.2.13.10 Creating a STL File

To create/export to a STL file click the Tools→Export→  button.

A STL file contains a description of the PCB in 3D and is suitable for 3D printing.

STL (Stereolithography) is a file format commonly used for representing 3D models in the field of 3D printing, computer-aided design (CAD), and computer graphics. STL files are used to describe the surface geometry of a 3D object using a mesh of triangles. This format is widely supported by various 3D modeling software, slicing software for 3D printing, and other applications that work with 3D models.

Key features of STL files include:

- **Triangle Mesh Representation:** An STL file describes a 3D model's surface using a collection of triangular facets. Each facet is defined by three vertices and a normal vector indicating the orientation of the triangle.

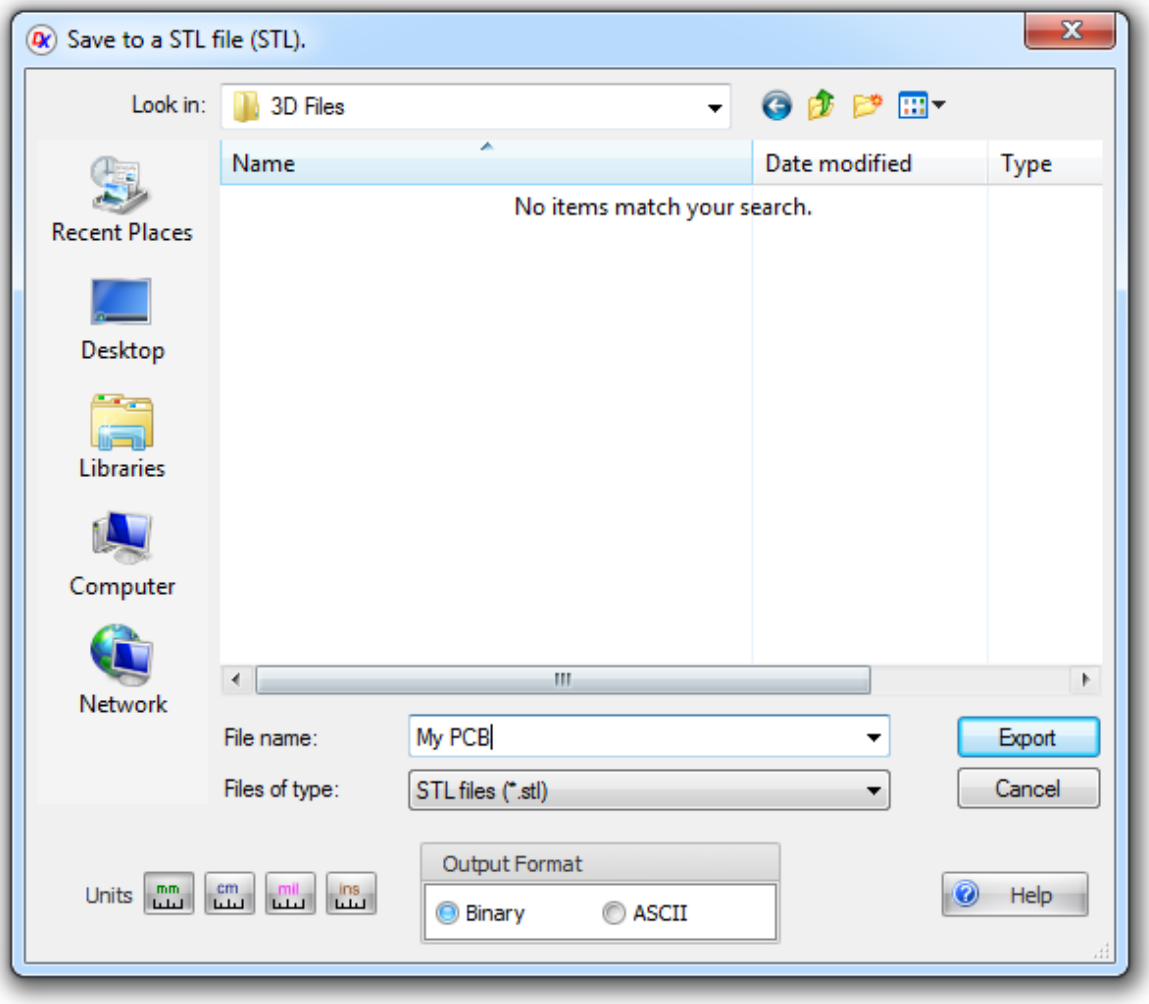
- **Binary and ASCII Formats:** STL files come in two formats: binary and ASCII. Binary STL files store the triangle data in a compact binary format, making them more efficient for storage and transmission. ASCII STL files use plain text to represent the triangle data, making them human-readable but larger in size.
- **Facet Normal:** Each facet in an STL file includes a normal vector that defines the orientation of the triangle. This information is essential for rendering and printing.
- **Vertex Coordinates:** For each facet, the coordinates of its three vertices are provided. These vertices define the shape and structure of the 3D model.
- **Color and Texture Information:** STL files primarily focus on geometry and don't inherently store color, texture, or material information. Color and texture details are usually added using external methods or in applications that support additional metadata.

STL files are commonly used in various applications:

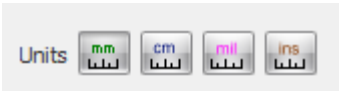
- **3D Printing:** STL files are the standard format for 3D printers. Slicing software converts the model's triangles into layers suitable for printing.
- **CAD Software:** STL files are widely supported by CAD software for exporting 3D models. They allow engineers and designers to share their designs with colleagues and partners.
- **Computer Graphics:** STL files are used in computer graphics applications for visualization, simulations, and animations.
- **Rapid Prototyping:** STL files are essential for creating prototypes using rapid prototyping techniques like 3D printing.

It's important to note that while STL files are great for representing 3D geometry, they may lack certain information like color, texture, and assembly structure. Other formats like COLLADA or OBJ may be better suited for scenarios that require more comprehensive model information.

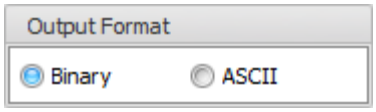
The Save to STL dialog is shown below.



Save to STL Dialog



Select your output units. **NOTE:** The STL files do not contain unit definitions so you will need to tell the person using the STL file what units they are in. It is common practice to use units of mm.



You can create a binary file or human readable ASCII text file. The binary file will be much smaller and will be easier to transit electronically over the Internet.

STL files have no color.

1.3 The User Interface

The AutoTRAX DEX workspace has had years to develop into a fine tuned working environment, and with bigger monitors and faster processors, working with AutoTRAX DEX has only become more fun. With all the room that larger display options give you, you can easily organize the panels, viewports and tools in the workspace to provide an efficient designing environment.

What is DPI and why does it matter?

DPI stands for dots per inch, where a dot represents a physical device pixel. (The nomenclature comes from printing, where dots are the smallest ink dot that a printing process can produce). HDPI stands for high dots per inch.

Historically, monitors shipped with 96 pixels per inch. The Windows operating system drew a bitmap that represented 96 DPI as 100%. But as display technology progressed, that DPI threshold was surpassed. Monitors started shipping with display panels close to 300 DPI or higher.

While higher pixel density produces sharper images, some sort of display scaling is required to size elements on the screen properly. Otherwise, user interface (UI) elements and text are too tiny to use effectively and can overlap. To help remedy this, Windows automatically scales the UI percentile to match the DPI setting. For example, a DPI setting of 100% represents 96 DPI, where 125% is 120 DPI, and 150% is 144 DPI. This automatic scaling affects text, graphics, controls, and window sizes.

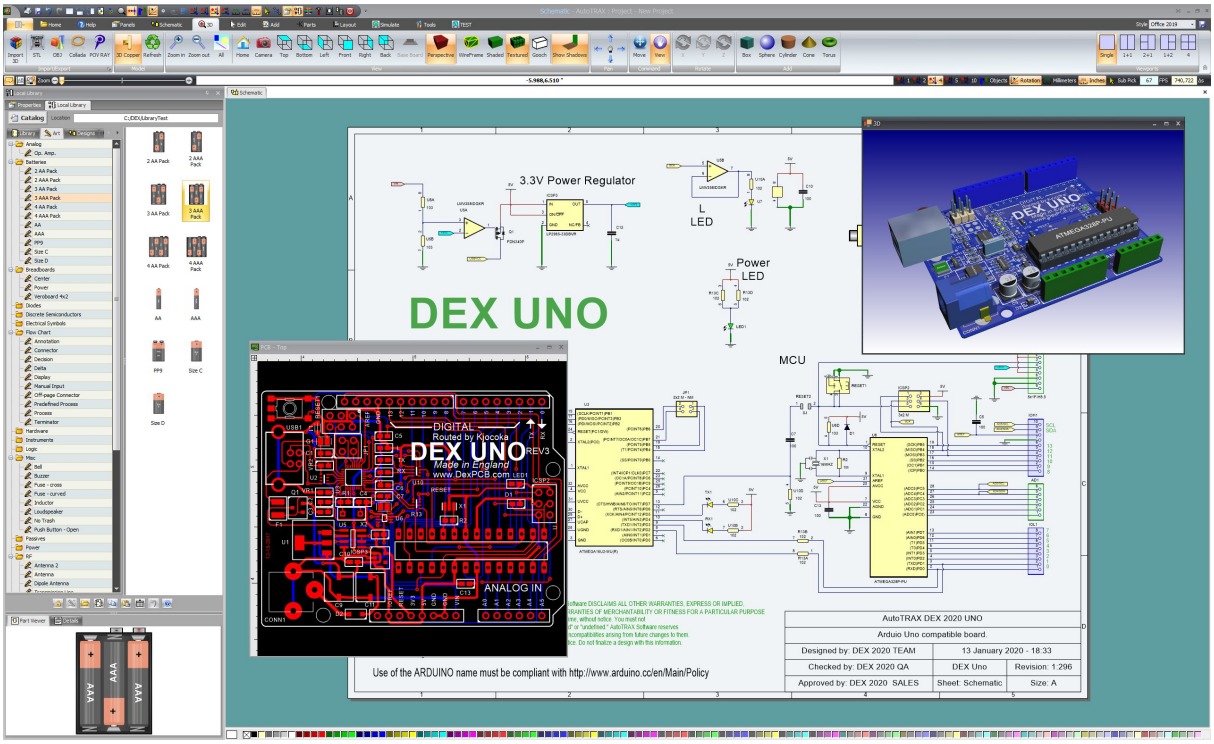
Here's where DPI-aware vs. DPI-unaware comes in. When an application declares itself to be DPI-aware, it's a statement specifying that the app behaves well at higher DPI settings and so Windows can apply auto-scaling. Conversely, DPI-unaware applications render at a fixed DPI value of 96 pixels per inch, or 100%, and so auto-scaling isn't applied.

AutoTRAX is now fully DPI-aware

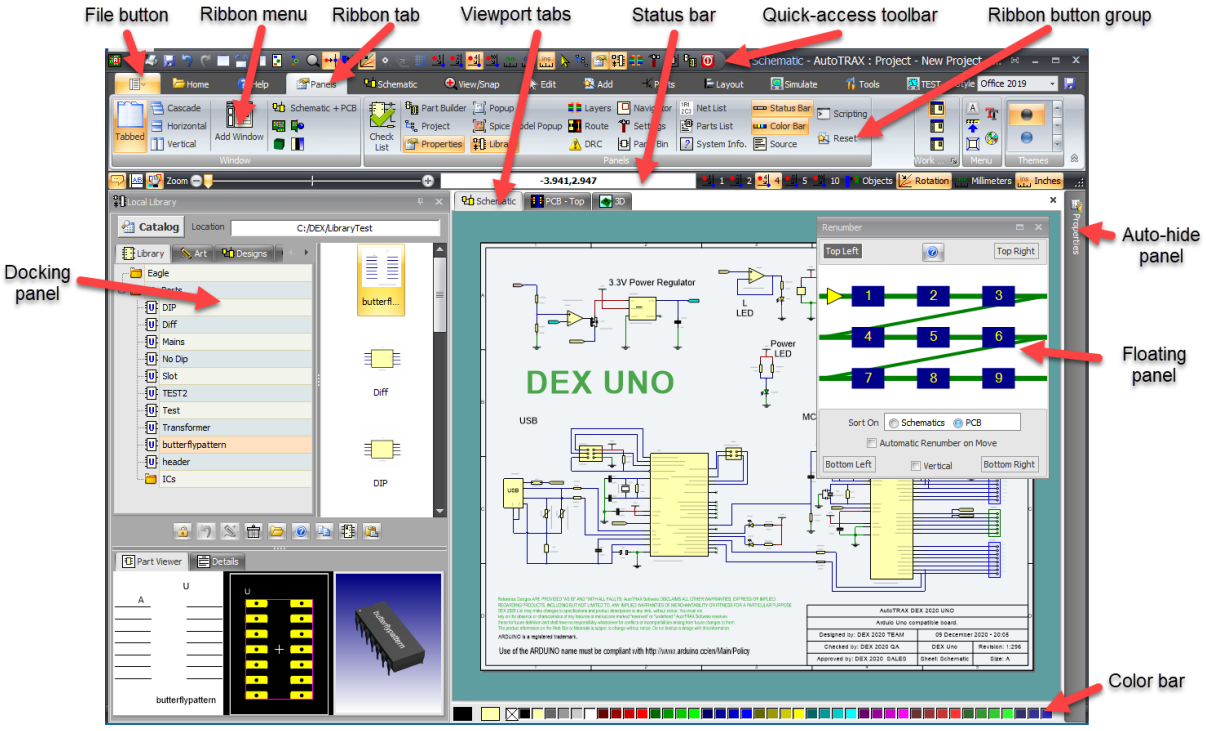
AutoTRAX DEX provides a consistent, single, highly configurable program interface to enable you to create parts, PCBs and even engineering artwork.

With all its features, AutoTRAX DEX can be a bit daunting at first. The purpose of this chapter is to familiarize you with the AutoTRAX DEX workspace and how to navigate around AutoTRAX DEX, find tools, customize settings, and set the working environment so it best suits you.

So, without further ado let's look at the AutoTRAX DEX workspace as shown below.



Typical Layout



Parts of aTypical Layout

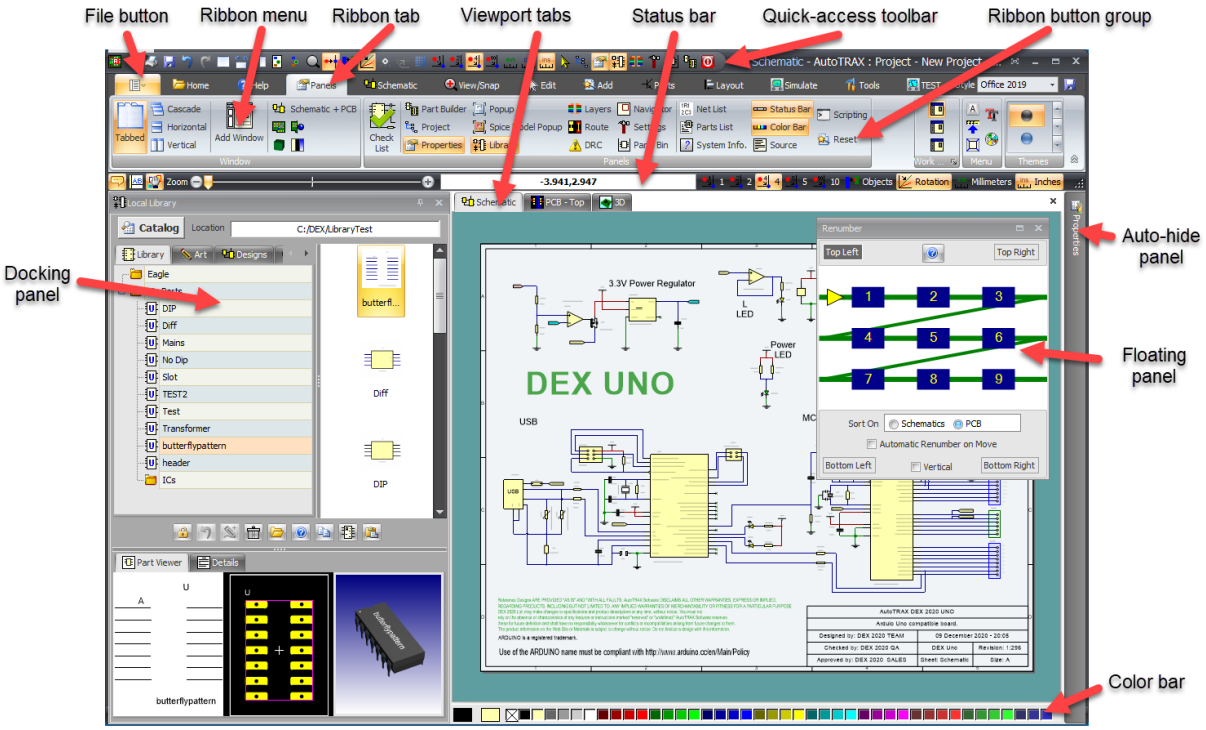
1.3.1 The Application Layout

The Ribbon Menu is displayed at the top of the AutoTRAX DEX main window as shown below.

The aim of the AutoTRAX DEX Ribbon is to enhance usability by consolidating AutoTRAX DEX's functions and commands in an easily recognizable place. You need not look through multiple levels of hierarchical menus, toolbars, or task panes before finding the right command.

The Ribbon consists of a pane that contains controls (such as buttons and icons) organized into a set of tabs, each one containing a grouping of relevant commands.

Some tabs, called contextual tabs, appear only after you select an object or a sheet. Contextual tabs expose functionality specific only to the object/sheet with focus.



File Button

Clicking this shows the file menu.

Ribbon Menu

This is the top menu which is similar to the ribbon menu you will see in Microsoft Office 2010 and later. It is a series of ribbon tabs which contain button groups. This is a modern substitute to the class drop down menu.

Ribbon Tab

A collection of ribbon button groups.

Ribbon Button Group

A collection of control buttons.

Quick Access Toolbar

At the top of AutoTRAX DEX is the Quick Access toolbar which contains a grouping of ribbon control buttons that you can configure.

Viewport Tabs

This area contains viewports in tabbed control. Each tab shows you a different sheet.

Viewport

Viewports are views of sheets. Sheets can be schematic sheet, symbol sheets, footprint sheet, PCB sheets or text document sheets.

Panels

Around the side of the viewports you can arrange Panels which are collections of controls to do a specific task.

Docking Panels

These are panels that are docked to the side of AutoTRAX DEX. Normally you will dock them to the left or the right of the AutoTRAX DEX main application window.

Floating Panels

These are panels that are free to move about the screen. You can even place then on different screens.

Auto-Hidden Panels

These are panels that automatically collapse and expand so you can see more of the viewports when you are working on them.

Status Bar

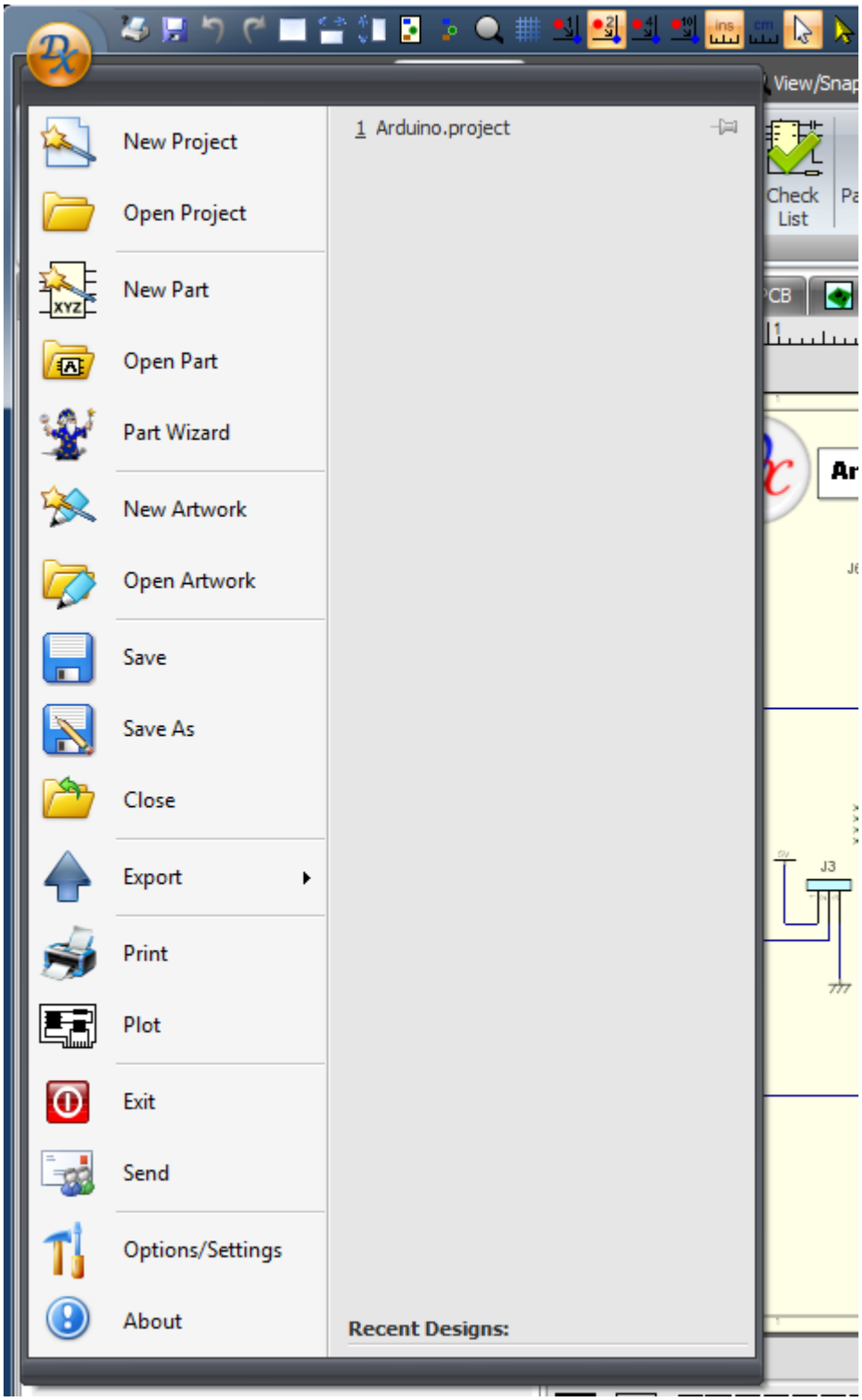
The status bar is always at the bottom of the AutoTRAX DEX window. It shows the status prompts and lets you zoom in and out.

The Color Bar

This is a bar at the bottom of the AutoTRAX DEX window that lets you set the fill and line color of selected objects.

1.3.1.1 The File Menu

The File menu contains a list of commands for creating, opening, plotting, printing and sending projects, parts and artwork. To show the file menu click on the  image at the top left of the AutoTRAX DEX main application window.



Recent Designs

At the right of the menu is the Recent Designs list. This displays recent designs on which you have been working. Click on any design to open it.

New Project

Click to create a new project.

Open Project

Click to open an existing project.

New Part

Click to create a new part.

Open Part

Click to create open an existing part.

Part Wizard

Click to start the Part Wizard

New Artwork

Click to create a new artwork.

Open Artwork

Click to open an existing project.

Save

Click to save the current design. If it has never been saved you will be prompted for a new name, this is the same as the Save As command below.

Save As

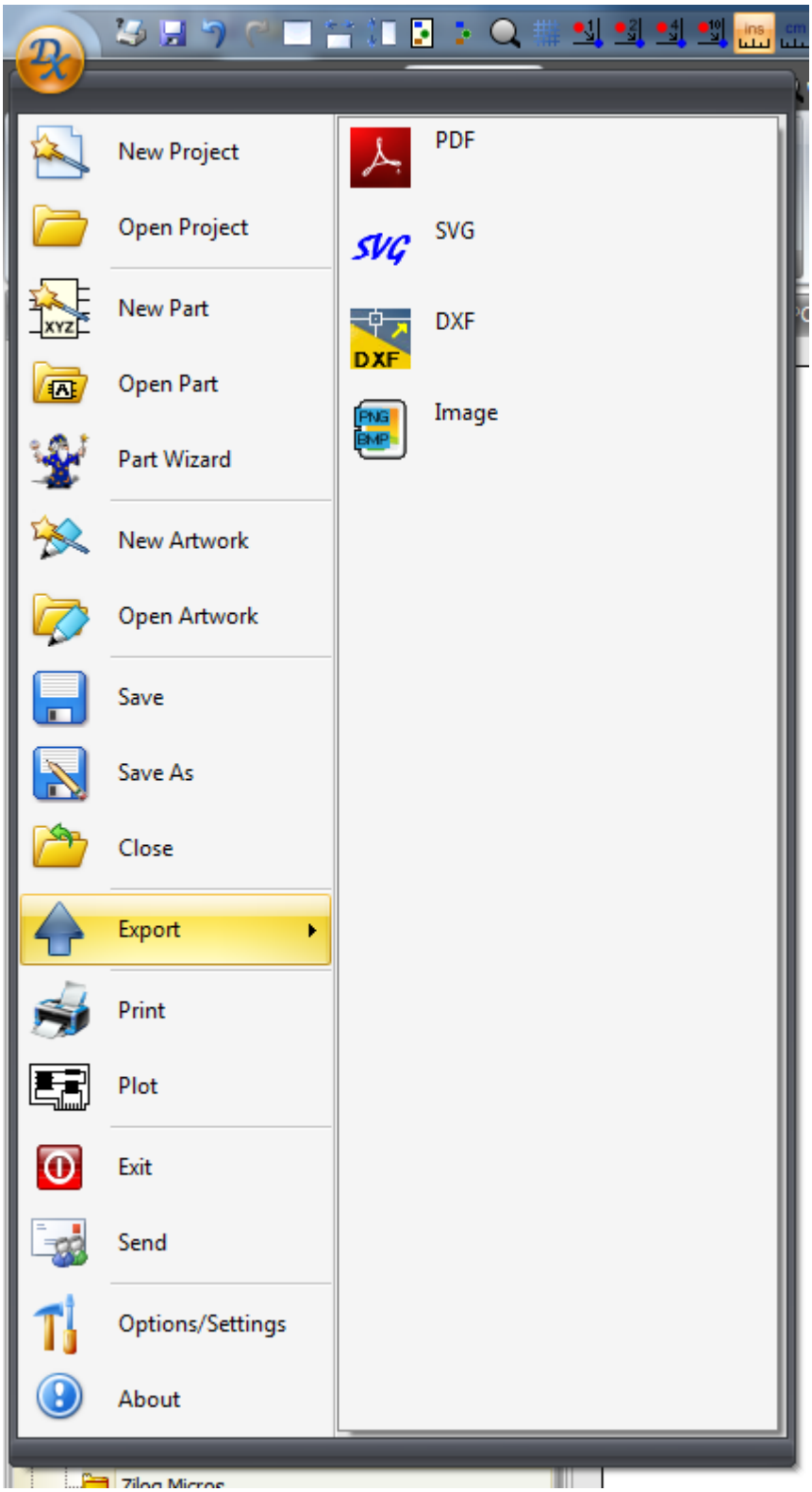
This will prompt you for a new file name.

Close

This will close the current design. If you have unsaved work, you will be prompted to save it.

Export

This will reveal a pop-out menu list of different formats to which you can save.



Print

This will display the print preview dialog.

Plot

This will display the plot preview dialog.

Exit

This will close AutoTRAX DEX. If you have unmodified work, you will be prompted to save it before AutoTRAX DEX closes.

Send

Click this to send the design as an email attachment to a work colleague. It will open a send email dialog for your email program.

Options/Settings

This will display the options and settings control.

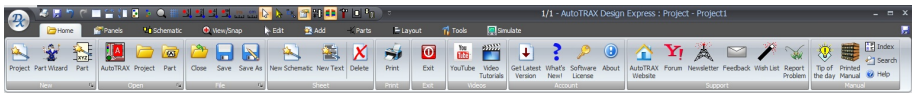
About

This will display the About dialog box that gives you details about the AutoTRAX DEX program and its version number.

1.3.1.2 Ribbon Menu

A ribbon menu is a portion of a graphical user interface where a set of toolbars are placed on tabs in a tab bar. Microsoft software released since 2007 have popularized a form of modular ribbon as their main interface, where large toolbars filled with graphical button and other controls are grouped by functionality. Ribbons use tabs to expose different sets of controls, eliminating the need for many parallel tool bars. Contextual tabs are tabs that appear only when a user needs them. For instance, in a word processor, an image-related tab may appear when the user selects an image in a document, allowing the user to interact with that image.

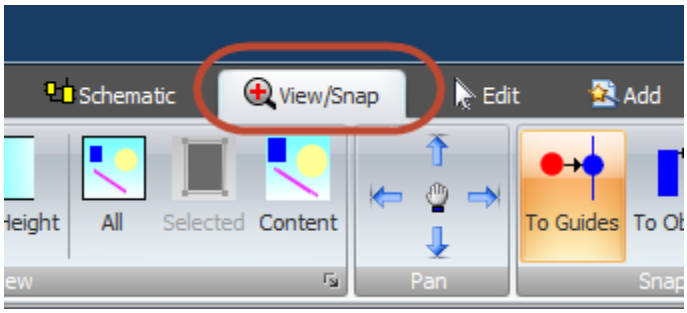
The ribbon menu consists of a collection of Ribbon Tabs with each tab containing several ribbon button groups which contain related command buttons and controls.



AutoTRAX DEX Ribbon Menu with Home Tab Selected

Ribbon Tab

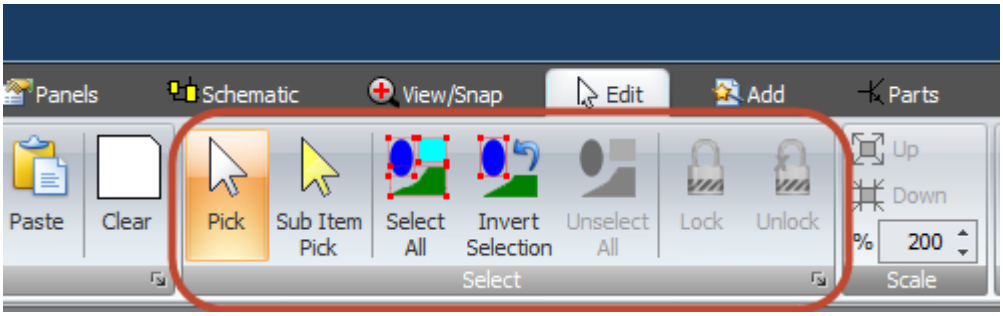
A ribbon tab is a collection of ribbon button groups. Clicking on a tab will show a collection of related button groups.



Ribbon Button Groups

A ribbon button group is a collection of buttons with a common border.

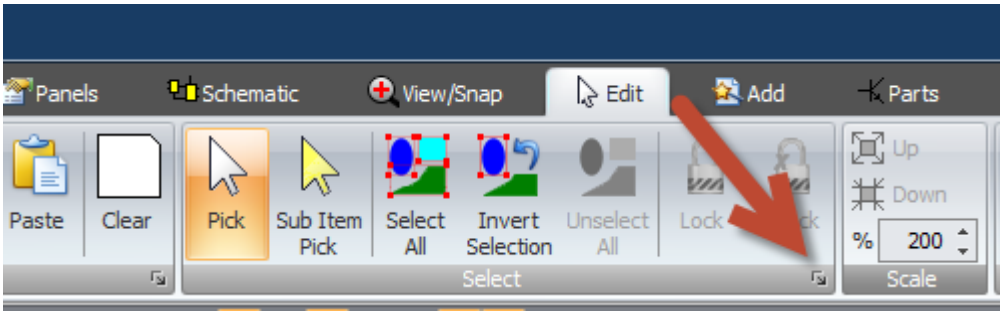
Below is a collection of buttons arranged together in a button group that is named 'Select'.



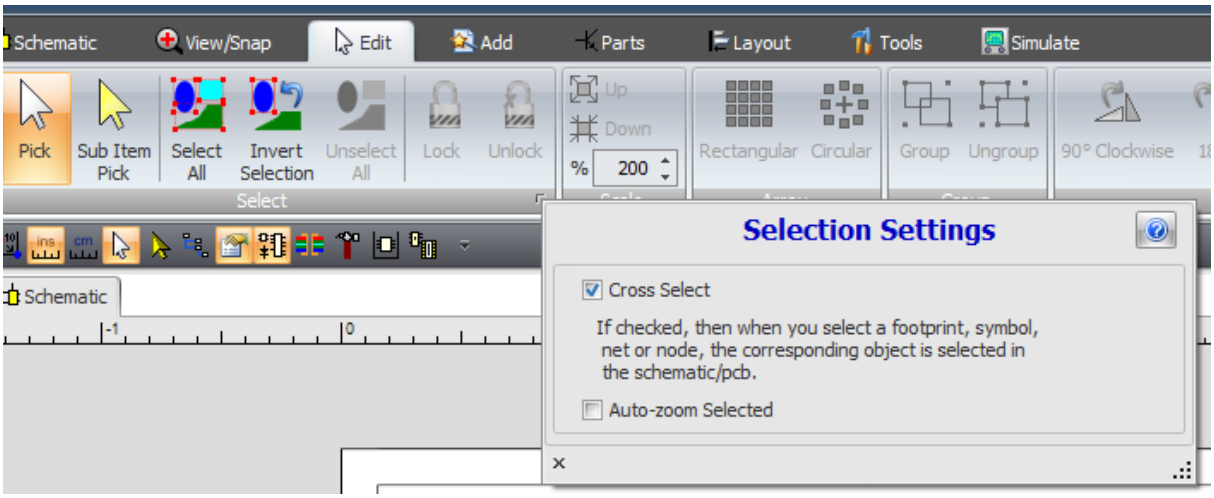
Ribbon Button Group

Ribbon Button Group Pop-up Menu

At the base of some button groups you will see a small button.  If you click on this a pop up dialog will appear that lets you set settings related to the commands in the button group.



Options Pop-up Button



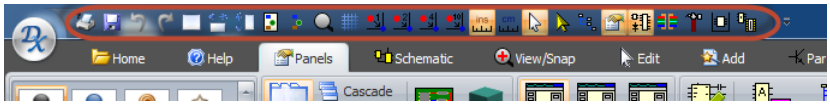
Pop-up dialog for the select button group

1.3.1.3 Quick Access Toolbar

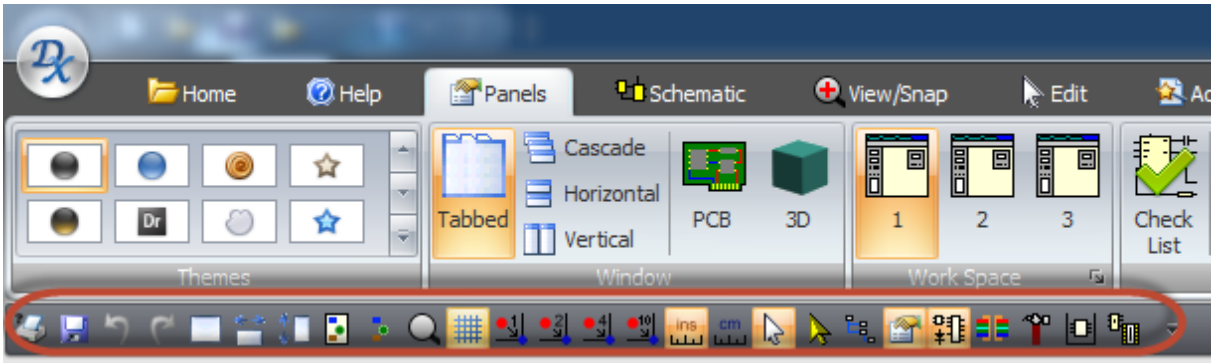
The Quick Access Toolbar is a customizable toolbar that contains a set of commands that are independent of the tab that is currently displayed. You can move the Quick Access Toolbar from one of the two possible locations, and you can add buttons that represent commands to the Quick Access Toolbar.

The Quick Access Toolbar can be located in one of two places:

Upper-left corner next to the AutoTRAX DEX Button



Below the Ribbon

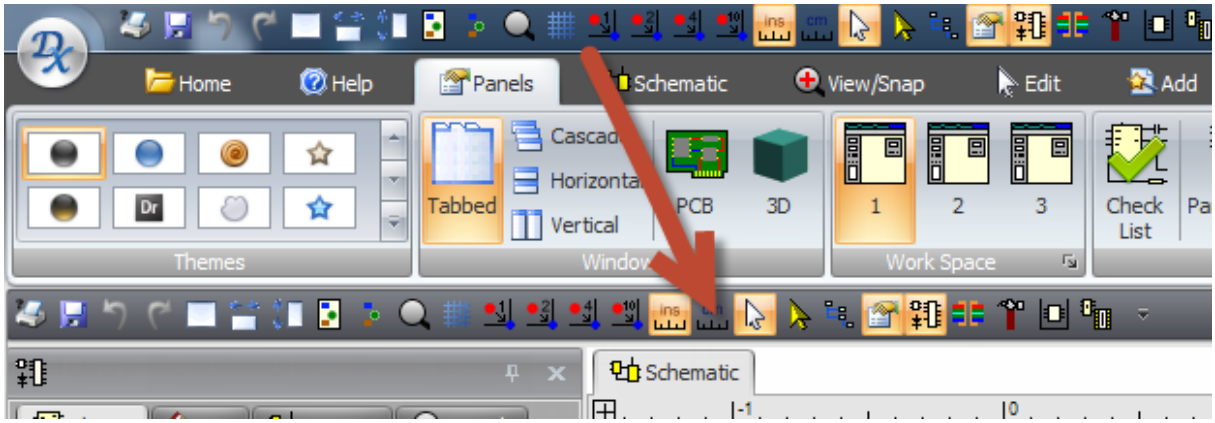


If you don't want the Quick Access Toolbar to be displayed in its current location, you can move it to another location. If you find that the default location next to the AutoTRAX DEX Button is too far from your work area to be convenient, you may want to move it closer to your work area. Since location below the Ribbon

encroaches on the work area, to maximize the work area, you may want to keep the Quick Access Toolbar in its default location.

Click Customize Quick Access Toolbar 

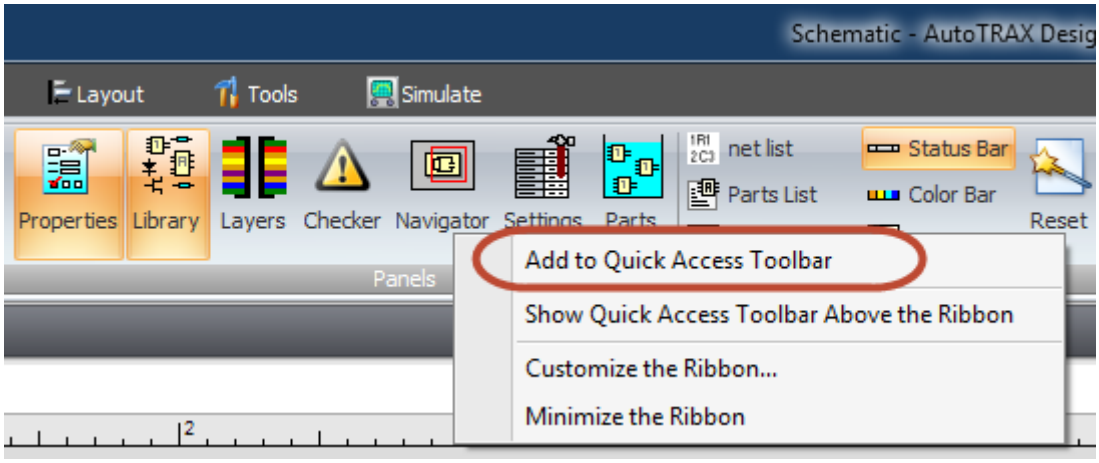
In the menu item list, click **Show Below the Ribbon** or **Show Above the Ribbon.**



Alternatively, you can ***right-click*** on any icon in the Quick Access Toolbar and select **Show Below the Ribbon or Show Above the Ribbon** from the shortcut menu.

Add a command to the Quick Access Toolbar

You can add a command to the Quick Access Toolbar directly from commands that are displayed on the Ribbon menu.

1. On the Ribbon, click the appropriate tab or group to display the command that you want to add to the Quick Access Toolbar.
2. Right-click the command, and then click Add to Quick Access Toolbar on the shortcut menu.
3. 

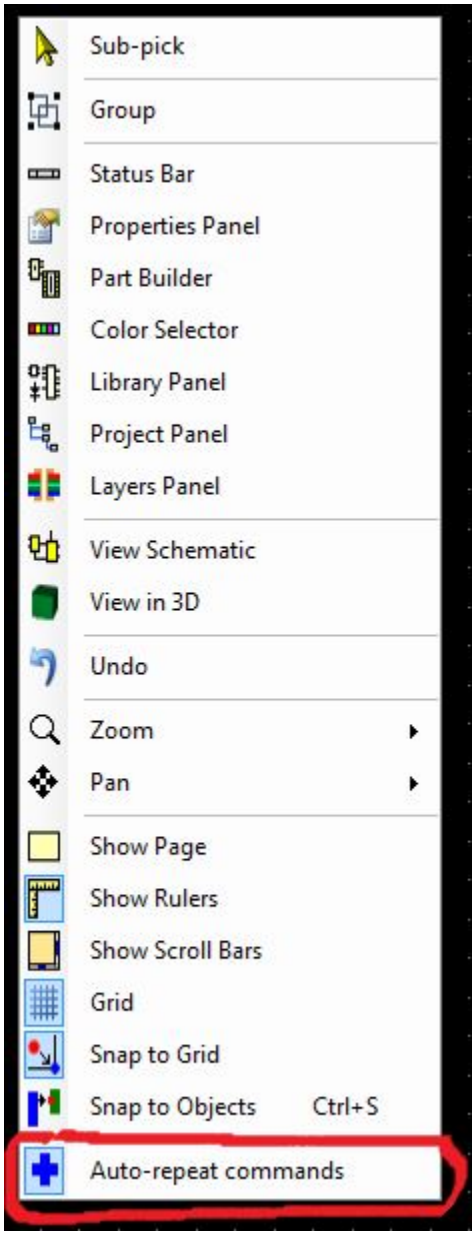
NOTES

- You cannot increase the size of the buttons representing the commands by an option in AutoTRAX DEX. The only way to increase the size of the buttons is to lower the screen resolution you use.
- You cannot display the Quick Access Toolbar on multiple lines.
- Only commands can be added to the Quick Access Toolbar. The contents of most lists, such as indent and spacing values and individual styles, which also appear on the Ribbon, cannot be added to the Quick Access Toolbar.

1.3.1.4 Auto-Repeat command

In AutoTRAX DEX, there are times when it is convenient to repeat the last command immediately. For example, you may want to place the same part on a graphical sheet more than once. The **Auto-Repeat commands** feature in AutoTRAX DEX is quite handy for this. When enabled, the last command completed will automatically be repeated. When this feature is enable to quit repeating the last command hit the ***ESC*** key.

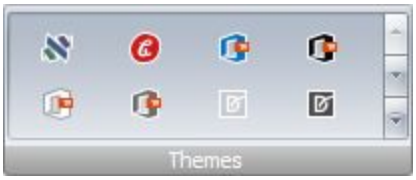
To enable/disable the **Auto-Repeat commands** feature, in any viewport, ***right-click*** and select **Auto-Repeat commands** from the context menu displayed. See below...



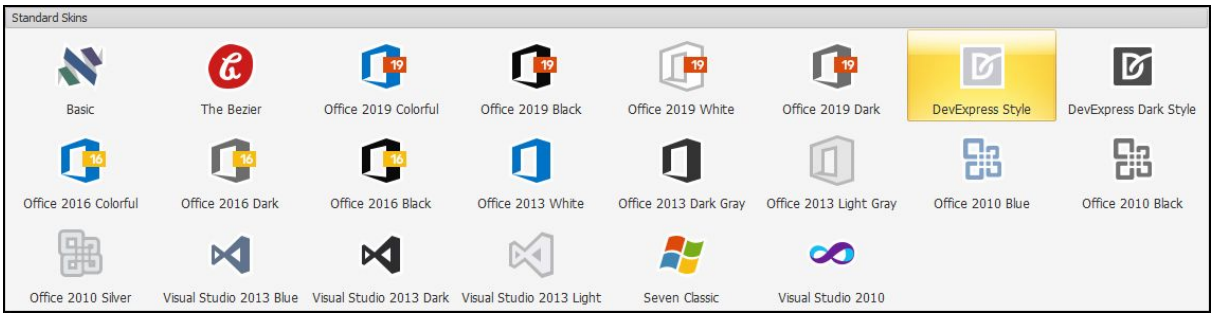
1.3.1.5 Themes

AutoTRAX DEX comes with many attractive themes (Skins) that you can choose to match your mode and add that bit of variety to your life.

The Skins are selected via the Panels→Skins menu. Select the theme from the Skin gallery.

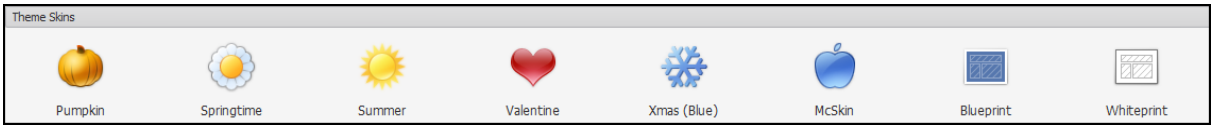


Themes ribbon button group

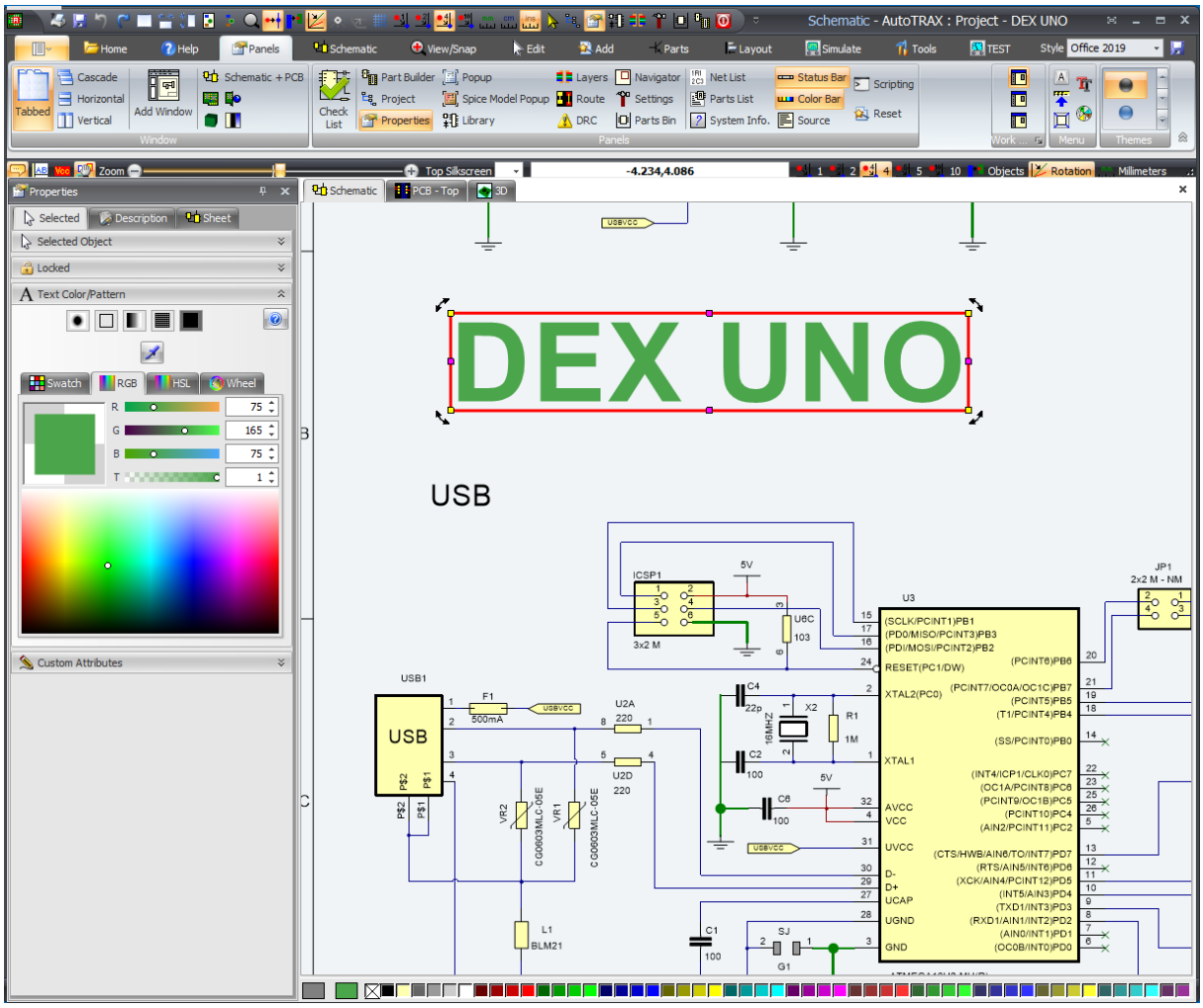


Standard Skins

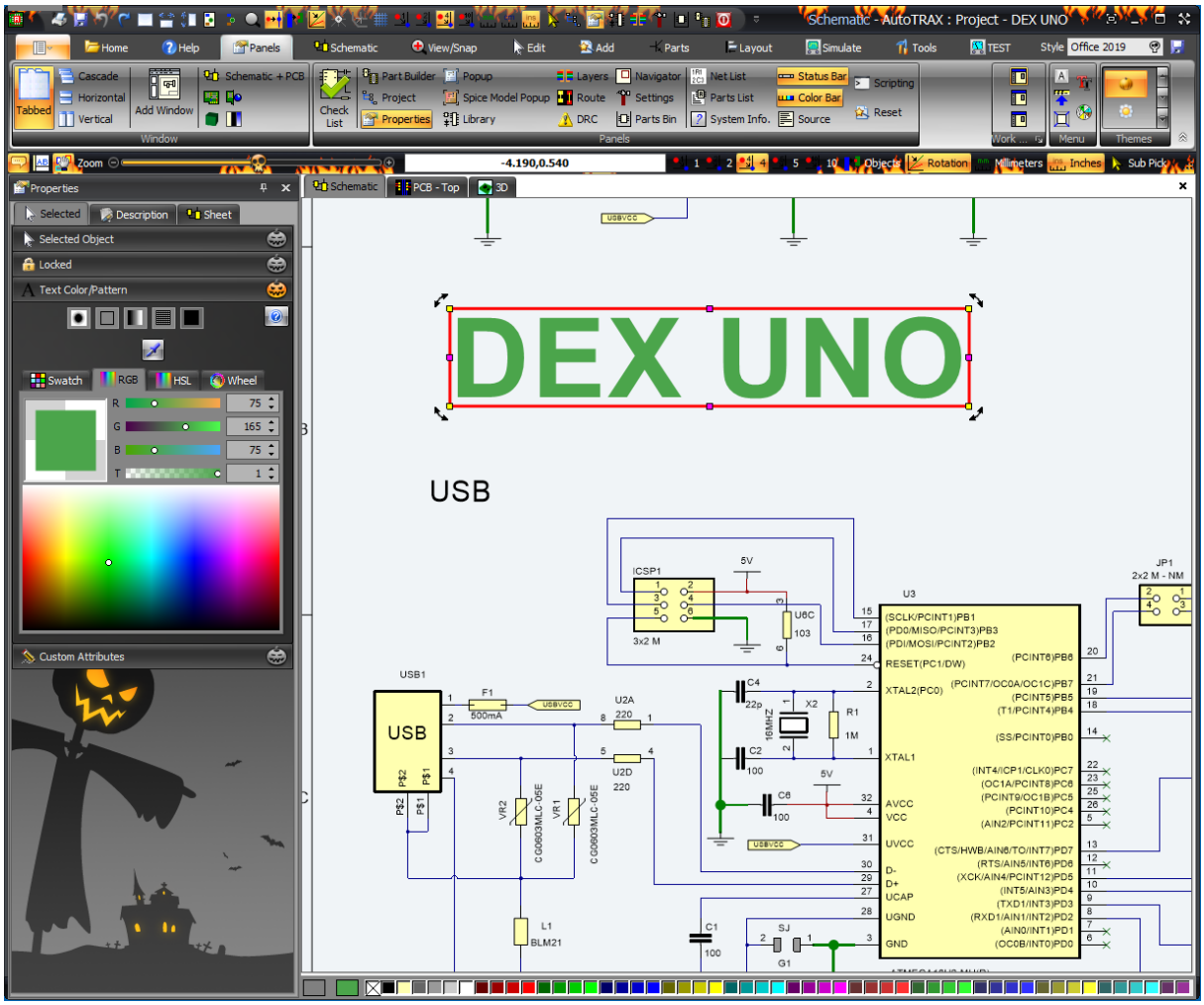
Bonus Skins



Theme Skins



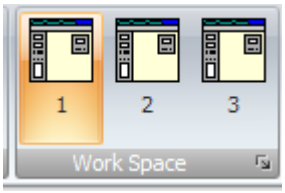
Default Theme



Halloween Theme

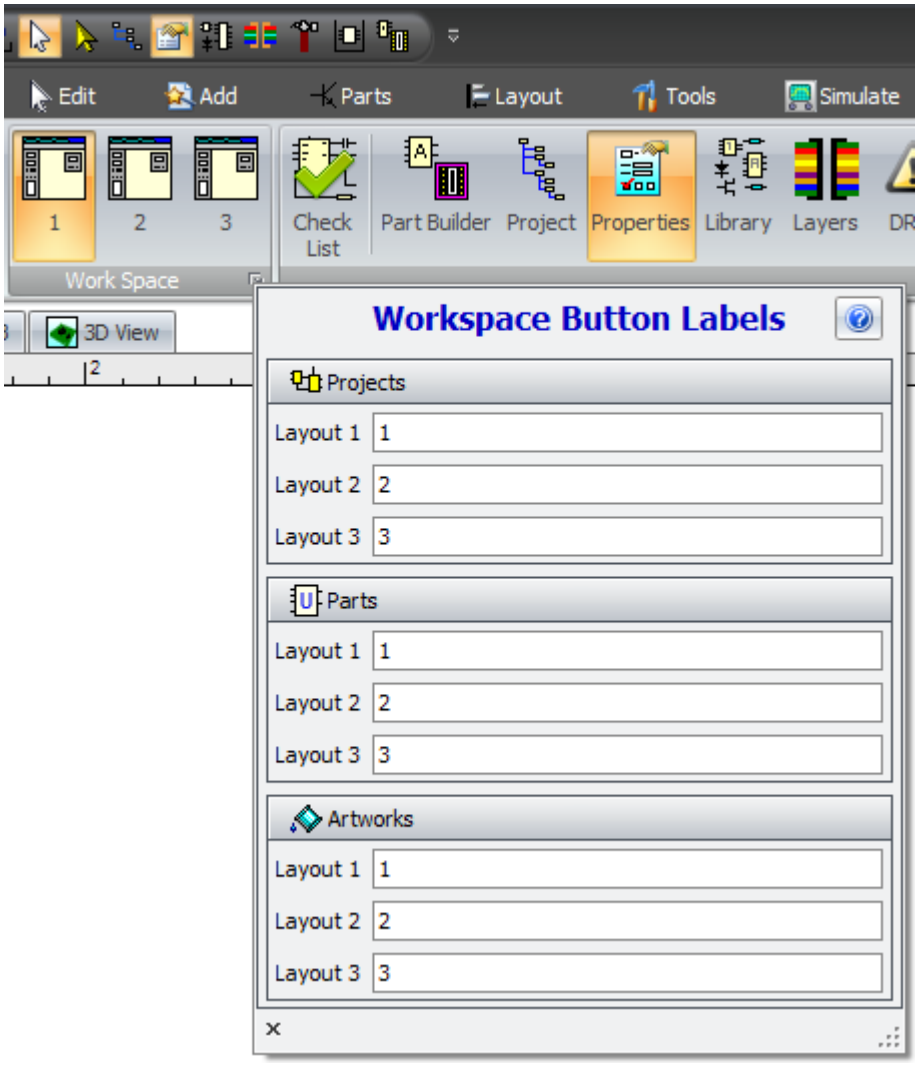
1.3.1.6 The Workspace Setttings

AutoTRAX DEX has 3 workspaces for each file type. When you click on any of the 3 work space buttons in the Panels ribbon tab, the layout of panels will be saved to the current work space and the panel layout for the clicked work space will be restored.



Workspace Buttons group

Click on the small button  at the bottom left of the workspace button group to display the workspace settings pop up dialog. You can set the text name for each workspace button.

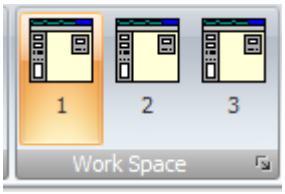


1.3.1.7 Workspaces

The AutoTRAX DEX workspace consists of:

- A top menu. This can be a Microsoft Office style ribbon menu or the class drop-down menu with tool bars.
- A viewport area. This can be either a tabbed collection of viewports or a Multiple Document Interface (MDI) displaying a collection of one or more viewports displaying schematics, PCBs, 3D views and text documents.
- An optional status bar at the base of the viewport area.

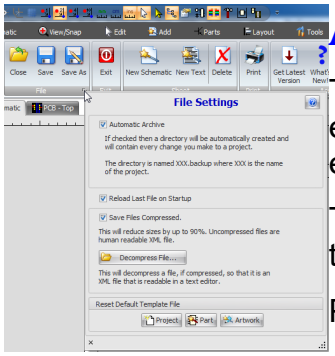
- An optional color chooser, again at the base of the viewport area.
- One of several optional dockable/floatable control panels that contain controls to help you modify/control your design.



Workspace Buttons group

1.3.1.8 The File Settings

The File button group has a pop up dialog that allows you to set the settings related to file open and save commands.



Automatic Archive

This creates a directory where a separate project file is saved every time you make a change to a projects (Zooms, pans etc. do not create a saved project file).

This gives you a complete audit trail of all changes you make to a project.

Parts and Artworks are not archived.

Reload Last File on Startup

If checked, the last project, part or artwork saved will be reloaded on start-up.

Save File Compressed

If not checked, AutoTRAX DEX files are uncompressed XML and can be views with any text editor. Compressed files can be less than 5% of the size of an uncompressed file. You are recommended to compress your files.

Decompress File

Click this to decompress a AutoTRAX DEX file. A pop up dialog will appear where you can select a file to decompress.

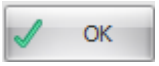
Reset Default

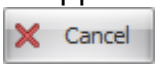
Reset the default template file for projects, parts and artworks to the factory default. You can always set the default again.

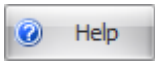
1.3.1.9 The Color Bar

The Color Bar is shown below. You can hide/show the color bar from the Panels ribbon menu group in the main menu.

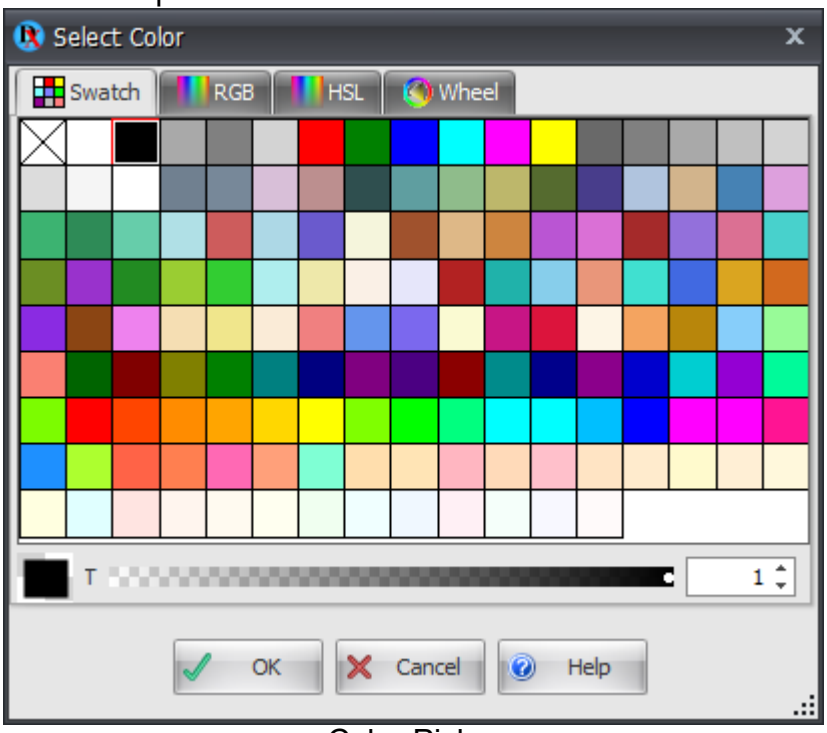
You can select the fill or pen/line color for selected object by clicking on one of the color buttons in the Color Bar.

 The button on the left of the color bar is the fill color. The button to the right of the fill color is the line or border color. If you click on either of these buttons a color chooser will appear as shown below. Pick a color and click on the 

button. Click the 

button to ignore your changes. Press the 

key to get help on this color picker.



Color Picker

Left-click on a button to set the fill color.

Right-click to set the pen/line color.

If the selected object can only have a fill color, or a pen/line color, then clicking the left or right mouse button will set the fill or pen/line color.

If no object is selected then the selected color will be used as the default color when adding more objects.

1.3.1.10 The Status Bar

The status bar gives you timely status messages and details as you work on your design. By default the status bar is placed underneath the top menu. You can place the status bar at the bottom of the application just below the viewport. To change the position of the status bar right click on the status bar and select the position of the status bar from the context menu.

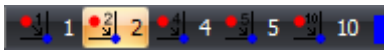
The status bar cannot be floated by the panels but you can I hide it by selecting hide from the status bars context menu. To reassure the status bar click on the view status bar in the panels menu.

This is the text status area. The content changes for each command in AutoTRAX DEX.

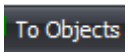
 Check/uncheck this button to enable/disable tooltips.

 Check/uncheck this button to enable/disable smart panning.

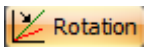
Drag the slider to zoom in and out of the selected viewport.



These buttons let you set the number of snaps per grid.



This button lets you turn stuck to objects on or off.



You can turn snap rotation on/off by clicking this button.

You can quickly change the units from millimeters to inches or vice versa by clicking on one of these two buttons.

You can change the pickup mode from top-level pick to sub pick or vice versa by clicking on one of these two buttons. Sub picking allows you to select objects that are inside other objects.

1.3.2 The Ribbon Menu

The Ribbon menu is home to hundreds of commands. It consists of several ribbon pages filled with commands and appears at the top of AutoTRAX DEX's main window.

You make changes to your design using:

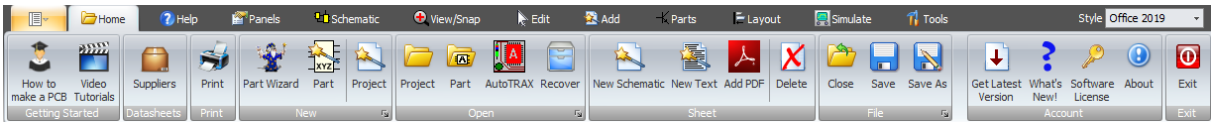
- The Ribbon Menu
- Interacting with the viewports using the mouse and keyboard.
- Making changes using one of the many panels.

The ribbon pages and their content are context sensitive and will only appear where applicable, for instance, the PCB Ribbon Page will only appear when editing a PCB in a project.

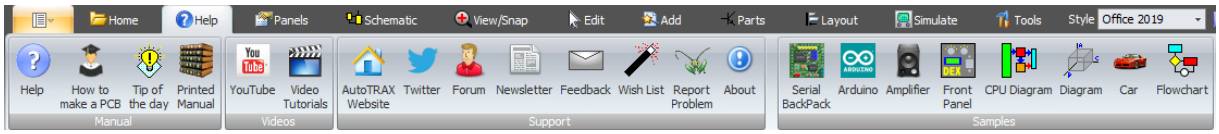
You can easily customize the command keyboard shortcuts using the Shortcuts Editor.

The Ribbon menu consists of the following ribbon pages:

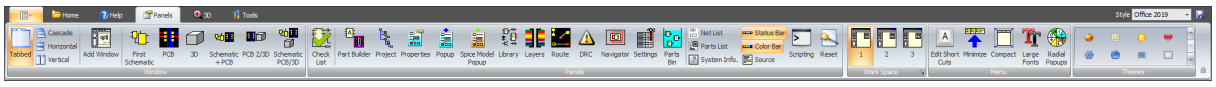
The Home Ribbon Page



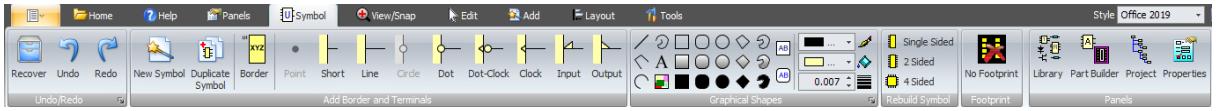
The Help Ribbon Page



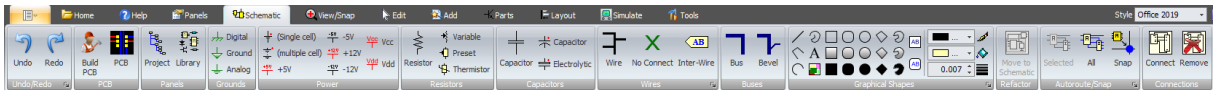
The Panels Ribbon Page



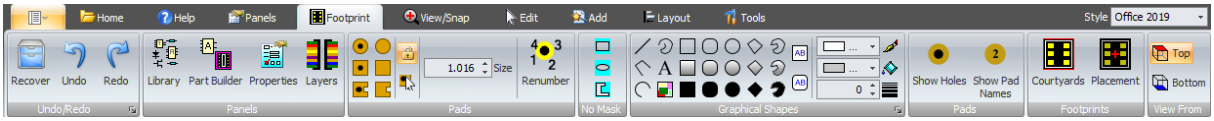
The Symbol Ribbon Page



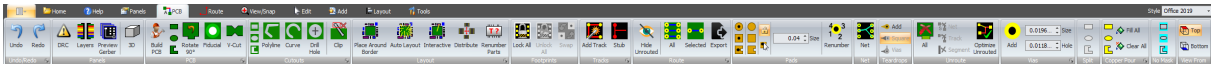
The Schematic Ribbon Page



The Footprint Ribbon Page

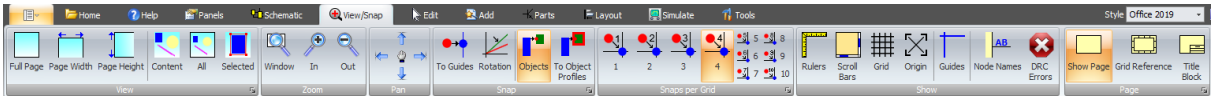


The PCB Ribbon Page

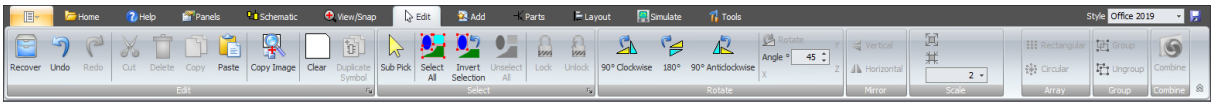


The Route Ribbon Page

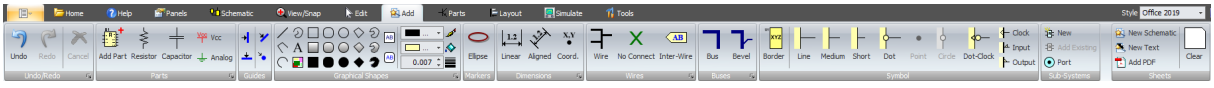
The View/Snap Ribbon Page



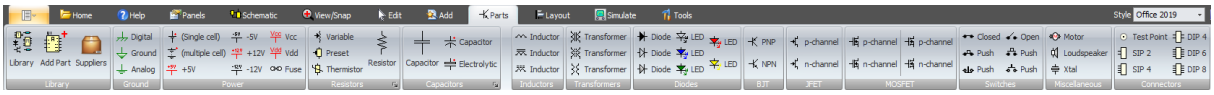
The Edit Ribbon Page



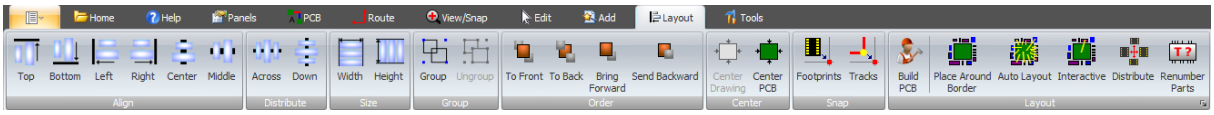
The Add Ribbon Page



The Parts Ribbon Page

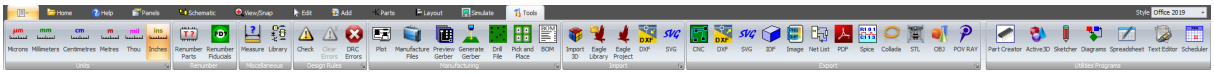


The Layout Ribbon Page

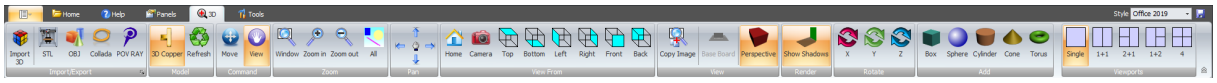


The Simulate Schematic Page

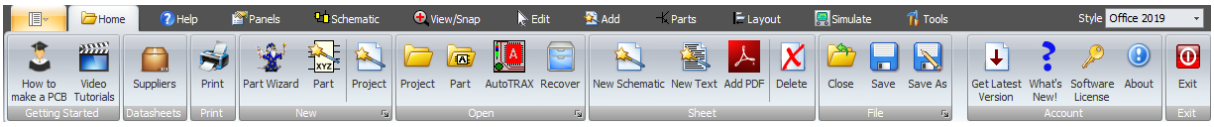
The Tools Ribbon Page



[The 3D Ribbon Page](#)



1.3.2.1 The Home Ribbon Page

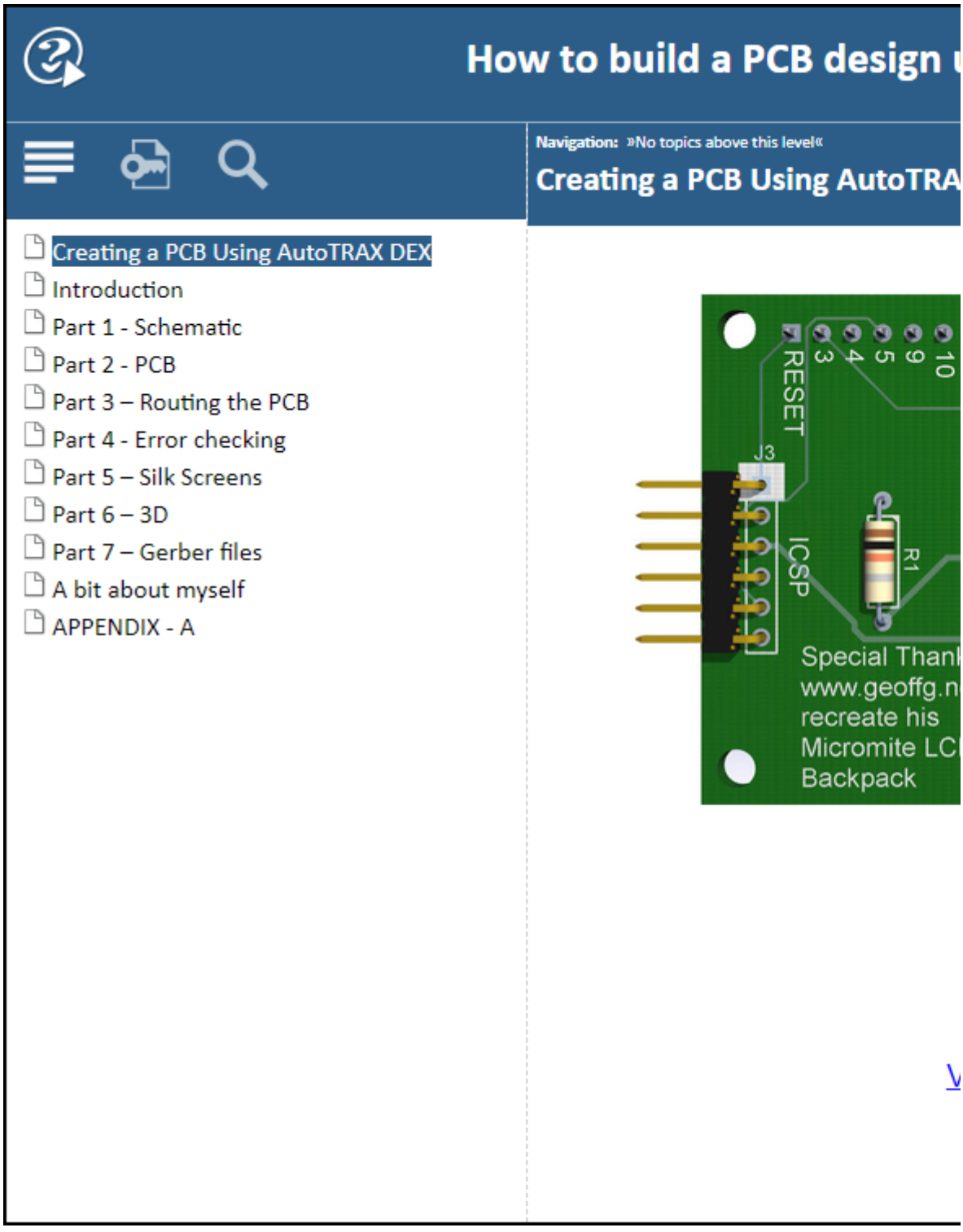


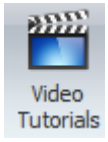
The Home Ribbon Page

The Getting Started Command Group

How to Make a PCB

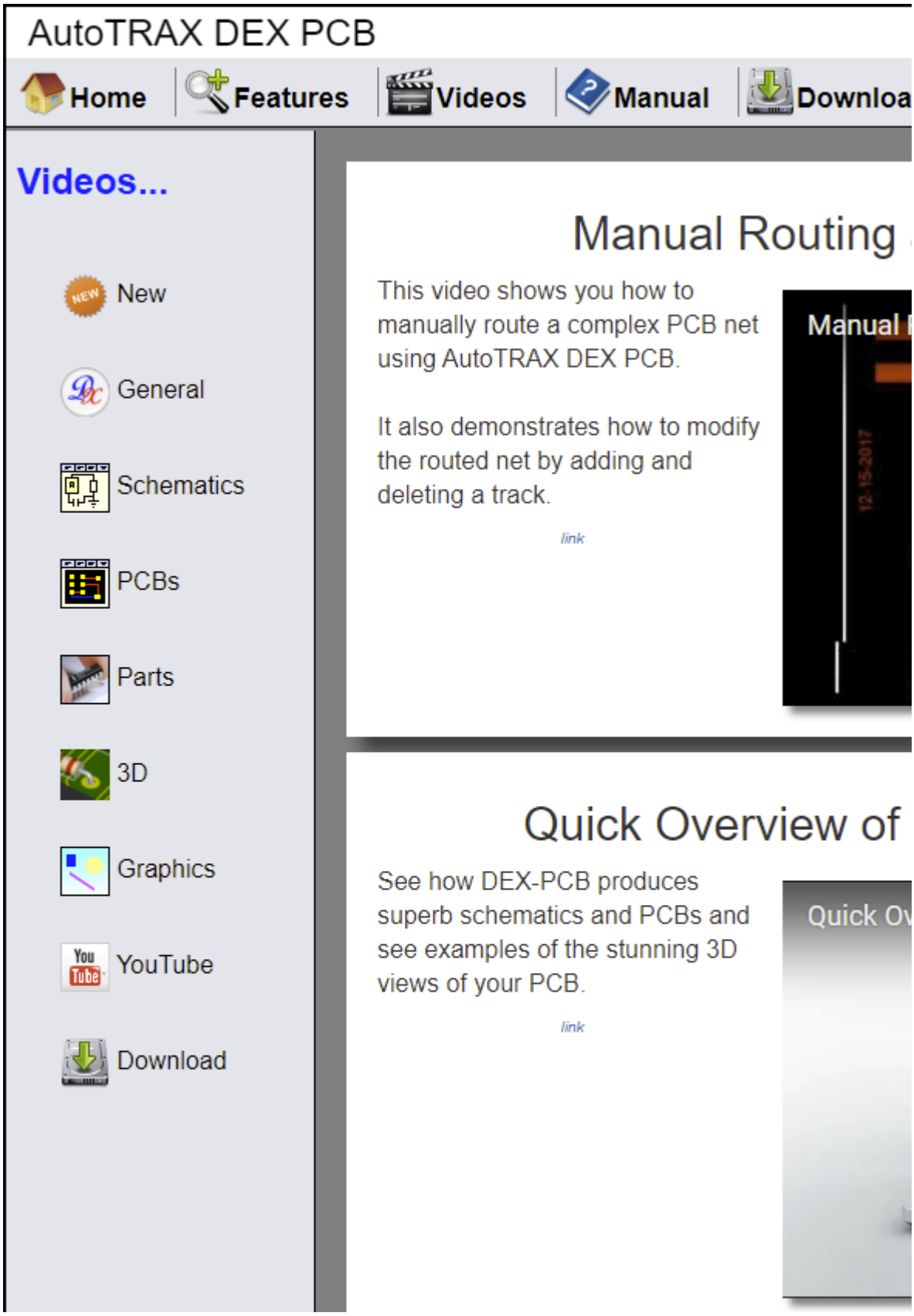
Click to see a quick and [enlightening tutorial](#) on how to make a PCB.





Video Tutorials

Click to view a useful collection of video tutorials that will guide you through the process of PCB design.



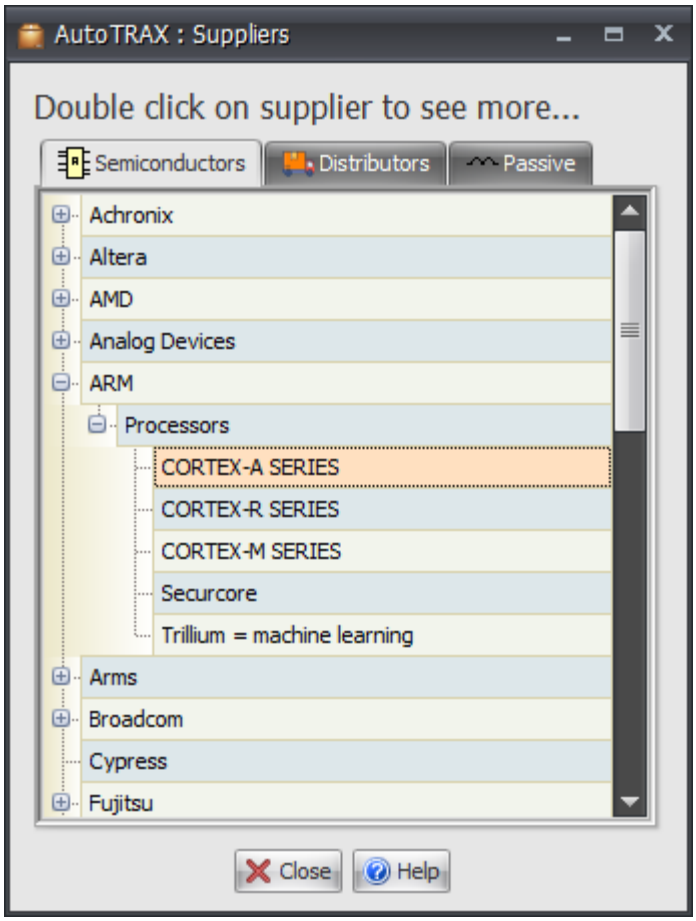
AutoTRAX DEX Video Tutorials Web Page

Getting Started Commands

The Datasheets Command Group

Suppliers

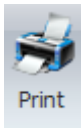
This display valuable links to suppliers of electronic parts. Read more...



Suppliers

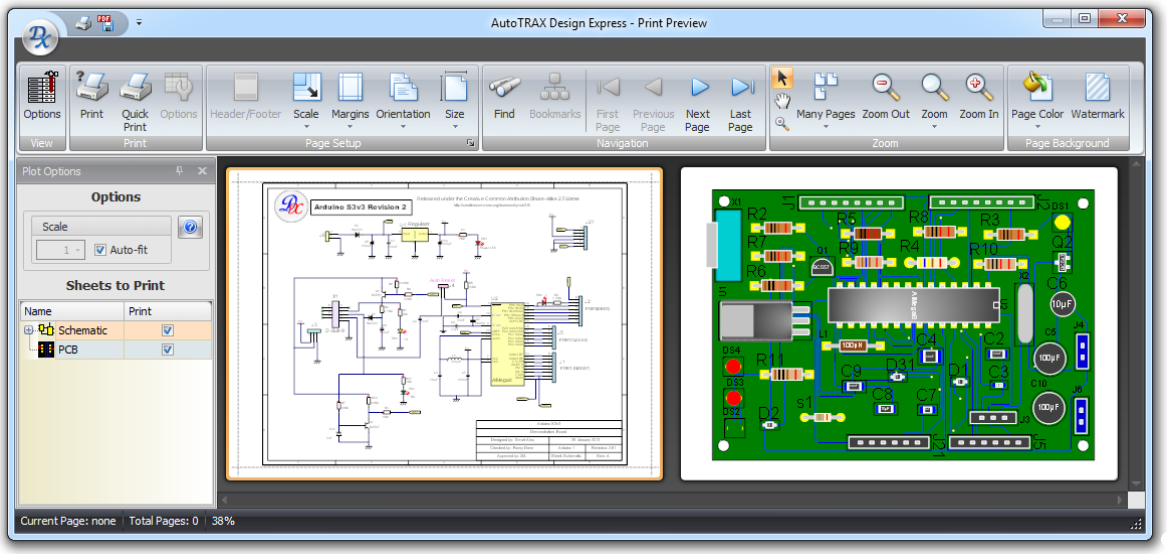
Suppliers Commands

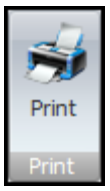
The Print Command Group



Print the Design

This will print your design. Read more...



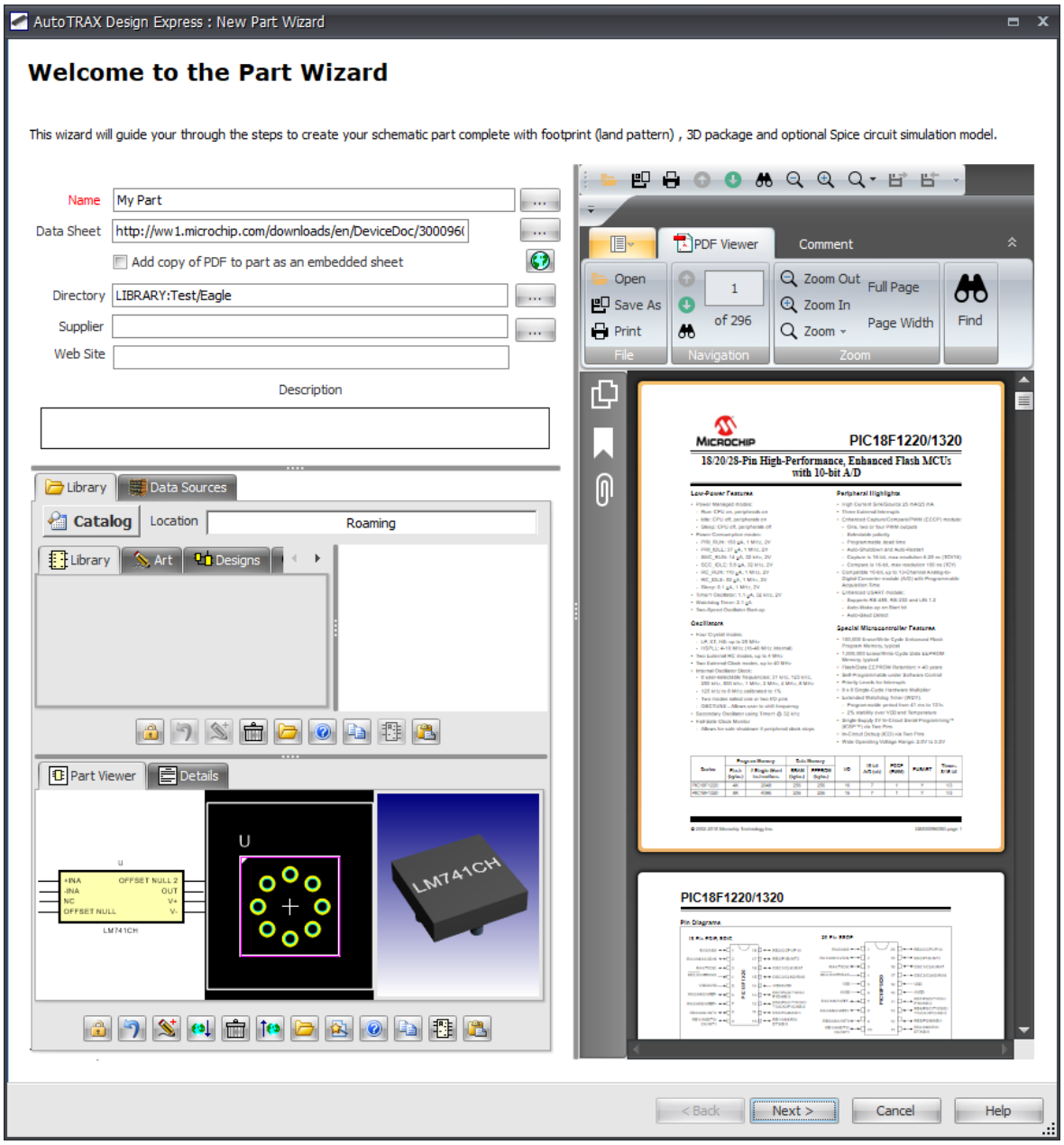


Print Commands

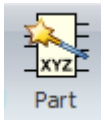
The New Command Group

Run the Part Wizard

This will run the Part Wizard what will guide you through the process of creating a custom part. Read more...



The Part Wizard

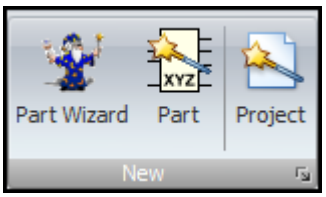


Create a New Part

Click to create a new blank part. Read more about creating parts...

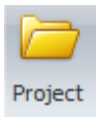
Create a New Project

This will create a new project. Read more about projects...



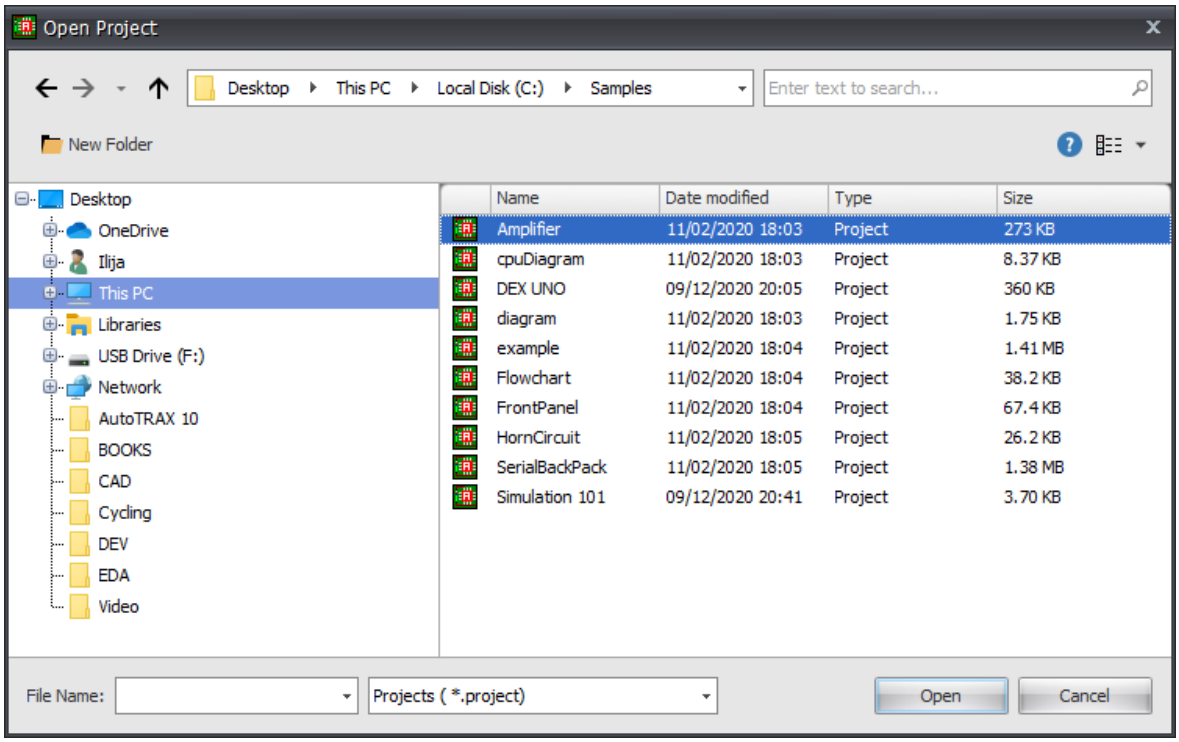
New Commands

The Open Command Group

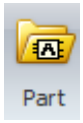


Open an Existing Project

Click to open an existing project.

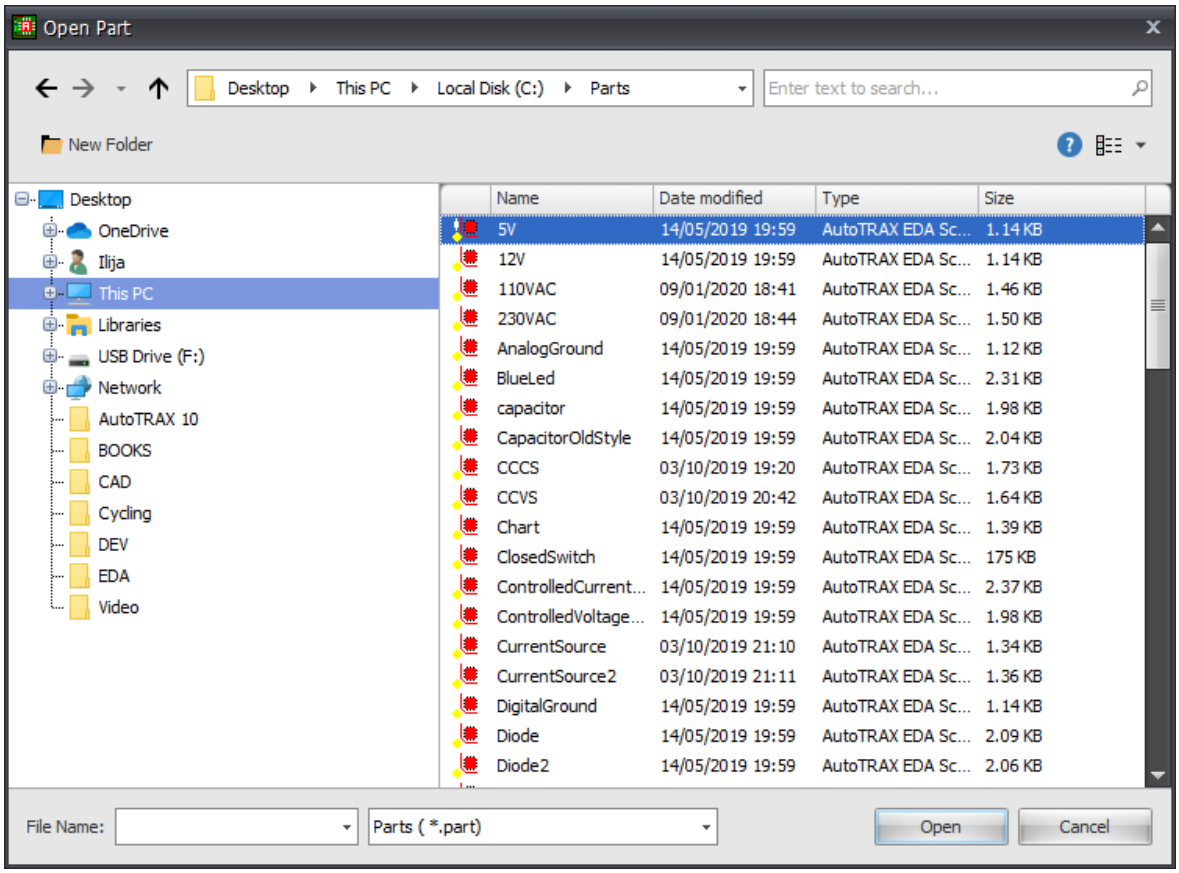


Open an Existing Project



Open an Existing Part

Click to open an existing part.

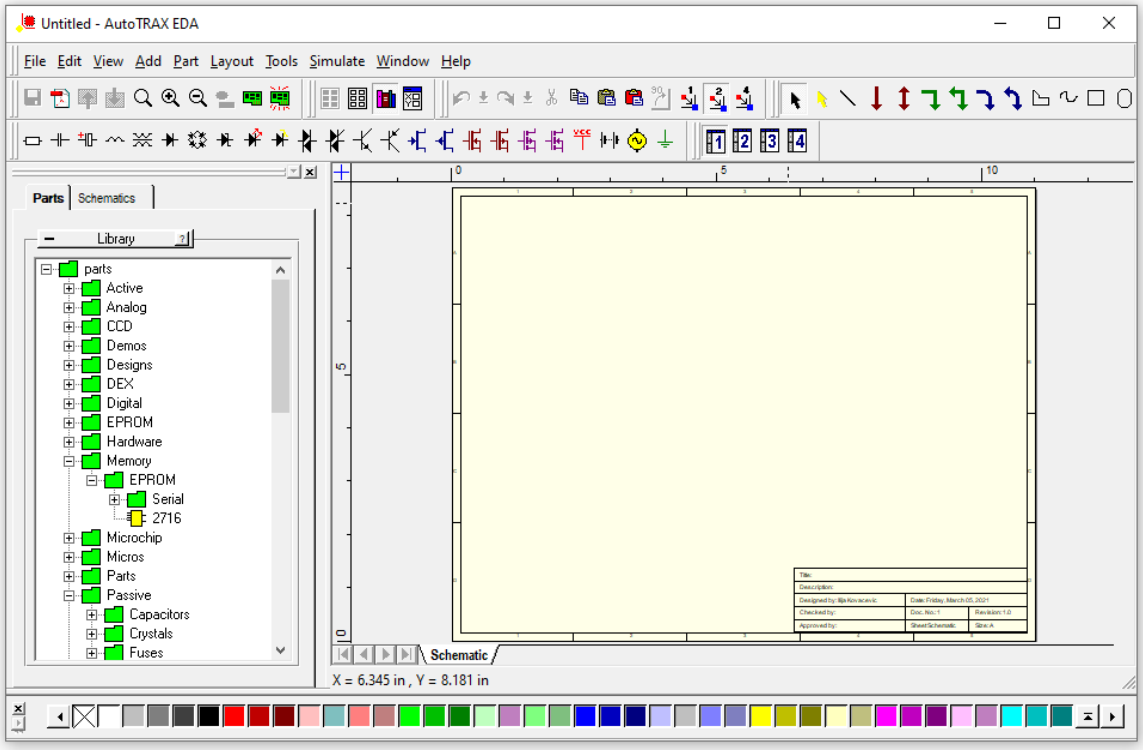


Open an Existing Part

Open an Existing AutoTRAX DEX EDA Design

Click to open an existing legacy AutoTRAX DEX EDA project. This will open a legacy schematic and PCB design. AutoTRAX DEX EDA was discontinued

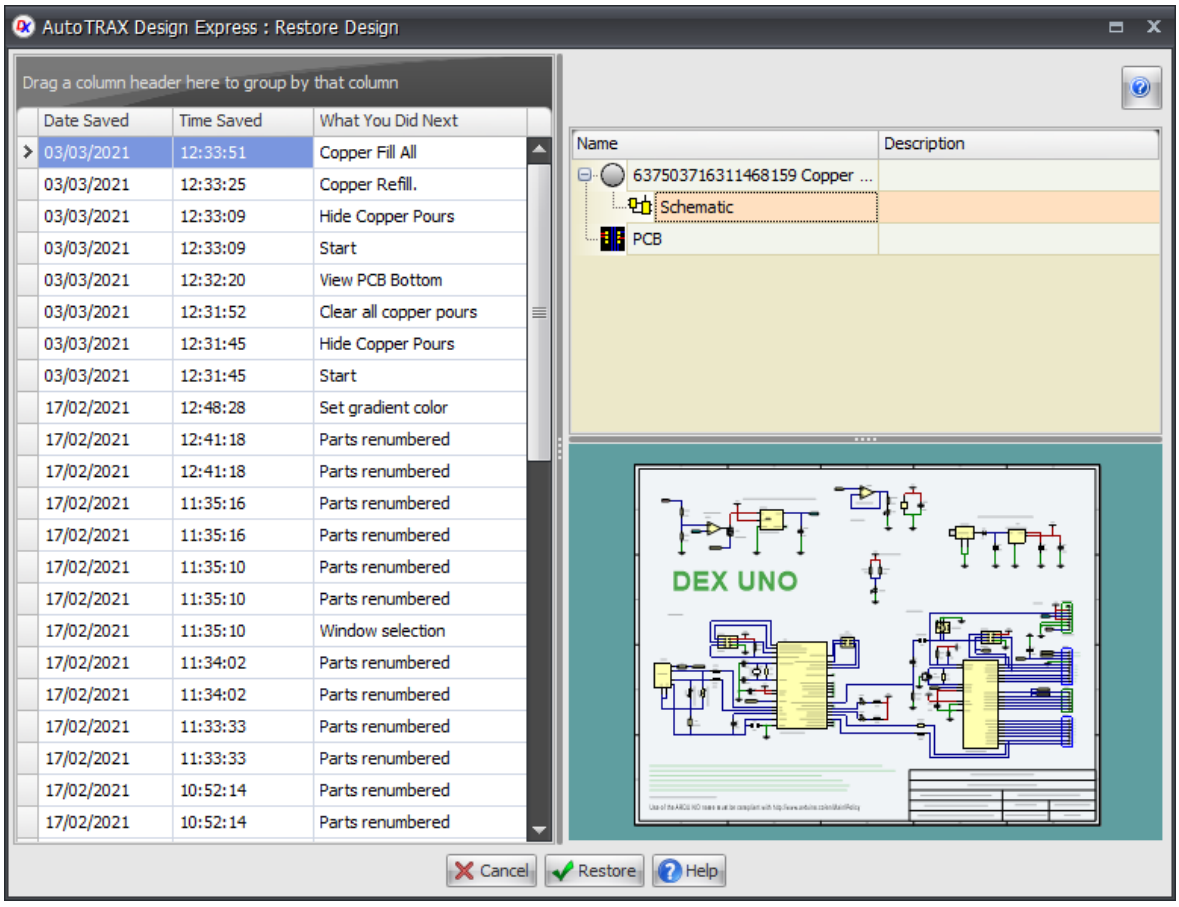
You can download AutoTRAX DEX EDA from here...



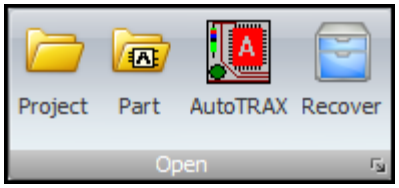
AutoTRAX DEX EDA

Recover a Past/Future Version

This displays the Restore Design dialog that list all changed that you have made to a project or part. You can easily revert to a past design change. Read more...

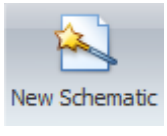


The Restore Design Dialog



Open Commands

The Sheet Command Group



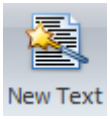
Create a New Schematic

This adds a new schematic sheet to your project..

Create a New Symbol

In parts, schematics act as symbol schematics. So, if you have a part that needs two symbols, then you would have two schematic symbol sheets.

If you currently have a part open, then the additional schematic symbol sheet will be a sub-part. For instance, if you are creating a part which has 2 op-amps, you would have 2 schematics symbol sheets in total; one for each op-amp. You might also add an additional schematic symbol sheet for the power connections.

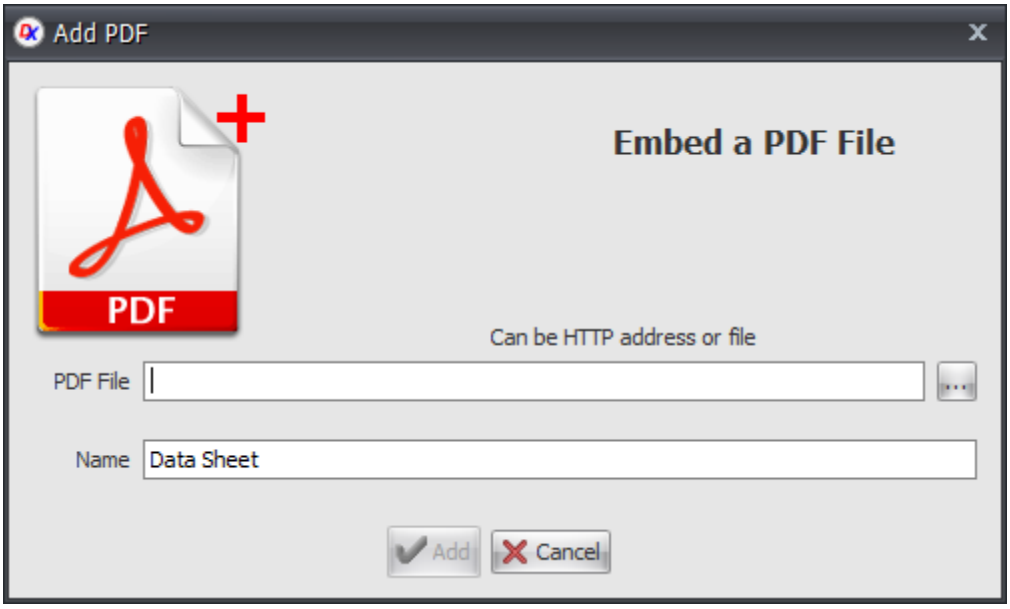


Create a New Text

This will add a new text/document sheet to your design.

Add a PDF to the Design

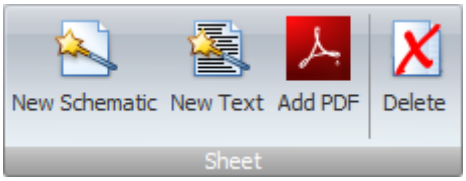
Clicking this will allow you to embed an Adobe PDF file in your project/part.



Add a PDF to the Design

Delete the Current Sheet

This will delete the currently select sheet.

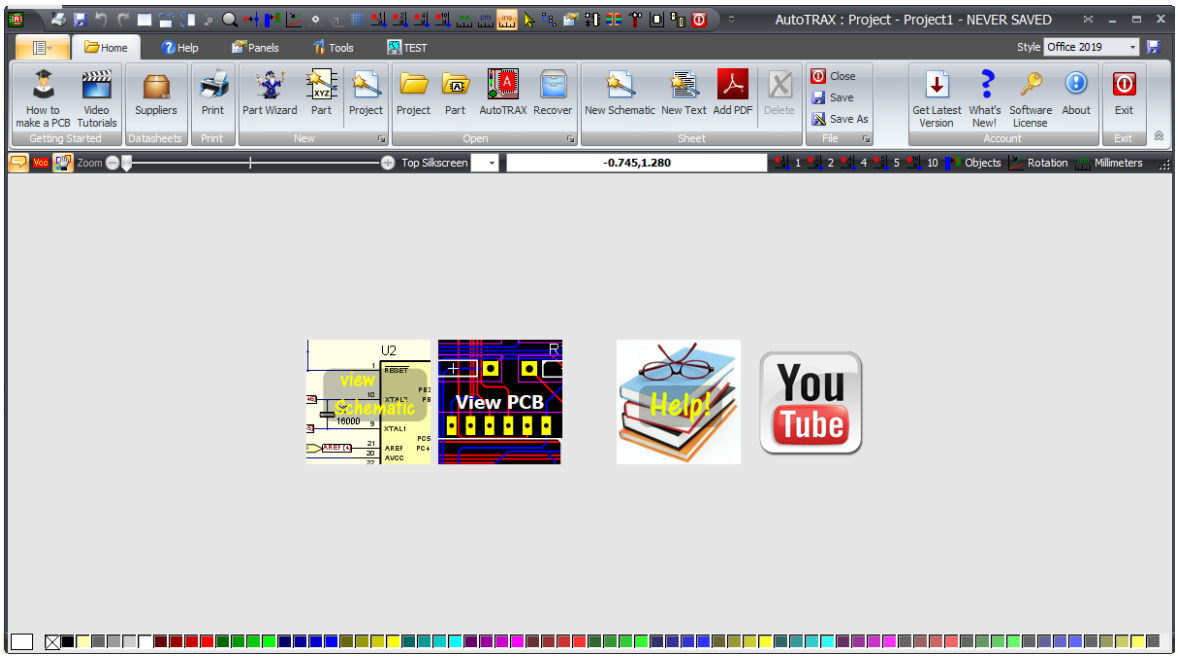


Sheet Commands

The File Command Group

Close

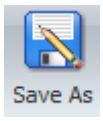
This will close the current project/part. A new unsaved project/part will then be created.



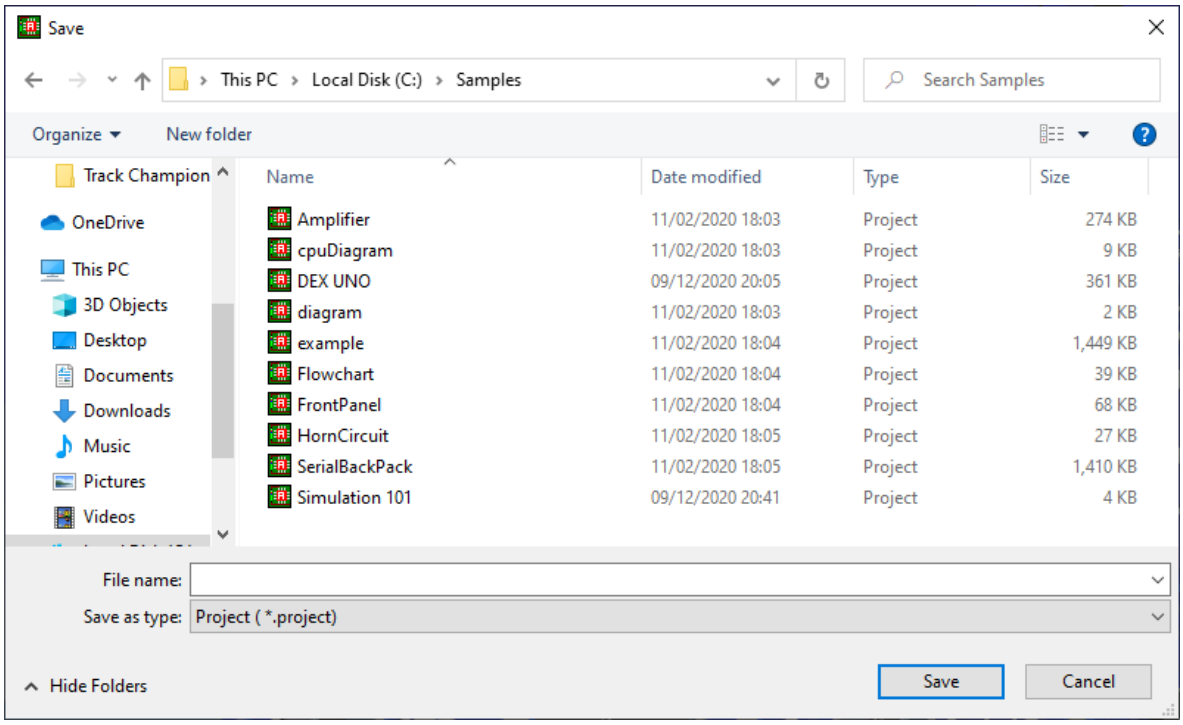
New Design After Closing Current Design

Save

This will save the design. If it has never been save you will be prompted for a save location and a file name.



Save As



Save As Dialog

This will save the design to a new location and file name.

File Commands

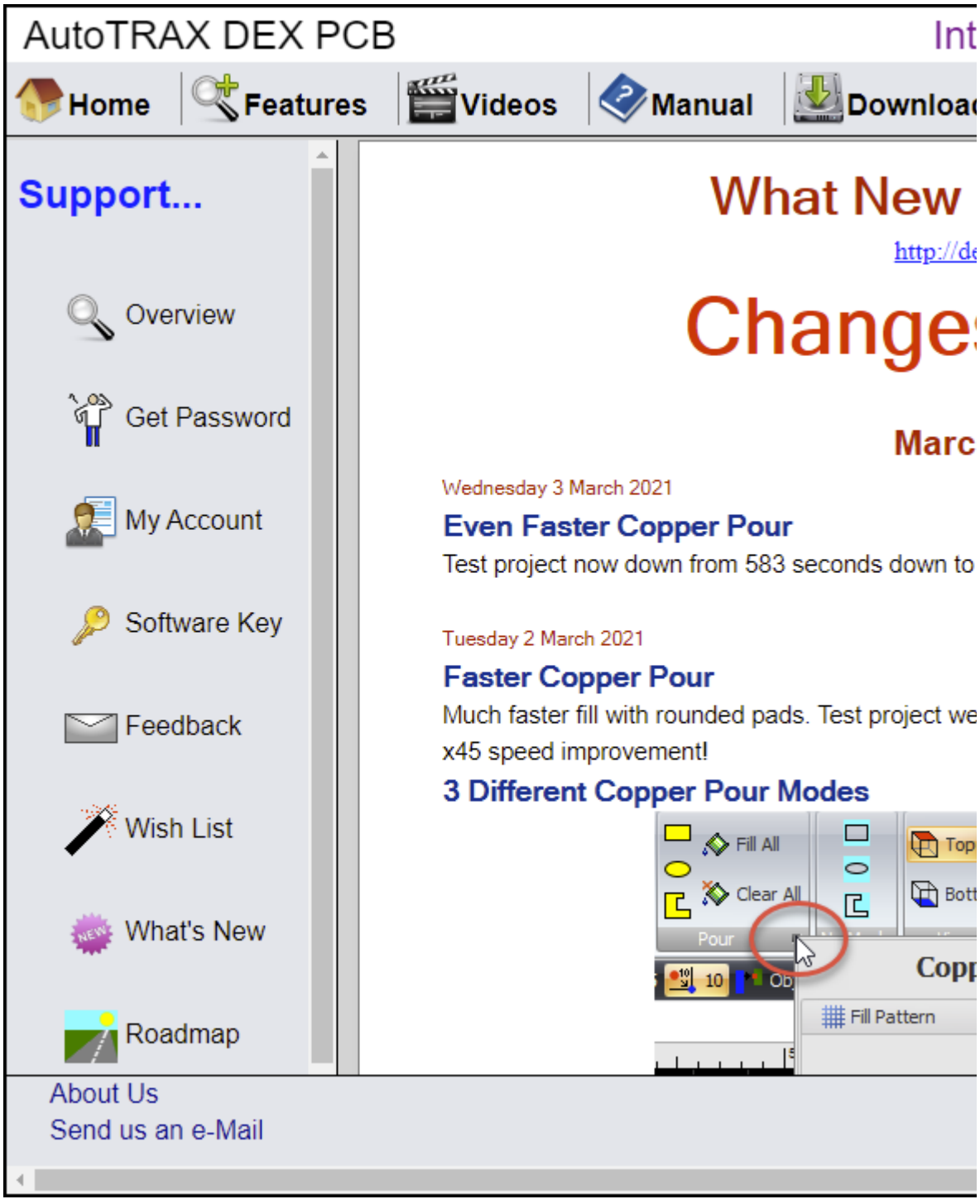
The Account Command Group

Get Latest Version

Click to get the latest version of AutoTRAX DEX. Read more...

What's New

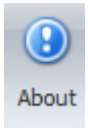
Display the What's New web-page detailing changes.



What's New

Software License

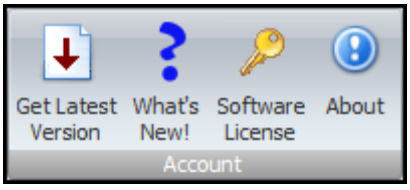
This will display you software license details. Read more...



About

This will display the About AutoTRAX DEX dialog box. It displays version details and the version of OpenGL installed on your machine.

The About Dialog Box



Account Commands

The Exit Command Group

Close AutoTRAX DEX

This will close AutoTRAX DEX. If you have an unsaved edit then you will be prompted to save it.

What if you exit AutoTRAX DEX but did not save it? Well, fear not:

When you restart AutoTRAX DEX if you click on the Redo buttons you can recover each change, one by one.

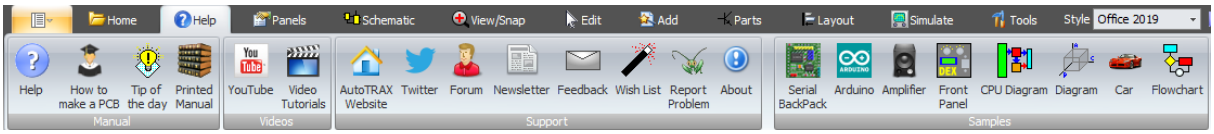
Alternatively you can click the 

button and select a state to recover to.

Find out more...

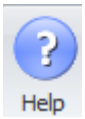
The Exit Commands

1.3.2.2 The Help Ribbon Page



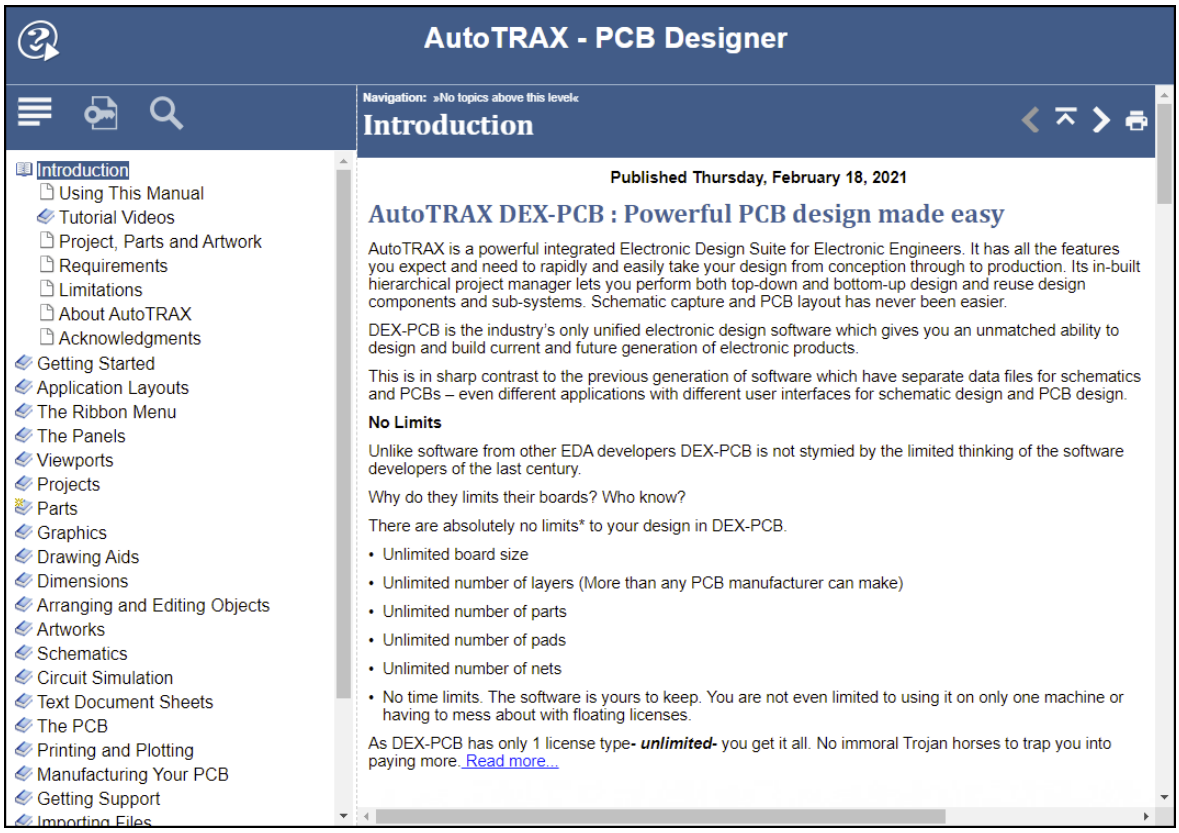
The Help Ribbon Page

The Manual Command Group



Help

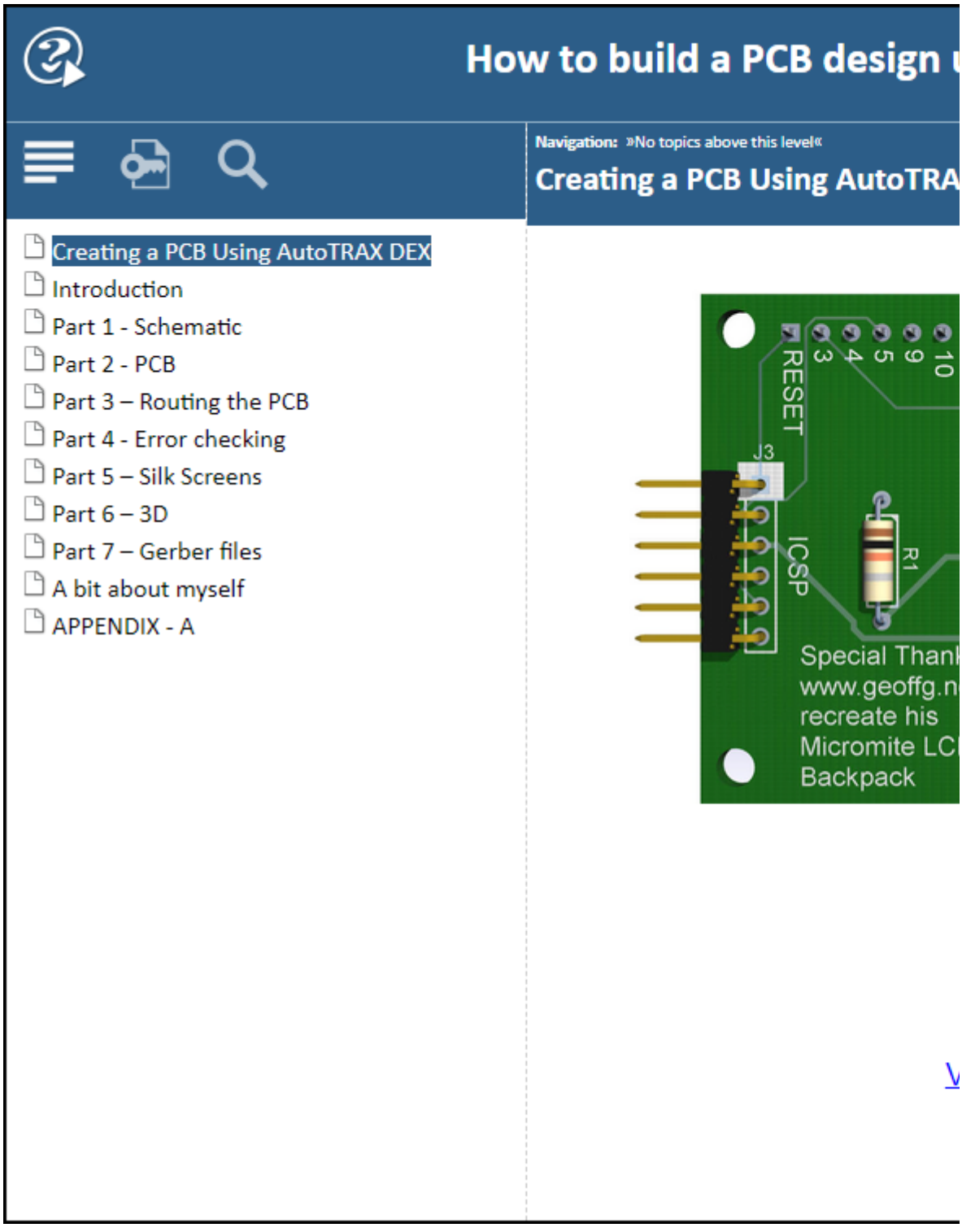
This will display the online manual.



The On-line Manual

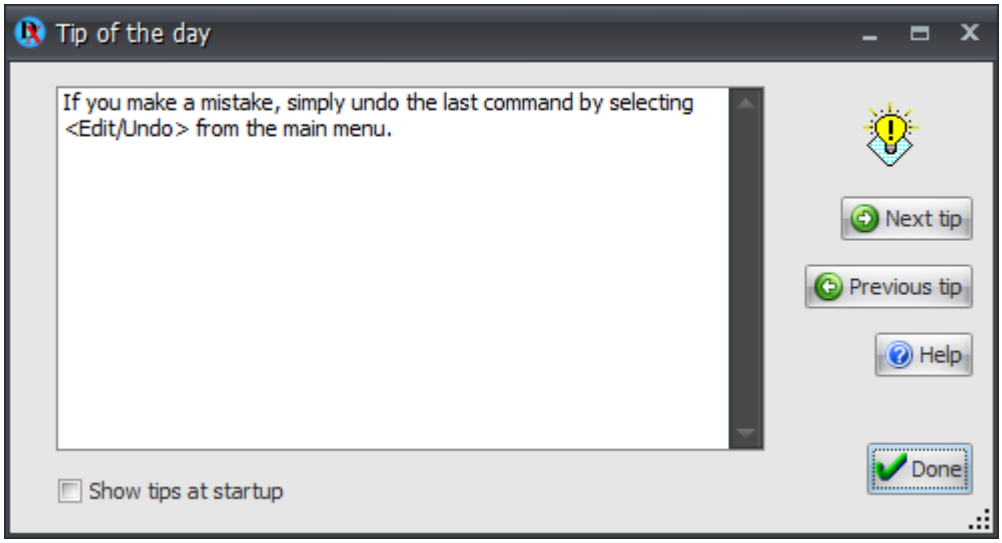
How to Make a Video Tutorial

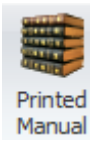
Click to display the on-line tutorial on how to create a PCB using AutoTRAX DEX.



Tip of the Day

Show the Tip of the Day modal dialog.

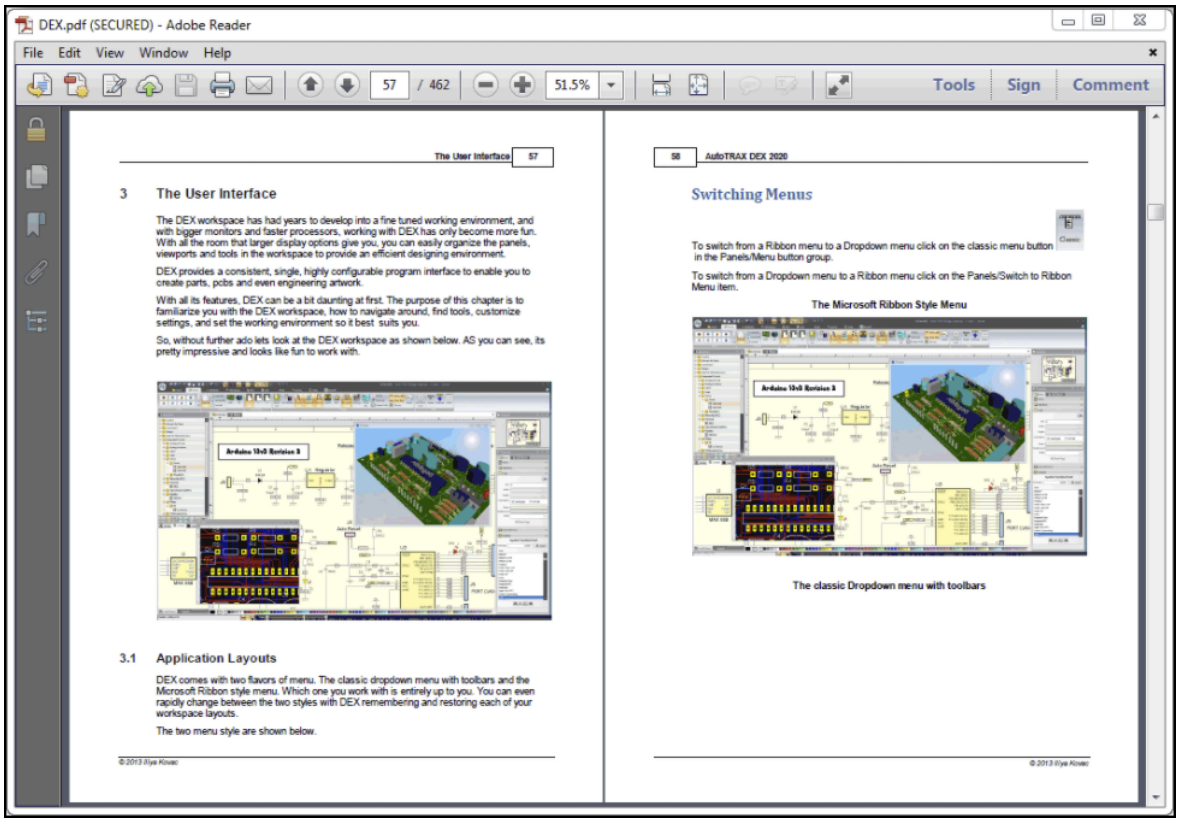




Download a Printable Manual

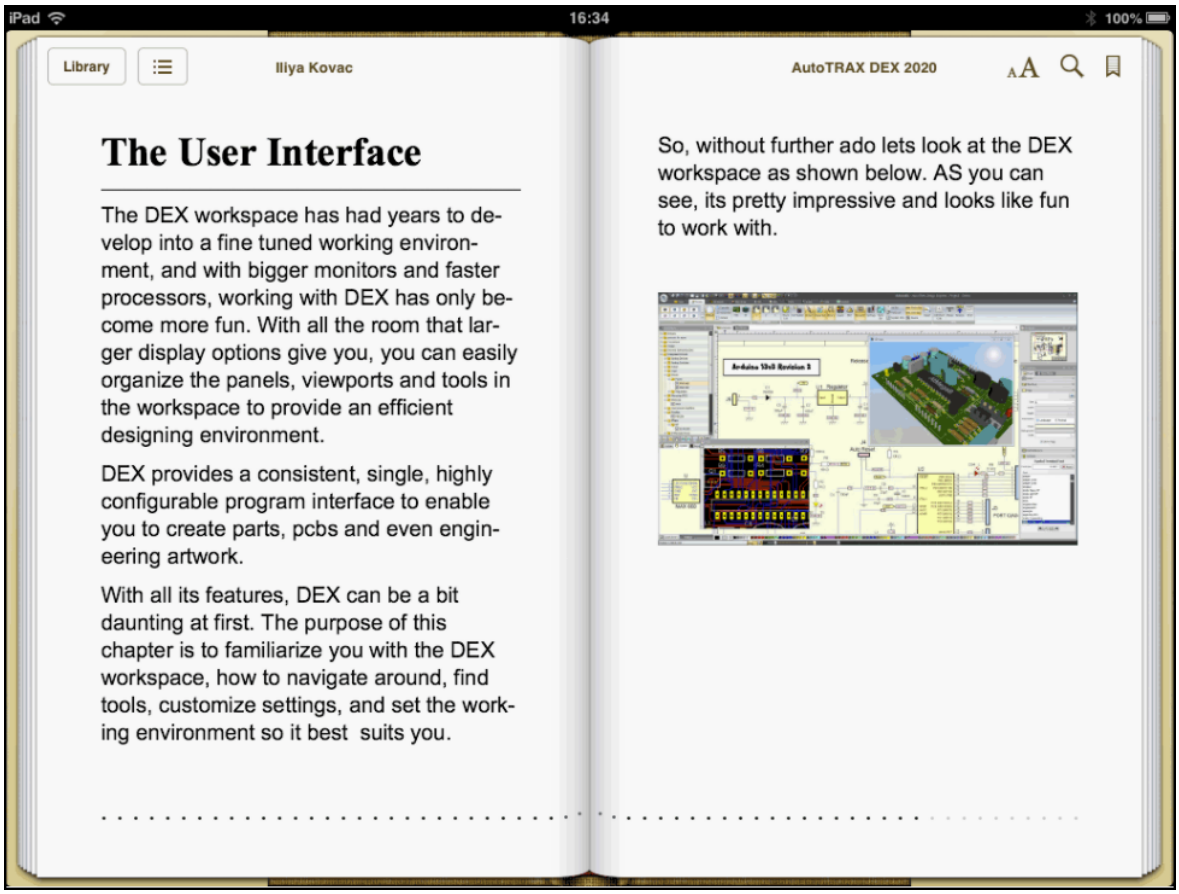
Click to download the AutoTRAX DEX manual as:

- An Adobe PDF file or

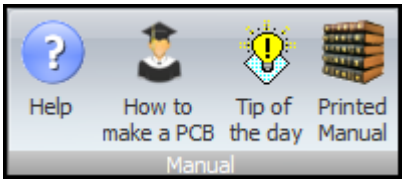


Adobe PDF Manual

- An Apple ePub file



Apple ePub Manual

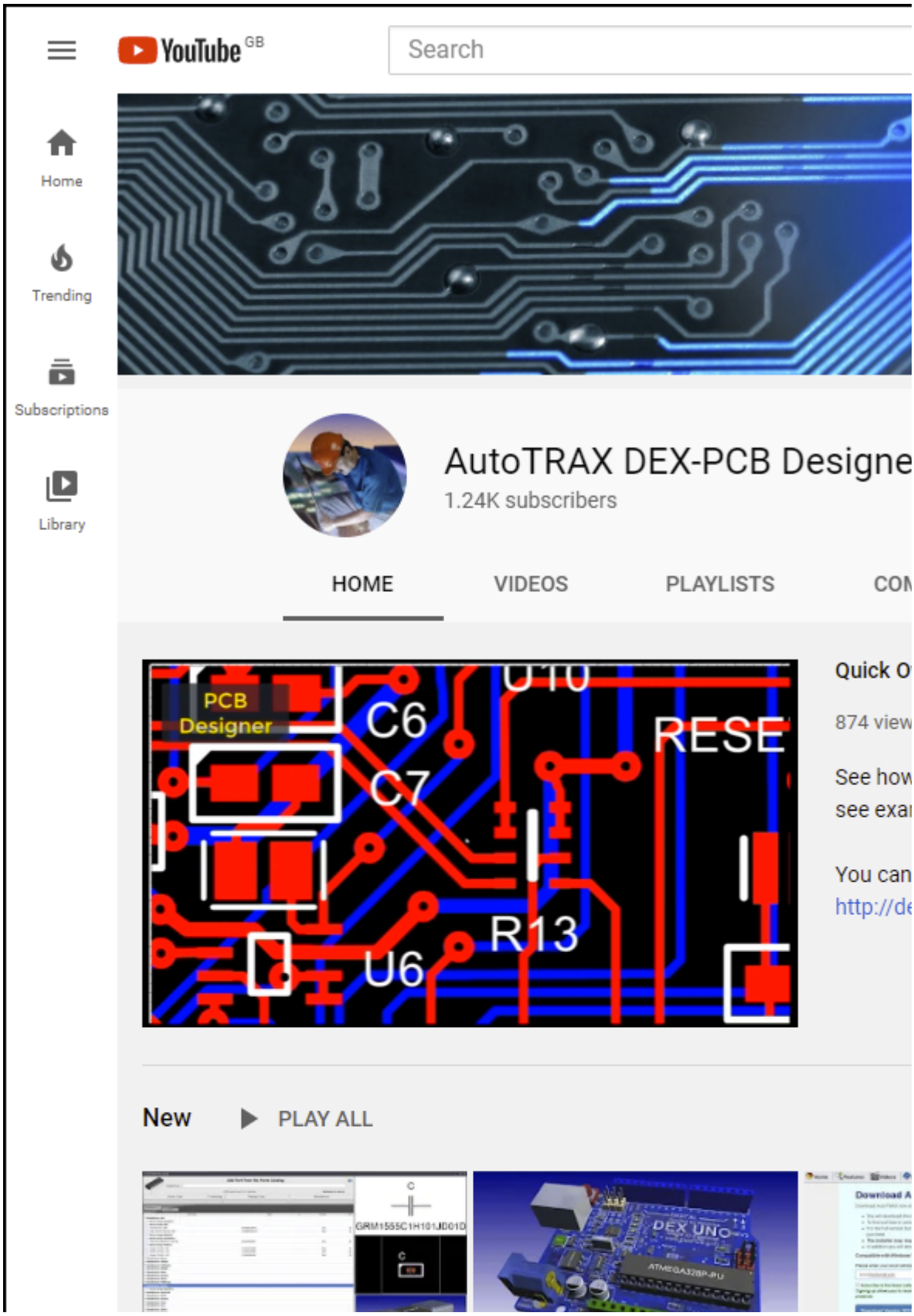


Manual Commands

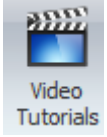
The Videos Command Group

View the YouTube Videos

This will display the AutoTRAX DEX YouTube web site.

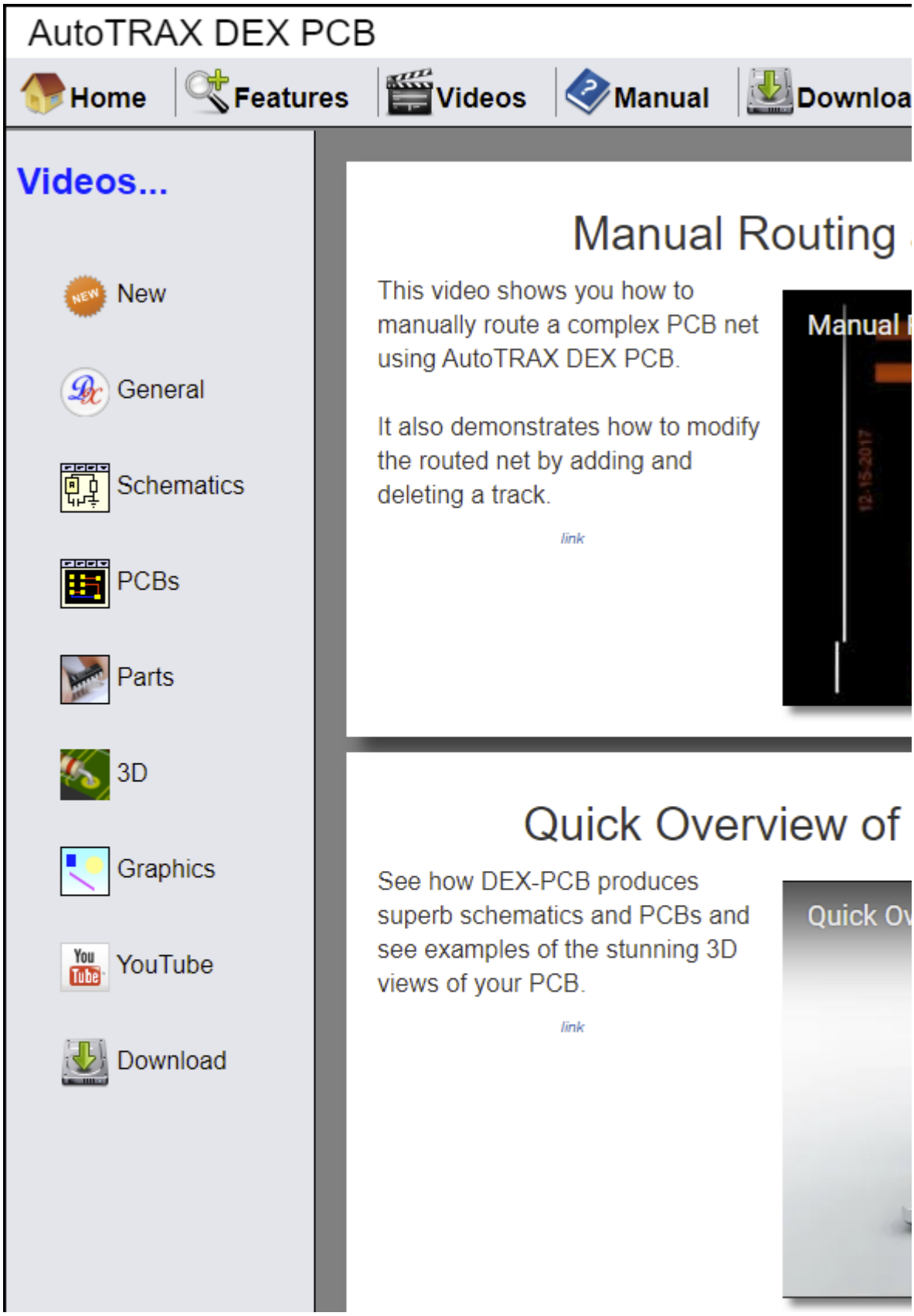


The AutoTRAX DEX YouTube Website



View the Video Tutorials

This will display the AutoTRAX DEX video tutorials web page.



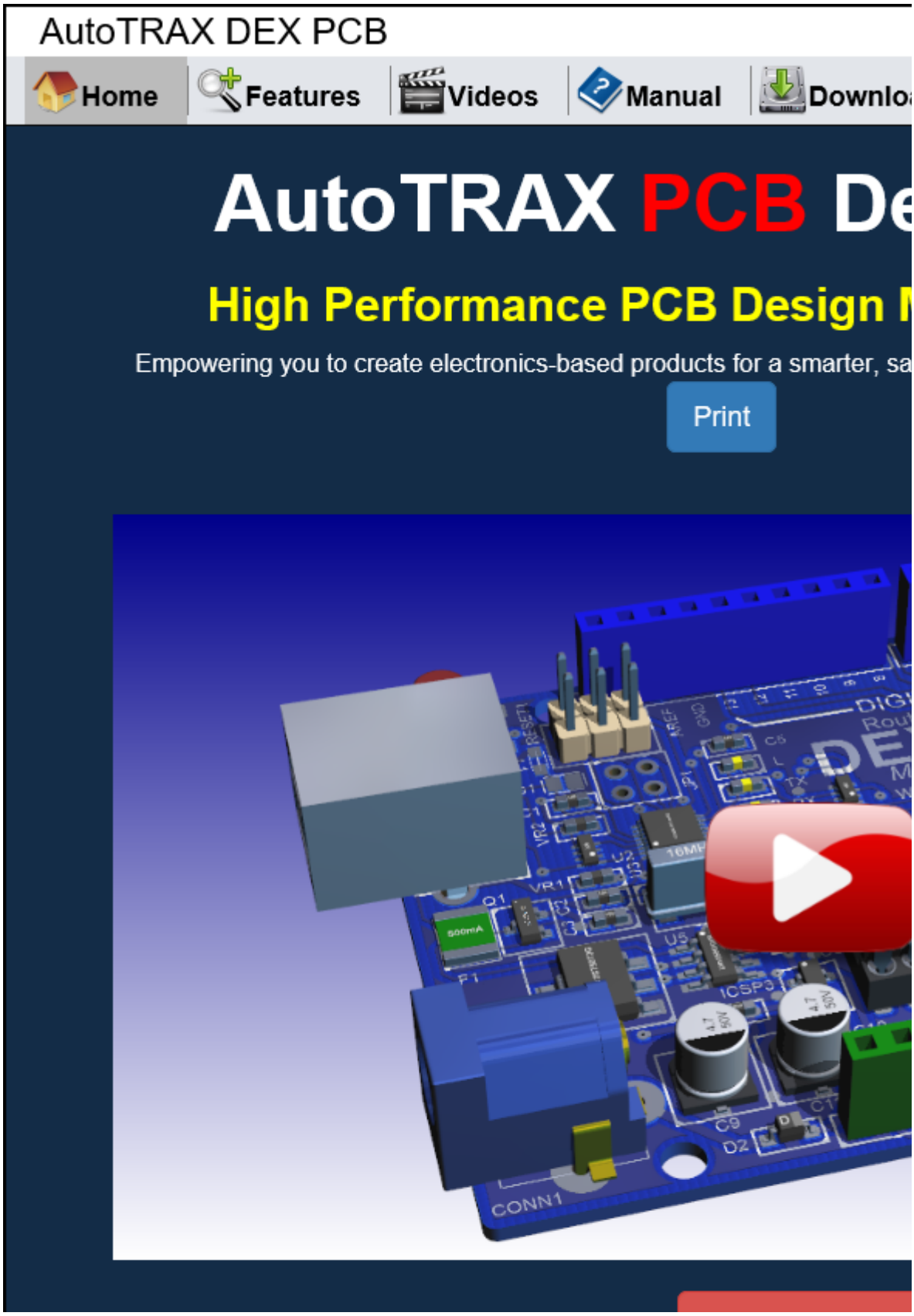
AutoTRAX DEX Video Tutorials Web Page

Video Commands

The Support Command Group

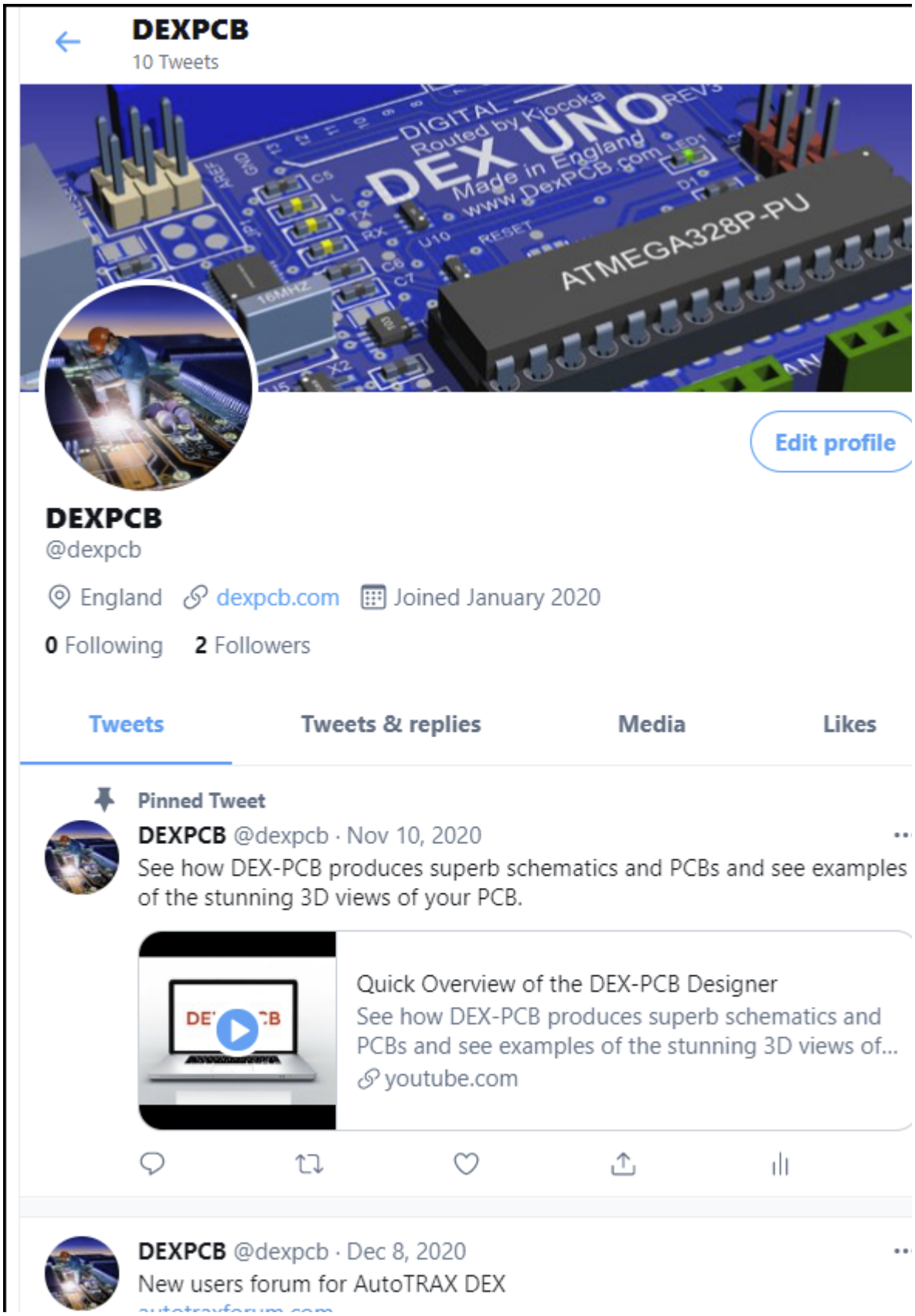
AutoTRAX DEX Website

Click to go to the AutoTRAX DEX Website.



AutoTRAX DEX Twitter Website

Click to go to the AutoTRAX DEX Twitter page.



The AutoTRAX DEX Twitter Page

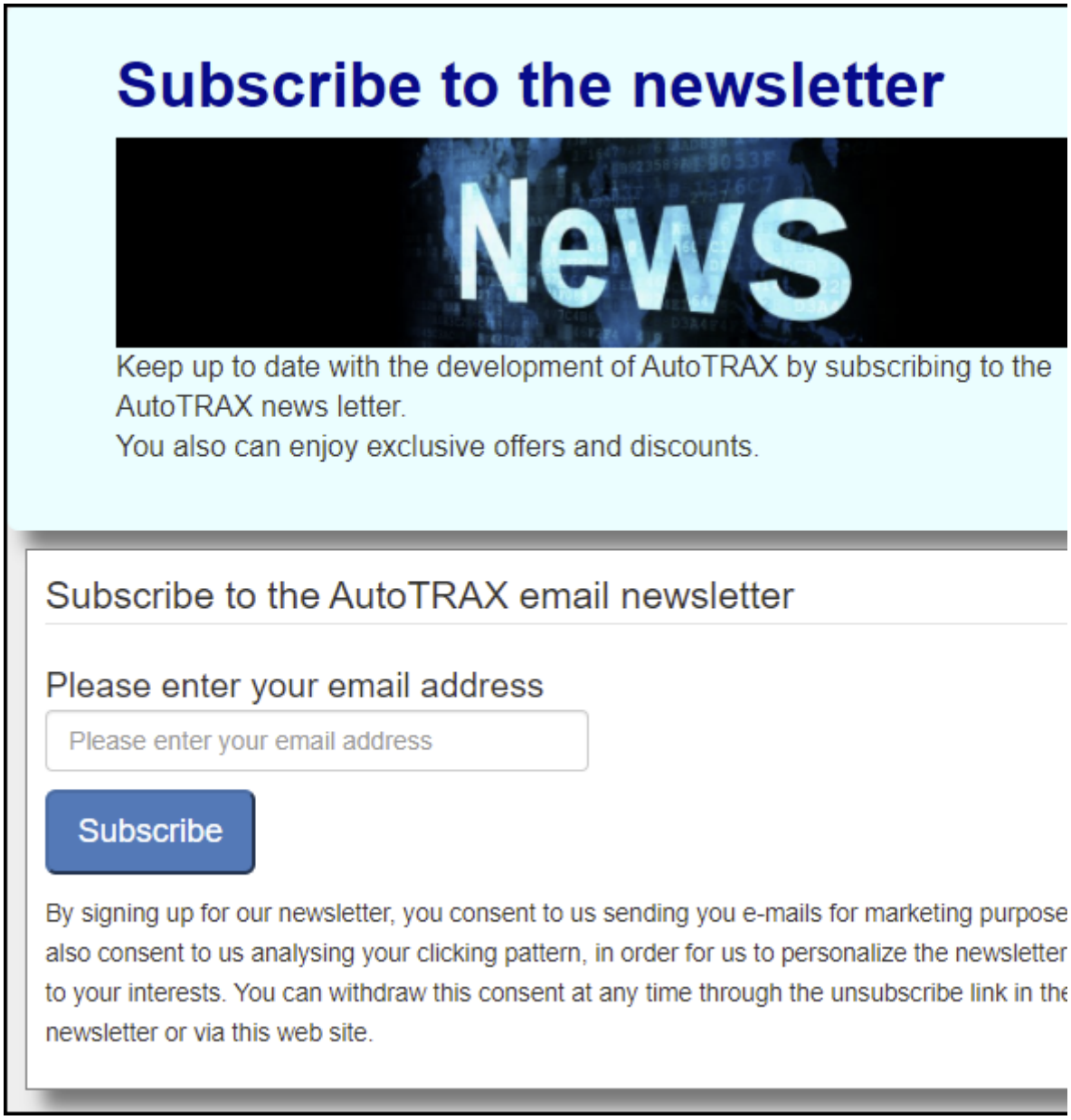
The AutoTRAX DEX Users Forum

Find out more...

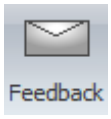
The Users Forum Application

Subscribe to the Newsletter

Click to subscribe to the AutoTRAX DEX Newsletter.



Subscribing to the Newsletter



Send Feedback

Send us feedback.

AutoTRAX DEX-PCB: Feedback

Send us feedback.

Your email address

Please enter your email address

Your name

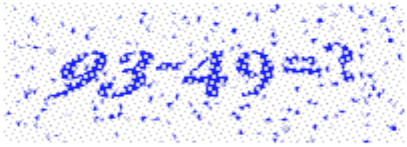
Please enter your full name

The subject

Please enter a subject

Your message

Please enter your feedback



Refresh

The answer is

Sending Feedback

Send Us Your Wishlist

Please tell use what you would like to se
happen

AutoTRAX is a leading developer of innovative electronic design software f
designers, with worldwide sales in over 60 countries.

Your Wish-List

Your email address

Please enter your email address

Your name

Please enter your full name

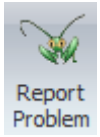
The subject

Please enter a title for your wish

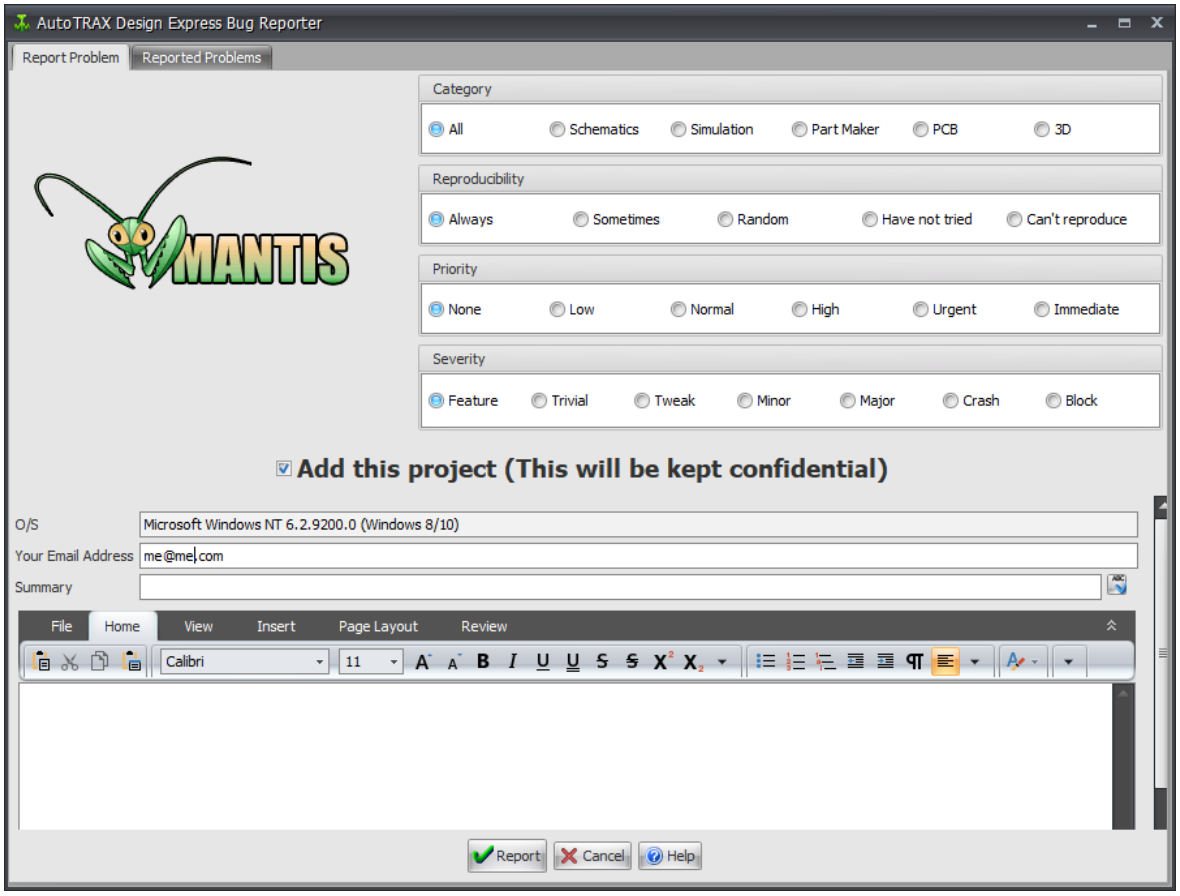
Your Wish?

Please enter your wish

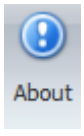
Sending Your Wishlist



Report a Problem



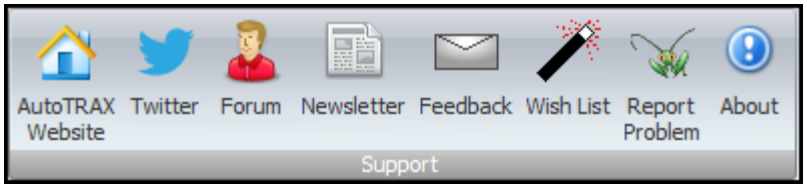
Reporting a Problem



Show the About Dialog

This will display the About AutoTRAX DEX dialog box. It displays version details and the version of OpenGL installed on your machine.

The About Dialog Box

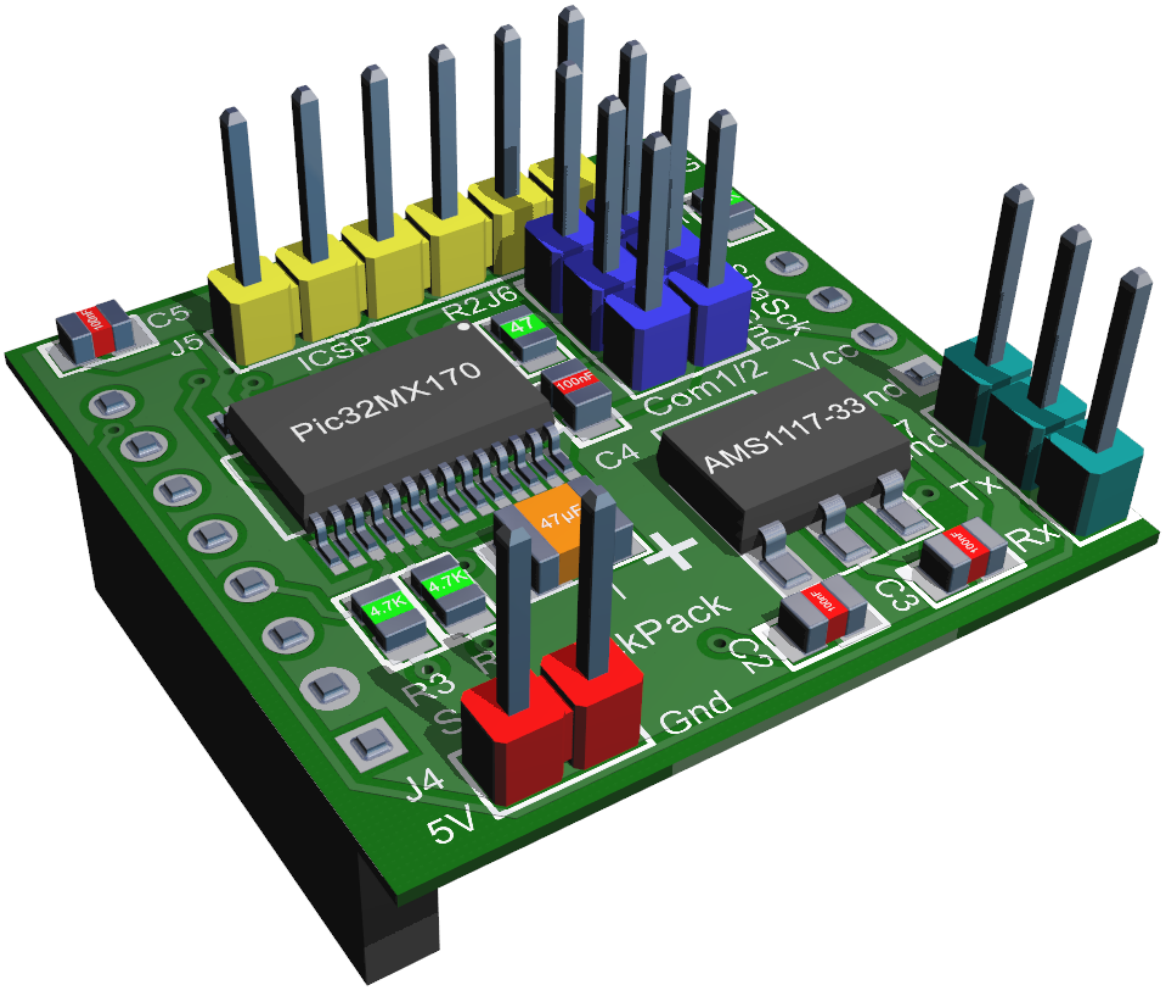


Support Commands

The Samples Command Group

The Serial Backpack

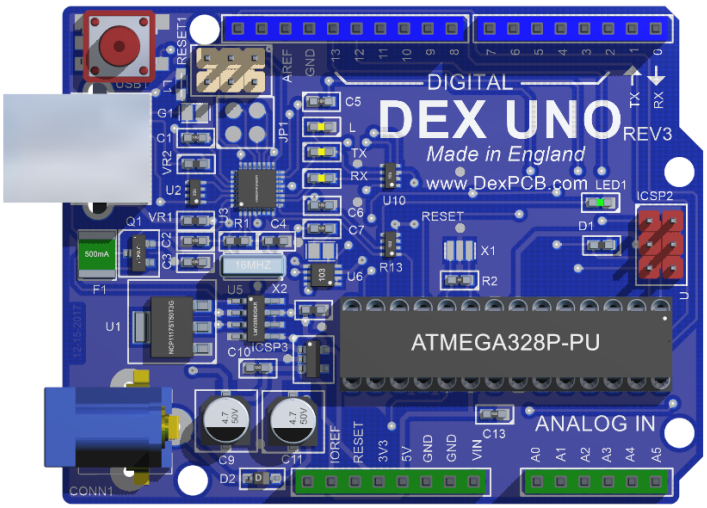
Open the Serial Backpack project donated by Mick Gulovsen from Melbourne, Australia.



The Serial Backpack

AutoTRAX DEX UNO

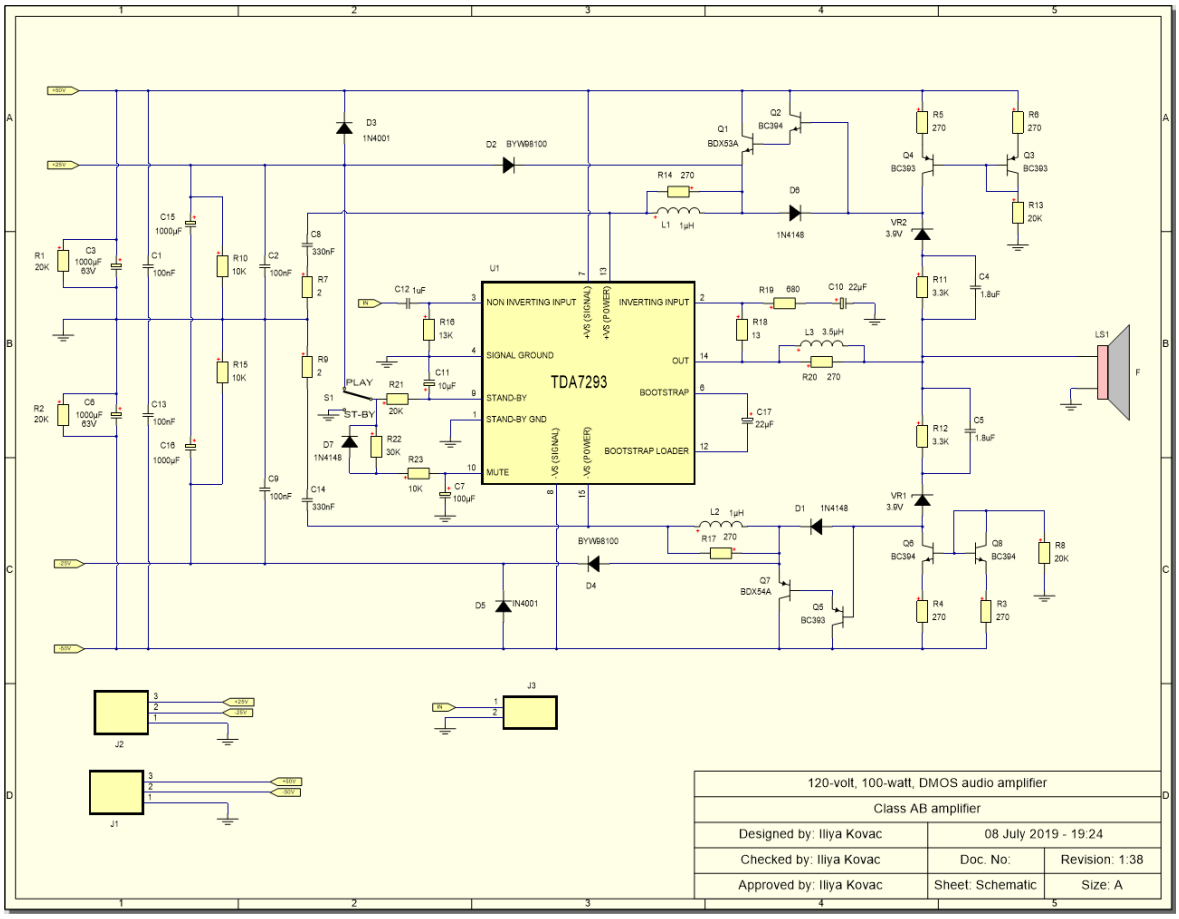
Click to open the **AutoTRAX DEX UNO** sample project.



AutoTRAX DEX UNO

Amplifier

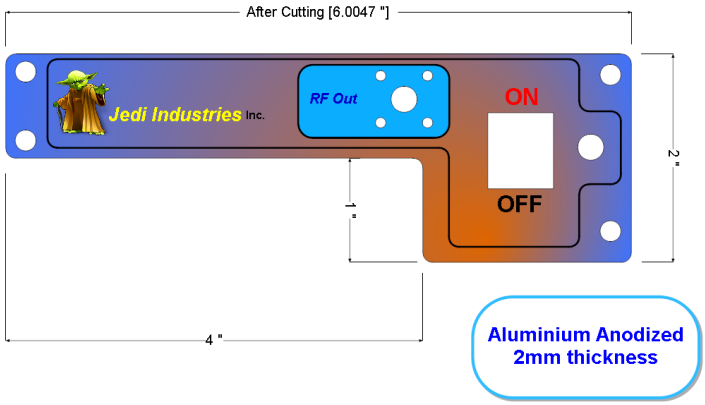
An amplifier.



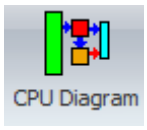
The Main Schematic for the Amplifier

Front Panel

A front panel demonstrating AutoTRAX DEX's graphics capability.

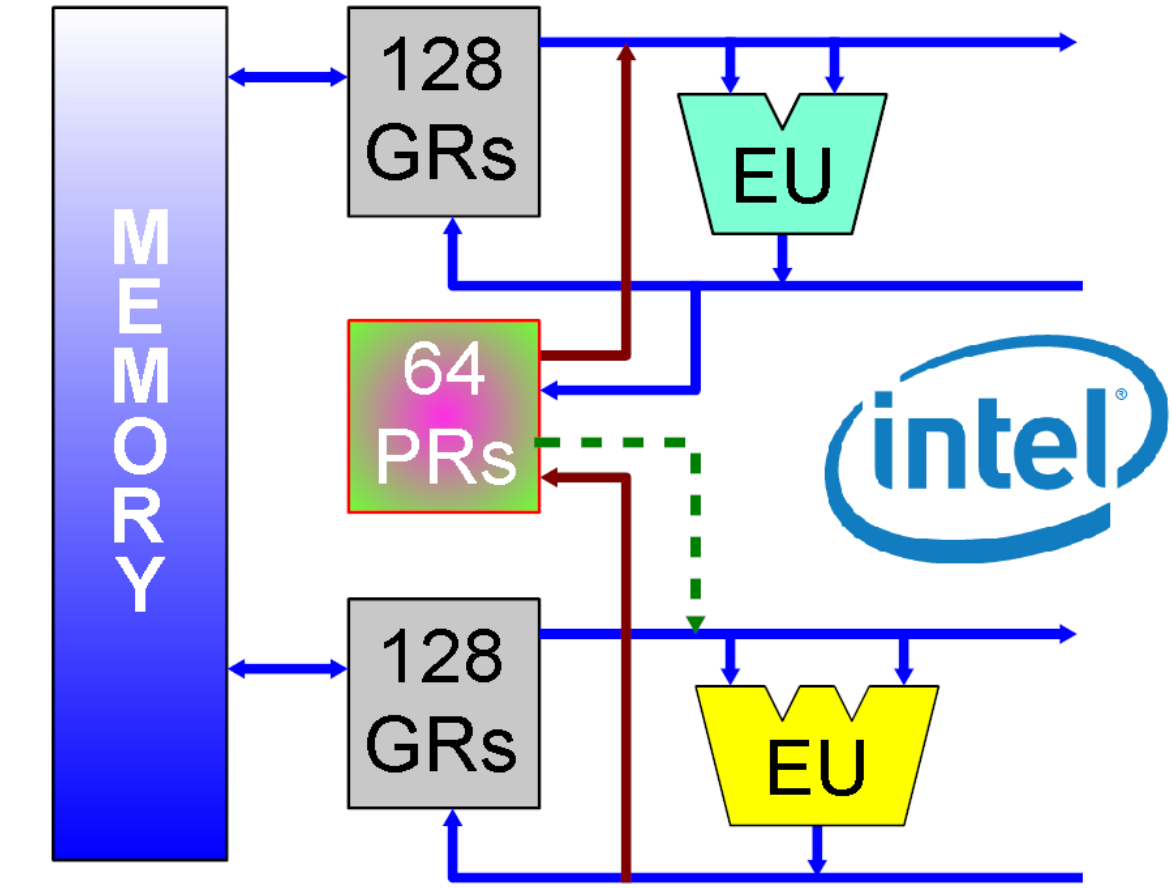


Front Panel

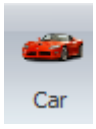


CPU Logic Diagram

A CPU logic diagram demonstrating AutoTRAX DEX's graphics capability.

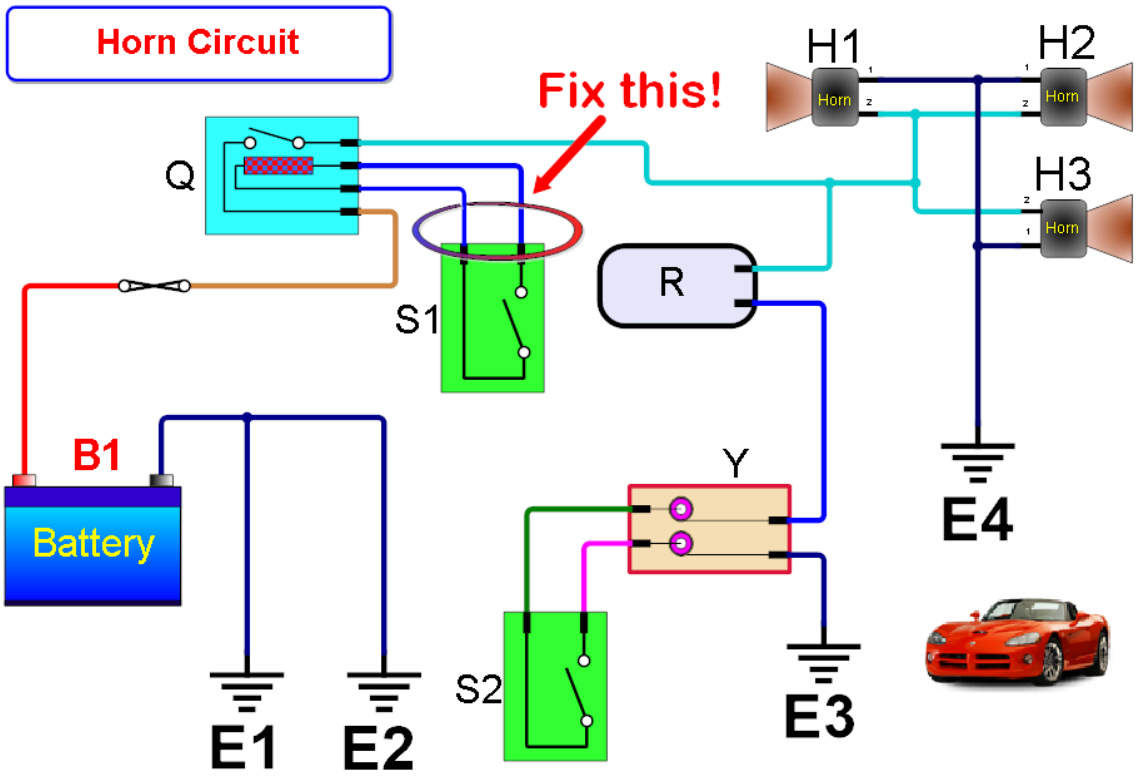


CPU Diagram

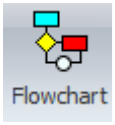


Automotive Car Harness

This project shows you part of the electrical circuit in an automobile such as you might see in a typical user manual.

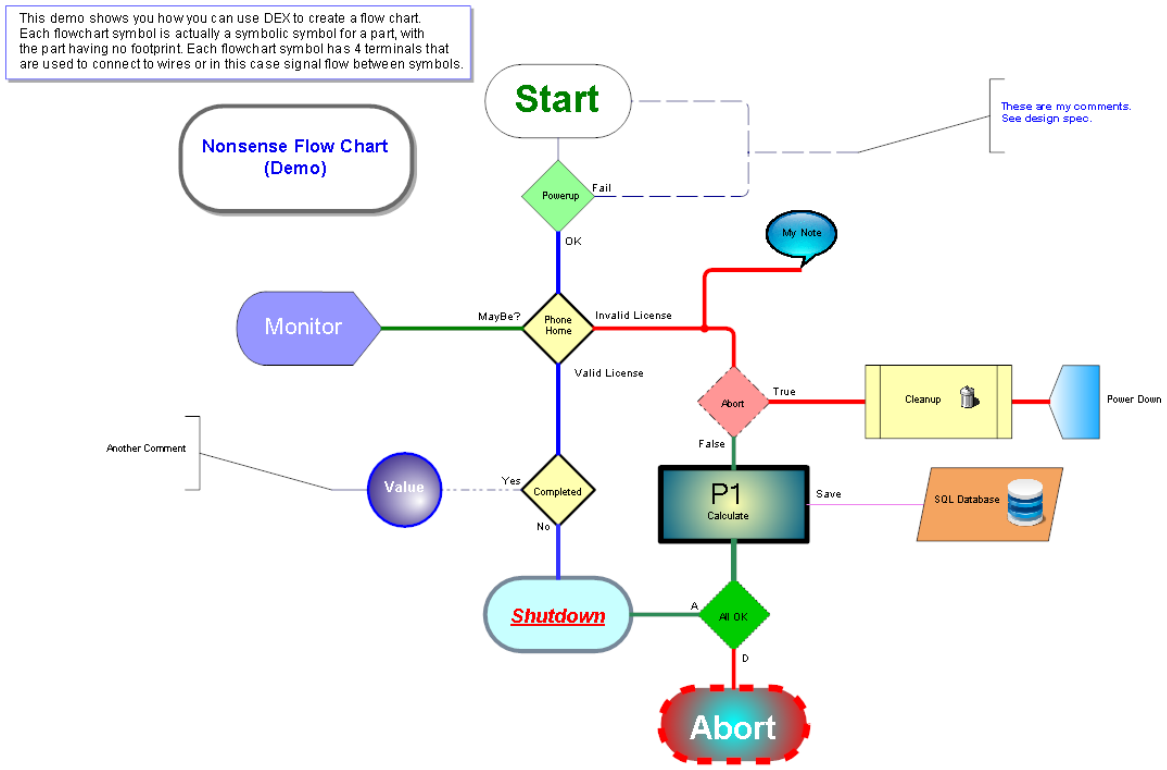


Car Electrical Harness

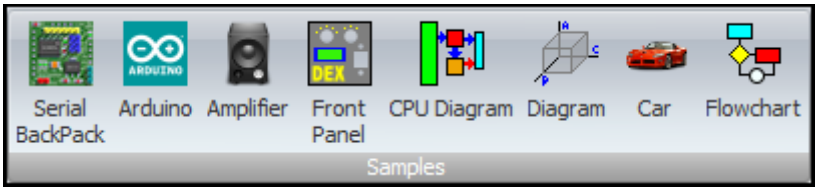


Flowchart

A flowchart demonstrating AutoTRAX DEX's graphics capability.



Flowchart



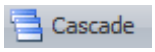
Sample Commands

1.3.2.3 The Panels Ribbon Page

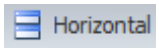
The Panels Ribbon Page

The Windows Command Group

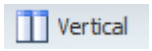
Tabbled



Cascade

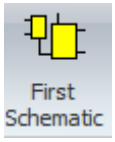


Horizontal

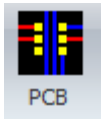


Vertical

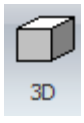
Add Window



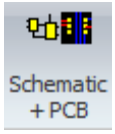
First Schematic



PCB

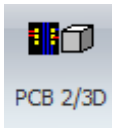


3D

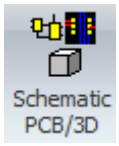


Schematic and PCB

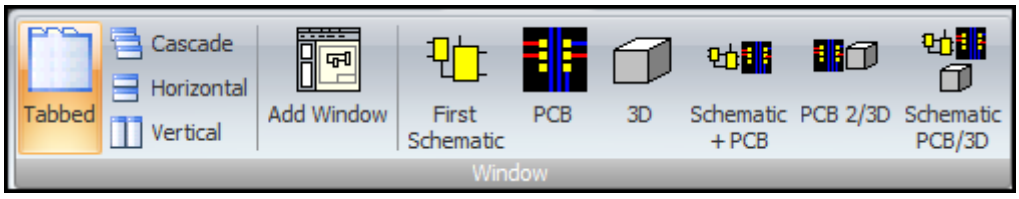
TODO



PCB and 3D

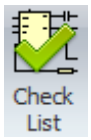


Schematic, PCB and 3D



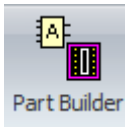
Window Commands

The Panels Command Group



Checklist

Read More...

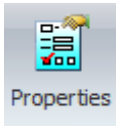


Part Builder

Read More...

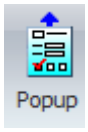
Project

Read More...



Properties

Read More...

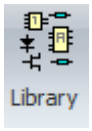


Popup Properties

Read More...

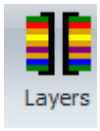
Spice Model Properties

Read More...



Library

Read More...



Layers

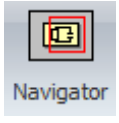
Read More...

Route

Read More...

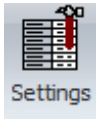
DRC

Read More...



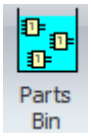
Navigator

Read More...



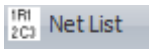
Settings

Read More...



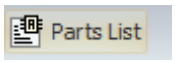
Parts Bin

Read More...



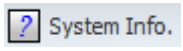
Net List

Read More...



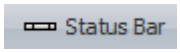
Parts List

Read More...



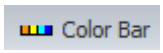
System Information

Read More...



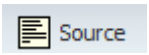
Status Bar

Read More...



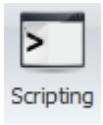
Color Bar

Read More...



Source

Read More...

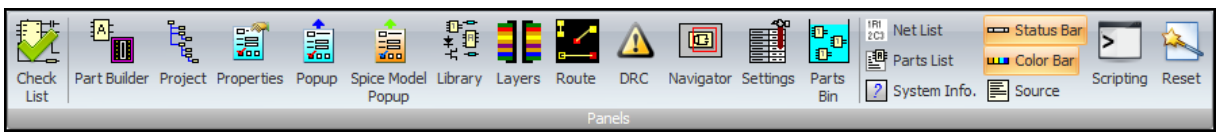


Scripting

Read More...

Reset

Click to reset all the panel layouts.



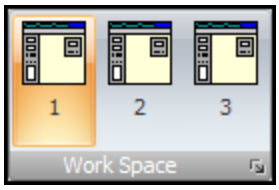
Panels Commands

TheWork Space Command Group

Workspace 1

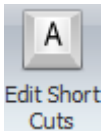
Workspace 2

Workspace 3



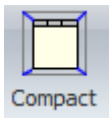
Workspace Commands

The Menu Command Group



Edit ShortCuts

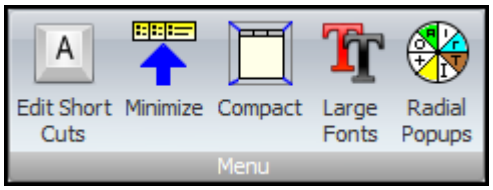
Minimize



Compact

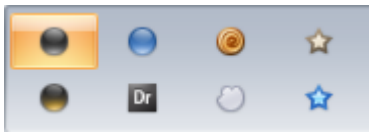
Large Fonts

Radial Popups

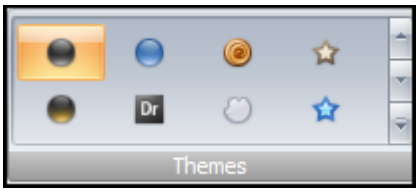


Menu Commands

The Themes Command Group

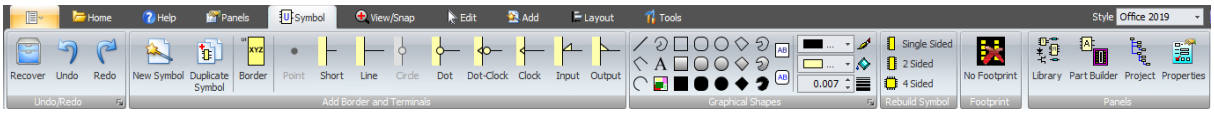


Set the Theme



Theme Commands

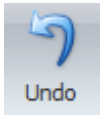
1.3.2.4 The Symbol Ribbon Page



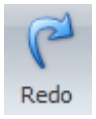
The Symbol Ribbon Page

Undo/Redo Command Group

Recover



Undo

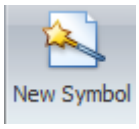


Redo

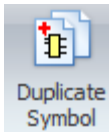
Click on the small button at the bottom right of the command group to display the Undo Popup.

Undo/Redo Commands

Add Border and Terminals Command Group

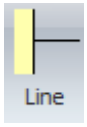


New Symbol



Duplicate Symbol

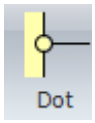
Border



Line

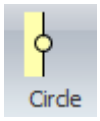
Medium

Short

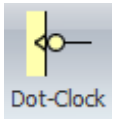


Dot

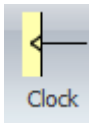
Point



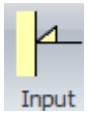
Circle



Dot-Clock

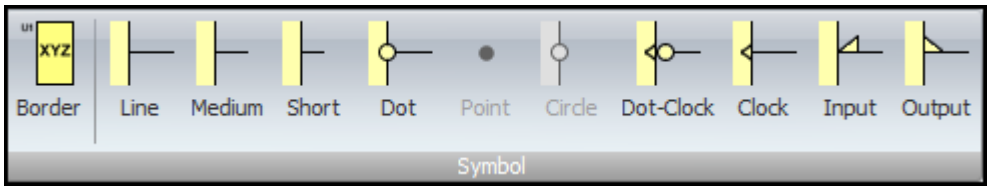


Clock



Input

Output



Add Terminals and Symbols Commands

Graphical Shapes Command Group

Line

Polyline

 **Arc**

 **Curve**

 **Text**

 **Image**

 **Hollow Rectangle**

 **Filled Rectangle**

 **Solid Rectangle**

 **Hollow Rounded Rectangle**

 **Filled Rounded Rectangle**

 **Solid Rounded Rectangle**

 **Hollow Ellipse / Circle**

 **Filled Ellipse / Circle**

 **Solid Ellipse / Circle**

 **Hollow Polygon**

 **Filled Polygon**

 **Solid Polygon**

 **Hollow Closed Curve**

 **Filled Closed Curve**

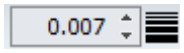
 **Solid Closed Curve**

 **Note**

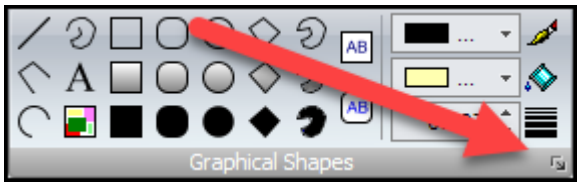
 **Note With Rounded Corners**

 **Line Style**

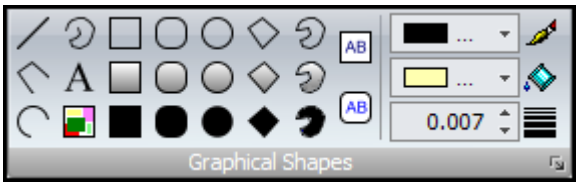
 **Fill Style**



Line Width

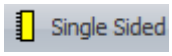


Click on the small button at the bottom right of the command group to display the Shapes Default Popup.

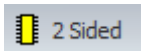


Graphical Shapes Commands

Rebuild Command Group

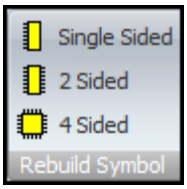


Single Sided



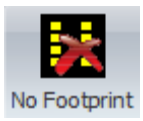
2 Sided

1 Sided

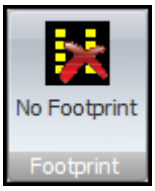


Rebuild Commands

Footprint Command Group

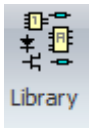


No Footprint



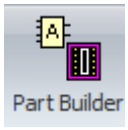
Footprint Commands

Panels Command Group



Library

Read More...

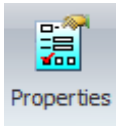


Part Builder

Read More...

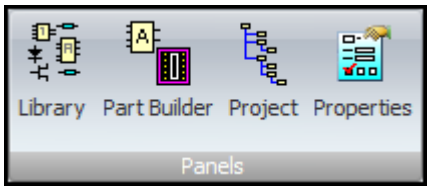
Project

Read More...



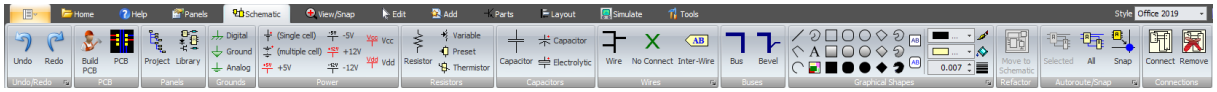
Properties

Read More...



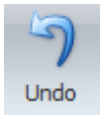
Panels Commands

1.3.2.5 The Schematic Ribbon Page

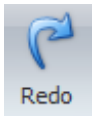


The Schematic Ribbon Page

Undo/Redo Command Group



Undo

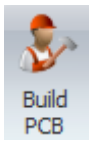


Redo

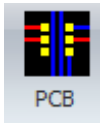
Click on the small button at the bottom right of the command group to display the Undo Popup

Undo/Redo Commands

PCB Command Group



Build PCB



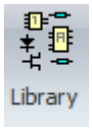
PCB

PCB Commands

Panels Command Group

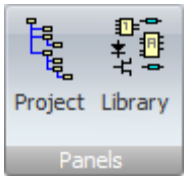
Project

Read More...



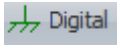
Library

Read More...

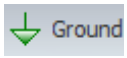


Panels Commands

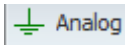
Ground Command Group



Digital



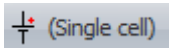
Ground



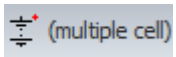
Analog

Ground Commands

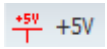
Power Command Group



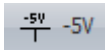
Single Cell



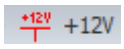
Multiple Cell



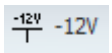
+5V



-5V

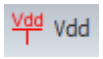


+12V



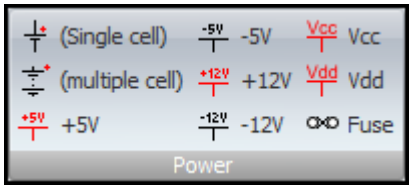
-12V

Vcc



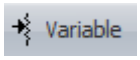
Vdd

Fuse

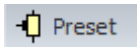


Power Commands

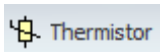
Resistors Command Group



Variable

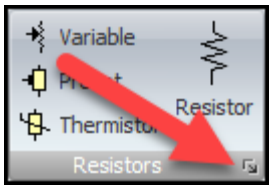


Preset

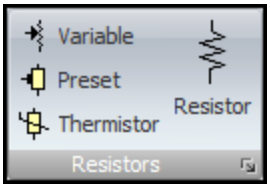


Thermistor

Resistor

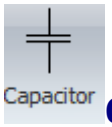


Click on the small button at the bottom right of the command group to display the Add Parts Settings Popup.

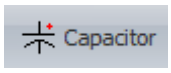


Resistors Commands

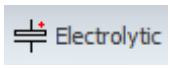
Capacitors Command Group



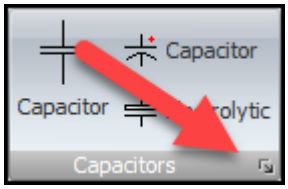
Capacitor



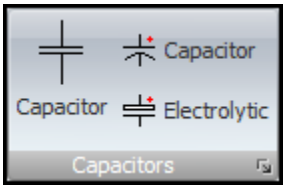
Capacitor



Electrolytic



Click on the small button at the bottom right of the command group to display the Add Parts Settings Popup..



Capacitors Commands

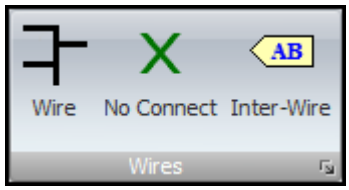
Wires Command Group

Wire

No Connect

Inter-Wire

Click on the small button at the bottom right of the command group to display the Wire and Bus Settings Popup.



Wires Commands

Buses Command Group

Bus

Bevel

Click on the small button at the bottom right of the command group to display the Wire and Bus Settings Popup.

Buses Commands

Graphical Shapes Command Group

Line

Polyline

Arc

Curve

Text

Image

Hollow Rectangle

Filled Rectangle

Solid Rectangle

Hollow Rounded Rectangle

 **Filled Rounded Rectangle**

 **Solid Rounded Rectangle**

 **Hollow Ellipse / Circle**

 **Filled Ellipse / Circle**

 **Solid Ellipse / Circle**

 **Hollow Polygon**

 **Filled Polygon**

 **Solid Polygon**

 **Hollow Closed Curve**

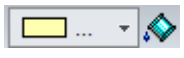
 **Filled Closed Curve**

 **Solid Closed Curve**

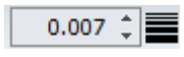
 **Note**

 **Note With Rounded Corners**

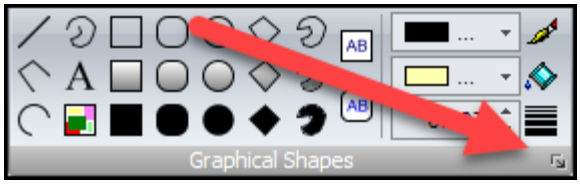
Line Style



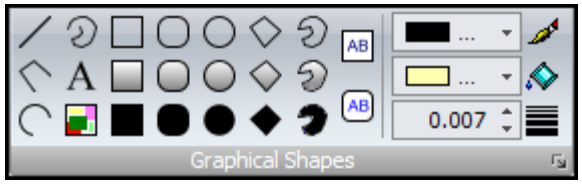
Fill Style



Line Width



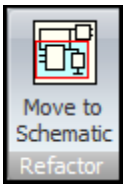
Click on the small button at the bottom right of the command group to display the Shapes Default Popup.



Graphical Shapes Commands

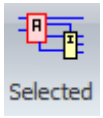
Refactor Command Group

Move to Schematic

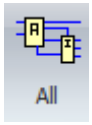


Refactor Commands

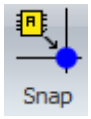
Autoroute/Snap Command Group



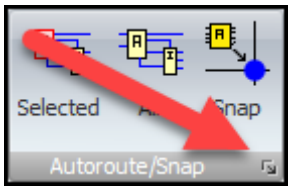
Selected



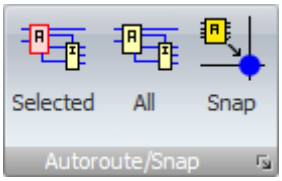
All



Snap

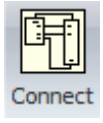


Click on the small button at the bottom right of the command group to display the Wire and Bus Settings Popup.

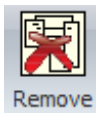


Autoroute/Snap Commands

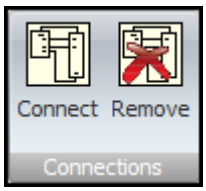
Connections Command Group



Connect

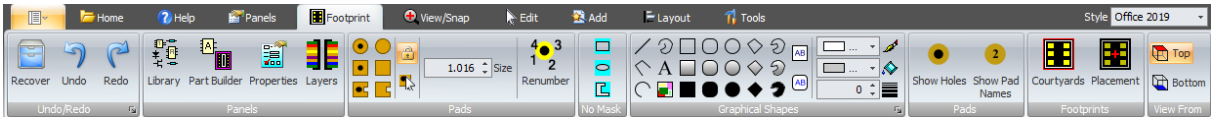


Remove



Connections Commands

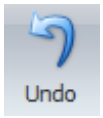
1.3.2.6 The Footprint Ribbon Page



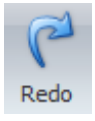
The Footprint Ribbon Page

Undo/Redo Command Group

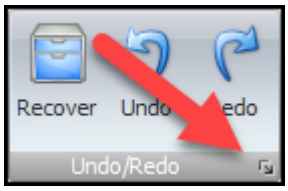
Recover



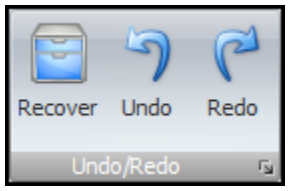
Undo



Redo

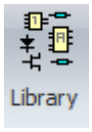


Click on the small button at the bottom right of the command group to display the Undo Popup.



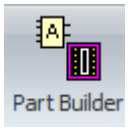
Undo/Redo Commands

The Panels Command Group



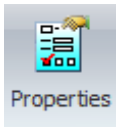
Library

Read More...



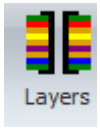
Part Builder

Read More...



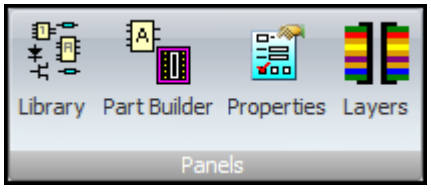
Properties

Read More...



Layers

Read More...



Panels Commands

Pads Command Group

 Round/Elliptical TPH Pad

 Square/Rectangular TPH Pad

 Polygonal TPH Pad

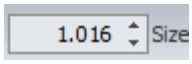
 Round/Elliptical SMT Pad

 Square/Rectangular SMT Pad

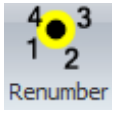
 Polygonal SMT Pad

 Lock Pad Width to Height

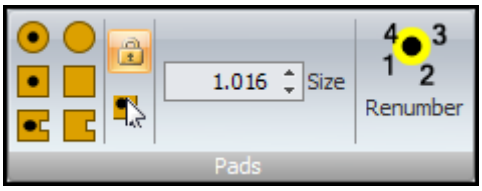
 Add Preset or Interactive



Pad Size Pad Size



Renumber



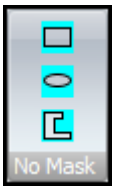
Pads Commands

No Mask Command Group

Square/Rectangular

Circle/Ellipse

Polygonal



No Mask Commands

Graphical Shapes Command Group

Line

Polyline

Arc

Curve

 **Text**

 **Image**

 **Hollow Rectangle**

 **Filled Rectangle**

 **Solid Rectangle**

 **Hollow Rounded Rectangle**

 **Filled Rounded Rectangle**

 **Solid Rounded Rectangle**

 **Hollow Ellipse / Circle**

 **Filled Ellipse / Circle**

 **Solid Ellipse / Circle**

 **Hollow Polygon**

 **Filled Polygon**

Solid Polygon

Hollow Closed Curve

Filled Closed Curve

Solid Closed Curve

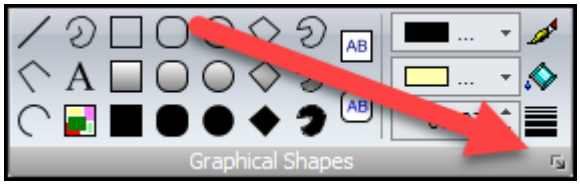
Note

Note With Rounded Corners

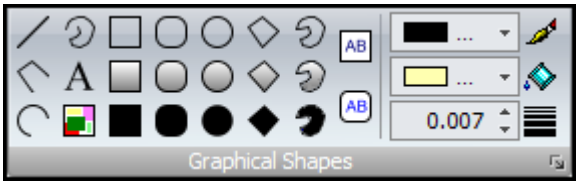
Line Style

Fill Style

Line Width

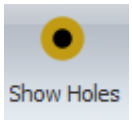


Click on the small button at the bottom right of the command group to display the Shapes Default Popup.

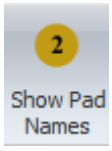


Graphical Shapes Commands

Pads Command Group



Show Holes



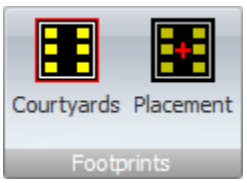
Show Pad Names

Pads Commands

Footprints Command Group

Courtyards

Placement

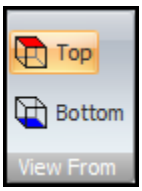


Footprints Commands

View From Command Group

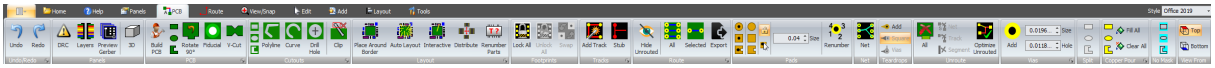
Top

Bottom



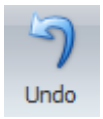
View From Commands

1.3.2.7 The PCB Ribbon Page

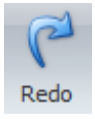


The PCB Ribbon Page

Undo/Redo Command Group



Undo



Redo

Click on the small button at the bottom right of the command group to display the Undo Popup.

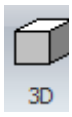
Undo/Redo Commands

Panels Command Group

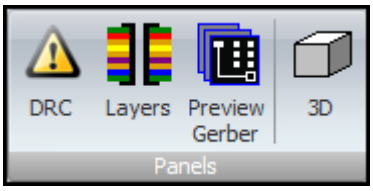
DRC

Layers

Preview Gerber

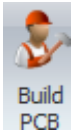


3D



Panels Commands

PCB Command Group

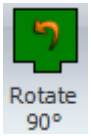


Build PCB

Rectangular PCB

Circular/Elliptical PCB

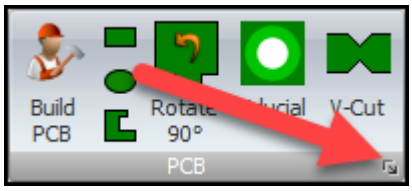
Polygonal PCB



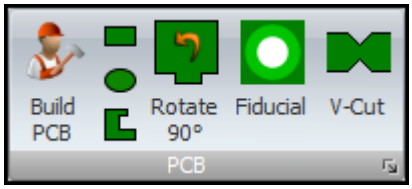
Rotate 90°

Fiducial

V-Cut



Click on the small button at the bottom right of the command group to display the Fiducial Popup.



PCB Commands

Cutouts Command Group

Retabgular

Circular/Elliptical

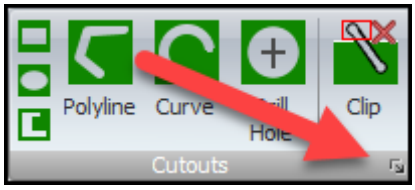
Polygon

Polyline

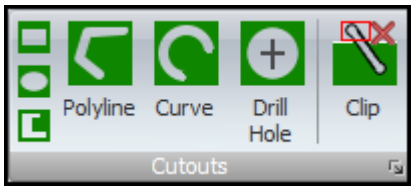
Curve

Drill Hole

Clip



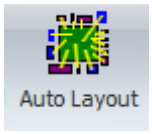
Click on the small button at the bottom right of the command group to display the Cutouts Popup.



Cutouts Commands

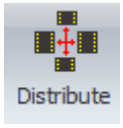
Layout Command Group

Place Around Border



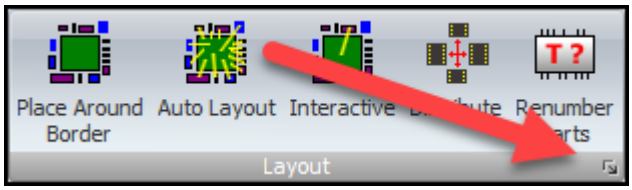
Auto-Layout

Interactive

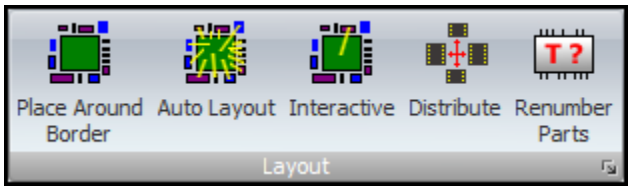


Distribute

Renumber Parts



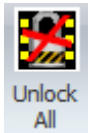
Click on the small button at the bottom right of the command group to display the Layout Settings Popup.



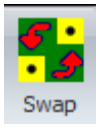
Layout Commands

Footprints Command Group

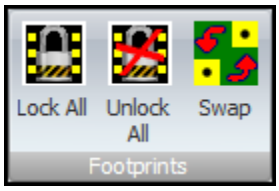
Lock



Unlock All

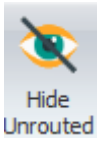


Swap



Footprints Commands

Route Command Group



Hide Unrouted

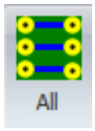
When manually routing a PCB, it often difficult to find a path because the un-routed track segments from other nets obscure you vision; You cannot *'see the wood for the trees'*. This video on the right shows you how AutoTRAX DEX solves this problem by using auto-dimming and auto-hiding.

If the button is checked and a net/track has been chosen, then all unrouted track segments on all nets are hidden except for the chosen net/track.

You can chose a net/track by:

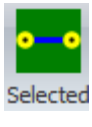
- Clicking on the net in the viewport or
- Click on the net's row in the Route Panel.

If unchecked the all unrouted track segments will be visible if no net/track is selected.



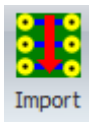
All

Auto-route all nets.



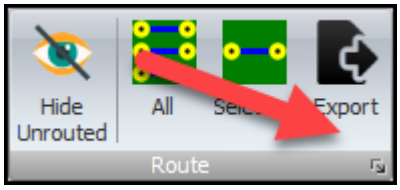
Selected

Auto-route the selected net.

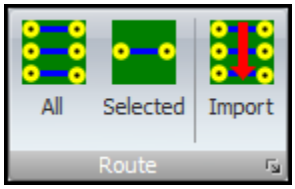


Import

Import an Electra route file.



Click on the small button at the bottom right of the command group to display the Router Settings Popup.



Route Commands

Pads Command Group

Round/Elliptical TPH Pad

Square/Rectangular TPH Pad

Polygonal TPH Pad

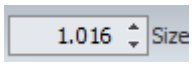
Round/Elliptical SMT Pad

Square/Rectangular SMT Pad

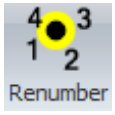
Polygonal SMT Pad

Lock Pad Width to Height

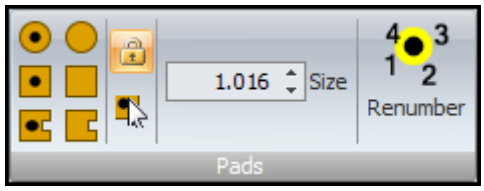
Add Preset or Interactive



Pad Size Pad Size

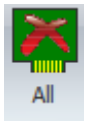


Renumber



Pads Commands

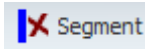
Unroute Command Group



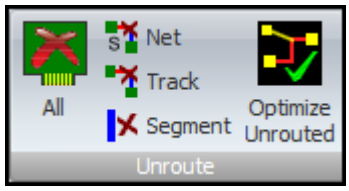
All

Net

Track

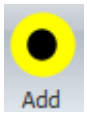


Segment

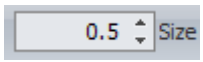


Unroute Commands

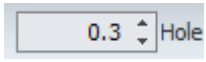
Vias Group



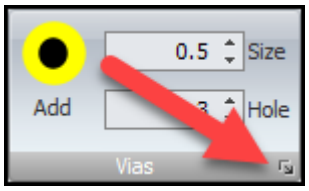
Add



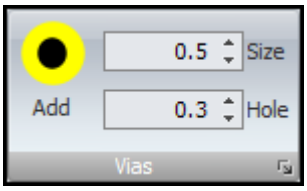
Size



X



Click on the small button at the bottom right of the command group to display the Track Via Settings Popup.



Vias Commands

Split Command Group

Rectangular Split Power Plane

Circular/Elliptical Split Power Plane

Polygonal Split Power Plane

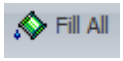
Split Commands

Copper Pour Command Group

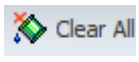
Rectangular Copper Pour

Circular/Elliptical Copper Pour

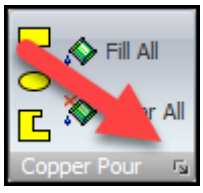
Polygonal Copper Pour



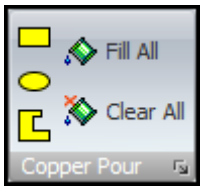
Fill All



Clear All



Click on the small button at the bottom right of the command group to display the Copper Pour Settings Popup.



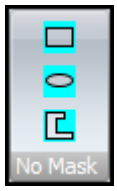
Copper Pour Commands

No Mask Command Group

 **Square/Rectangular**

 **Circle/Ellipse**

 **Polygonal**

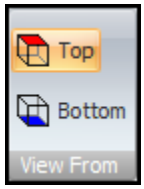


No Mask Commands

View From Command Group

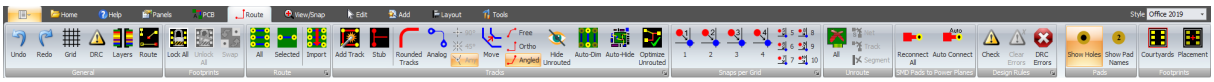
Top

Bottom



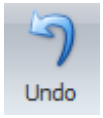
View From Commands

1.3.2.8 The Route Ribbon Page

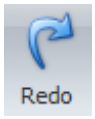


The Route Ribbon Page

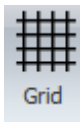
General Command Group



Undo



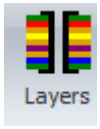
Redo



Show/Hide the Grid

DRC

Read More...

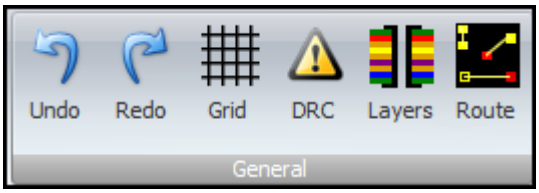


Layers

Read More...

Route

Read More...

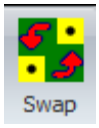


General Commands

Footprints Command Group

Lock

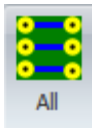
Unlock All



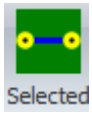
Swap

Footprints Commands

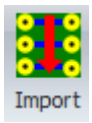
Route Command Group



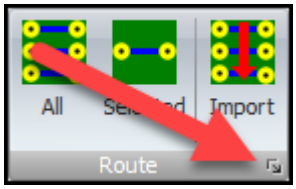
All

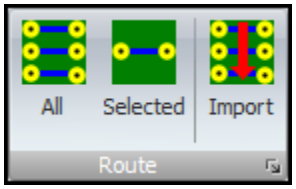


Selected



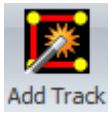
Import





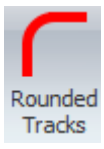
Route Commands

Tracks Command Group

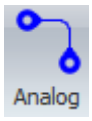


Add Track

Stub



Rounded Tracks



Analog

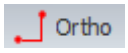
90°

45°

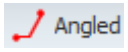
Any

Move

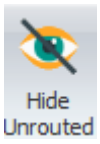
Free



Ortho



Angled



Hide Unrouted

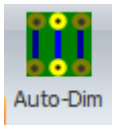
When manually routing a PCB, it often difficult to find a path because the un-routed track segments from other nets obscure you vision; You cannot *'see the wood for the trees'*. This video on the right shows you how AutoTRAX DEX solves this problem by using auto-dimming and auto-hiding.

If the button is checked and a net/track has been chosen, then all unrouted track segments on all nets are hidden except for the chosen net/track.

You can chose a net/track by:

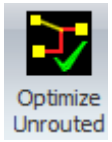
- Clicking on the net in the viewport or
- Click on the net's row in the Route Panel.

If unchecked the all unrouted track segments will be visible if no net/track is selected.



Auto-Dim

Auto-Hide

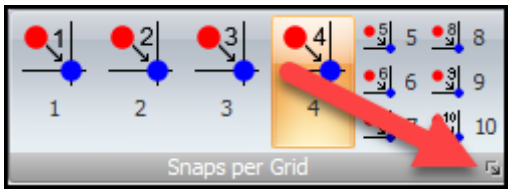


Optimize Unrouted

Tracks Commands

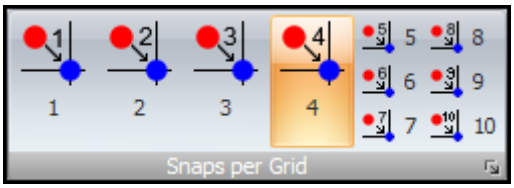
Snaps per Grid Command Group

Each button displays the number of snaps per grid.



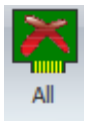
Snap Settings

Click on the small button at the bottom right of the command group to display the Snap Settings popup.



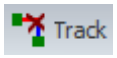
Snaps per Grid Commands

Unroute Command Group

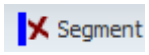


All

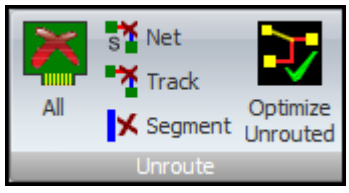
Net



Track

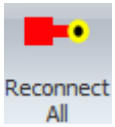


Segment

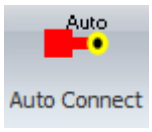


Unroute Commands

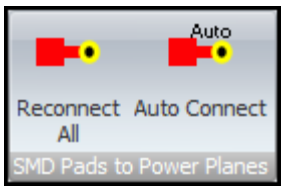
SMD Pads to Powe Planes Command Group



Reconnect All



Auto Connect

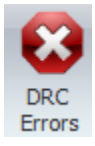


SMD Pads to Power Planes Commands

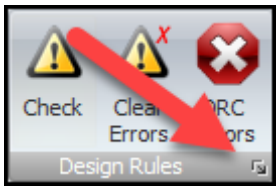
Design Rules Command Group

Check

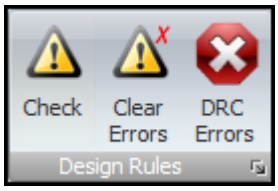
Clear



DRC Errors

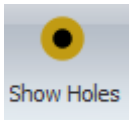


Click on the small button at the bottom right of the command group to display the Design Rules Popup.

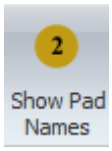


Design Rules Commands

Pads Command Group



Show Holes

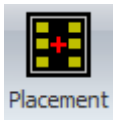


Show Pad Names

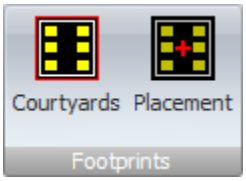
Pads Commands

Footprints Command Group

Courtyards

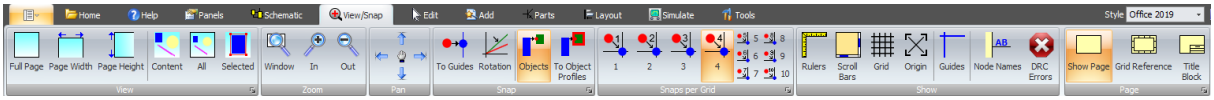


Placement



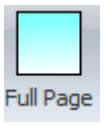
Footprints Commands

1.3.2.9 The View/Snap Ribbon Page

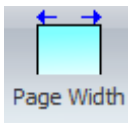


The View/Snap Ribbon Page

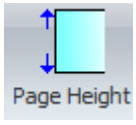
View Command Group



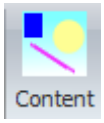
View Full Page



View Page Width

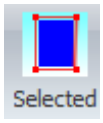


View Page Height

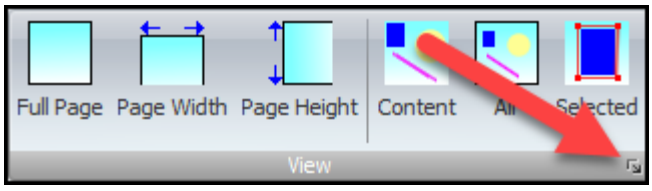


View Content

View All

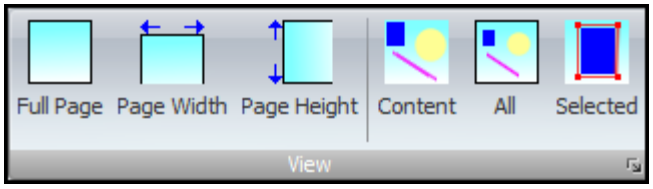


View Selected



Page Settings

Click on the small button at the bottom right of the command group to display the Page Settings popup.

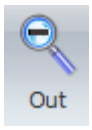


View Commands

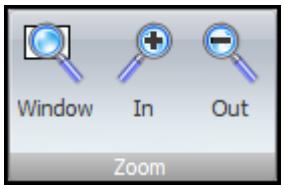
Zoom Command Group

Zoom Window

Zoom In



Zoom Out



Zoom Commands

Pan Command Group

Pan Left

Pan Right

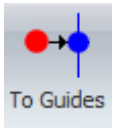
Pan Up

Pan Down

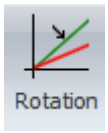
Pan

Pan Commands

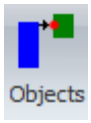
Snap Command Group



Snap to Guides



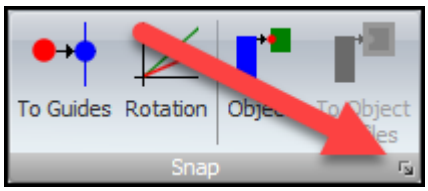
Snap Rotation



Snap to Objects

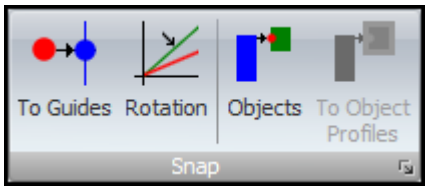
Snap to Object Profiles

This is only enabled if Snap to Objects is enabled.



Snap Settings

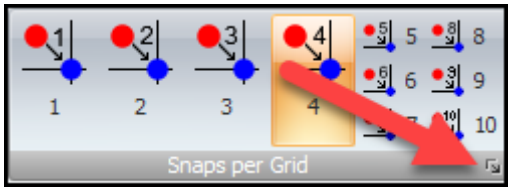
Click on the small button at the bottom right of the command group to display the Snap Settings popup.



Snap Commands

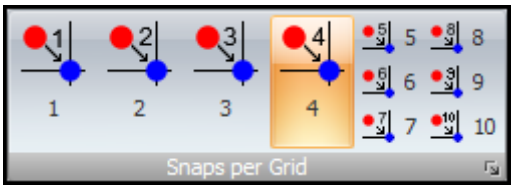
Snaps per Grid Command Group

Each button displays the number of snaps per grid.



Snap Settings

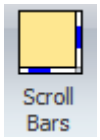
Click on the small button at the bottom right of the command group to display the Snap Settings popup.



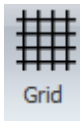
Snaps per Grid Commands

Show Command Group

Show/Hide Rulers



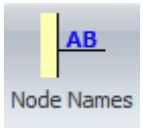
Show/Hide the Scroll Bars



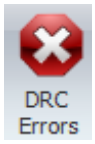
Show/Hide the Grid

Show/Hide the Origin

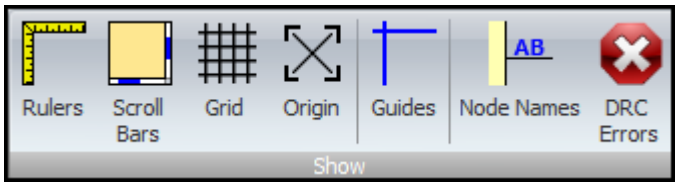
Show/Hide the Guides



Show/Hide Node Names

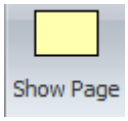


Show/Hide DRC Errors

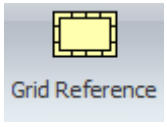


Show Commands

Page Command Group

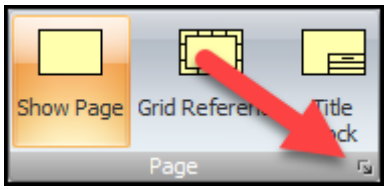


Show Page



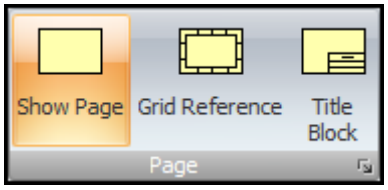
Show the Grid Reference

Show the Title Block



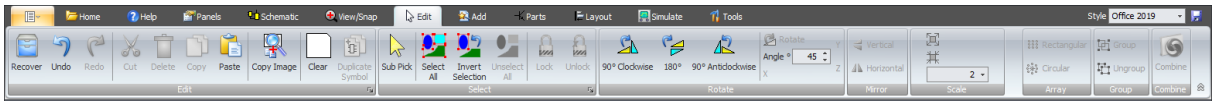
Page Settings

Click on the small button at the bottom right of the command group to display the Page Settings popup.



Page Commands

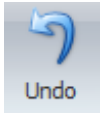
1.3.2.10 The Edit Ribbon Page



The Edit Ribbon Page

Edit Command Group

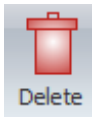
Recover



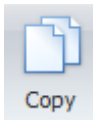
Undo

Redo

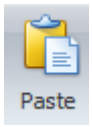
Cut



Delete



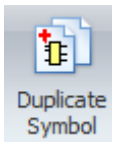
Copy



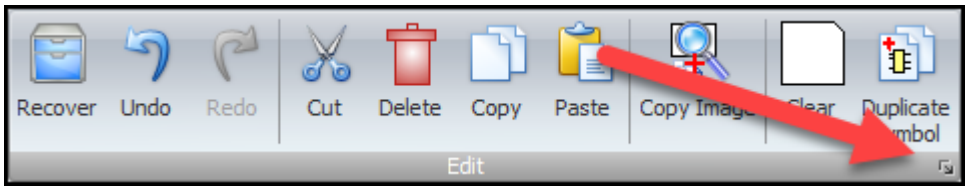
Paste

Copy Image

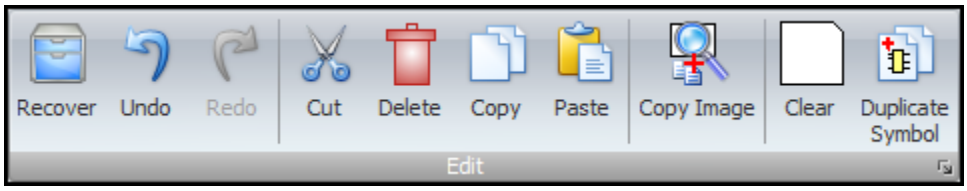
Clear



Duplicate Symbol



Click on the small button at the bottom right of the command group to display the Undo Popup.

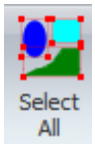


Edit Commands

Select/Lock Command Group

Sub Pick

Pick



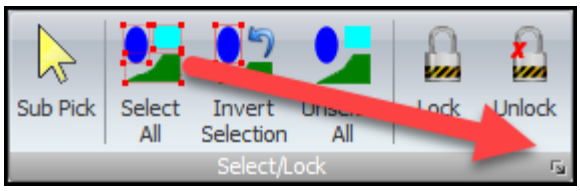
Select All

Invert Selection

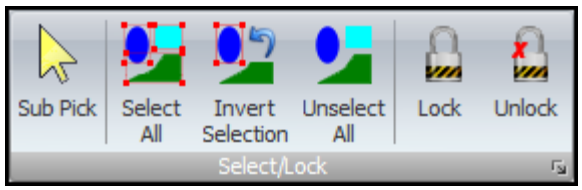
Deselect All

Lock

Unlock

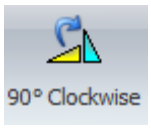


Click on the small button at the bottom right of the command group to display the Selection Settings Popup.



Select/Lock Commands

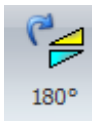
Rotate Command Group



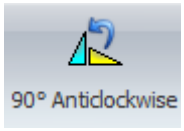
Rotate Clockwise 90°

To rotate an object by 90°

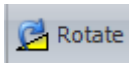
1. Select the object.
2. Click the button above.



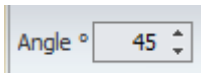
Rotate180°



Rotate Anticlockwise 90°

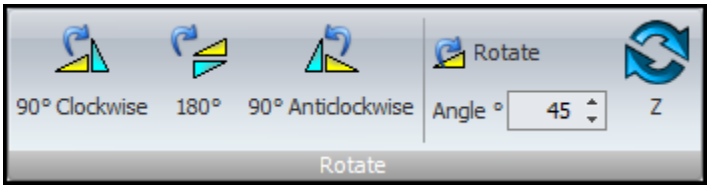


Turn Rotate Snap On/Off



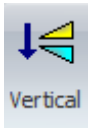
Rotate Snap Angle

Rotate About the Z Axis



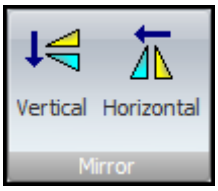
Rotate Commands

Mirror Command Group



Mirror About the Selected Entities Horizontal Axis

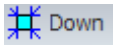
Mirror About the Selected Entities Vertical Axis



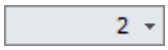
Mirror Commands

Scale Command Group

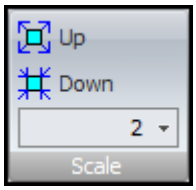
Scale Up



Scale Down

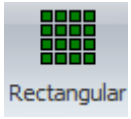


Scale Factor



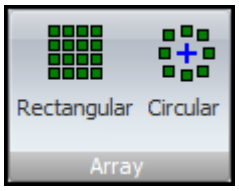
Scale Commands

Array Command Group



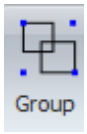
Create a Rectangular Array From the Selected Entities

Create a Circular Array From the Selected Entities

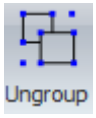


Array Commands

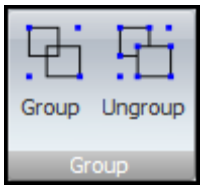
Group Command Group



Group the Selected Entities

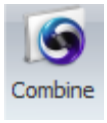


Ungroup the Selected Entities

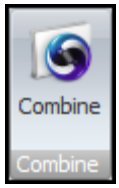


Group Commands

Combine Command Group

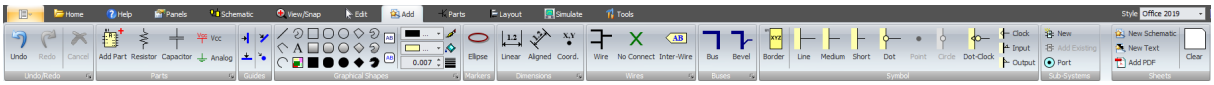


Combine the Selected Entities



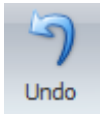
Combine Commands

1.3.2.11 The Add Ribbon Page

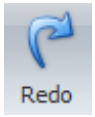


The Add Ribbon Page

Undo/Redo Command Group



Undo



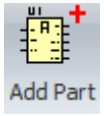
Redo

Cancel

Click on the small button at the bottom right of the command group to display the Undo Popup

Undo/Redo Commands

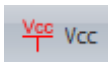
Parts Command Group



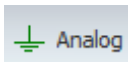
Add Part

Resistor

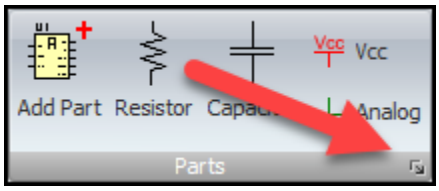
Capacitor



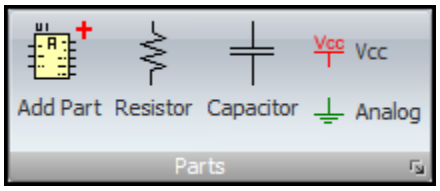
Vcc



Analog



Click on the small button at the bottom right of the command group to display the Add Parts Popup.



Parts Commands

Guides Command Group

Vertical Guide

Horizontal Guide

Angled Guide

Point Guide

Guides Commands

Graphical Shapes Command Group

Line

Polyline

Arc

Curve

Text

Image

 **Hollow Rectangle**

 **Filled Rectangle**

 **Solid Rectangle**

 **Hollow Rounded Rectangle**

 **Filled Rounded Rectangle**

 **Solid Rounded Rectangle**

 **Hollow Ellipse / Circle**

 **Filled Ellipse / Circle**

 **Solid Ellipse / Circle**

 **Hollow Polygon**

 **Filled Polygon**

 **Solid Polygon**

 **Hollow Closed Curve**

Filled Closed Curve

Solid Closed Curve

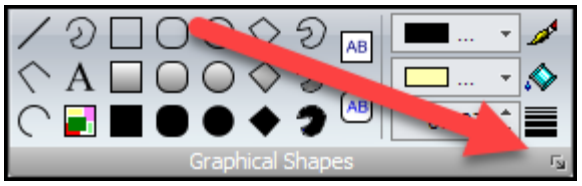
Note

Note With Rounded Corners

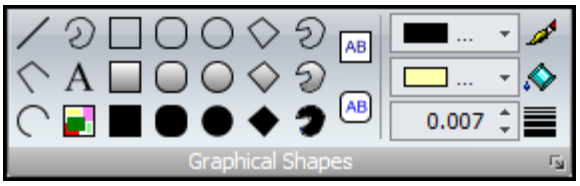
Line Style

Fill Style

Line Width

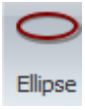


Click on the small button at the bottom right of the command group to display the Shapes Default Popup.

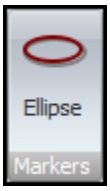


Graphical Shapes Commands

Markers Command Group

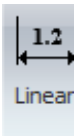


Ellipse



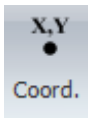
Markers Commands

Dimensions Command Group

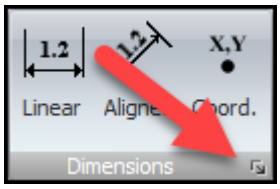


Linear

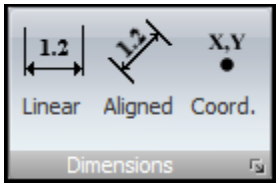
Aligned



Coord.



Click on the small button at the bottom right of the command group to display the Dimension Settings Popup.



Dimensions Commands

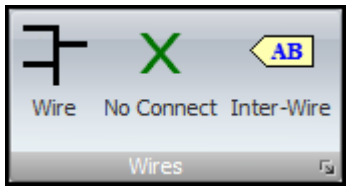
Wires Command Group

Wire

No Connect

Inter-Wire

Click on the small button at the bottom right of the command group to display the Wire and Bus Settings Popup.



Wires Commands

Buses Command Group

Bus

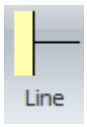
Bevel

Click on the small button at the bottom right of the command group to display the Wire and Bus Settings Popup.

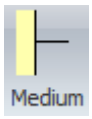
Buses Commands

Symbol Command Group

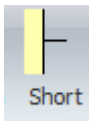
Border



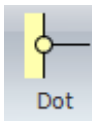
Line



Medium

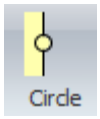


Short

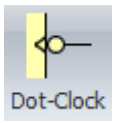


Dot

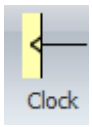
Point



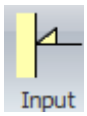
Circle



Dot-Clock

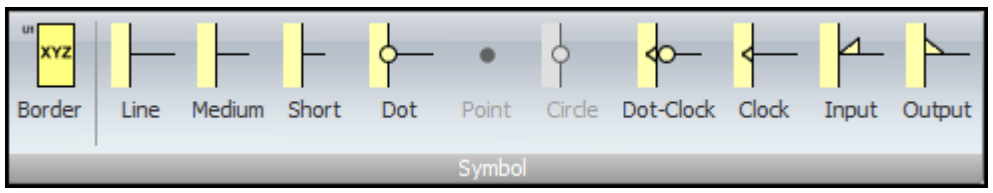


Clock



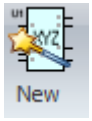
Input

Output

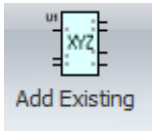


Symbol Commands

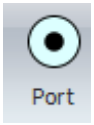
Sub Command Group



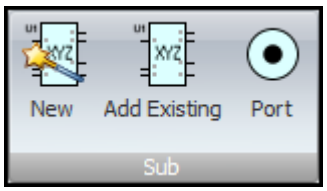
New



Add Existing

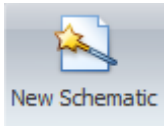


Port



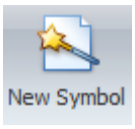
Sub Commands

Sheet Command Group



Create a New Schematic

This adds a new schematic sheet to your project..

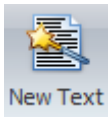


Create a New Symbol

In parts, schematics act as symbol schematics. So, if you have a part that needs two symbols, then you would have two schematic symbol sheets.

If you currently have a part open, then the additional schematic symbol sheet will be a sub-part. For instance, if you are creating a part which has 2 op-amps, you

would have 2 schematics symbol sheets in total; one for each op-amp. You might also add an additional schematic symbol sheet for the power connections.

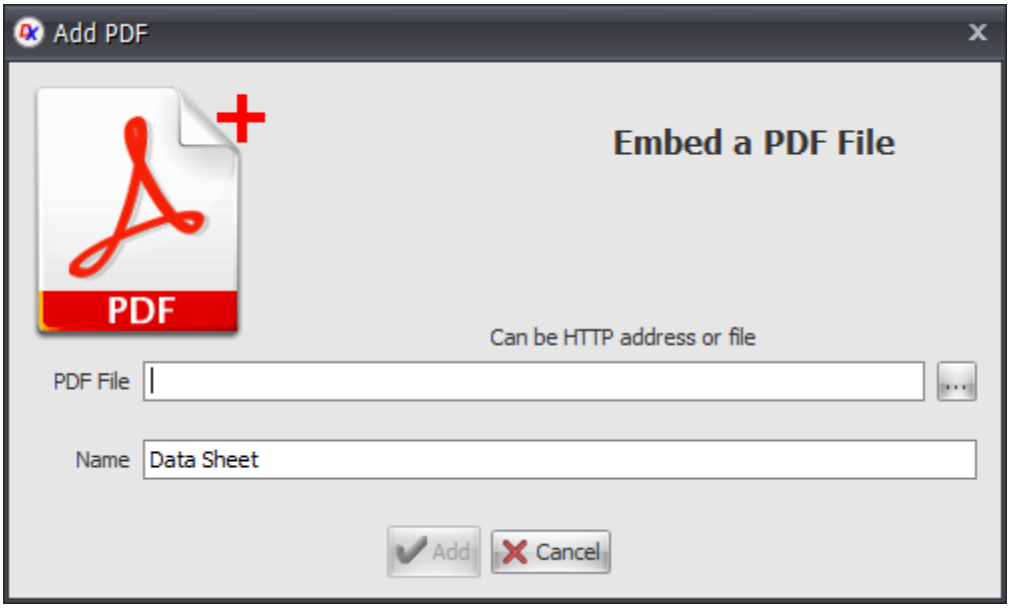


Create a New Text

This will add a new text/document sheet to your design.

Add a PDF to the Design

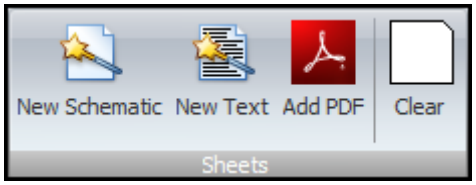
Clicking this will allow you to embed an Adobe PDF file in your project/part.



Add a PDF to the Design

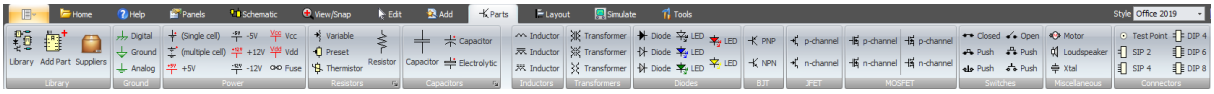
Clear

Clears the current sheet.



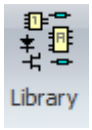
Sheet Commands

1.3.2.12 The Parts Ribbon Page

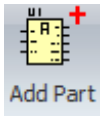


The Parts Ribbon Page

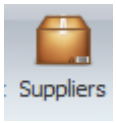
Library Command Group



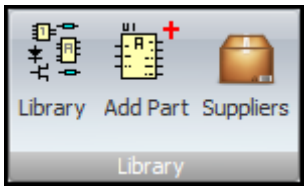
Library



Add Part

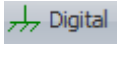


Suppliers

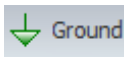


Library Commands

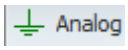
Ground Command Group



Digital



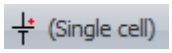
Ground



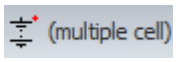
Analog

Ground Commands

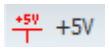
Power Command Group



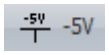
Single Cell



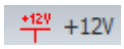
Multiple Cell



+5V



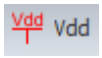
-5V



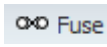
+12V

-12V

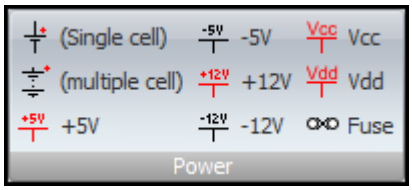
Vcc



Vdd

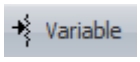


Fuse

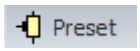


Power Commands

Resistors Command Group



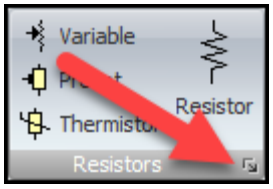
Variable



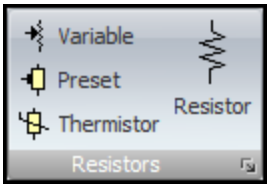
Preset

Thermistor

Resistor

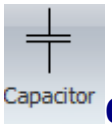


Click on the small button at the bottom right of the command group to display the Add Parts Settings Popup.

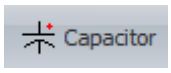


Resistors Commands

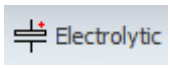
Capacitors Command Group



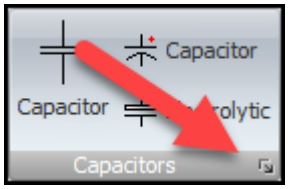
Capacitor



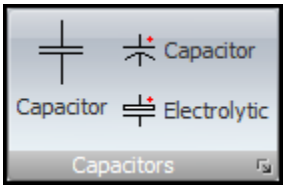
Capacitor



Electrolytic

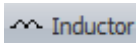


Click on the small button at the bottom right of the command group to display the Add Parts Settings Popup..

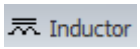


Capacitors Commands

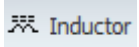
Inductors Command Group



Inductor



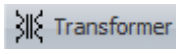
Inductor



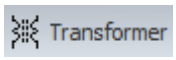
Inductor

Inductors Commands

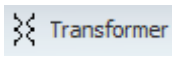
Transformers Command Group



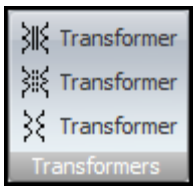
Transformer



Transformer

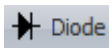


Transformer



Transformers Commands

Diodes Command Group



Diode

Diode

Diode

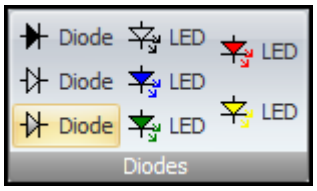
LED

LED

LED

LED

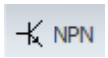
LED



Diodes Commands

BJT Command Group

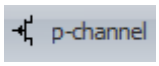
PNP



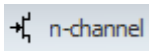
NPN

BJT Commands

JFET Command Group



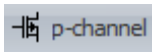
P-Channel



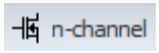
N-Channel

JFET Commands

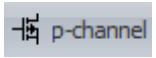
NMOS Command Group



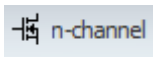
P-Channel



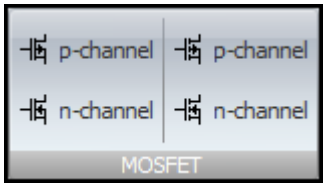
N-Channel



P-Channel

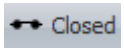


N-Channel



NMOS Commands

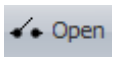
Switches Command Group



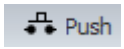
Closed

Push

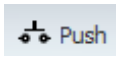
Push



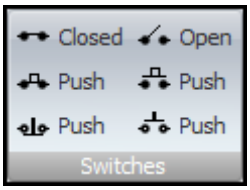
Open



Push

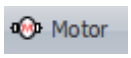


Push

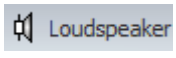


Switches Commands

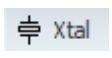
Miscellaneous Command Group



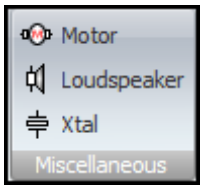
Motor



Loudspeaker

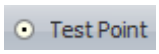


Xtal

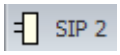


Miscellaneous Commands

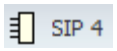
Connectors Command Group



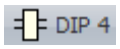
Test Point



SIP 2

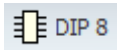


SIP 4

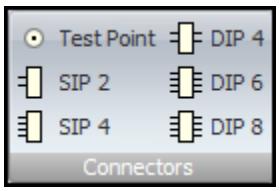


DIP 4

DIP 6

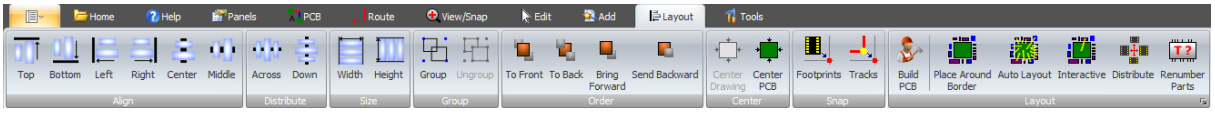


DIP 8



Connectors Commands

1.3.2.13 The Layout Ribbon Page



The Layout Ribbon Page

Align Command Group

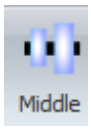
Top

Bottom

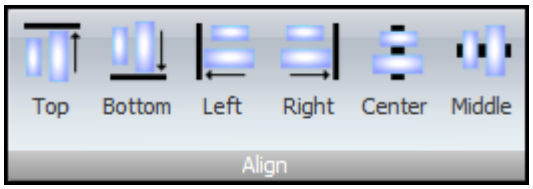
Left

Right

Center



Middle

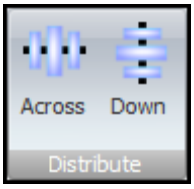


Align Commands

Distribute Command Group

Across

Down

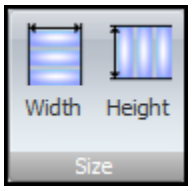


Distribute Commands

Size Command Group

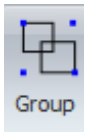
Width

Height

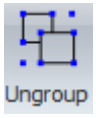


Size Commands

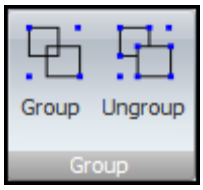
Group Command Group



Group the Selected Entities

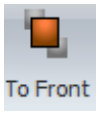


Ungroup the Selected Entities

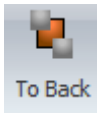


Group Commands

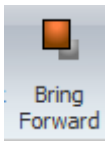
Order Command Group



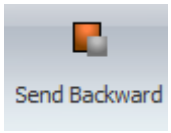
To Front



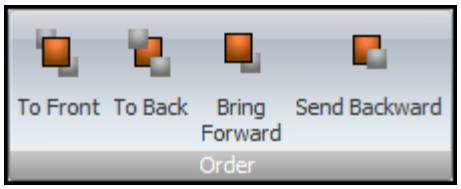
To Back



Bring Forward

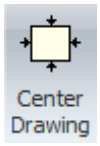


Send Backward

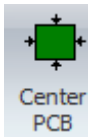


Order Commands

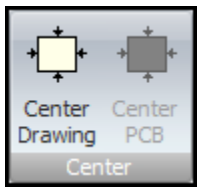
Center Command Group



Center Drawing

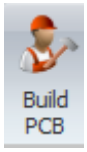


Center PCB



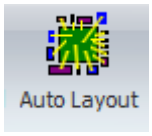
Center Commands

Layout Command Group



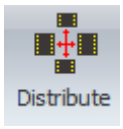
Build PCB

Place Around Border



Auto-Layout

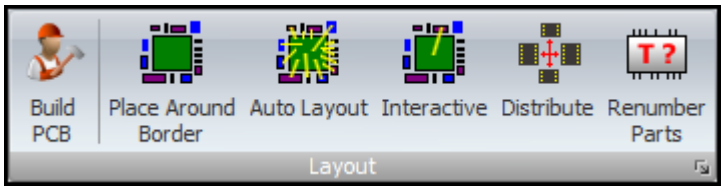
Interactive



Distribute

Renumber Parts

Click on the small button at the bottom right of the command group to display the Layout Settings Popup.



Layout Commands

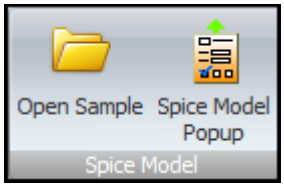
1.3.2.14 The Simulate Schematic Page

The Simulate Ribbon Page

Spice Model Command Group

Open Sample

Spice Model Popup

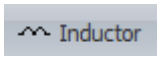


Spice Model Commands

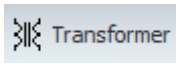
Passive Command Group

Resistor

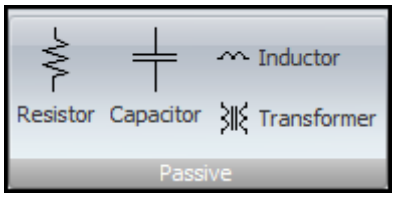
Capacitor



Inductor



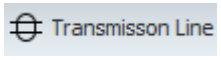
Transformer



Passive Commands

Diodes Command Group

Diode

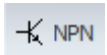


Transmission Line

Diodes Commands

BJT Command Group

PNP

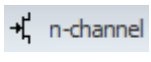


NPN

BJT Commands

JFET Command Group

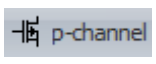
P-Channel



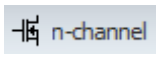
N-Channel

JFET Commands

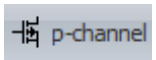
NMOS Command Group



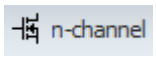
P-Channel



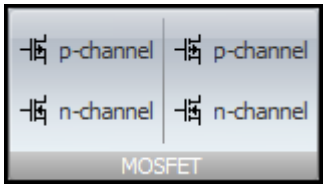
N-Channel



P-Channel

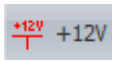


N-Channel

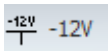


NMOS Commands

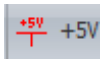
Power Command Group



+12V

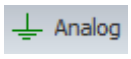


-12V

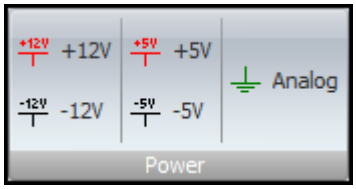


+5V

-5V



Analog

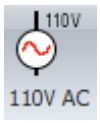


Power Commands

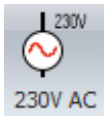
Source Command Group

Voltage

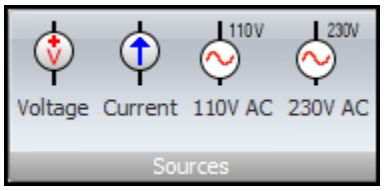
Current



110V AC

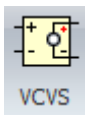


230V AC

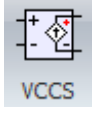


Sources Commands

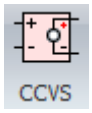
Dependent Sources Command Group



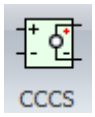
VCVS



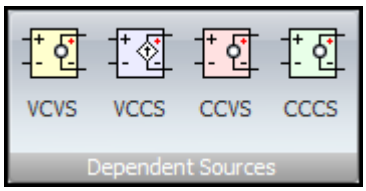
VCCS



CCVS



CCCS

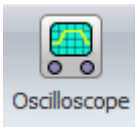


Dependent Sources Commands

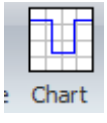
Instruments Command Group

Power Supply

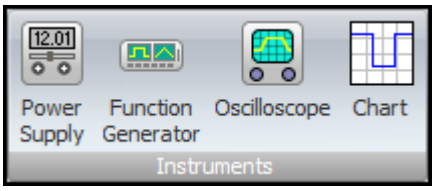
Function Generator



Oscilloscope

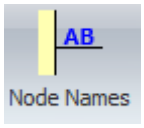


Chart

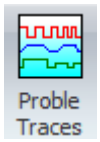


Instruments Commands

Analysis Command Group

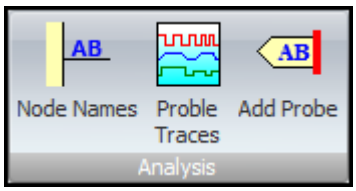


Node Names



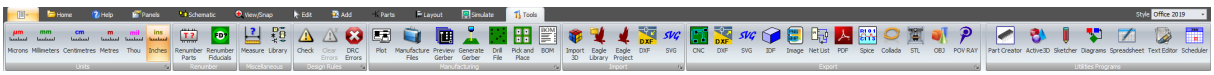
Probe Traces

Add Probe



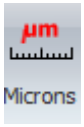
Analysis Commands

1.3.2.15 The Tools Ribbon Page



The Tools Ribbon Page

Units Command Group



Microns

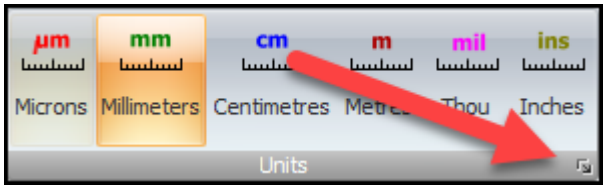
Millimetres

Centimetres

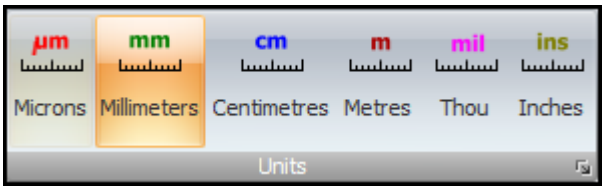
Metres

Thou

Inches



Click on the small button at the bottom right of the command group to display the Snap Settings Popup.



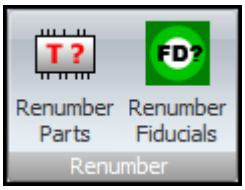
Units Commands

Renumber Command Group

Renumber Parts

Renumber Fiducials

.

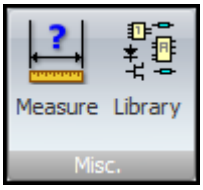


Renumber Commands

Miscellaneous Command Group

Measure

Library

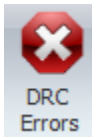


Miscellaneous Commands

Design Rules Command Group

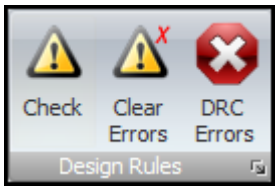
Check

Clear



DRC Errors

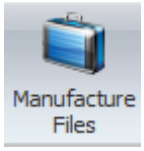
Click on the small button at the bottom right of the command group to display the Design Rules Popup.



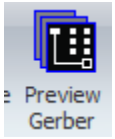
Design Rules Commands

Manufacturing Command Group

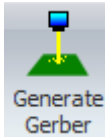
Plot



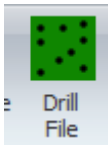
Manufacture Files



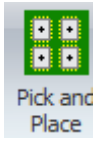
Preview Gerber



Generate Gerber

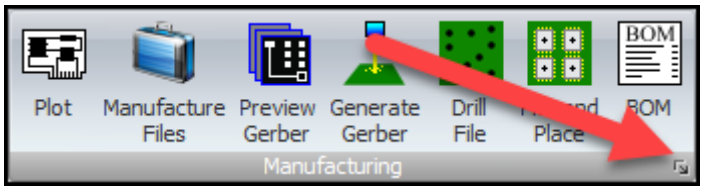


Dill File



Pick and Place

BOM

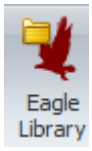


Click on the small button at the bottom right of the command group to display the Manufacturing Settings Popup.

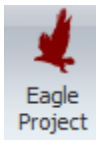
Manufacturing Commands

Import Command Group

Import 3D



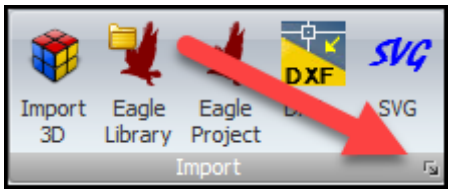
Eagle Library



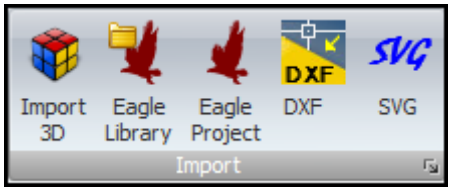
Eagle Project

DXF

SVG



Click on the small button at the bottom right of the command group to display the File Settings Popup.



Import Commands

Export Command Group

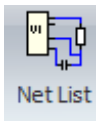
CNC

DXF

SVG

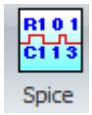
IDF

Image



Netlist

PDF



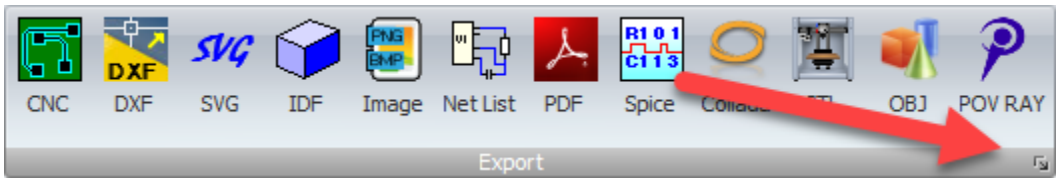
Spice

Collada

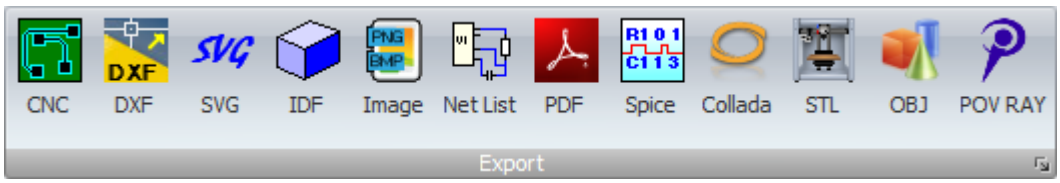
STL

OBJ

POV RAY

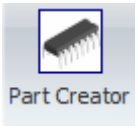


Click on the small button at the bottom right of the command group to display the File Settings Popup.

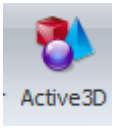


Export Commands

Utilities Programs Command Group

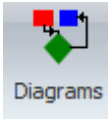


Part Creator

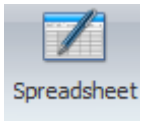


Active3D

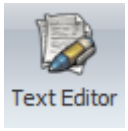
Sketcher



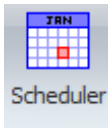
Diagrams



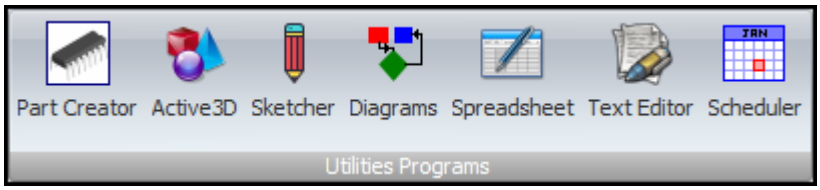
Spreadsheet



Text Editor



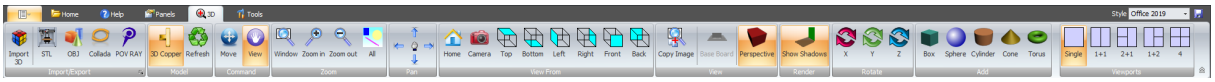
Scheduler



Utilities Commands

1.3.2.16 The 3D Ribbon Page

The 3D Ribbon page, which contains 43 separate commands, will only be visible if a 3D viewport is visible and selected: see the View/Edit ribbon page and the viewport context menu

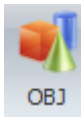


The 3D Ribbon Page

The Import/Export Command Group

Import 3D

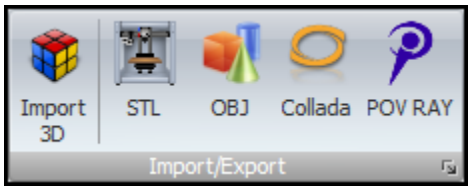
Export to STL



Export to OBJ

Export to Colada

Export to POV RAY



Import/Export Commands

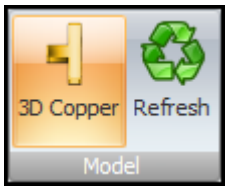
The Model Command Group

3D Copper

View copper as true 3D else view copper as a 2D image on the PCB. Viewing copper as 3D may make your 3D viewing slower - it depends on your graphics card.

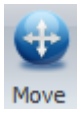
Refresh All 3D Models

Click to rebuild all the 3D content in the 3D viewport. You might do this if the 3D looks out of sync with your design. Generally, you should not need to do this as AutoTRAX DEX automatically rebuilds your 3D models if needed.

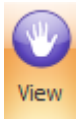


Model Commands

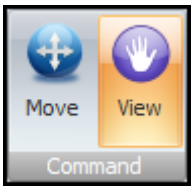
The Move/View Command Group



Move



View



Move/View Commands

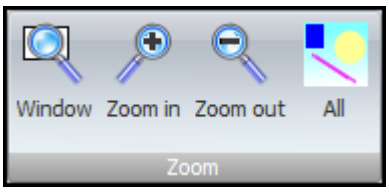
The Zoom Command Group

Zoom Window

Zoom In

Zoom Out

View All



Zoom Commands

The Pan Command Group

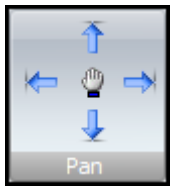
Pan Left

Pan Right

Pan Left

Pan Down

Pan



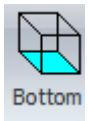
Pan Commands

The View From Command Group

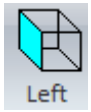
View from Standard Home Position

View from Preset Camera Position

View from the Top

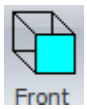


View from the Bottom

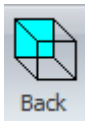


View from the Left

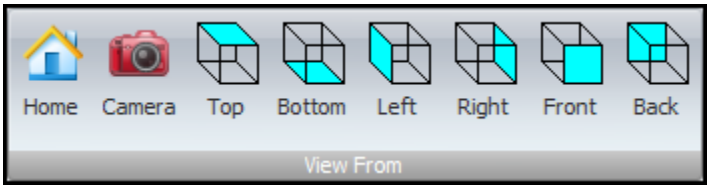
View from the Right



View from the Front



View from the Back



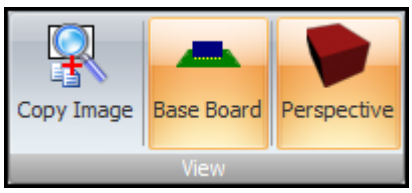
View From Commands

The View Command Group

Copy Inage to Clipboard

View/Hide the PCB Base Board

Toggle Perspective/Orthographic View



View Commands

The Render Command Group

Show Shadows

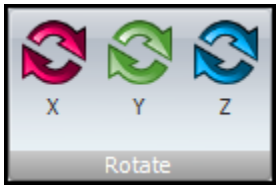
Render Commands

The Rotate Command Group

Rotate About the X Axis

Rotate About the Y Axis

Rotate About the Z Axis

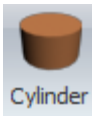


Rotate Commands

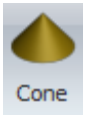
The 3D Add Command Group

Adding a Box

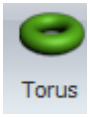
Adding a Sphere (Ball)



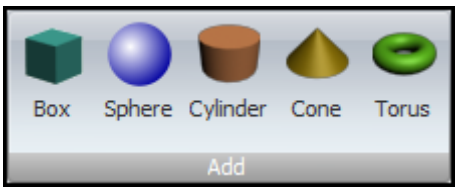
Adding a Cylinder



Adding a Cone



Adding a Torus (Donut)



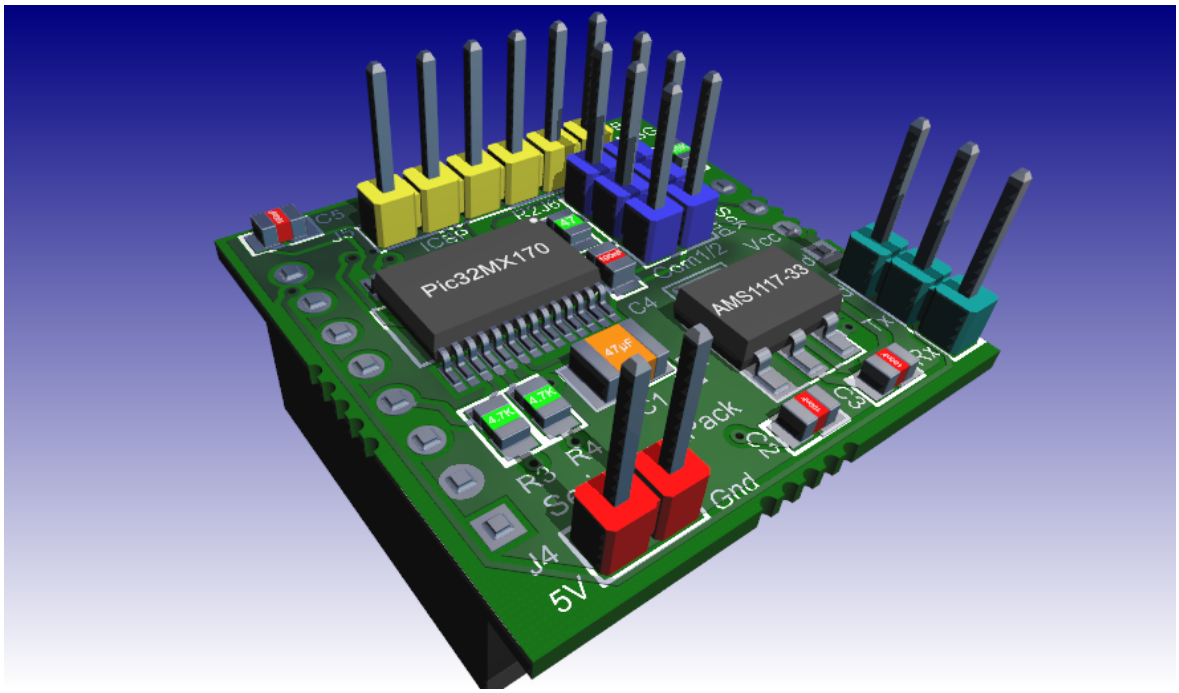
Add Commands

The 3D Viewports Command Group

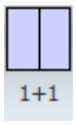
- You have 5 preset view configurations.
- If there is more than 1 view visible then the active view has a red border.

- View can have separators that you can drag to re-size.

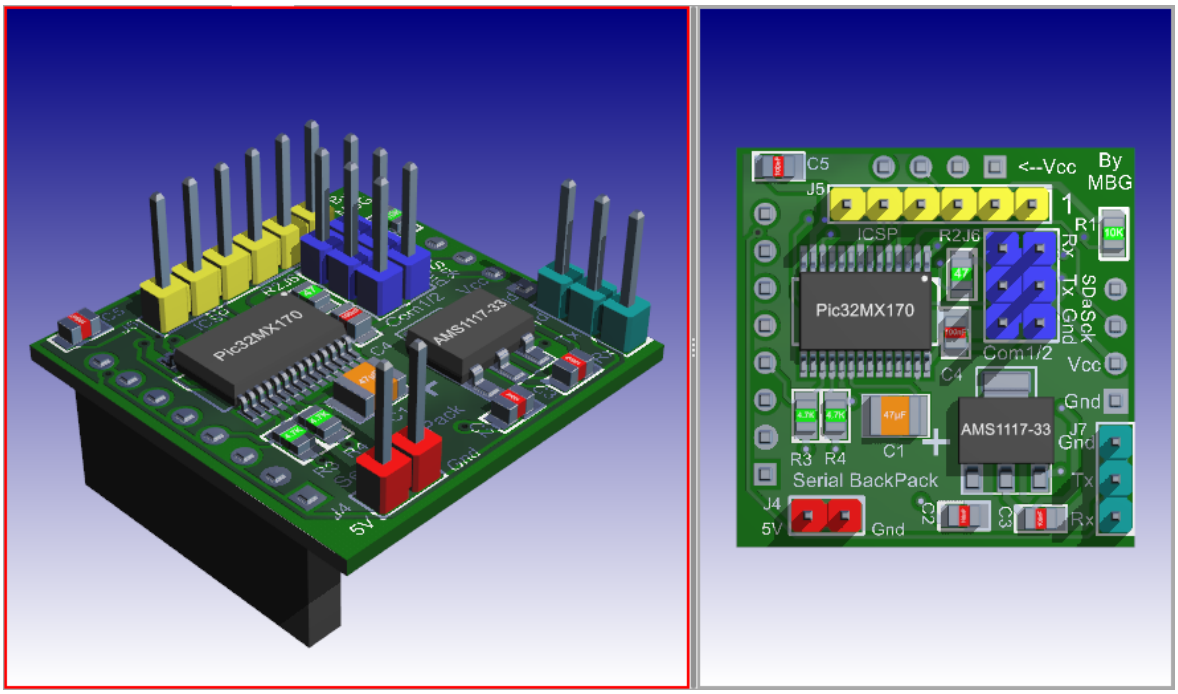
Viewing a Single 2D Viewport



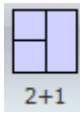
Single Viewport



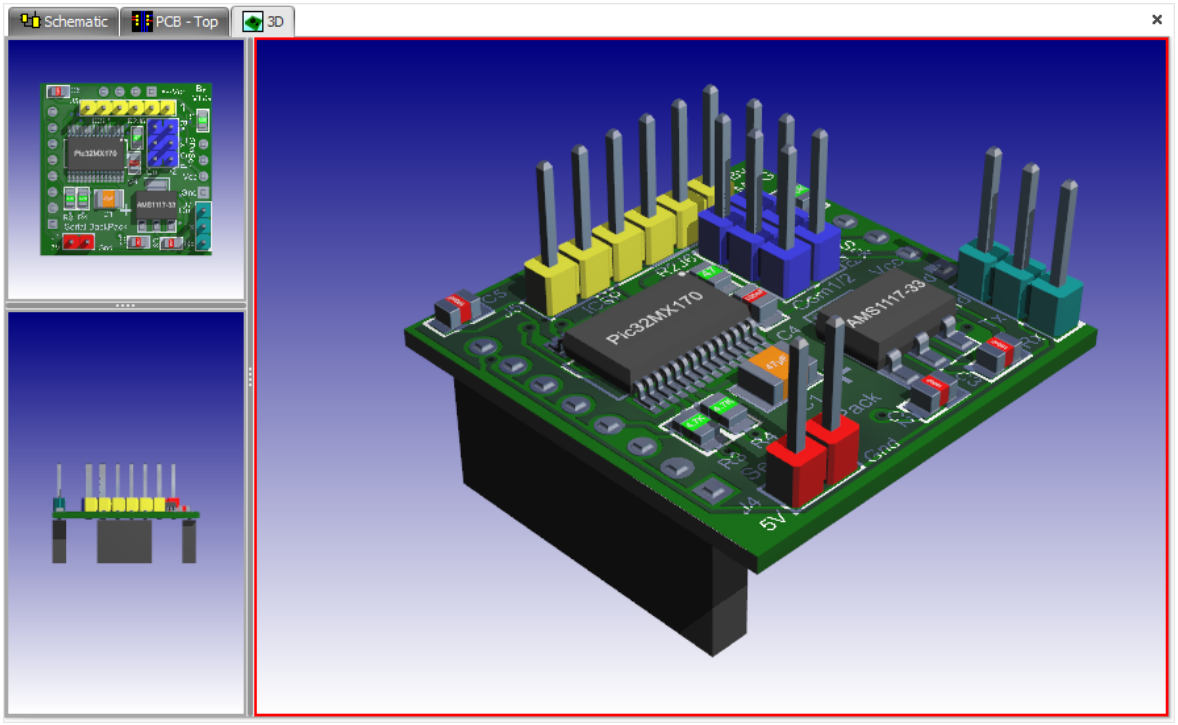
Viewing two (1+1) 3D Viewports



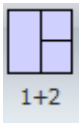
Two (1+1) Viewports



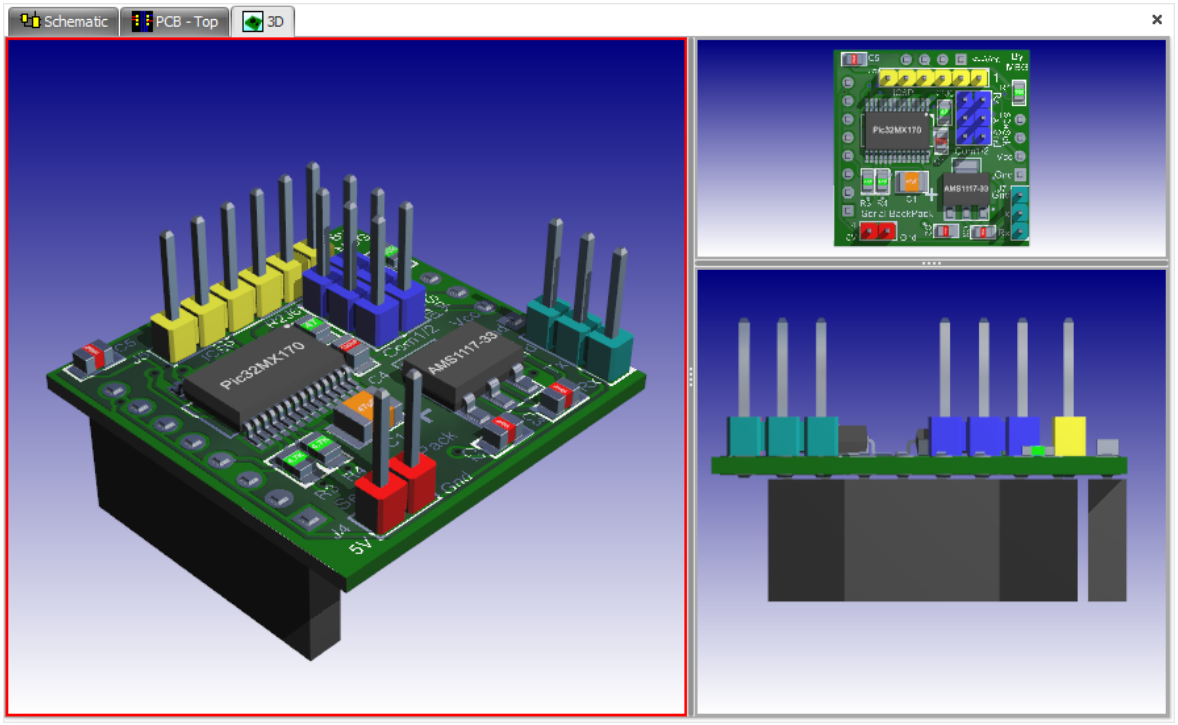
Viewing three (2+1) 3D Viewports



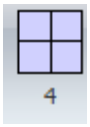
three (2+1) Viewports



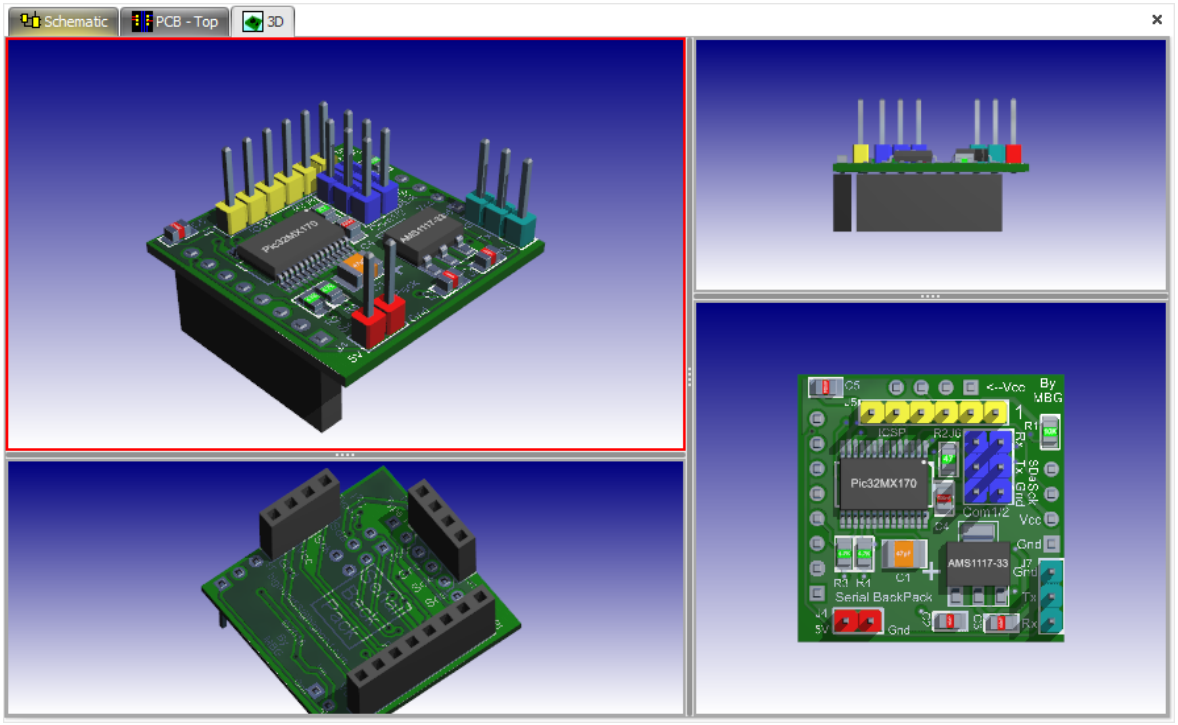
Viewing three (1+2) 3D Viewports



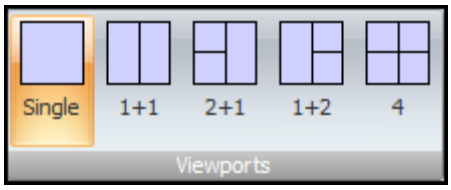
Three (1+2) Viewports



Viewing Four 3D Viewports



Four 3D Viewports



3D Viewport Commands

1.3.2.17 The 4 Ribbon Layout Modes

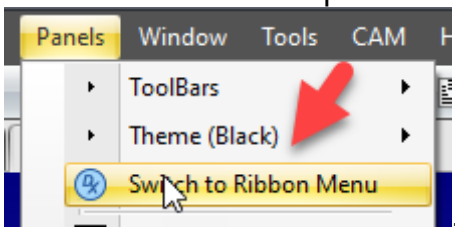
You now have a choice of a traditional drop-down menu or a Microsoft Office style ribbon menu.

Switching Between Drop-down and Ribbon Menus

To switch from the ribbon menu to the drop-down menu click on 

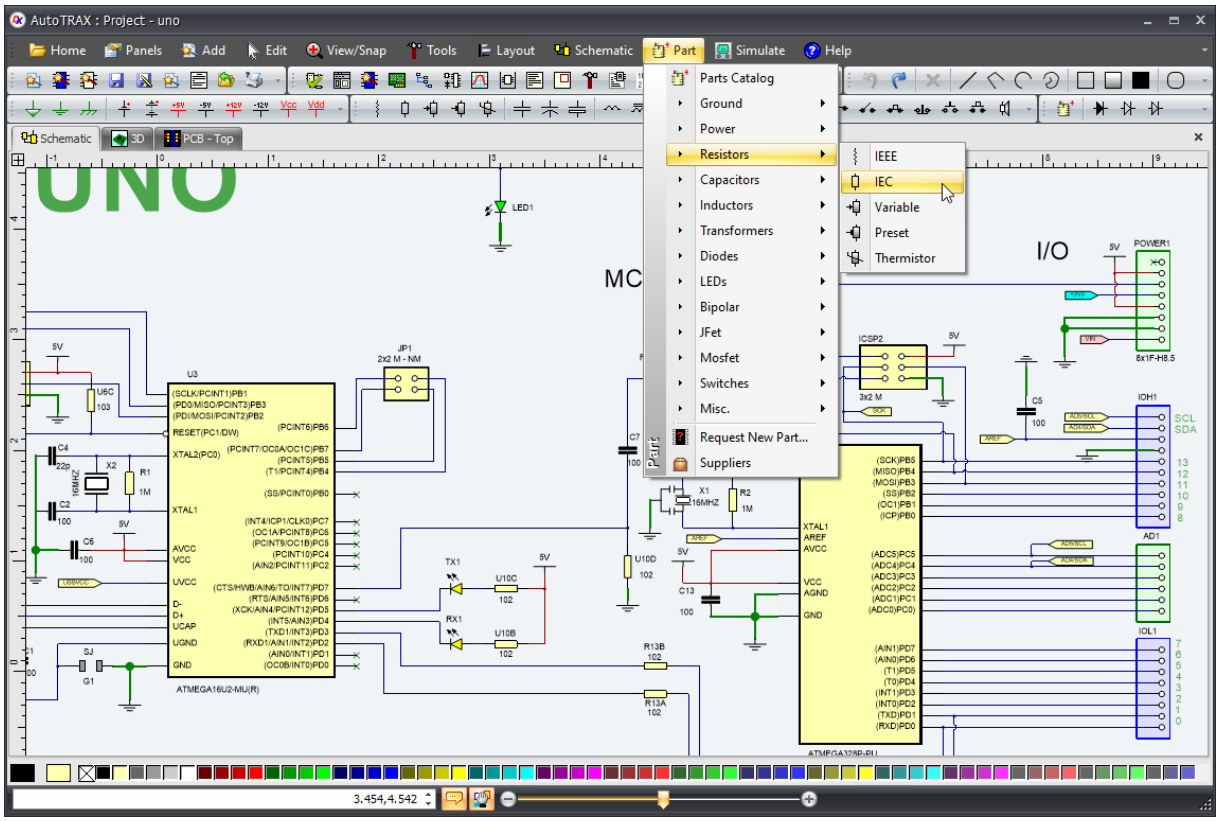
in the Panels->Menu ribbon button group.

To switch from the drop-down menu to the ribbon menu click on



Drop-Down Menu

You have a classic drop-down menu. This also comes with optional toolbars.



Ribbon Menu

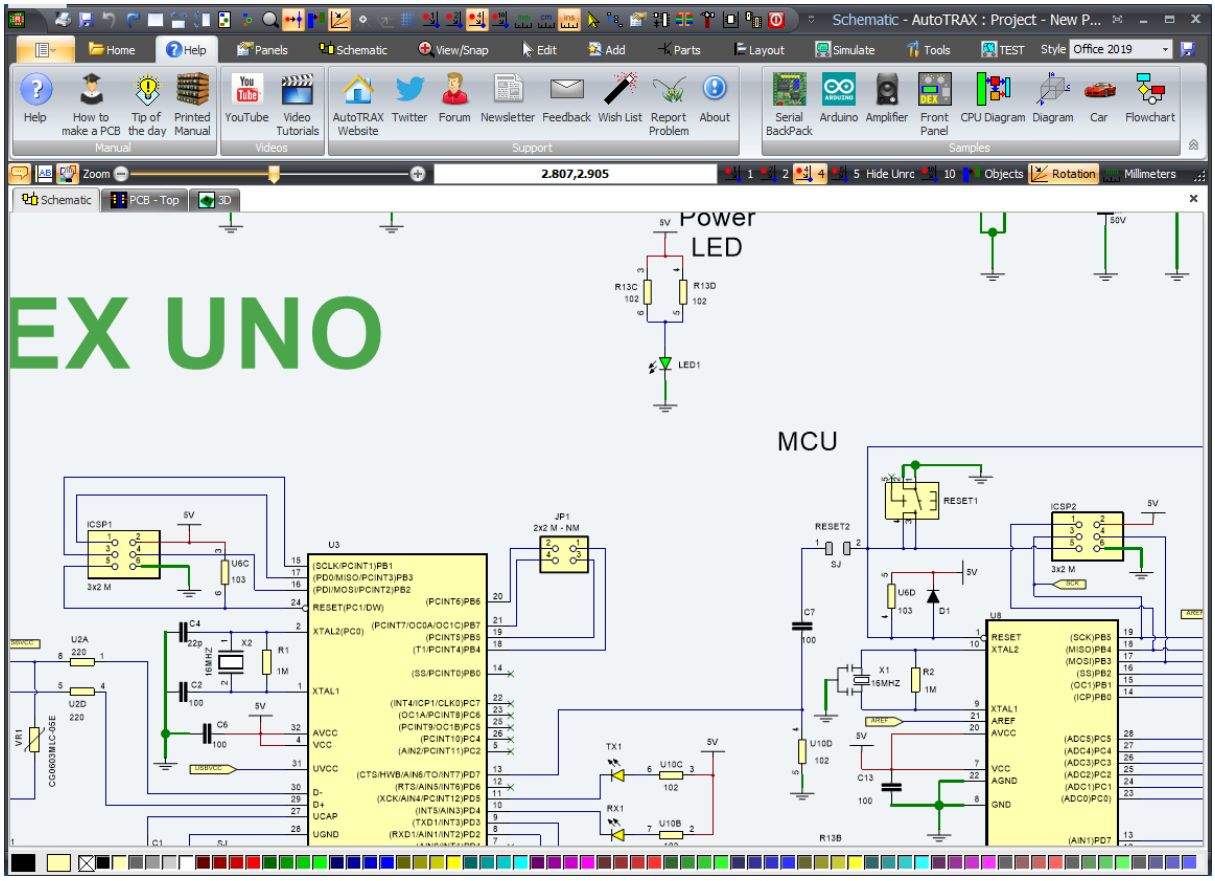
You can save some of the screen space normally occupied by the the Ribbon Menu.

The Ribbon menu has 3 display layout modes:

1. Classic Fully Expanded
2. Simplified
3. Minimized

Classic Fully Expanded Ribbon Menu

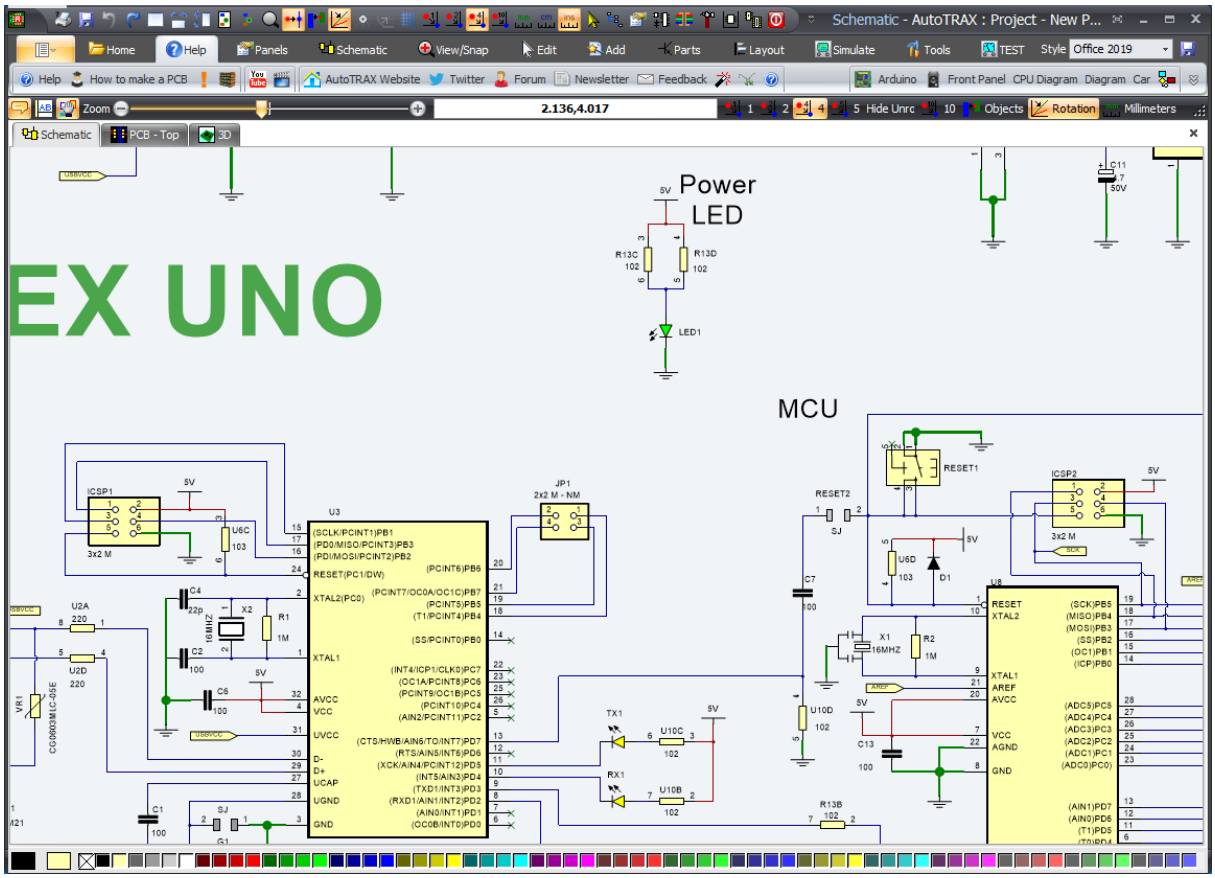
In this mode, all the ribbon page buttons groups are visible.



Classic Fully Expanded Ribbon Menu

Simplified Ribbon Layout

In this mode, the ribbon page button groups are compacted.

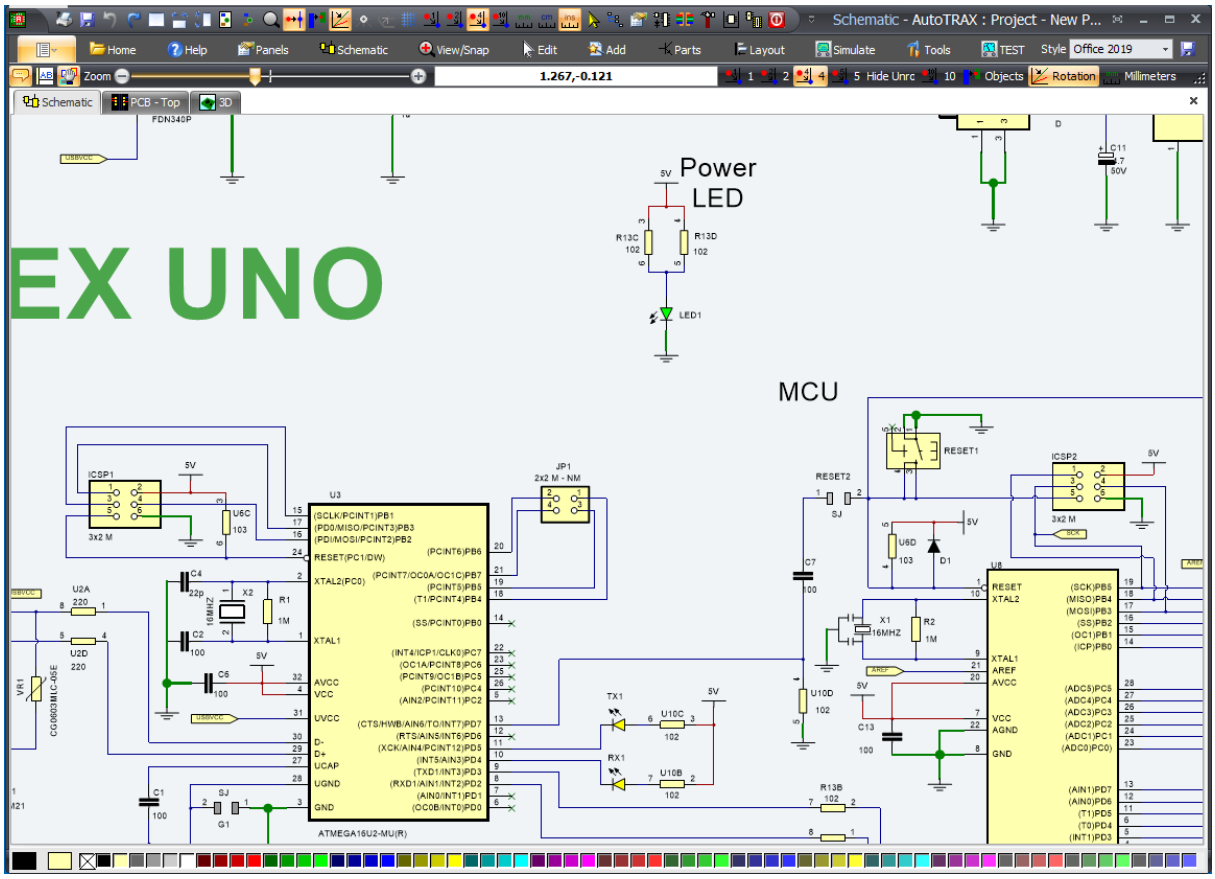


Simplified Ribbon Layout

To switch between the Classic and the Simplified layout: click on the arrows button  at the bottom left of the Ribbon Menu.

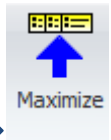
Minimized Ribbon Layout

In this layout mode, the ribbon button groups are hidden until made visible by clicking on a ribbon page heading.

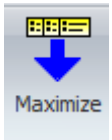


Minimized Ribbon Layout

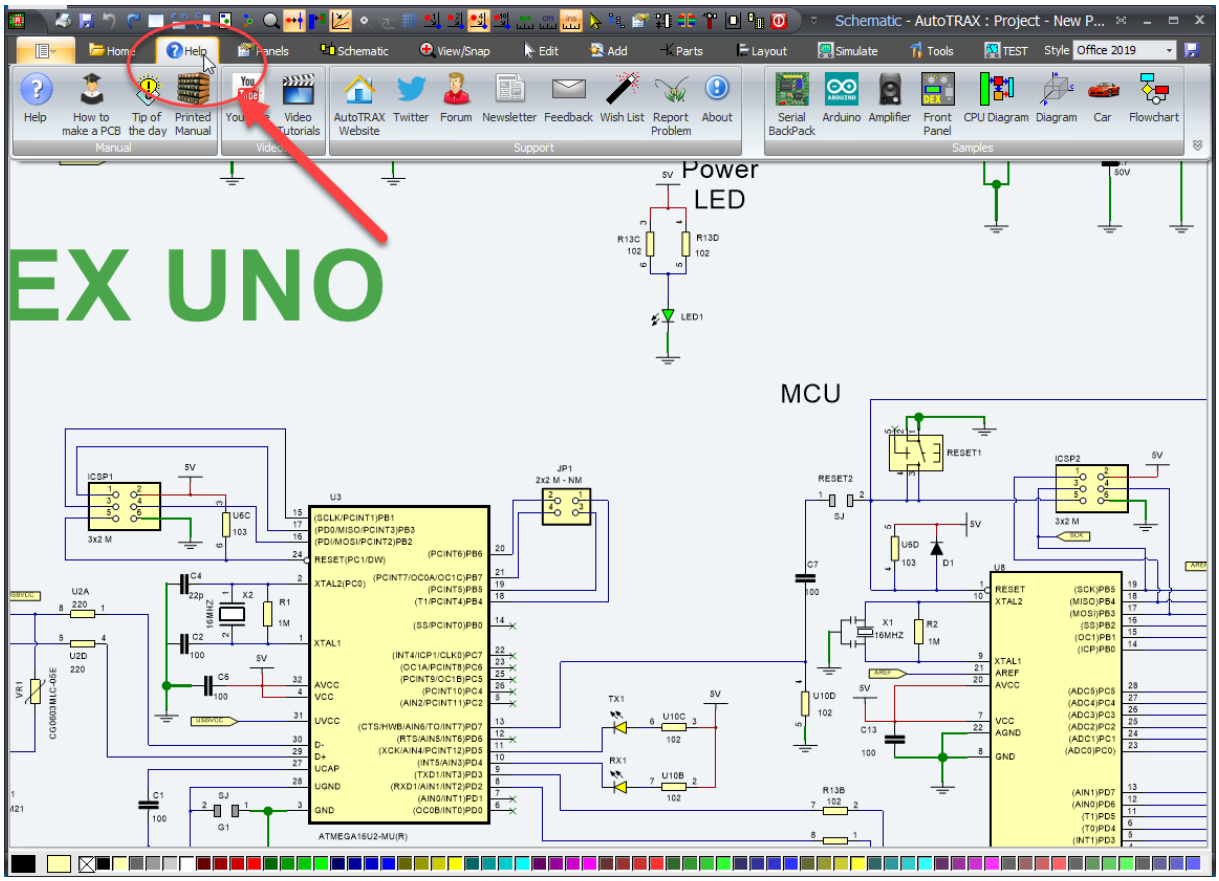
AutoTRAX DEX can optionally collapse panels and the ribbon menu to give you the maximum view of the design. The choice is yours and you can rapidly change your workstation style at the click of a mouse button.



To collapse the Ribbon Menu click the Maximixe in the Panels→ Menu→ button. When collapsed, the ribbon menu appears as shown above. When you click on a ribbon tab, the ribbon will expand down, as shown below, allowing you to select a command and will contract after the command starts. To restore the ribbon



to classic node: in the Panels menu button group click the Panels→ Menu→ button.



Ribbon Buttons Group Expanded in Minimized Ribbon Layout

Classic Dropdown Menu

1.3.2.18 Popup Settings

[Popup Settings]

[The Add Parts Popup]

[The CAM Settings Popup]

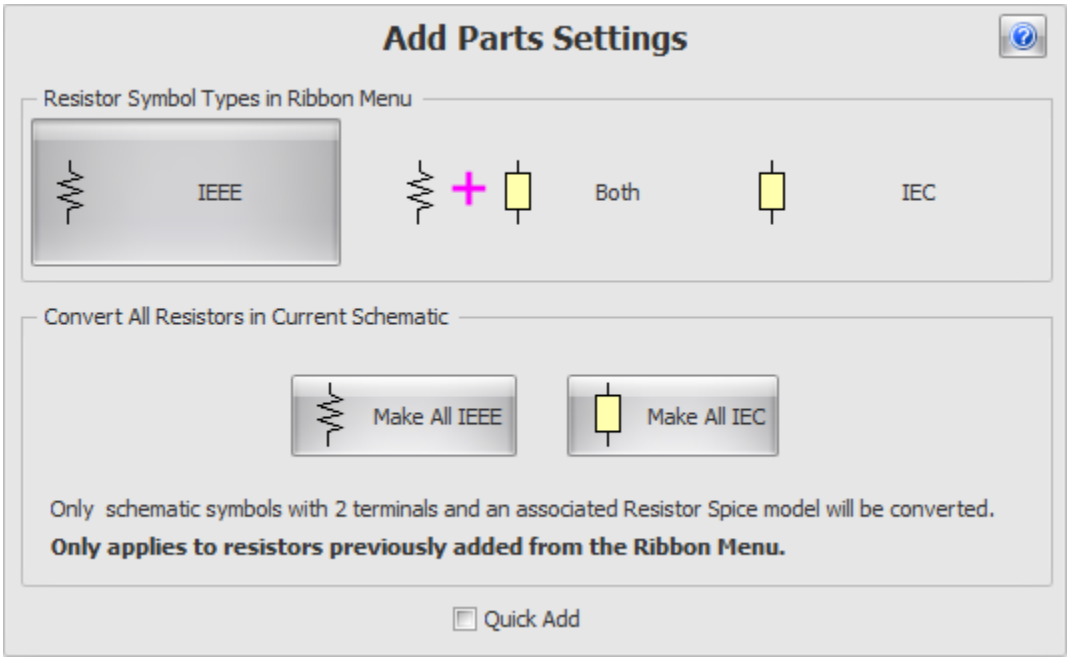
[The Copper Pour Settings Popup]

[The Cutouts Popup]

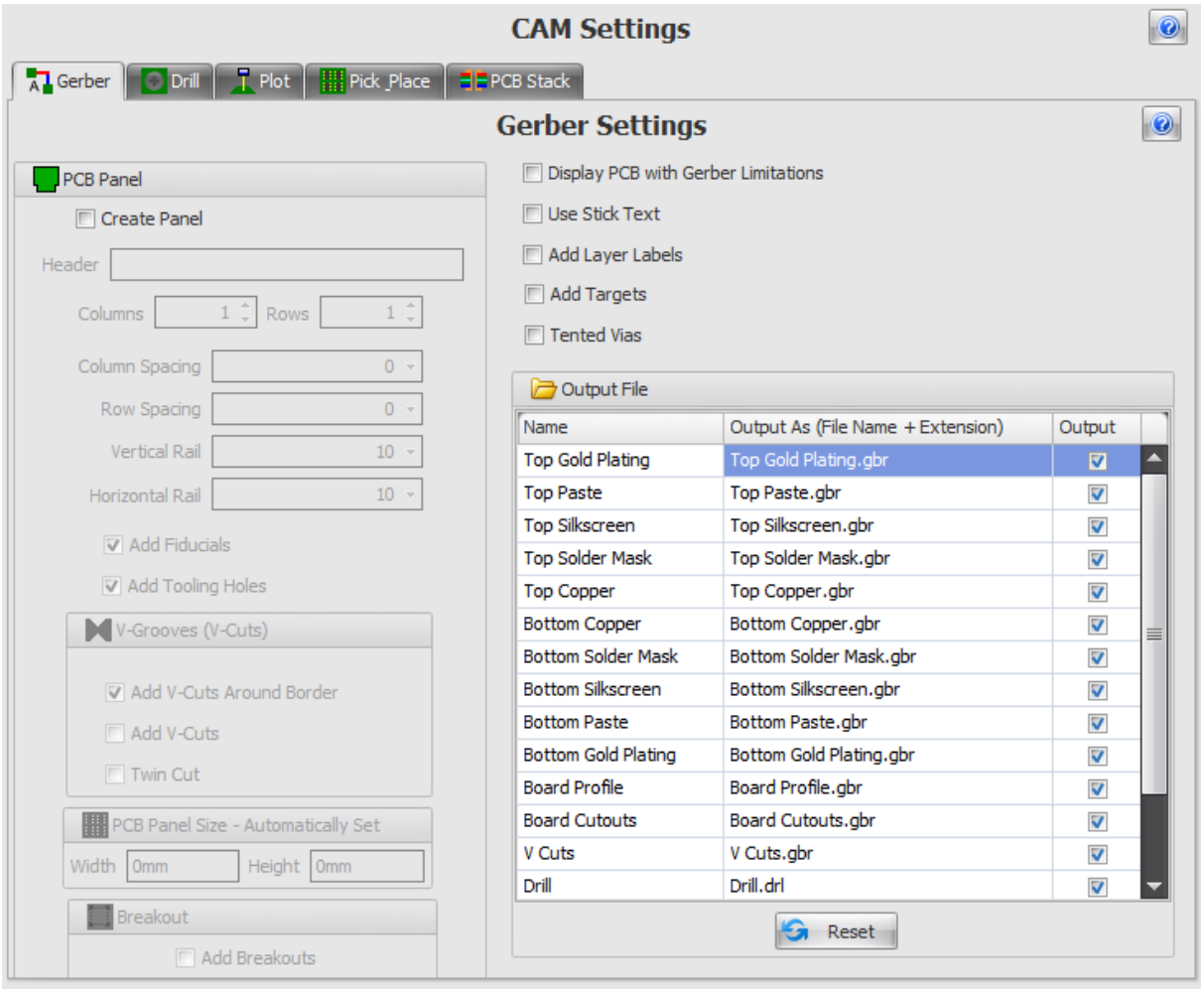
[The Design Rules Popup]

[The Dimension Settings Popup]

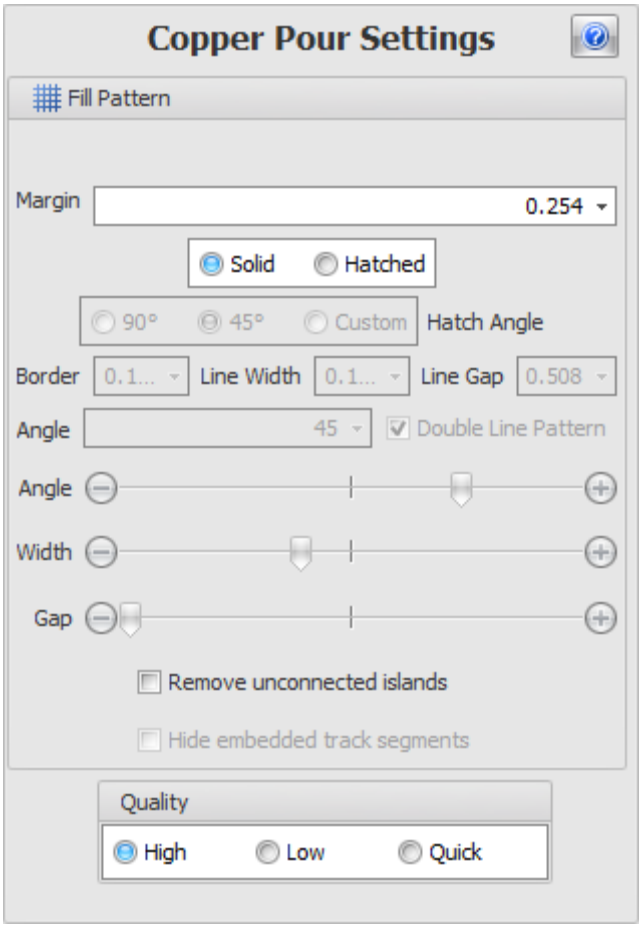
1.3.2.18.1 The Add Parts Popup



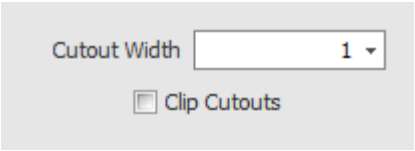
1.3.2.18.2 The CAM Settings Popup



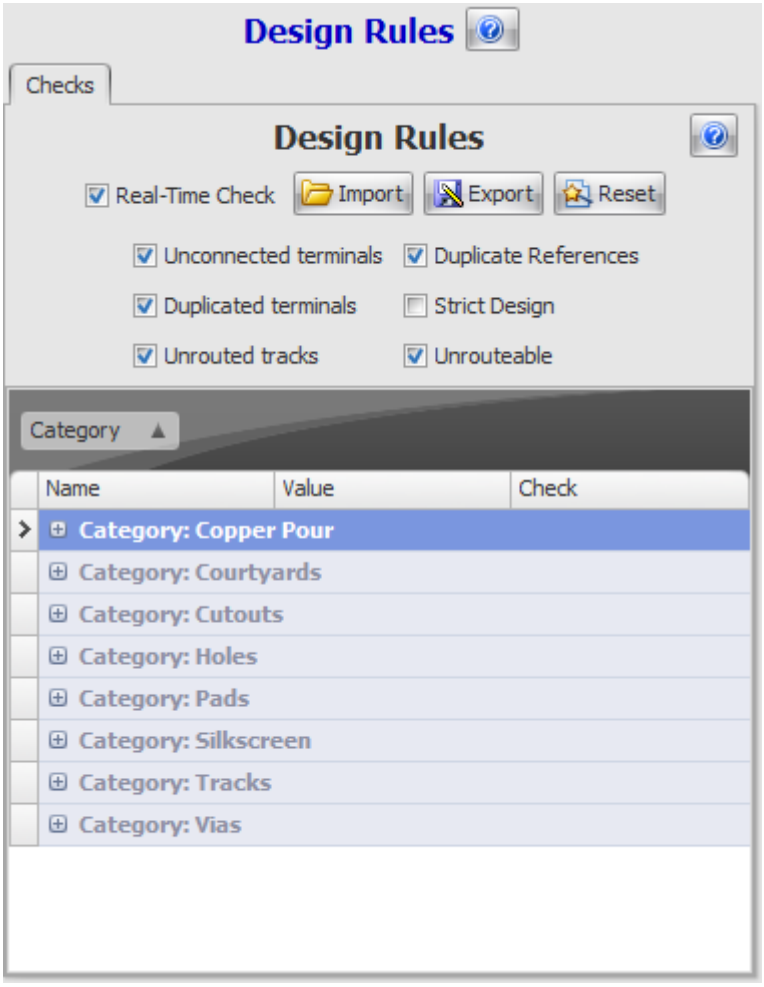
1.3.2.18.3 The Copper Pour Settings Popup



1.3.2.18.4 The Cutouts Popup



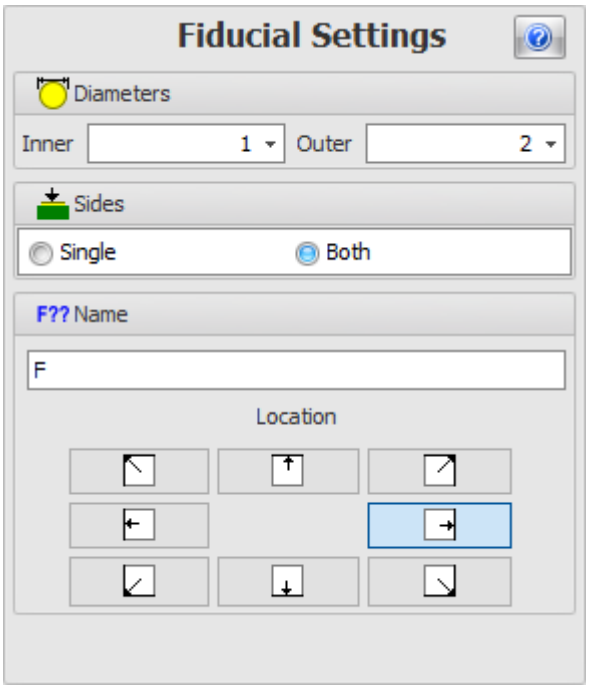
1.3.2.18.5 The Design Rules Popup



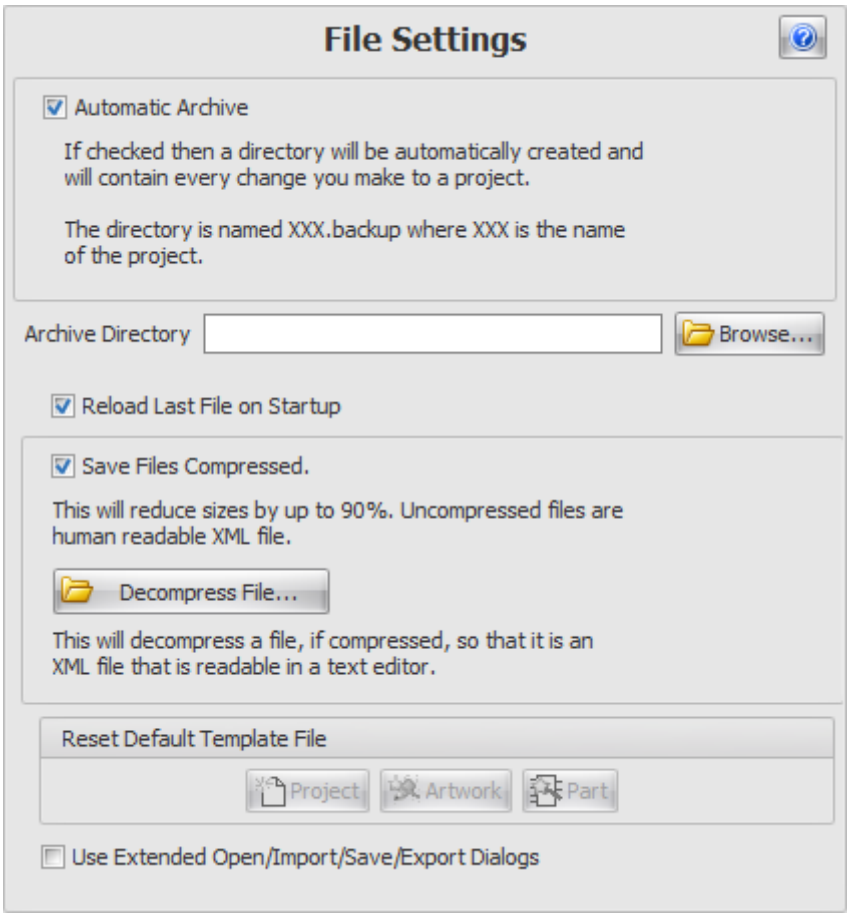
1.3.2.18.6 The Dimension Settings Popup

Enter topic text here.

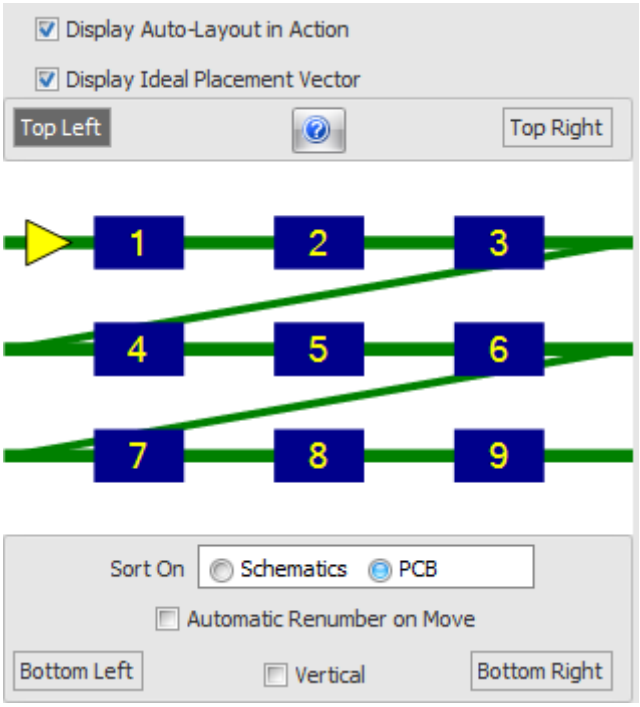
1.3.2.18.7 The Fiducial Popup



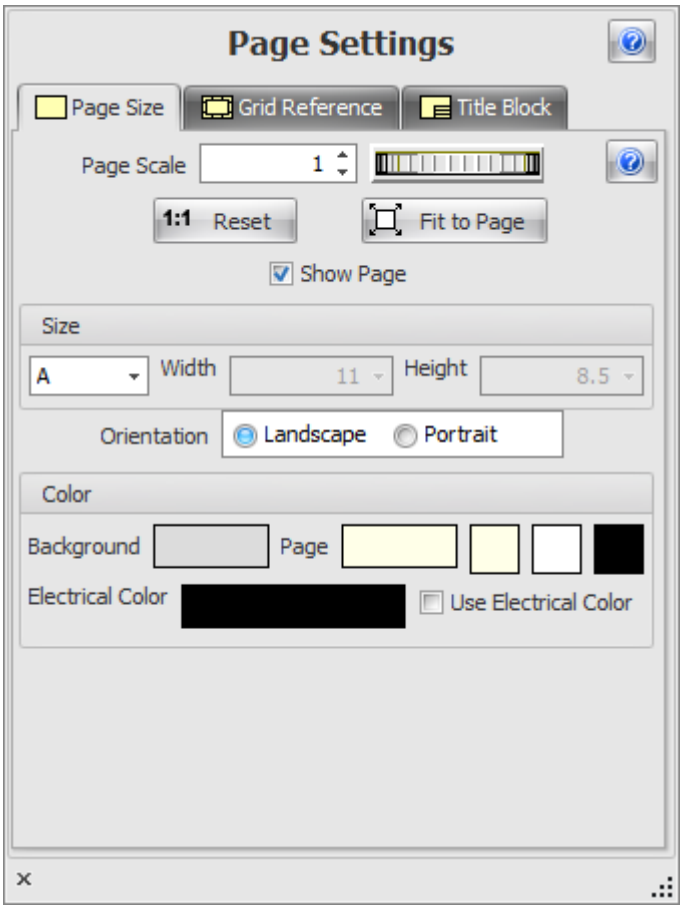
1.3.2.18.8 The File Settings Popup



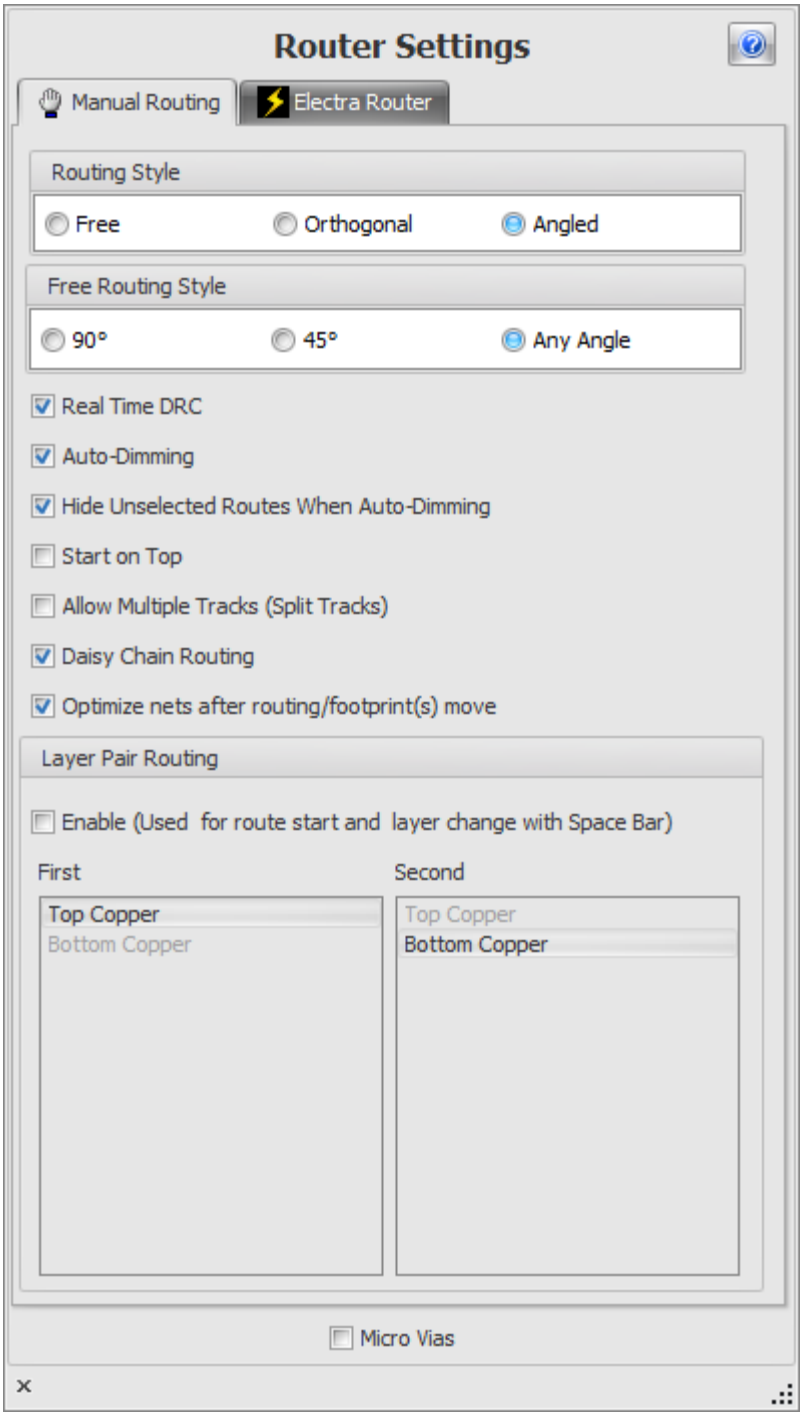
1.3.2.18.9 The Layout Settings Popup



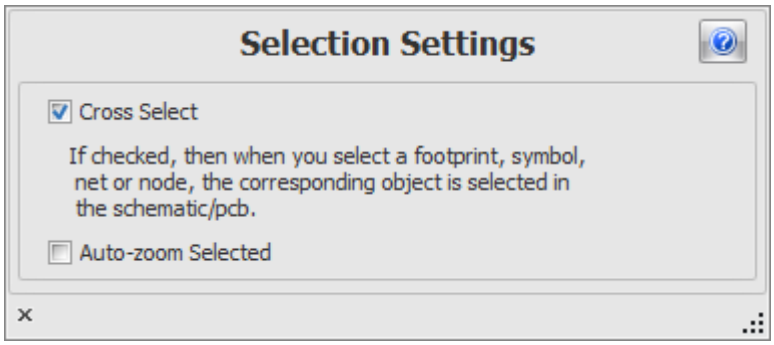
1.3.2.18.10 The Page Settings Popup



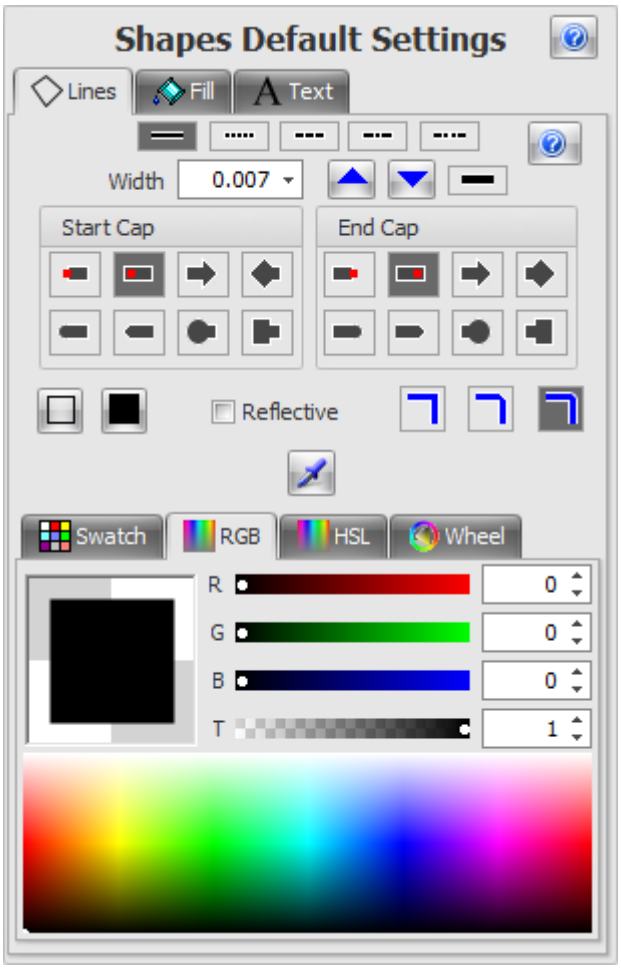
1.3.2.18.11 The Router Settings Popup



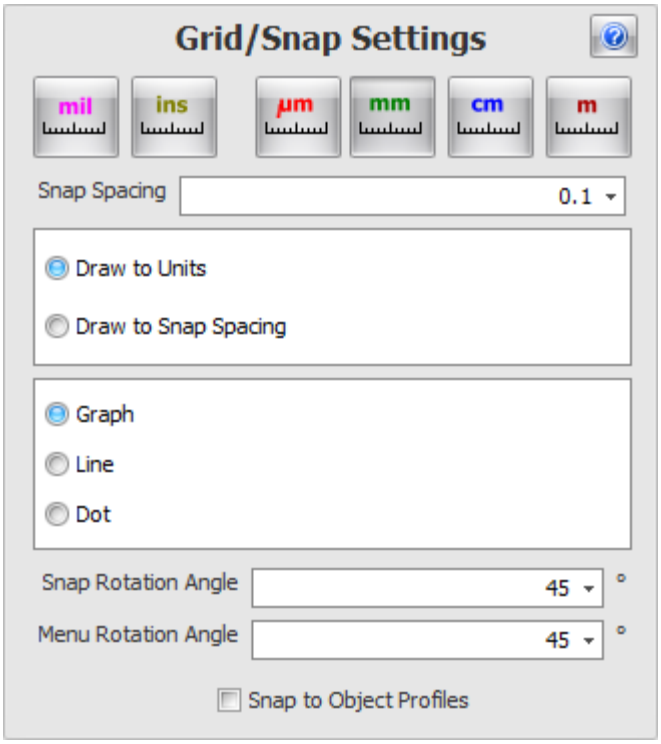
1.3.2.18.12 The Selection Settings Popup



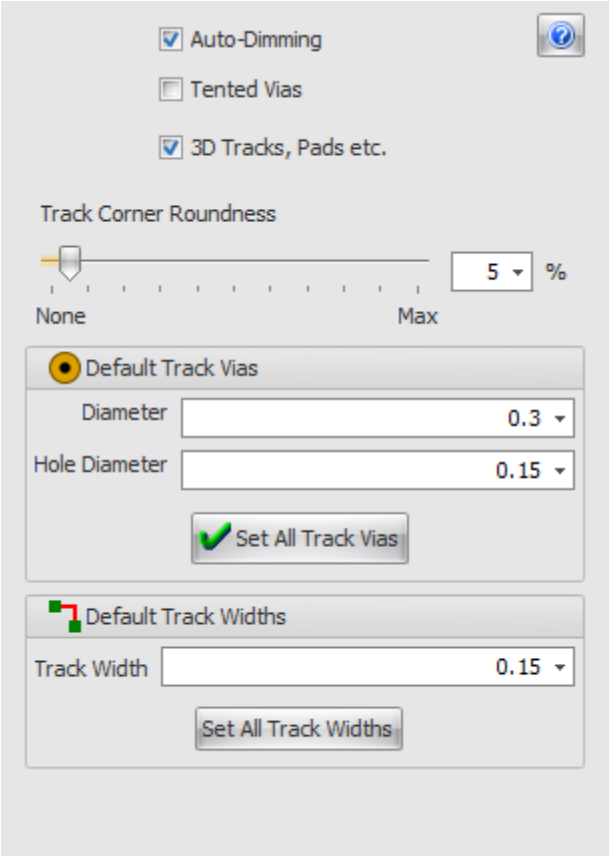
1.3.2.18.13 The Shapes Default Popup



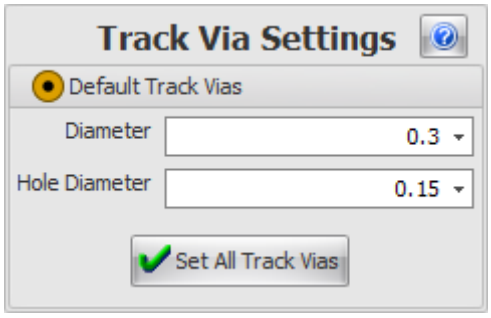
1.3.2.18.14 The Snap Settings Popup



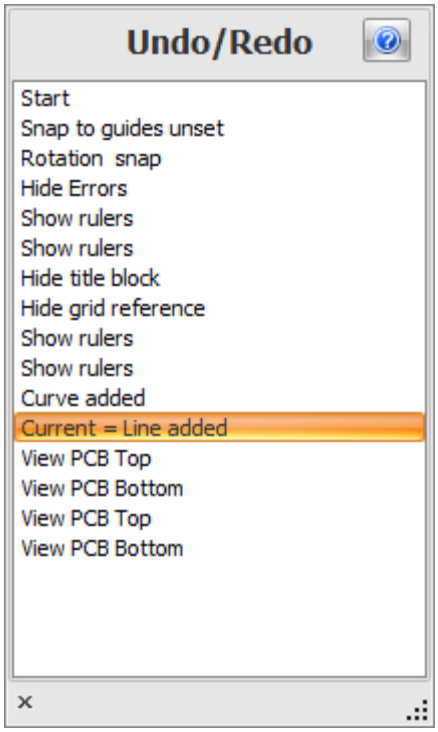
1.3.2.18.15 The Tracks Popup



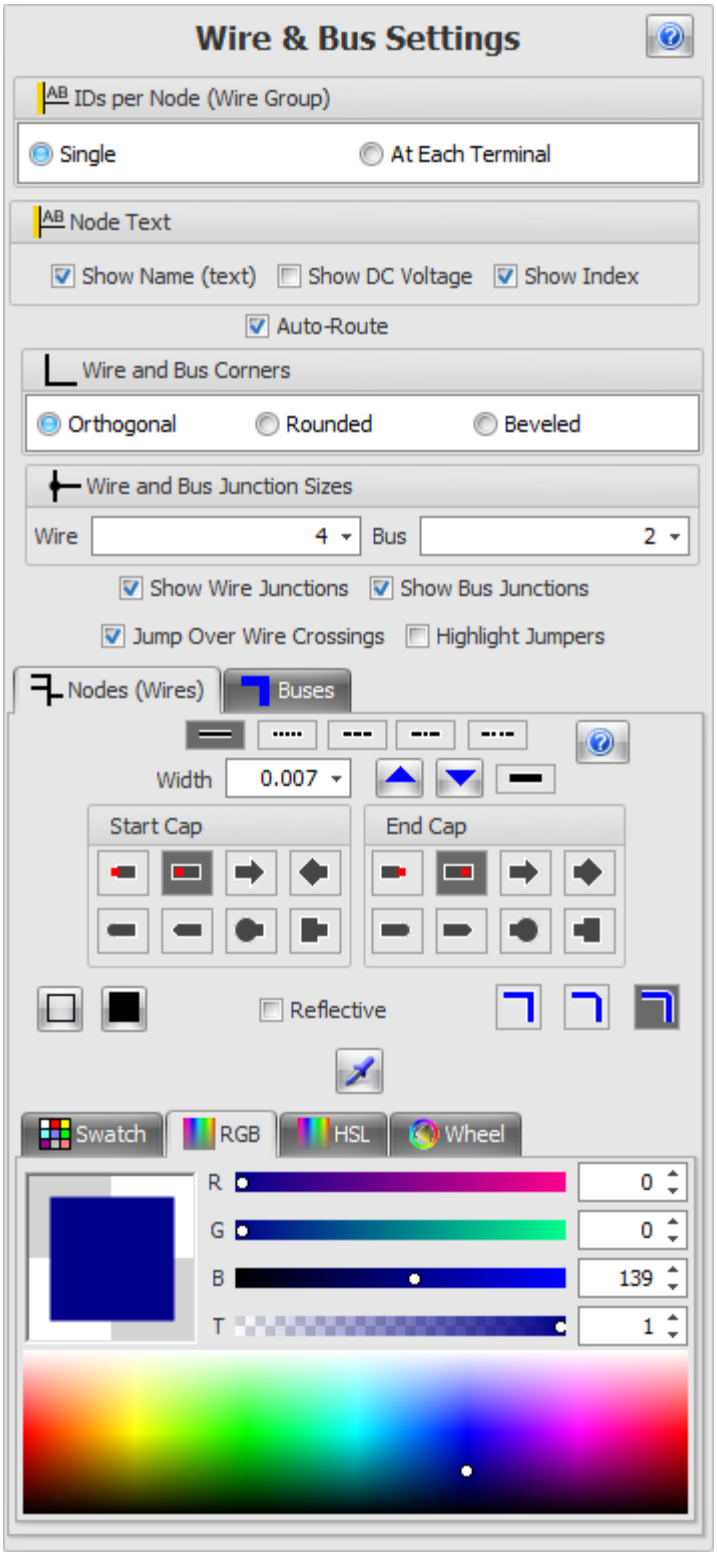
1.3.2.18.16 The Track Via Settings Popup



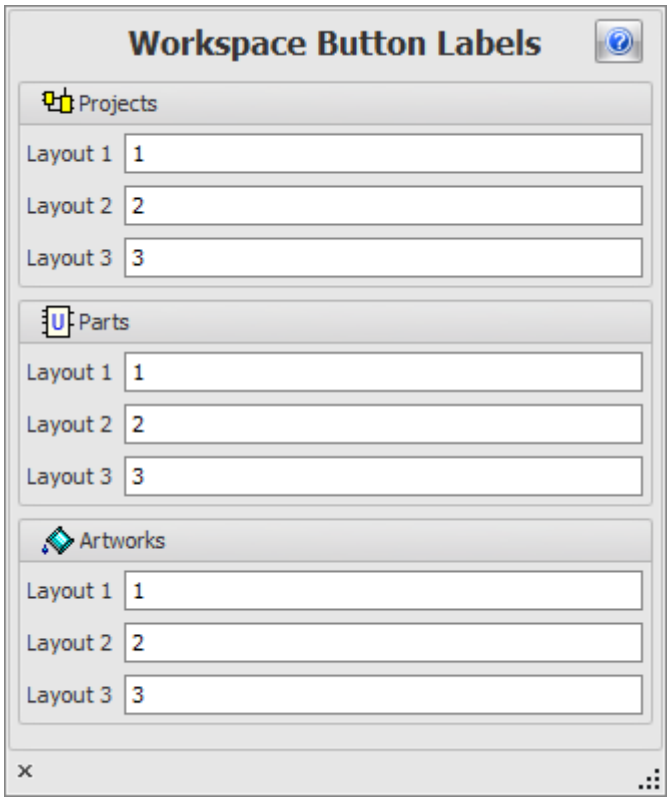
1.3.2.18.17 The Undo Popup



1.3.2.18.18 The Wire and Bus Settings Popup

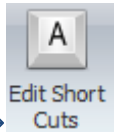


1.3.2.18.19 The Workspace Popup

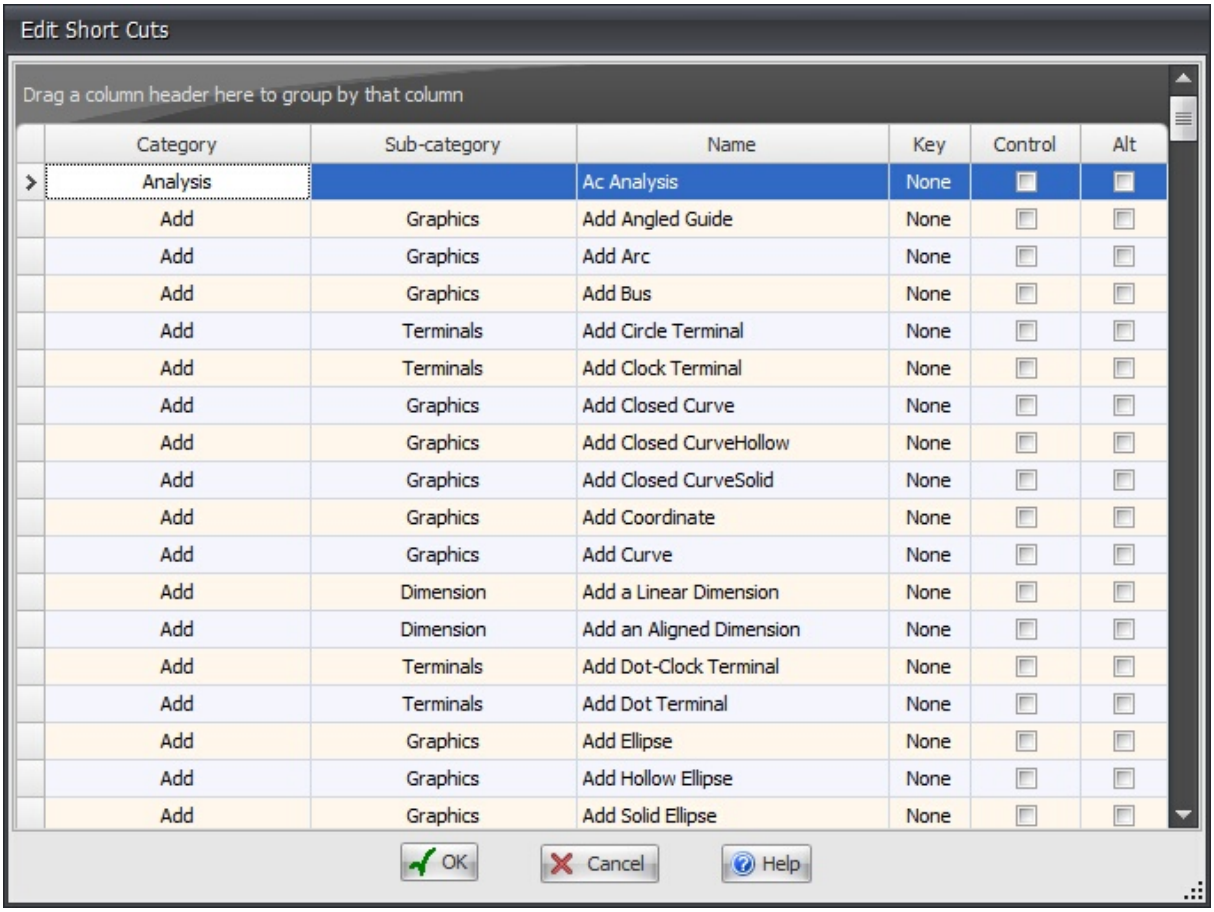


1.3.2.19 Menu Shortcuts

You can set you own shortcuts for the ribbon menu commands.

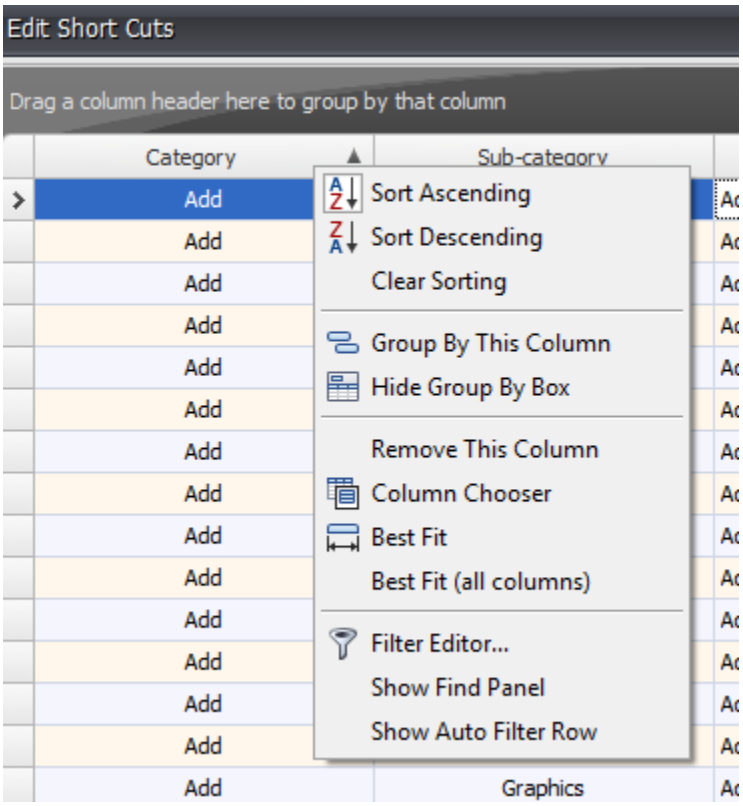
Click the **Panels→Menu→** 

button group. The Edit Short Cuts dialog box shown below will appear.



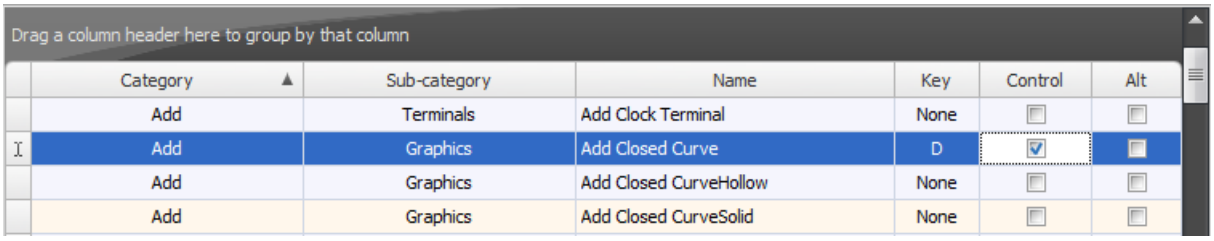
Click on any of the column headers to sort by that column.

Right-click on a column header to show the context menu.



To set a command pick a key in the Key column/Check the control checkbox to require the **CTRL** key to be pressed, similarly check the Alt checkbox to require the ***Alt*** key to be pressed. (Note – It is possible to require both the ***CTRL* and *Alt*** keys to be required)

Below the Add Closed Curve has a shortcut assigned to it which is the ***'D'*** key and the *CTRL* Key.



1.3.3 The Panels

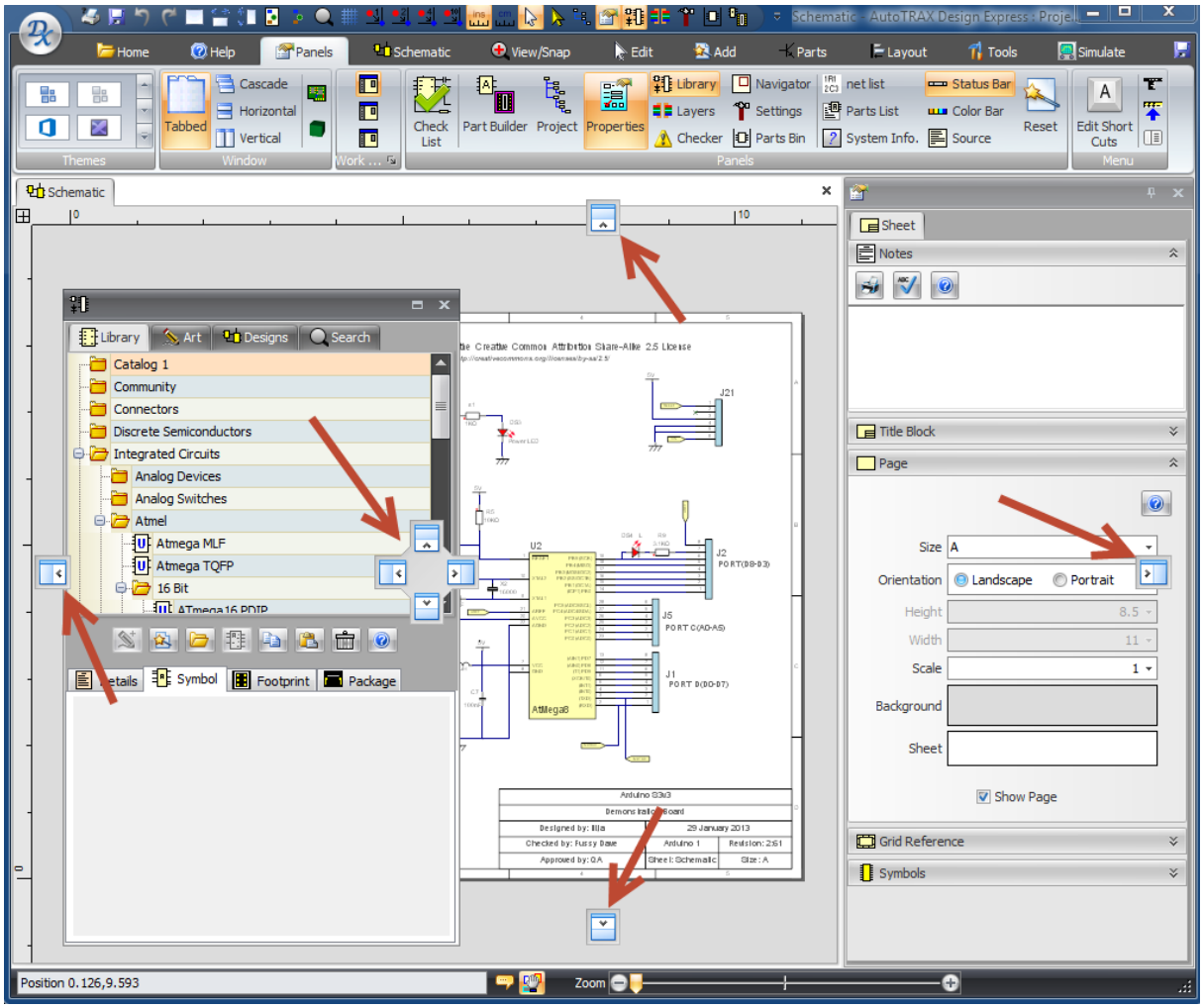
AutoTRAX DEX has the following panels which can be docked to the sides of the application or left floating.

- The Checklist Panel
- The Part Builder Panel
- The Project Panel
- The Properties Panel
- The Library Panel
- The Layers Panel
- The Design Rules Checker Panel
- The Navigator Panel
- The Settings Panel
- The Source View Panel
- The Parts Bin Panel
- The Netlist Panel
- The Parts List Panel
- The System Information Panel

Moving Panels

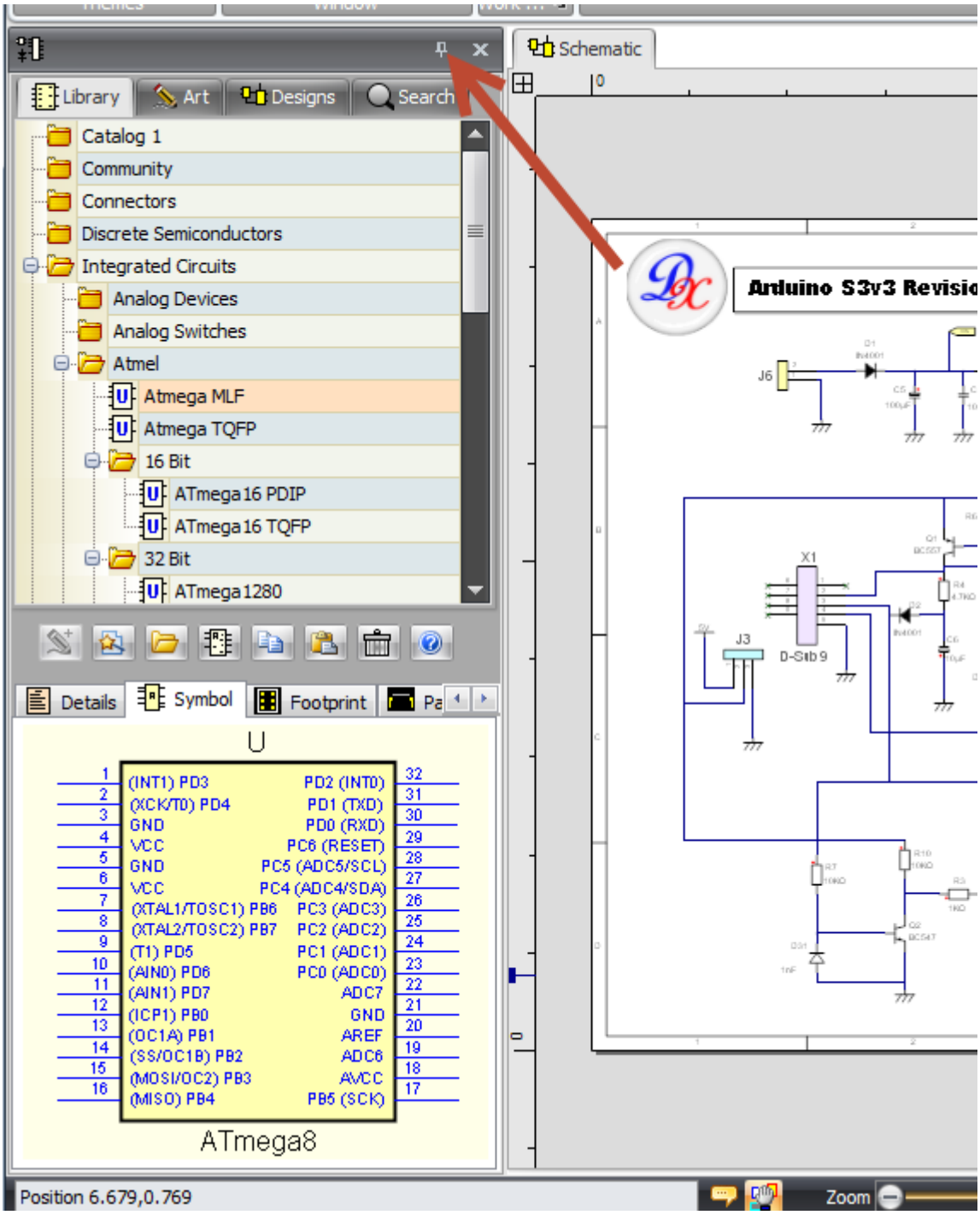
Dock panels are the main building blocks of the AutoTRAX DEX application. They represent dockable windows on which visual controls are placed. A panel can be docked to another panel or it can float. The panel's caption allows you to drag a panel to a new position and thus perform docking operations on it.

To move a panel hold down the left mouse button and drag the panel. As you move the panel you will see small rectangles appear showing you where you can dock the panel you are moving. Below the Library Panel is being dragged.

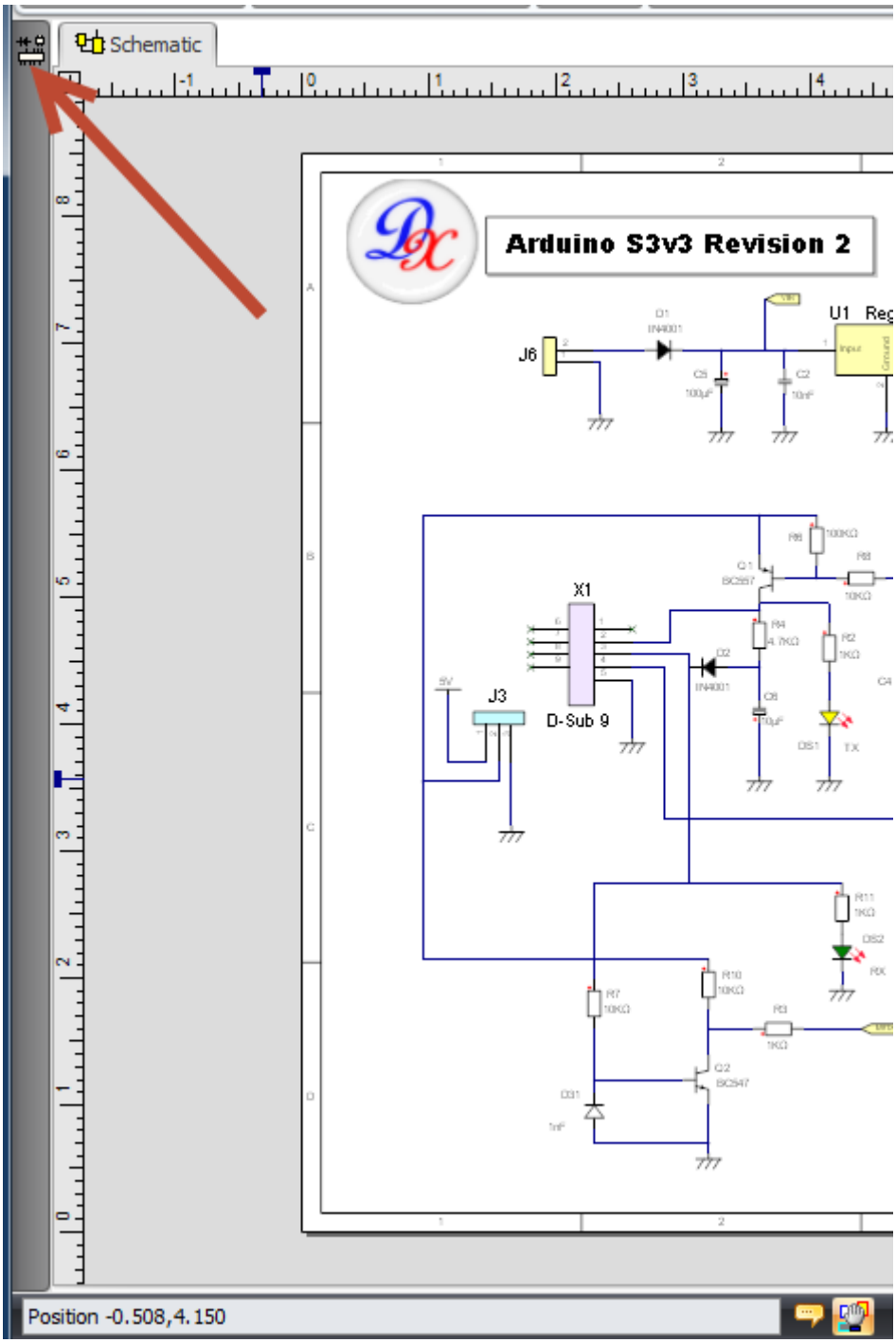


Collapsing Panels

To collapse a panel click the small pin in the top right of the panels caption bar.



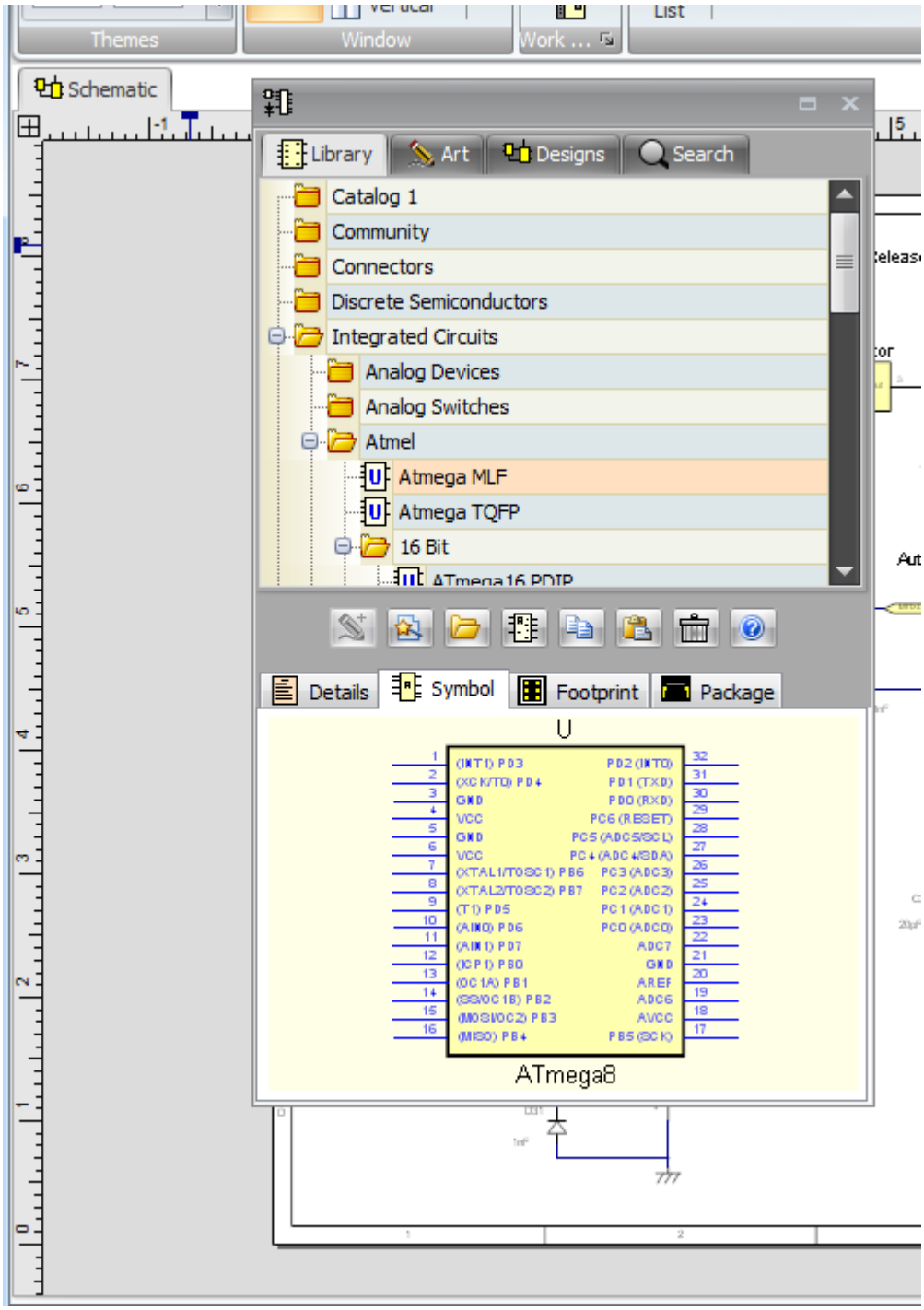
Non-collapsed Docked Panel



Collapsed Docked Panel

Floating Panels

Double-click on the panels caption bar to float it.

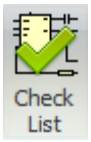


Floated Panel

1.3.3.1 The Checklist Panel

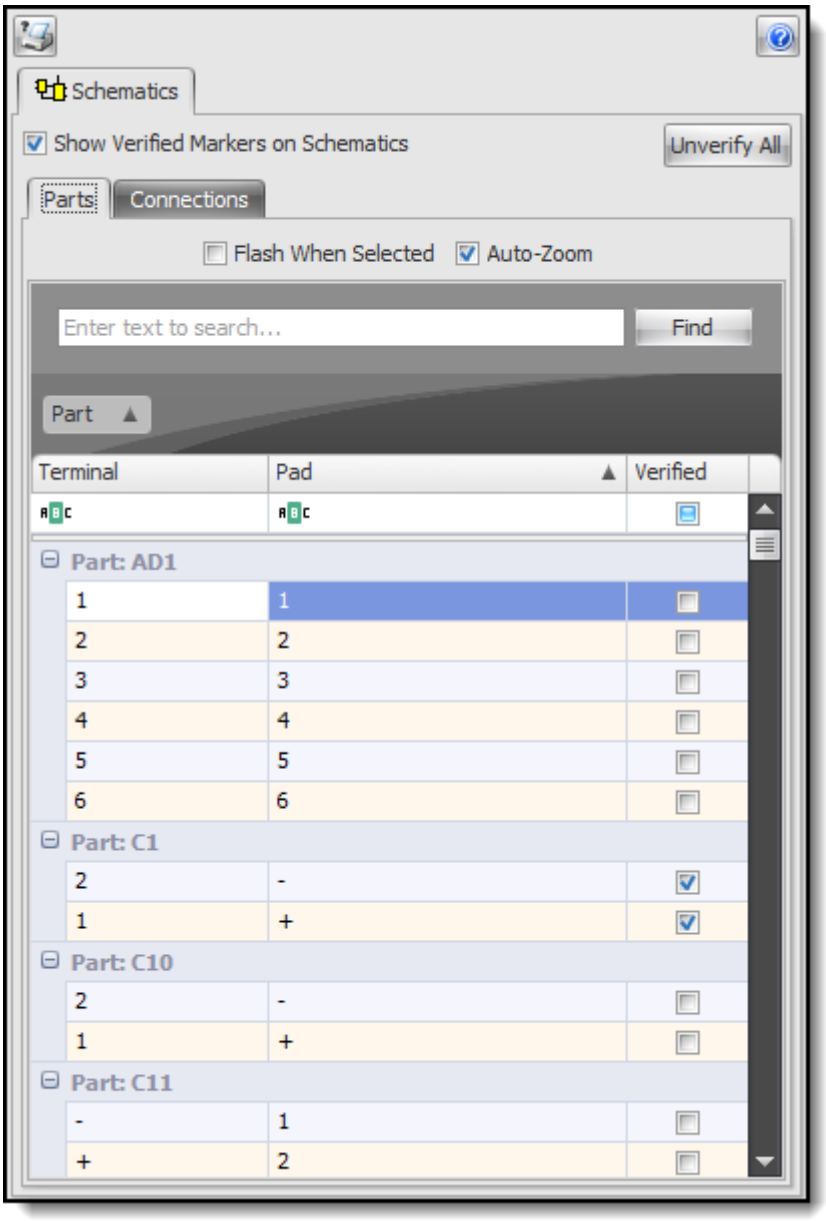
When you are at certain key stages in your design such as when you have completed the schematic design it is good practice to check your design. AutoTRAX DEX provides you with the Checklist panel for just such a purpose.

The Checklist panel is a collection of controls that let you 'check off' you design.

To view/hide the Checklist Information panel click the Panels→Panels→ 

button.

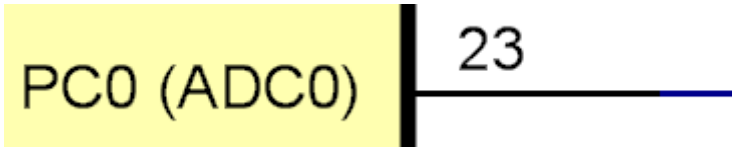
Parts



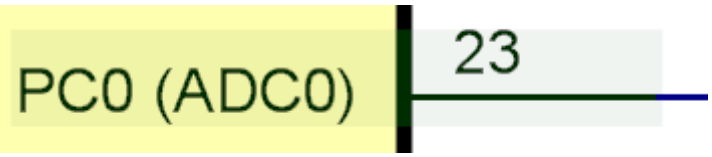
Use this to check off and mark parts as verified in the design.

Check the 

checkbox to display verified markers on your schematics. The markers are similar to marks left by a marker pen but much neater!



Terminal without verified marker



Terminal showing verified marker (grayed rectangle)

When you select an item it will be selected in the schematic

Selected Item

Click the  button to print the list.

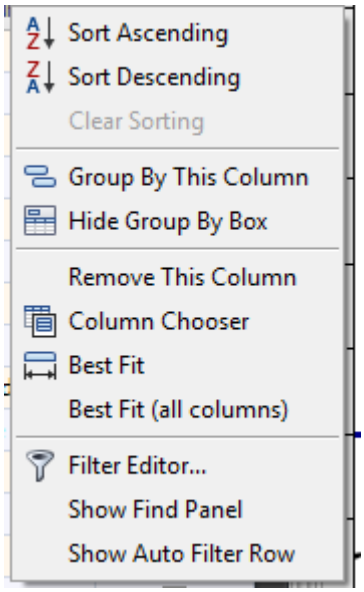
Click the 

button to remove all verified status for the design.

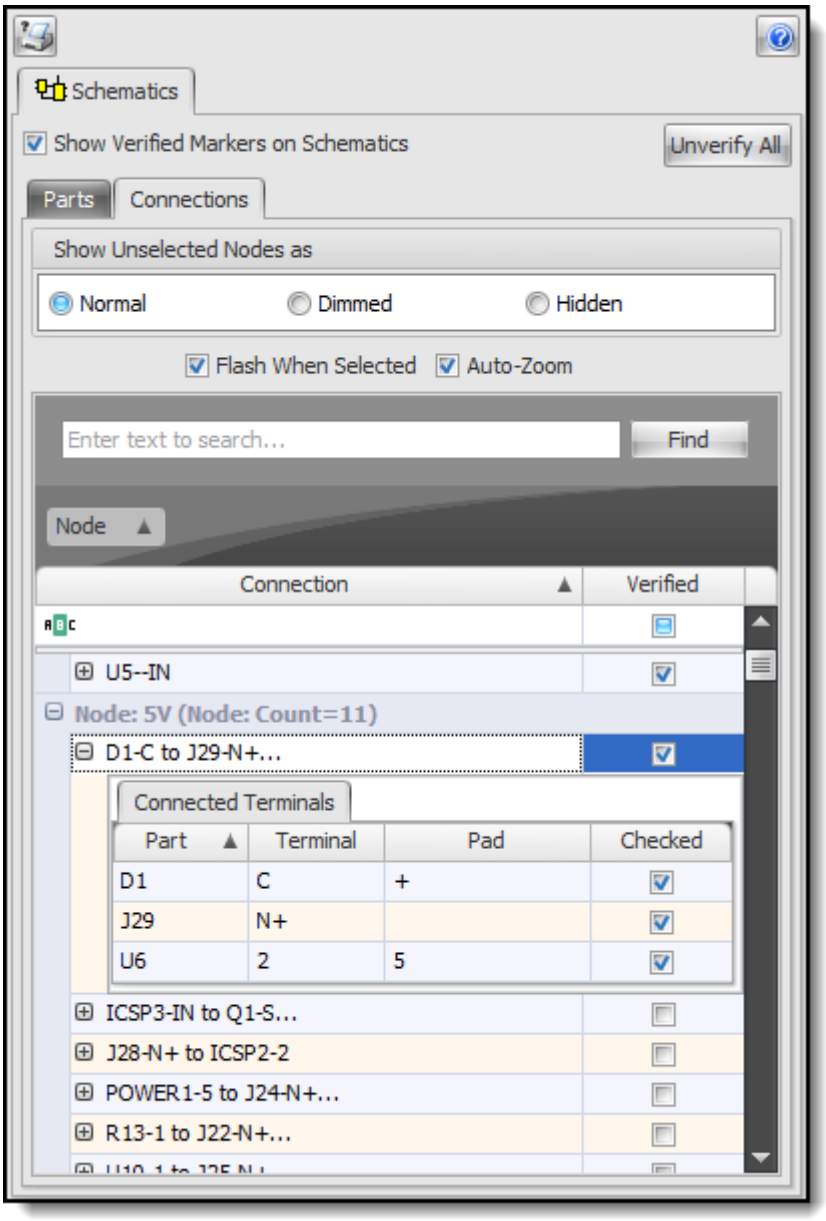
Click the  button to display this help topic.

You can *sort the list* by clicking on the column header.

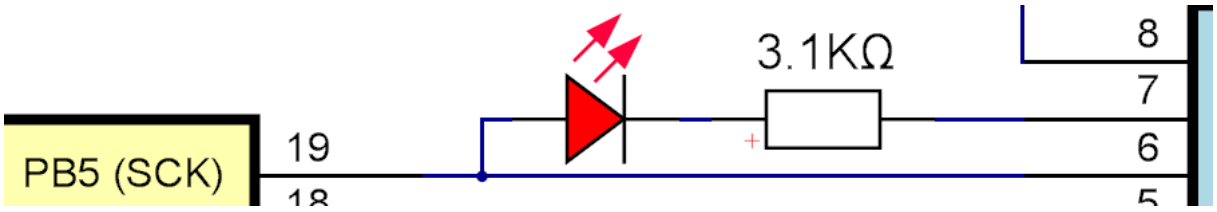
Right click on the column header to find **more options**.



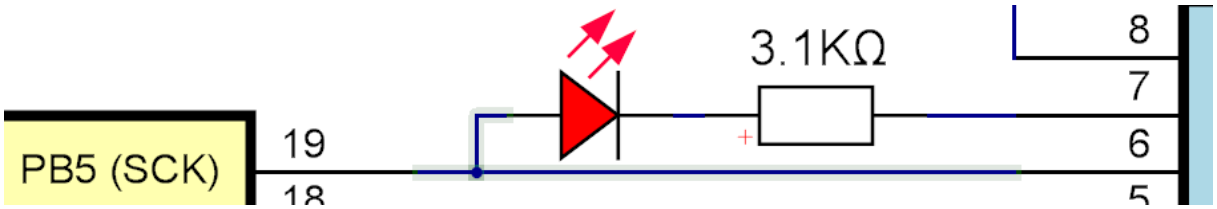
Connections



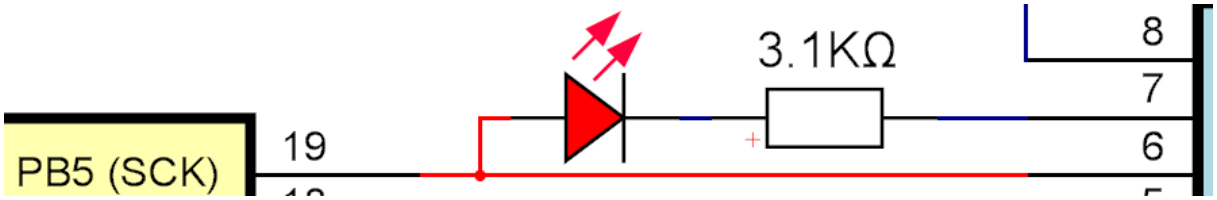
Use this to check off and mark electrical connections (wires/nodes) as verified in the design.



Connection showing without verified marker



Connection showing verified marker (grayed rectangles)



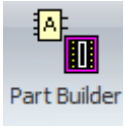
Selected Item

Check the Flash When Selected to flash the selected item.

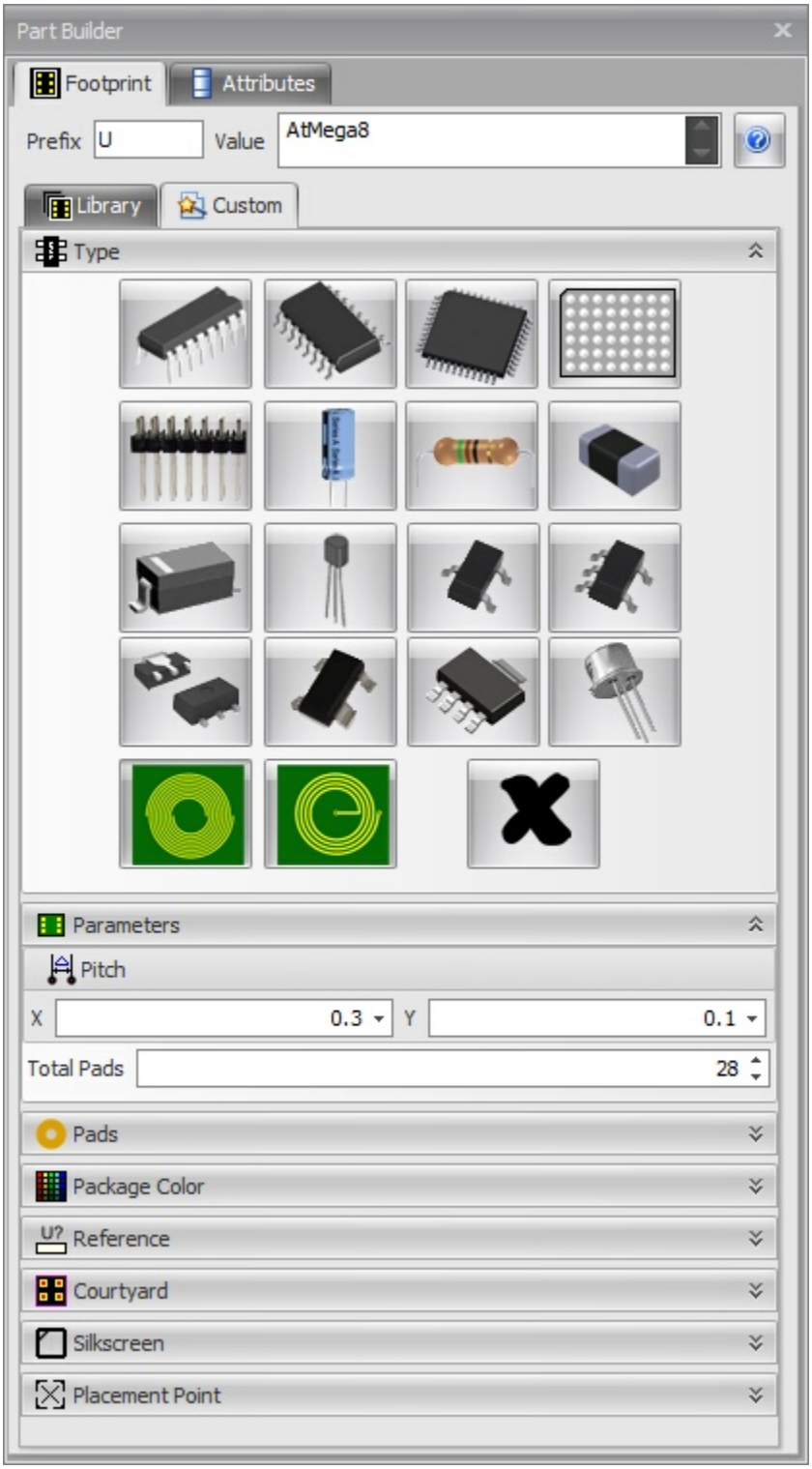
Check the Auto-Zoom to have the item zoomed into view when selected. This combined with Flash When Selected makes them easier to find.

1.3.3.2 The Part Builder Panel

The Parts Builder Panel groups several common controls and displays them based your current context. For instance, if you are editing a part's schematic symbol, the symbol wizard will be shown. If you are editing a part's footprint, the footprint wizard will appear.

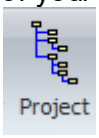
To view/hide the Parts Builder panel click the Panels→Panels→ 

button.

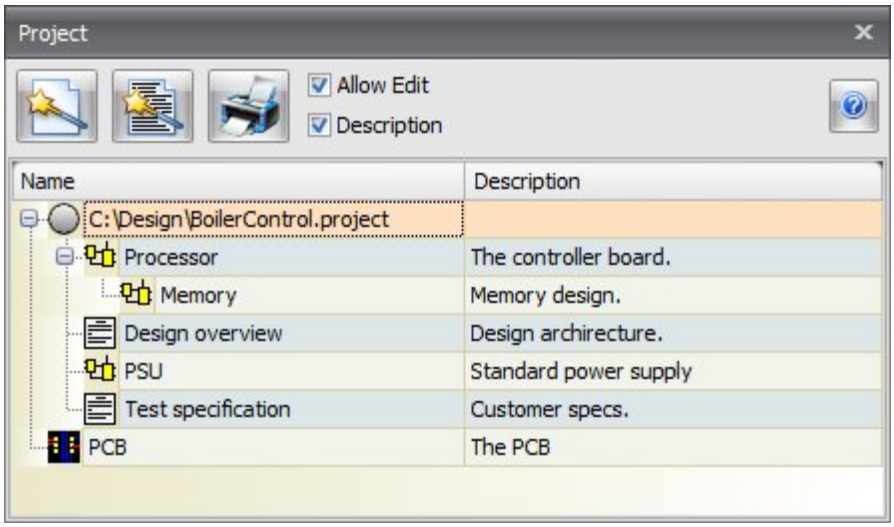


The Part Builder Panel

1.3.3.3 The Project Panel

The **Project Panel** displays a tree list of the contents of your projects. It is shown below. To display the **Project Panel** click on the Projects button 

in the Panels ribbon tab or ***right-click*** on a viewport and select **Project Panel** from the context menu.

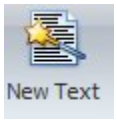


The Project Panel

Using the **Project Panel** you can add sheets and text documents as well as view and edit them.

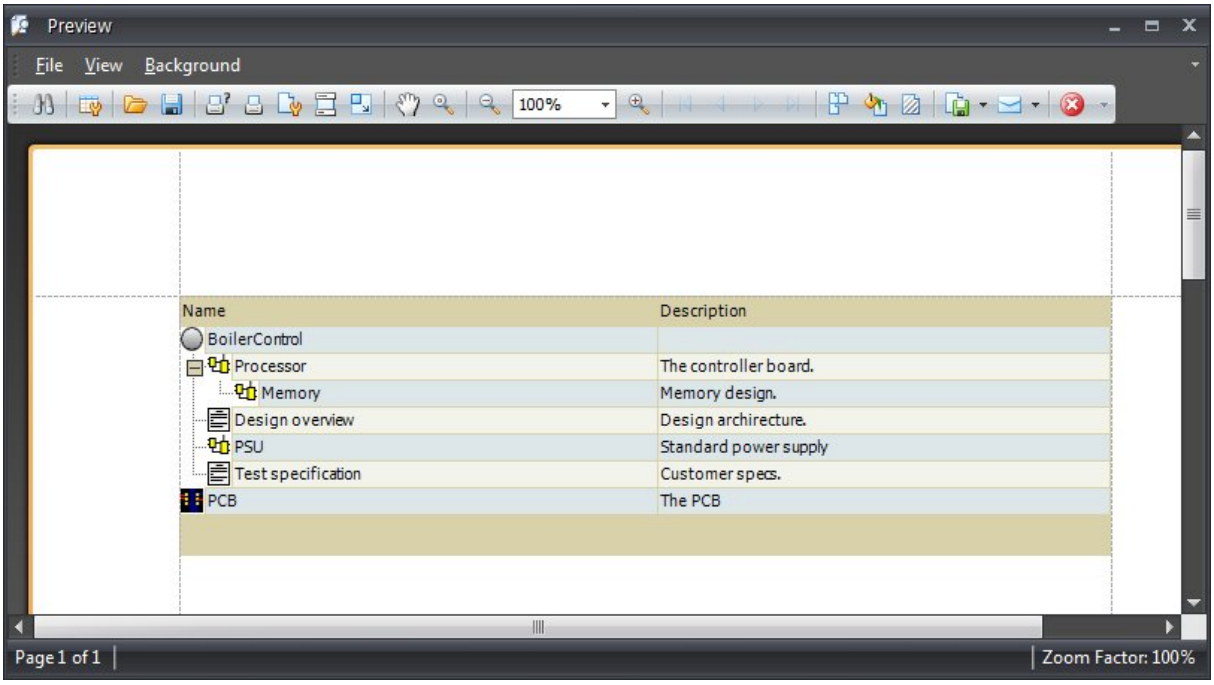
Double-click on a project item schematic, PCB or text document to view it in a viewport. If a viewport already exists that contains the item to view, then it will be brought to the front.

 Adds a new schematic sheet.



Adds a new text document.

 Prints the project hierarchy. This will display the print preview dialogs similar to the one below. You can print it or save it to PDF and several other formats.



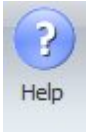
Allow Edit If checked then you can edit the name of the sheet or document as well as the description. Otherwise they are protected and read only.

Description Show/Hide the description column.

Both the Allow Edit and Descriptions settings are saved from session to session.

Expanded directories are also saved from session to session.

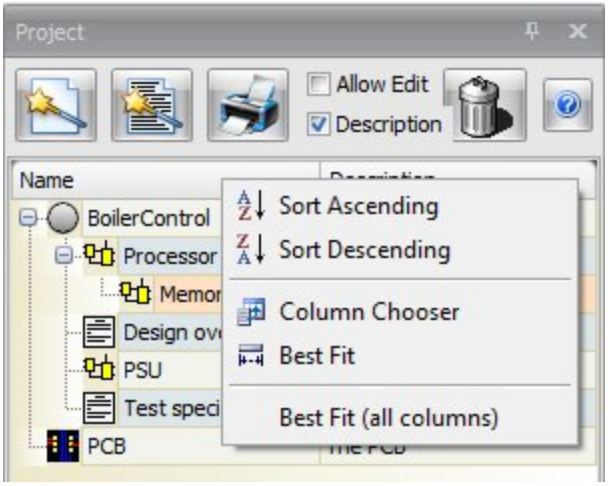
Right-click in the Project panel to display the context menu. This will allow you to add sheets and delete selected sheet.



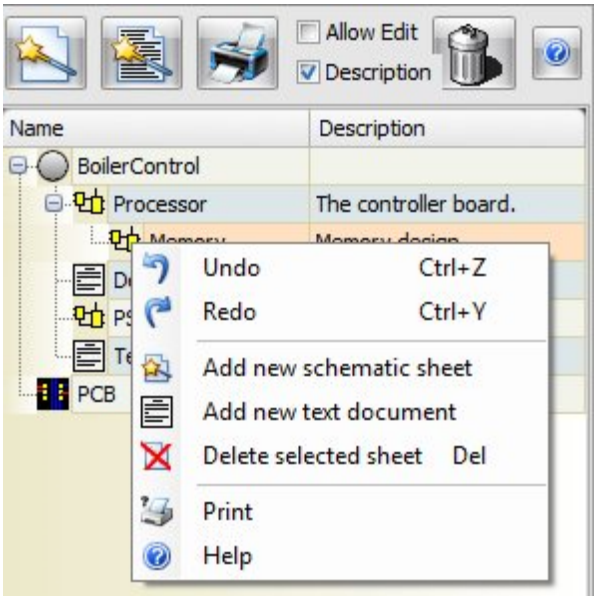
Displays this help page.

You can drag sheets to reposition them in the project hierarchy.

Right-click on the Name or Description column header to display the header context menu shown below. This will allow you to sort columns etc.

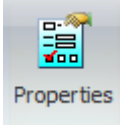


Right-click over a sheet to see the context menu as shown below.



1.3.3.4 The Properties Panel

The Properties Panel lets you edit the properties of selected objects on schematic and PCB sheets.

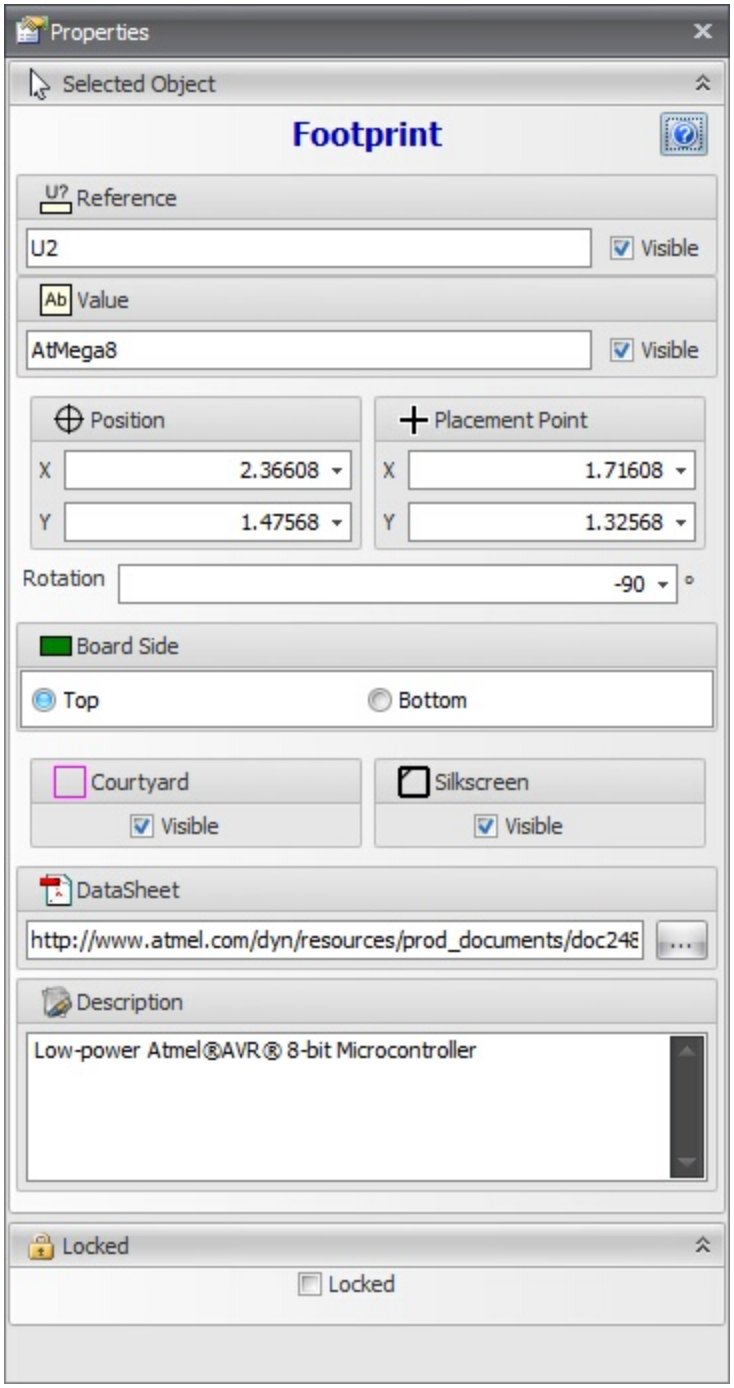
To view/hide the Properties panel click the Panels→Panels→ 

button.

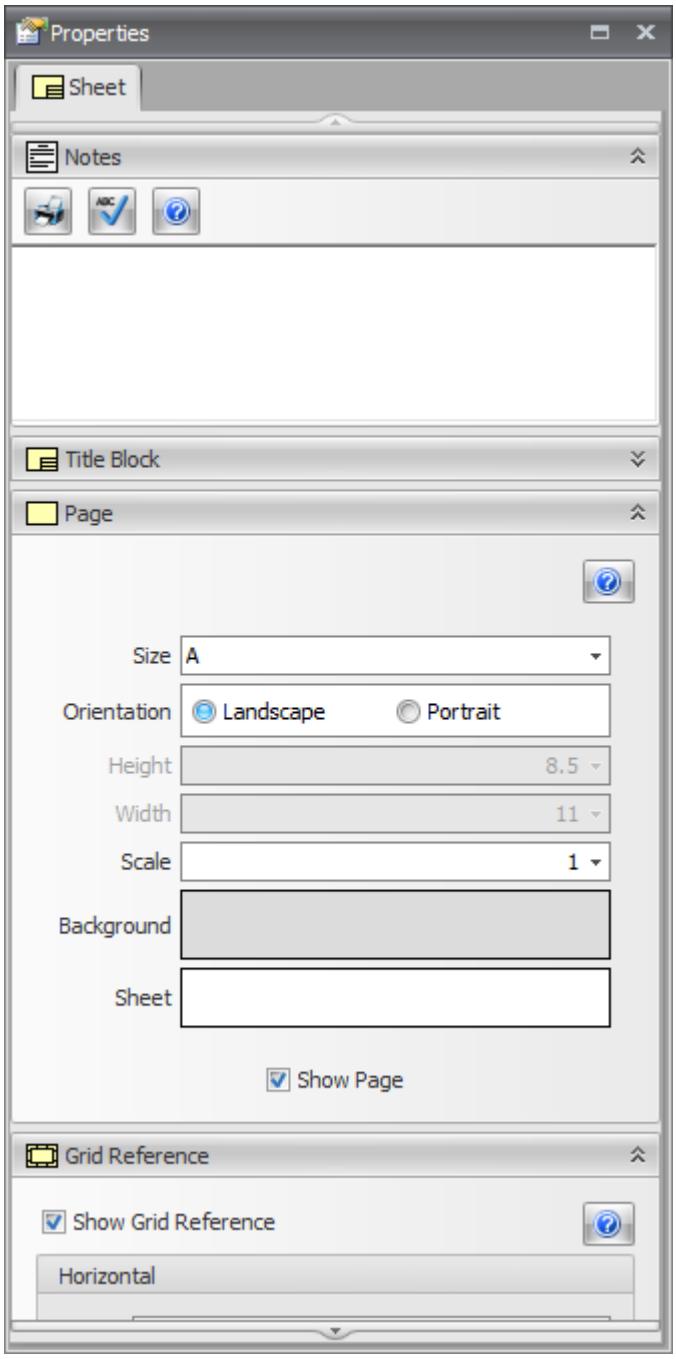
When you select an object on a schematic or PCB the properties panel will display an editor that is specifically tailored for the selected object. Below you can see the editor for a selected footprint. If nothing is selected then you will see an editor for the sheet in the currently selected viewport.

To find out more details of the editor for the selected object click the  button in the property panel.

To discover more about the sheet editor view the sheet editor details.

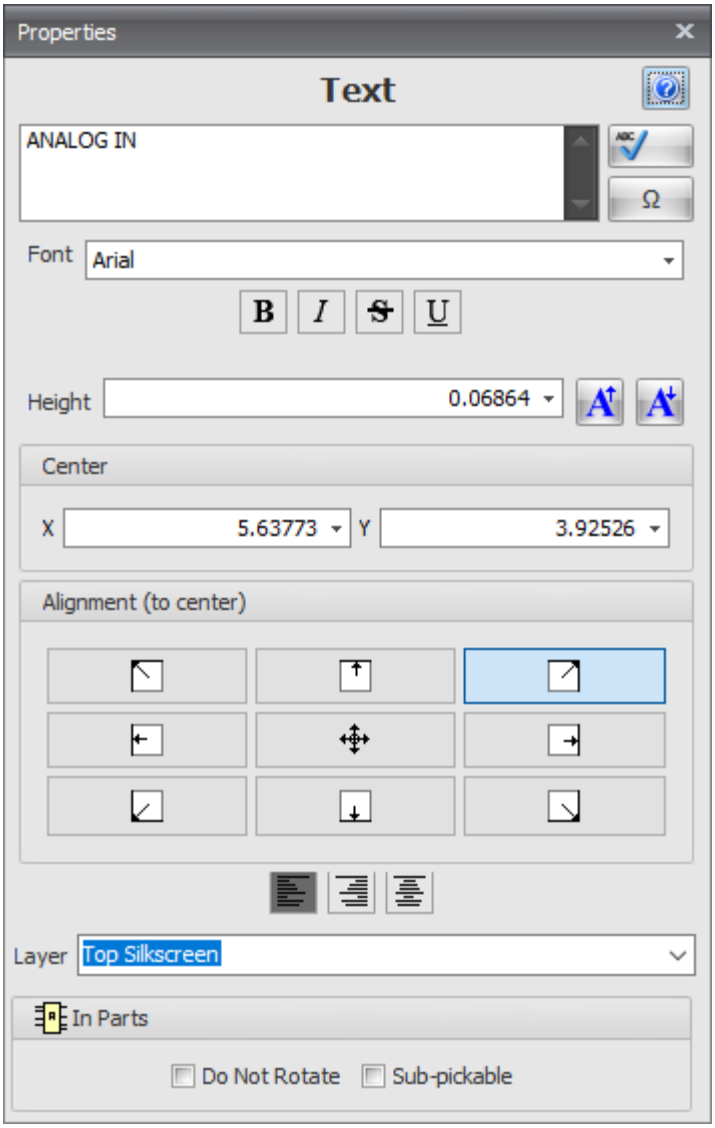


Editor for selected footprint



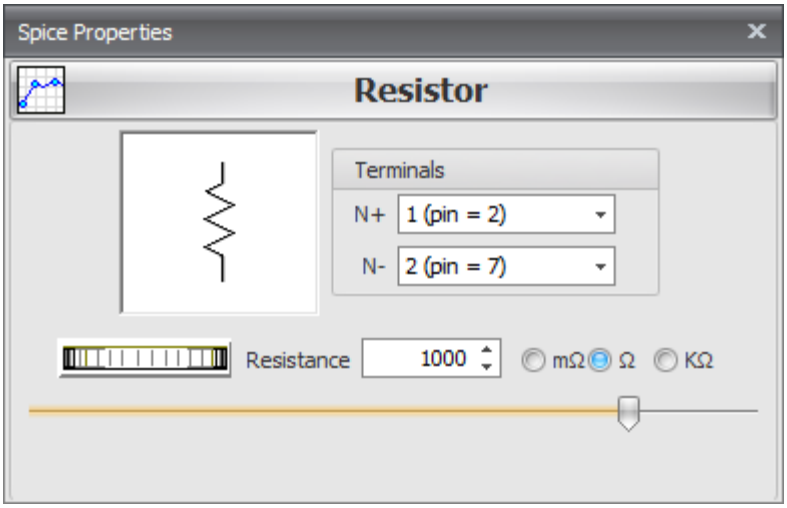
Editor for sheet (nothing selected)

1.3.3.5 The Properties Popup Panel



The Properties Popup Panel

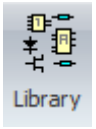
1.3.3.6 The Spice Model Properties Popup Panel



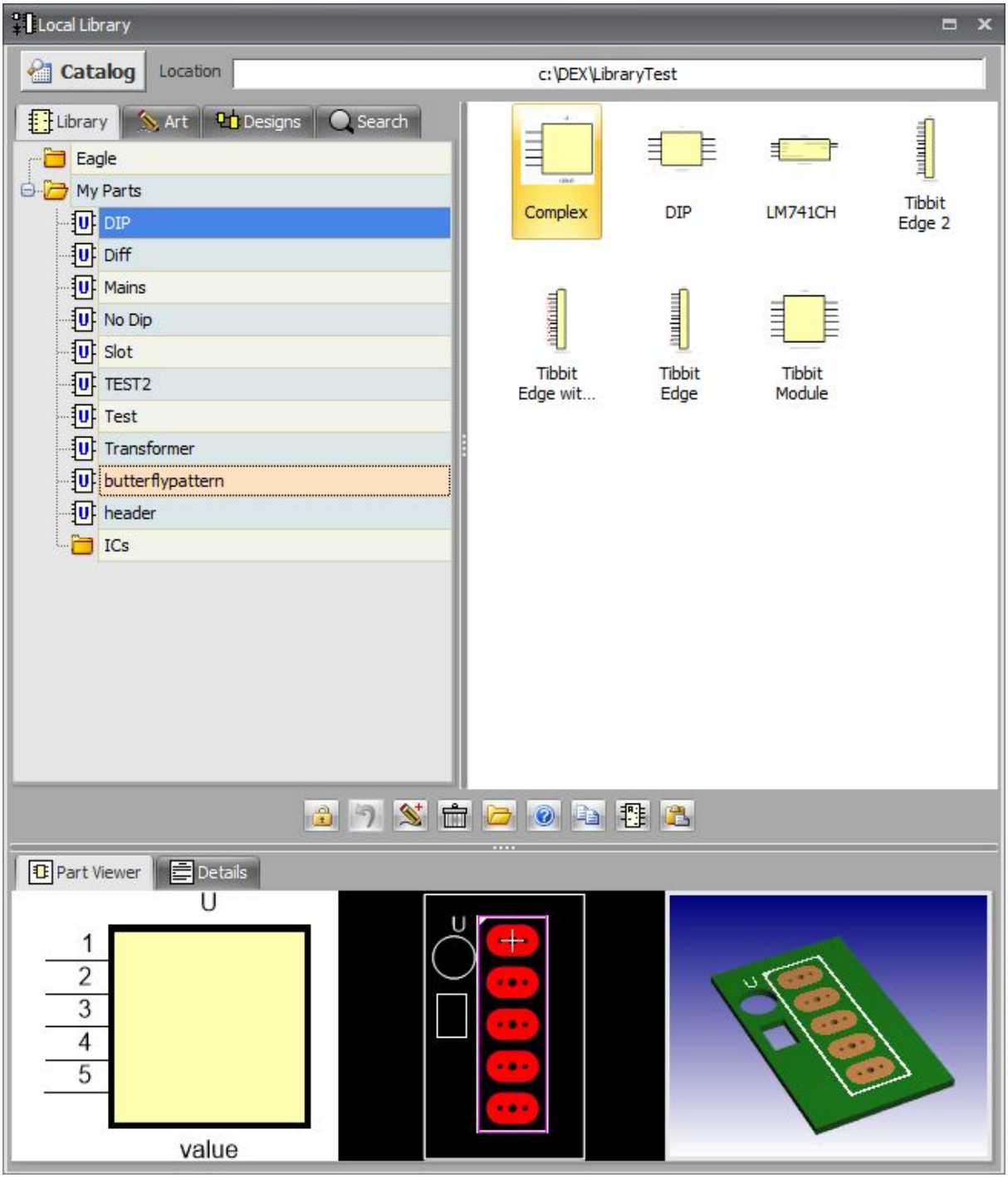
The Spice Model Properties Popup Panel

1.3.3.7 The Library Panel

The Local Library Panel is shown below.

To view/hide the Library panel click the Panels→Panels→ 

button.

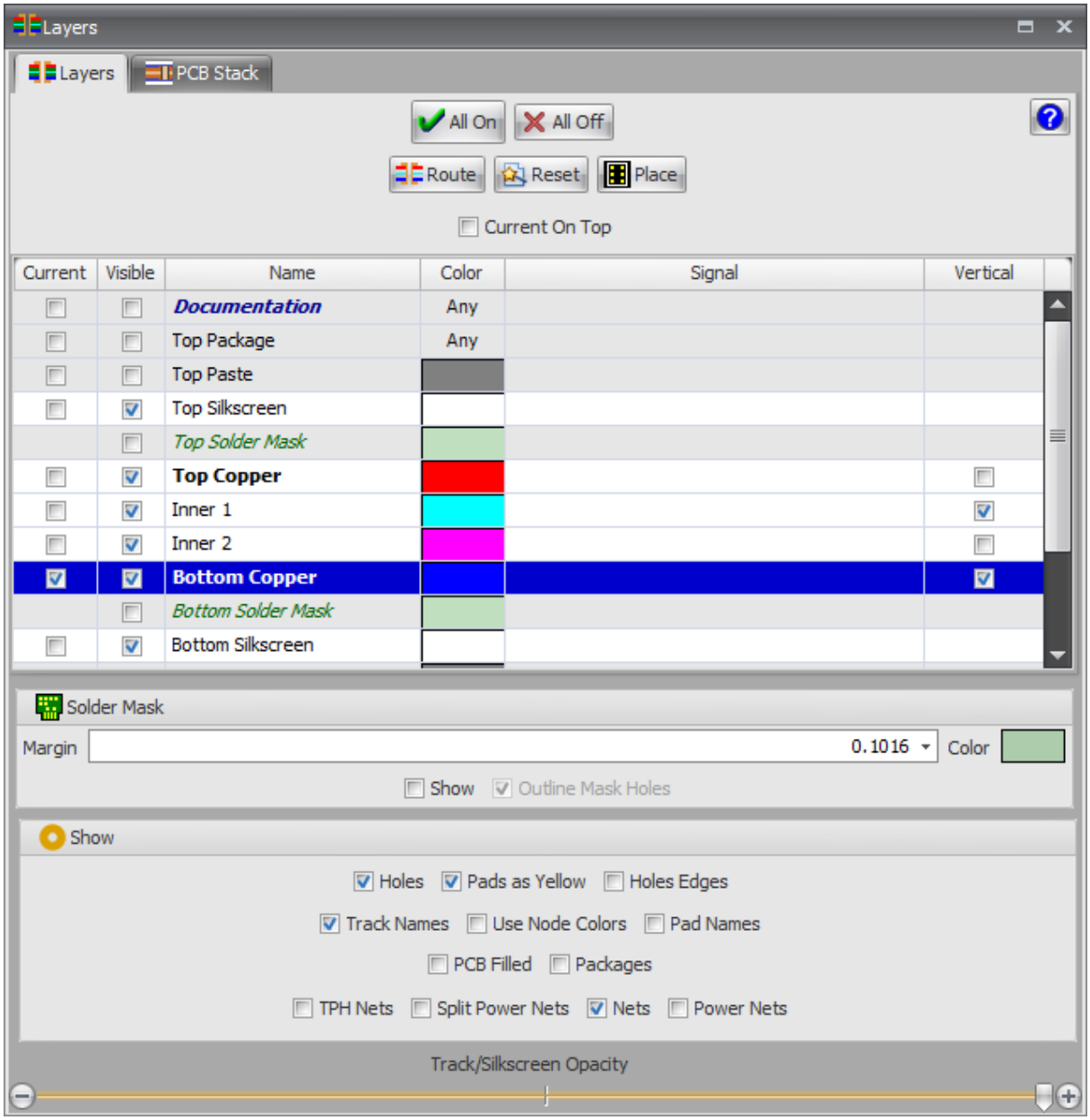


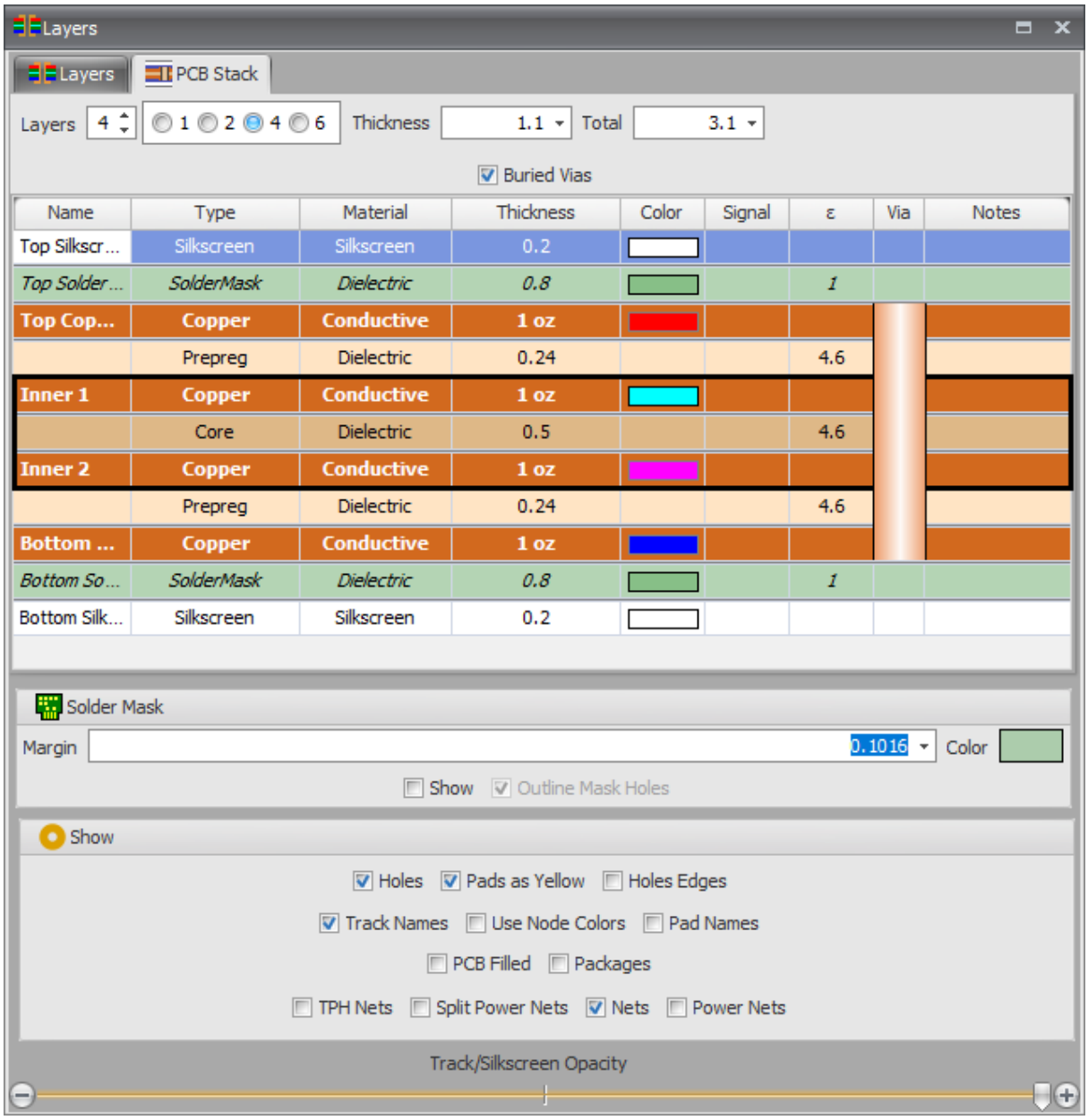
1.3.3.8 The Layers Panel

The Layers Panel is shown below.

To view/hide the Layers panel click the Panels→Panels→ 

button.





The Layers Popup

1.3.3.9 The Design Rules Checker Panel

The Design Rules Checker Panel is shown below.

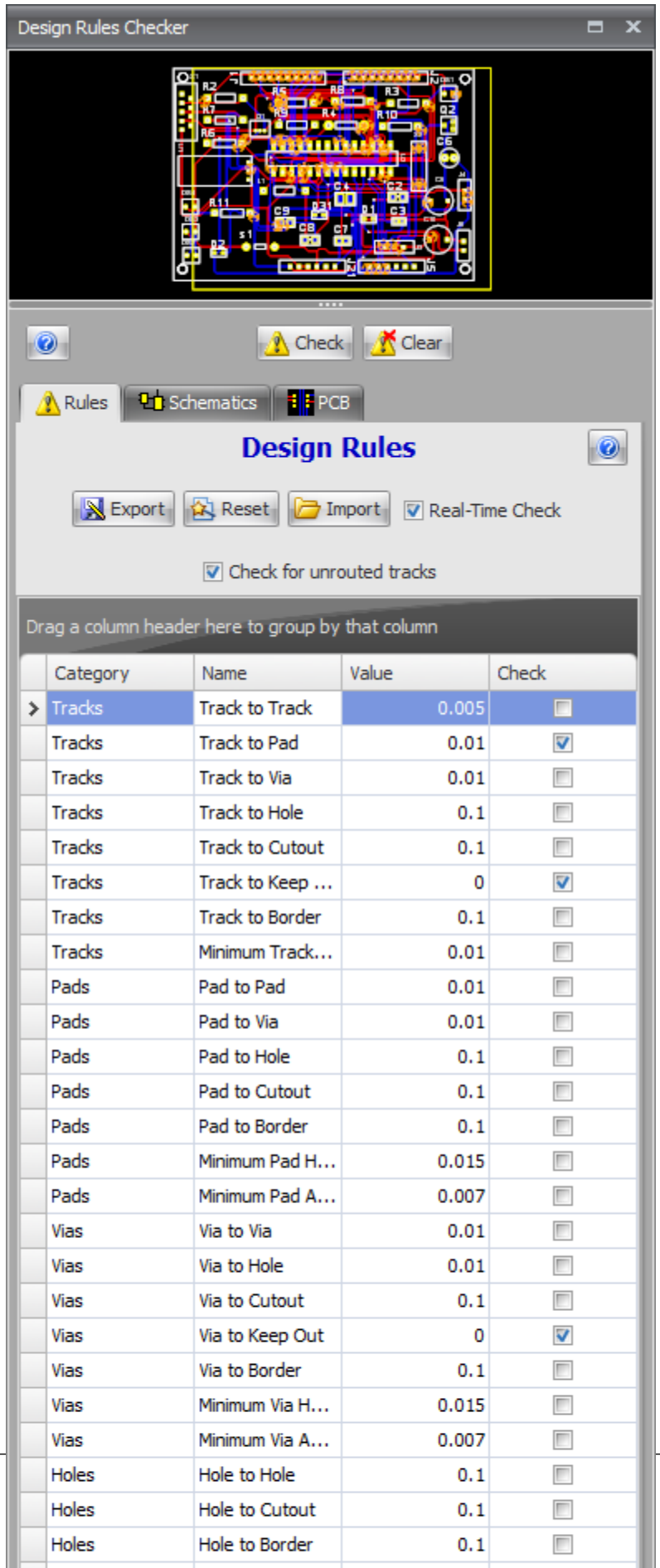
To view/hide the Design Rules Checker panel click the Panels→Panels→ 

button.

You can 

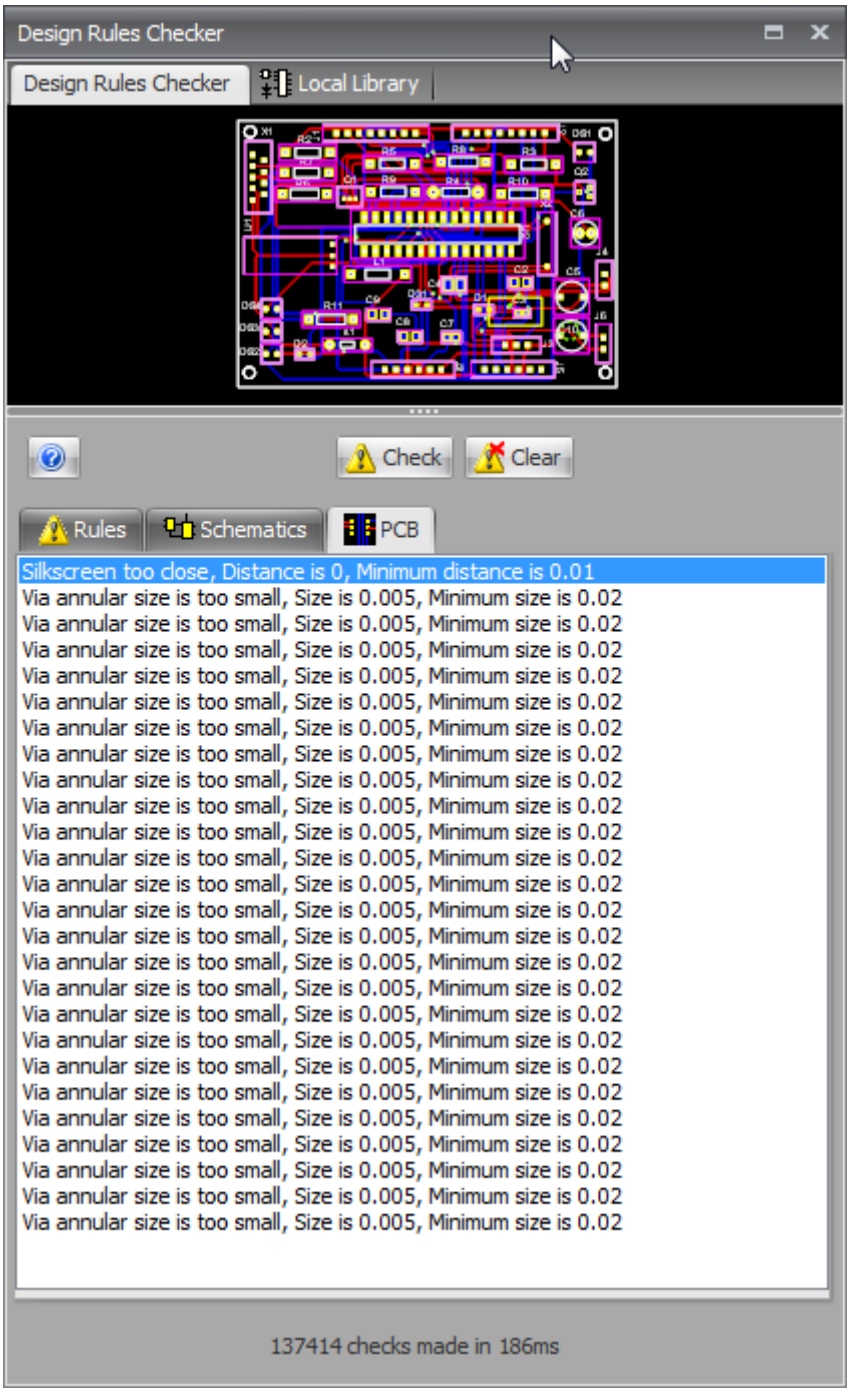
and 

the design rules.



Category	Name	Value	Check
Tracks	Track to Track	0.005	☐
Tracks	Track to Pad	0.01	☑
Tracks	Track to Via	0.01	☐
Tracks	Track to Hole	0.1	☐
Tracks	Track to Cutout	0.1	☐
Tracks	Track to Keep ...	0	☑
Tracks	Track to Border	0.1	☐
Tracks	Minimum Track...	0.01	☐
Pads	Pad to Pad	0.01	☐
Pads	Pad to Via	0.01	☐
Pads	Pad to Hole	0.1	☐
Pads	Pad to Cutout	0.1	☐
Pads	Pad to Border	0.1	☐
Pads	Minimum Pad H...	0.015	☐
Pads	Minimum Pad A...	0.007	☐
Vias	Via to Via	0.01	☐
Vias	Via to Hole	0.01	☐
Vias	Via to Cutout	0.1	☐
Vias	Via to Keep Out	0	☑
Vias	Via to Border	0.1	☐
Vias	Minimum Via H...	0.015	☐
Vias	Minimum Via A...	0.007	☐
Holes	Hole to Hole	0.1	☐
Holes	Hole to Cutout	0.1	☐
Holes	Hole to Border	0.1	☐

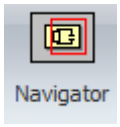
The Design Rule Checker Panel



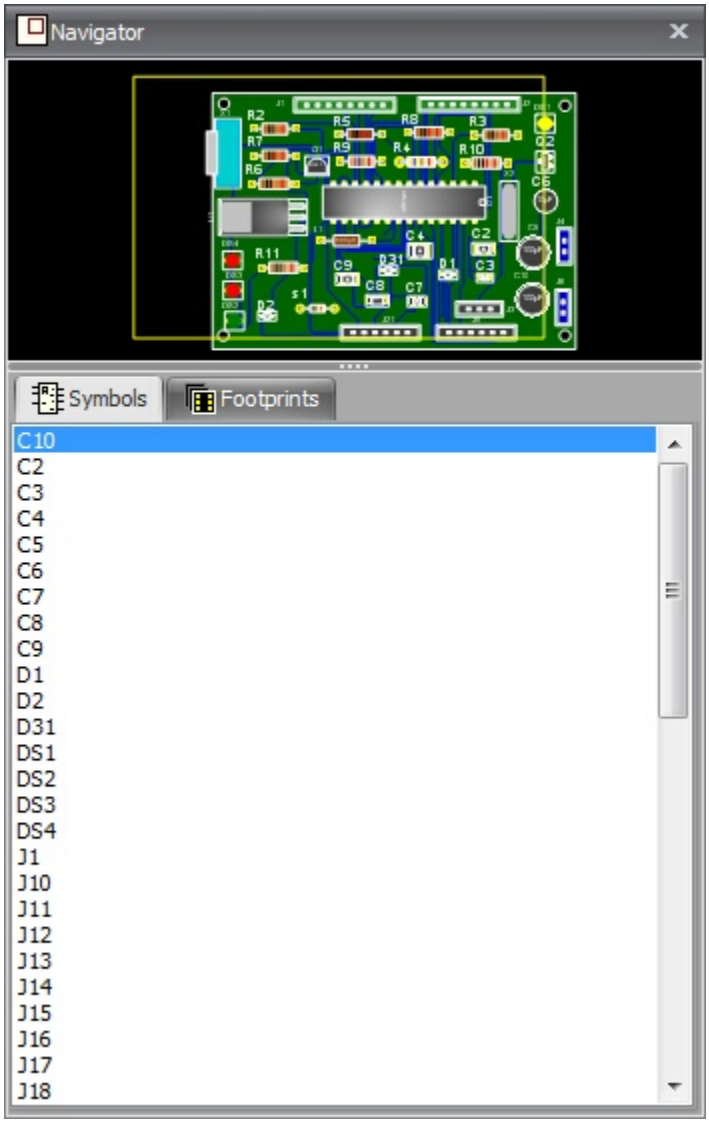
Typical PCB Error Display

1.3.3.10 The Navigator Panel

The Navigator Panel is shown below.

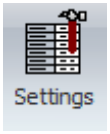
To view/hide the Navigator panel click the Panels→Panels→ 

button.

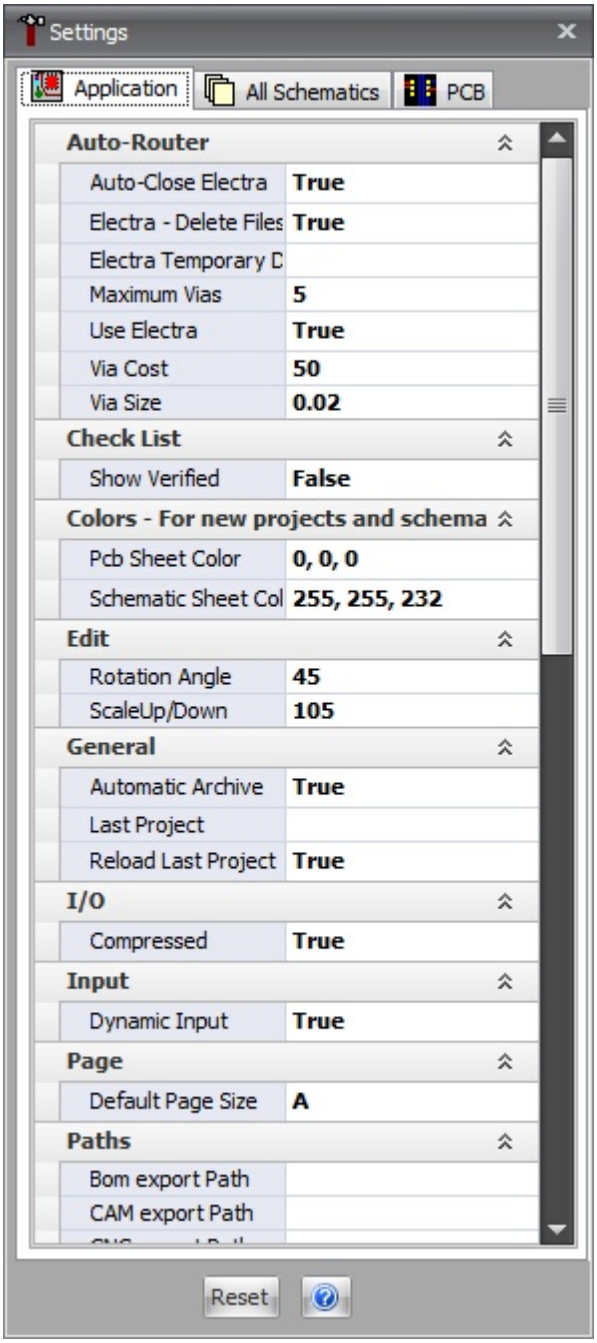


1.3.3.11 The Settings Panel

The Settings Panel lets you view and edit the individual settings and preferences for sheets and the project.

To view/hide the Settings panel click the Panels→Panels→ 

button.

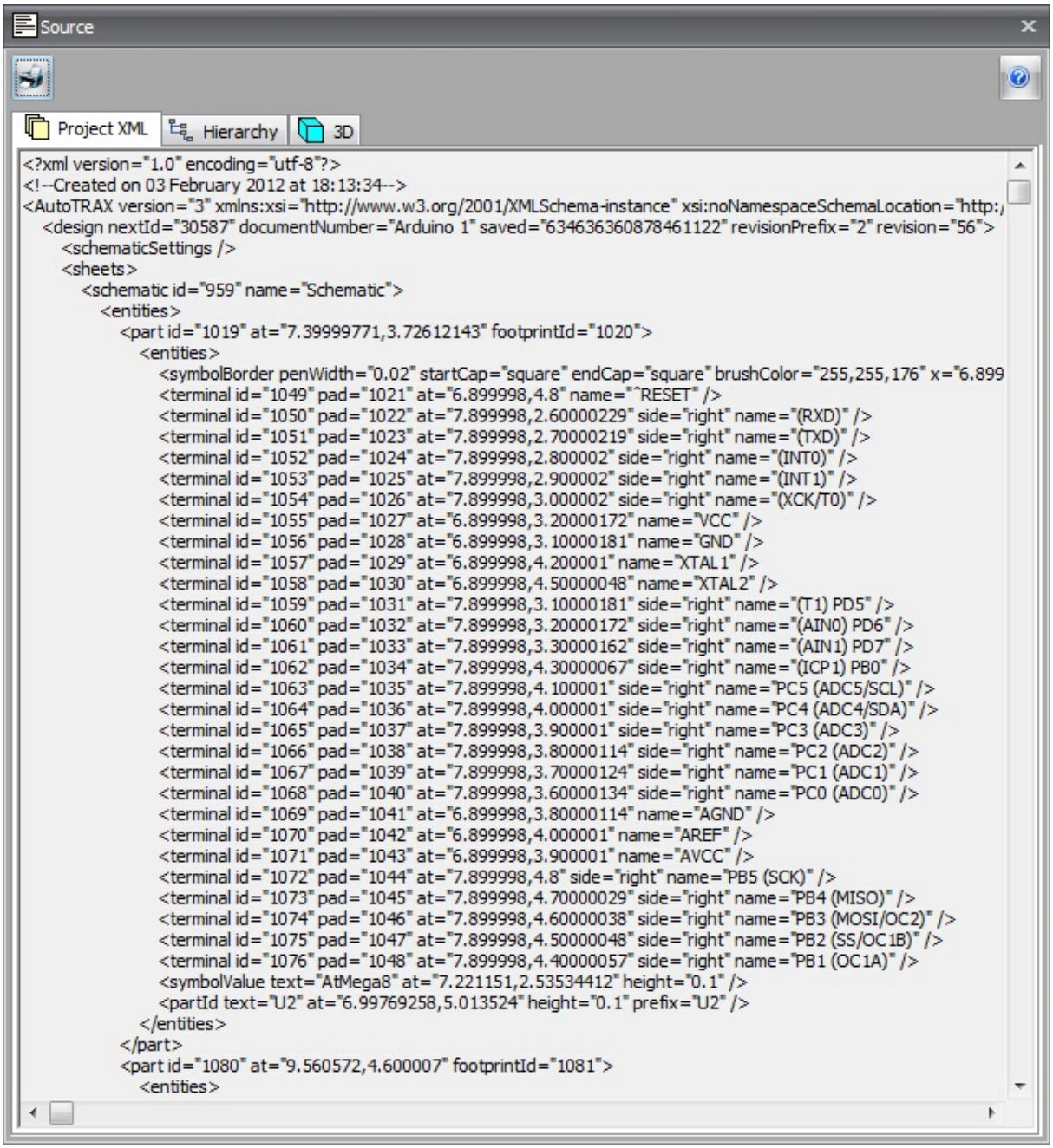


The AutoTRAX DEX Settings Panel is a central control that lets you view and change all the parameters that are used by AutoTRAX DEX.

1.3.3.12 The Source View Panel

The Source Panel displays the XML file data that describes your project.

To view/hide the Source panel click the Panels→Panels→ Source button.

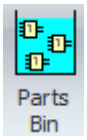


The file format for all project and part files in AutoTRAX DEX are industry standard XML files. The files are text files, readable by you. AutoTRAX DEX files are open and

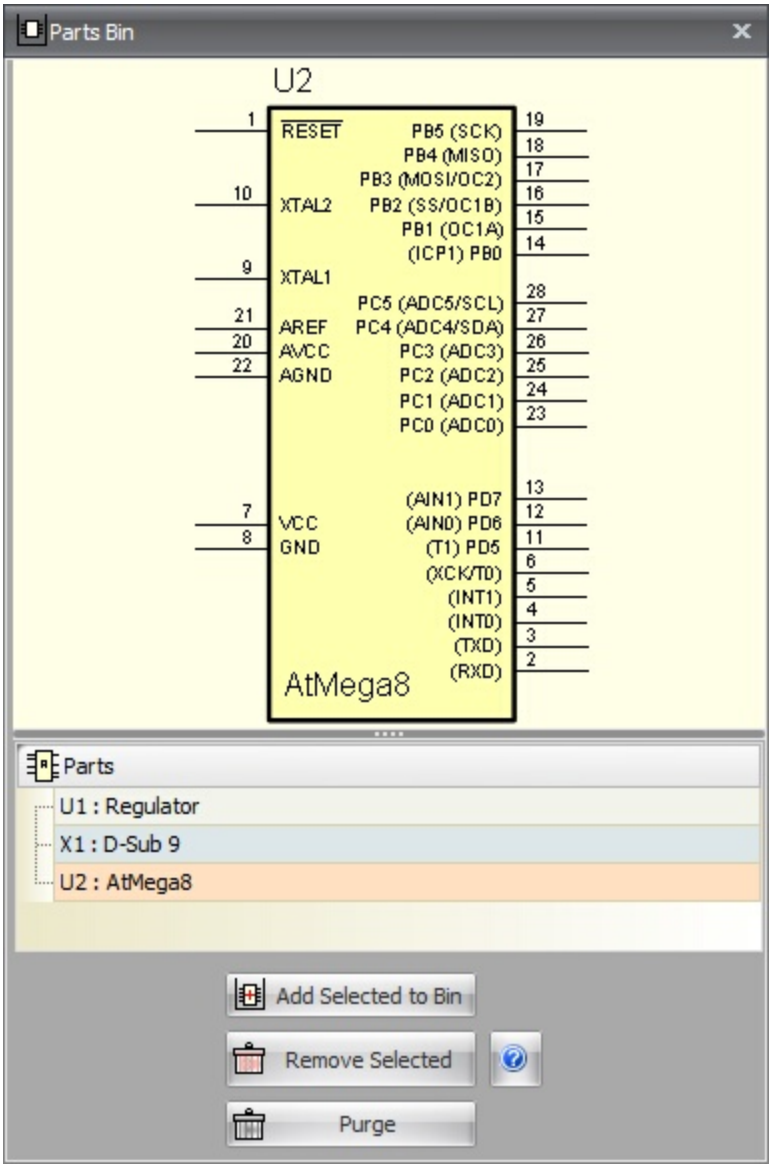
the specification is freely available from the AutoTRAX DEX web site. Even AutoTRAX DEX's config files are human readable XML files.

1.3.3.13 The Parts Bin Panel

The Parts Bin Panel contains parts that you do not wish to appear on schematic sheets. It acts as a holding bin for parts not yet needed in a design.

To view/hide the Parts Bin panel click the Panels→Panels→ 

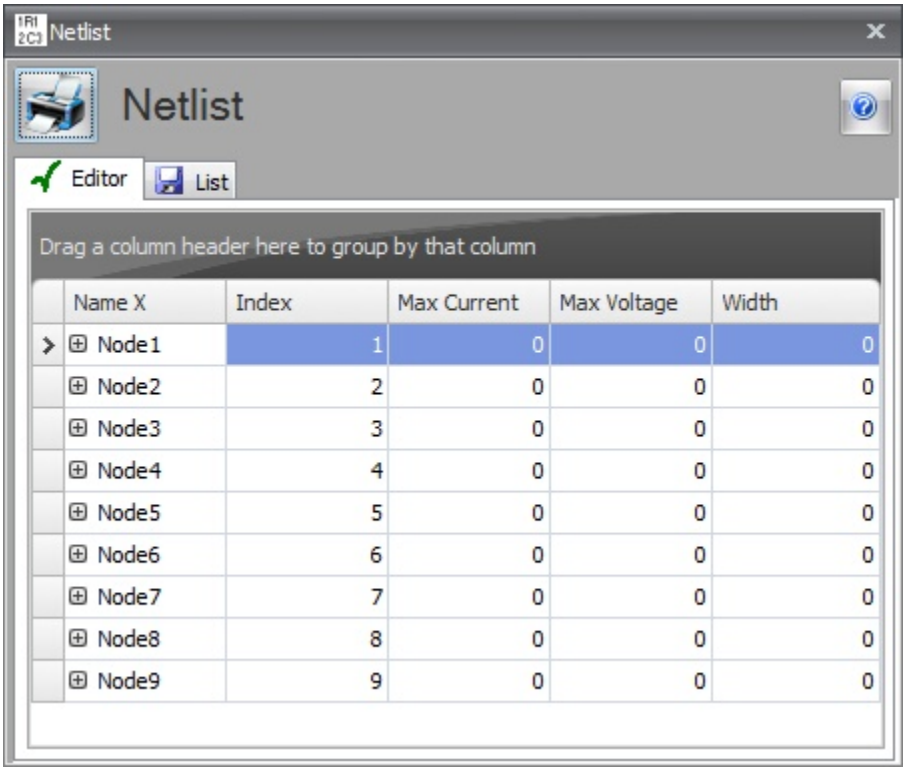
button.



1.3.3.14 The Netlist Panel

The Netlist Panel shows you the netlist, in a spreadsheet format that details the electrical connectivity of your design.

To view/hide the Netlist panel click the Panels→Panels→ net list button.



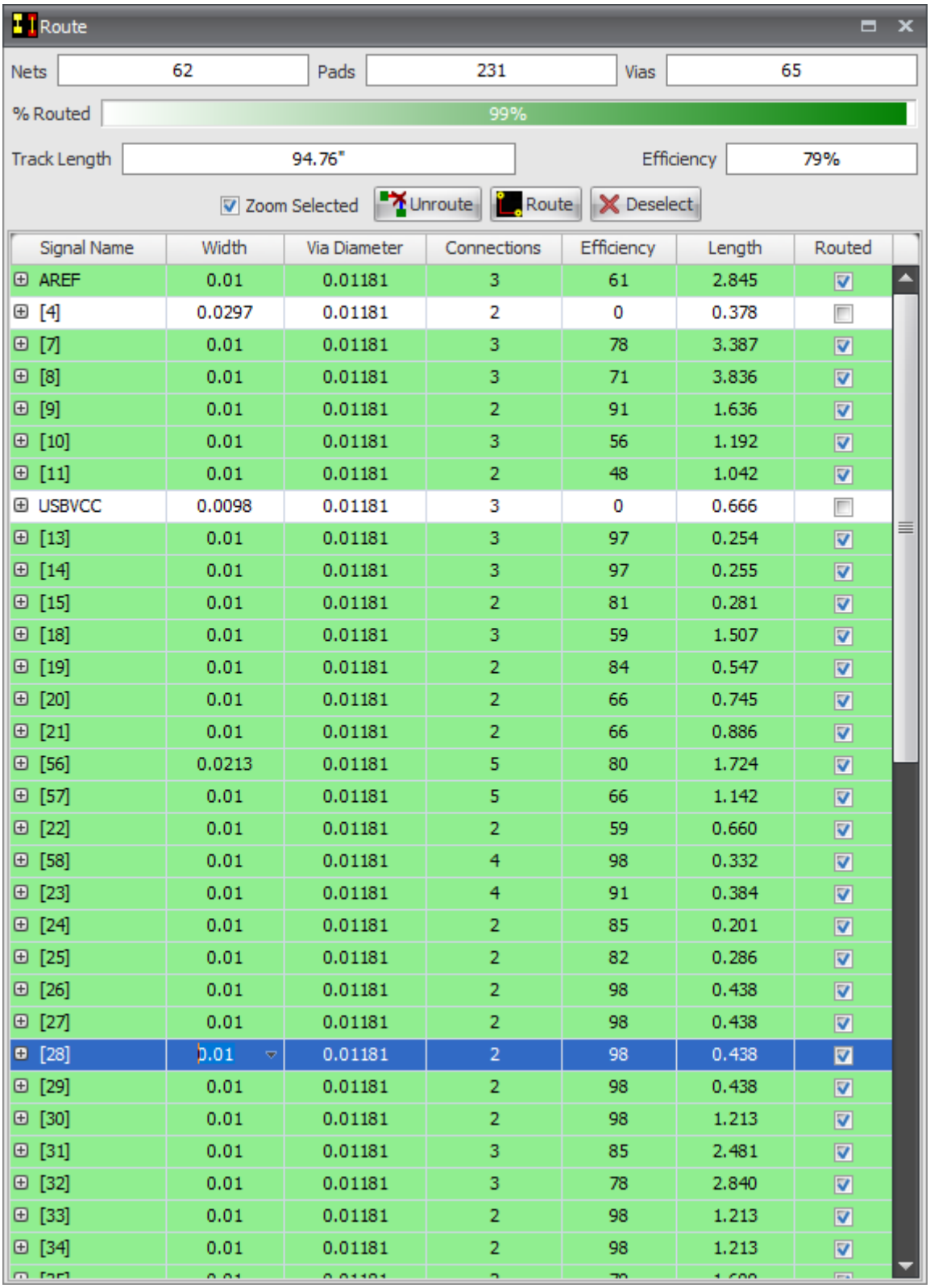
1.3.3.15 The Parts List Panel

The Parts List Panel displays the parts in your project.

To view/hide the System Information panel click the Panels→Panels→ Parts List button.

1.3.3.16 The Route Panel

Using the Route Panel to quickly locate schematic nodes and PCB nets in your design

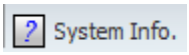


Signal Name	Width	Via Diameter	Connections	Efficiency	Length	Routed
AREF	0.01	0.01181	3	61	2.845	☑
[4]	0.0297	0.01181	2	0	0.378	☐
[7]	0.01	0.01181	3	78	3.387	☑
[8]	0.01	0.01181	3	71	3.836	☑
[9]	0.01	0.01181	2	91	1.636	☑
[10]	0.01	0.01181	3	56	1.192	☑
[11]	0.01	0.01181	2	48	1.042	☑
USBVCC	0.0098	0.01181	3	0	0.666	☐
[13]	0.01	0.01181	3	97	0.254	☑
[14]	0.01	0.01181	3	97	0.255	☑
[15]	0.01	0.01181	2	81	0.281	☑
[18]	0.01	0.01181	3	59	1.507	☑
[19]	0.01	0.01181	2	84	0.547	☑
[20]	0.01	0.01181	2	66	0.745	☑
[21]	0.01	0.01181	2	66	0.886	☑
[56]	0.0213	0.01181	5	80	1.724	☑
[57]	0.01	0.01181	5	66	1.142	☑
[22]	0.01	0.01181	2	59	0.660	☑
[58]	0.01	0.01181	4	98	0.332	☑
[23]	0.01	0.01181	4	91	0.384	☑
[24]	0.01	0.01181	2	85	0.201	☑
[25]	0.01	0.01181	2	82	0.286	☑
[26]	0.01	0.01181	2	98	0.438	☑
[27]	0.01	0.01181	2	98	0.438	☑
[28]	0.01	0.01181	2	98	0.438	☑
[29]	0.01	0.01181	2	98	0.438	☑
[30]	0.01	0.01181	2	98	1.213	☑
[31]	0.01	0.01181	3	85	2.481	☑
[32]	0.01	0.01181	3	78	2.840	☑
[33]	0.01	0.01181	2	98	1.213	☑
[34]	0.01	0.01181	2	98	1.213	☑

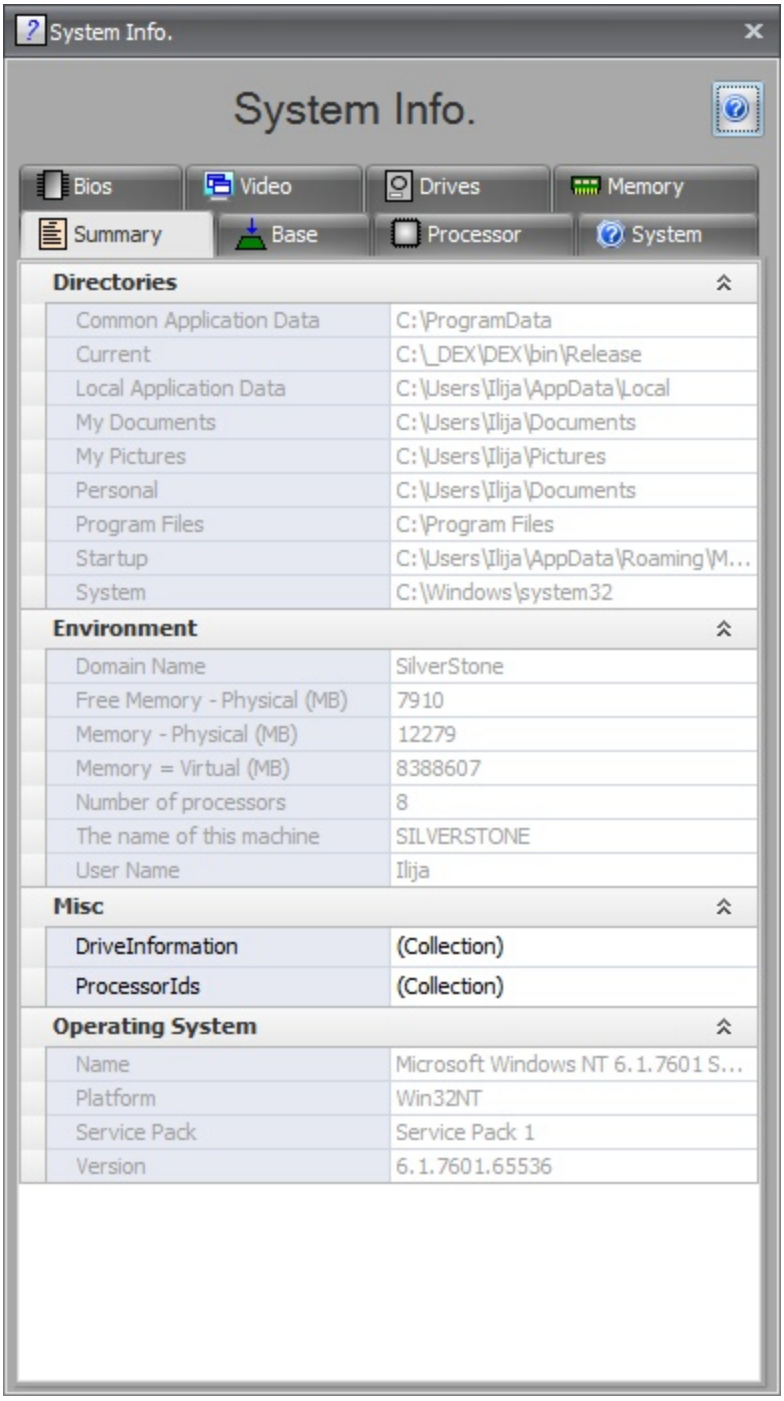
The Route Panel

1.3.3.17 The System Info. Panel

The System Information Panel is shown below. It provides extensive information about your machine and the operating system. It is a valuable tool to help you explore the capabilities of your machine and O/S. It can also help you track down problems with your machine. As the information it provides is vast and detailed the best way to find out what it has to offer is to click on each tab and view its contents. For more information about what each parameter represents I suggest you do a Google search.

To view/hide the System Information panel click the Panels→Panels→ 

button.



1.3.4 Viewports

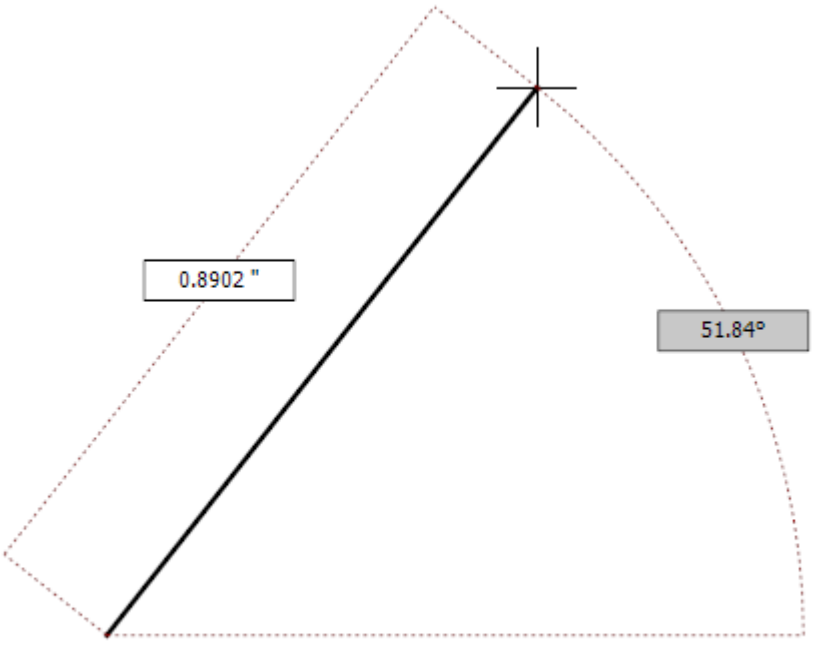
Viewports provide a graphical view into your design and can display:

- Schematic Sheets
- Symbols

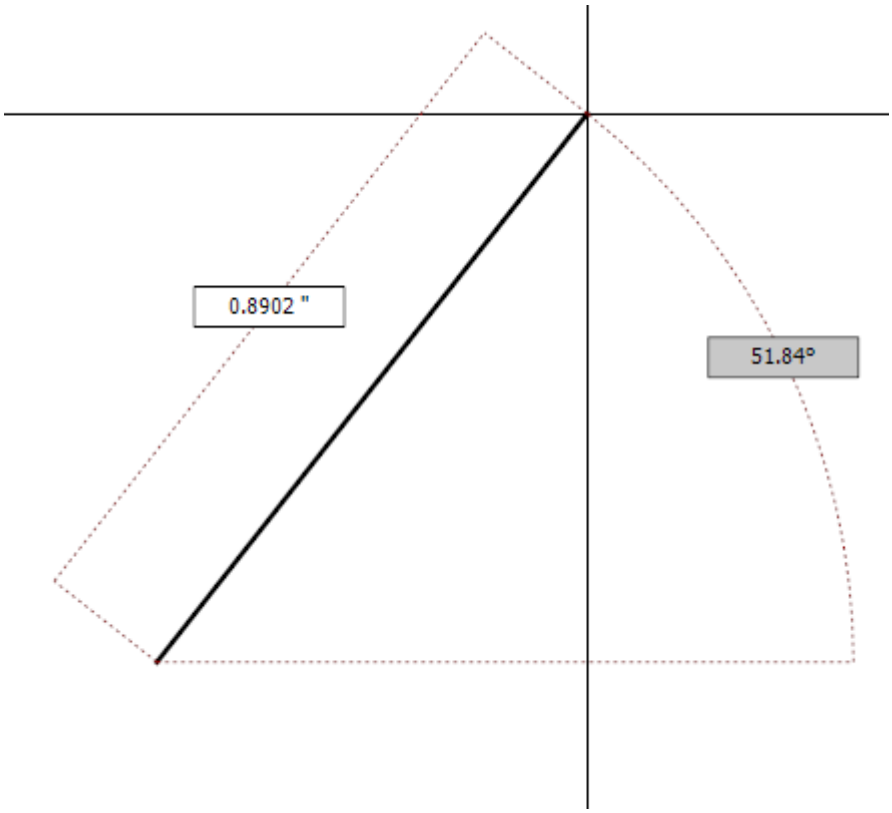
- Footprints
- PCBs
- Text documents
- PDF documents

1.3.4.1 Cross Cursors

AutoTRAX DEX can optionally display a cross cursor which is a horizontal line in the vertical line that meet at the position of the cursor and extend to the edges of the viewport. The cross cursor is only visible when adding and editing objects. An additional pair of diagonal lines can be added to the cross cursor.



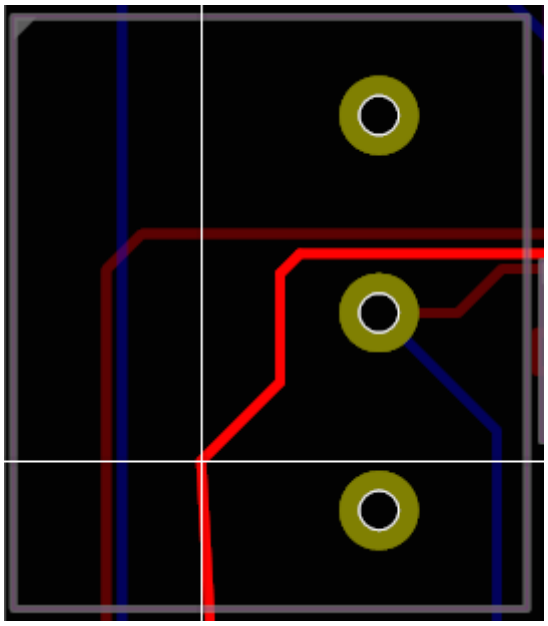
Cross cursor disabled



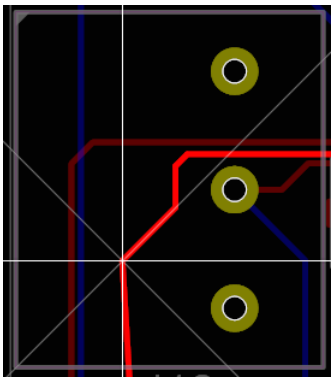
The cross cursor

1.3.4.2 The Diagonal Cursor

You can optionally display tool with dial lines in additional to the cross cursor when manually routing. When enabled you will see two lines at 45° emanating from the cursor position. This is useful when you are wanting to manually route tracks and keep them aligned horizontally, vertically or at 45°. You can turn the diagonal cursor on/off using the application settings in the settings panel.



Diagonal cursor disabled



Diagonal cursor enabled

1.3.4.3 Arranging Viewports

Viewports can be arranged in a tabbed layout or a tiled or cascaded layout.

Floating Viewport

Viewports can also be floated onto other screens or the current screen.

AutoTRAX DEX remembers the position of the viewports when the program ends and restores them when the program starts.

Floating viewports are only enabled when you have a tabbed layout.

To float a viewport double-click on the tab at the top of the viewport or hold down the left mouse button and drag the tab.

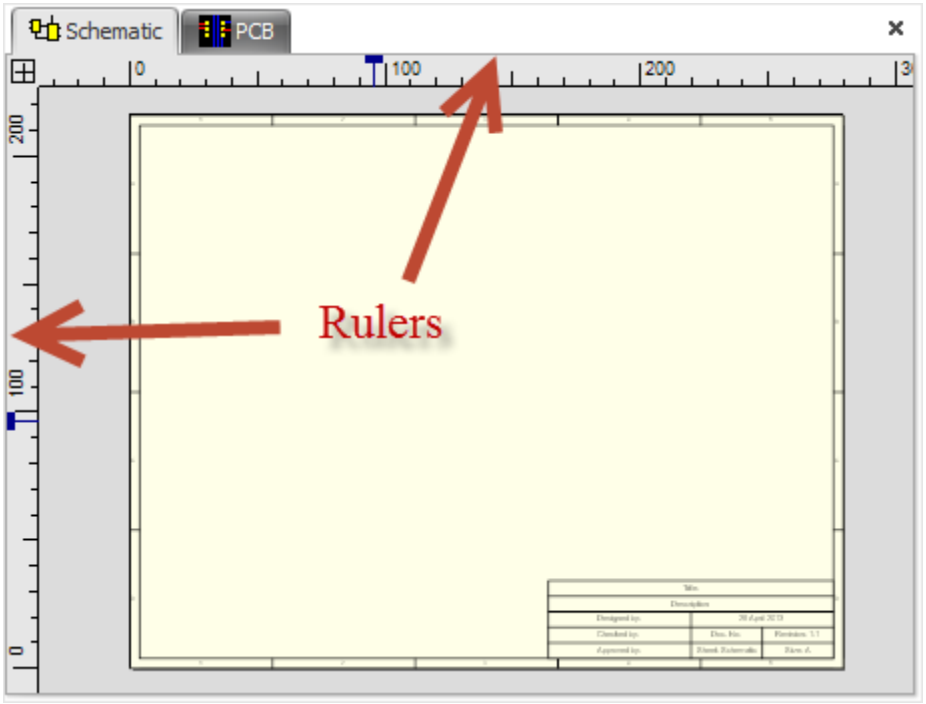
1.3.4.4 Rulers

You can optionally display rulers around graphical viewports.By default, schematic to not display a rulers while PCBs do. The rulers act as a measuring device changes when you change units. The rulers also have context menus which contain many useful commands.

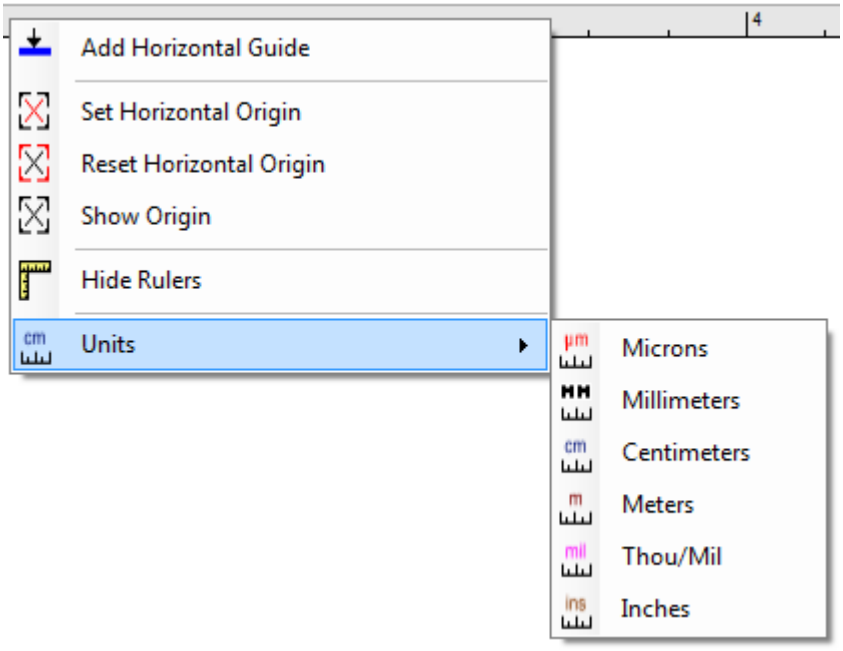
As you move your mouse on the viewport you will see both rulers display a line indicating the position of the mouse in the viewport.
At the junction of the two rulers, top left, you will see the origin box. The origin box has a similar context menu to the rulers.

Click the **View/Snap→View→** 

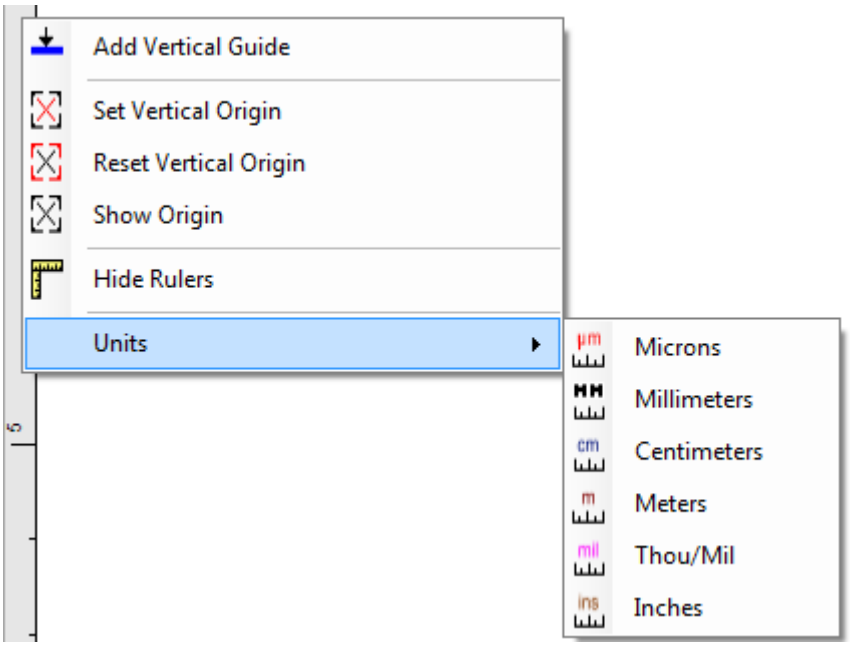
button to view/hide the rulers.



Both the top horizontal ruler and the vertical ruler on the left have context menus. To view the context menu right click on the ruler. The context menus for the horizontal ruler in the vertical ruler shown below. Click on one of the menu items to perform the indicator function.



The horizontal rulers context menu



The vertical rulers context menu

Adding Guides

You can add guides to the viewport by dragging from the origin box of the top/left rulers.

Dragging from the left ruler adds a vertical guide to the viewport.

Dragging from the top of ruler adds a vertical guide to the viewport.

Dragging from the origin box adds a point to guide to the viewport.

Setting the Origin

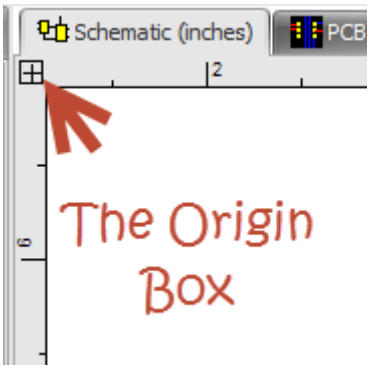
You can set the horizontal or vertical origin or both by selecting the appropriate command in the context menu. Then drag the mouse to where you want the origin to appear and left click to set the new origin. The rulers will update to show the new origin and if you are the grid displayed it will be updated.

Setting The Units

Click on any of the unit menu items to change the units for the current sheet. The rulers will update to show your changes.

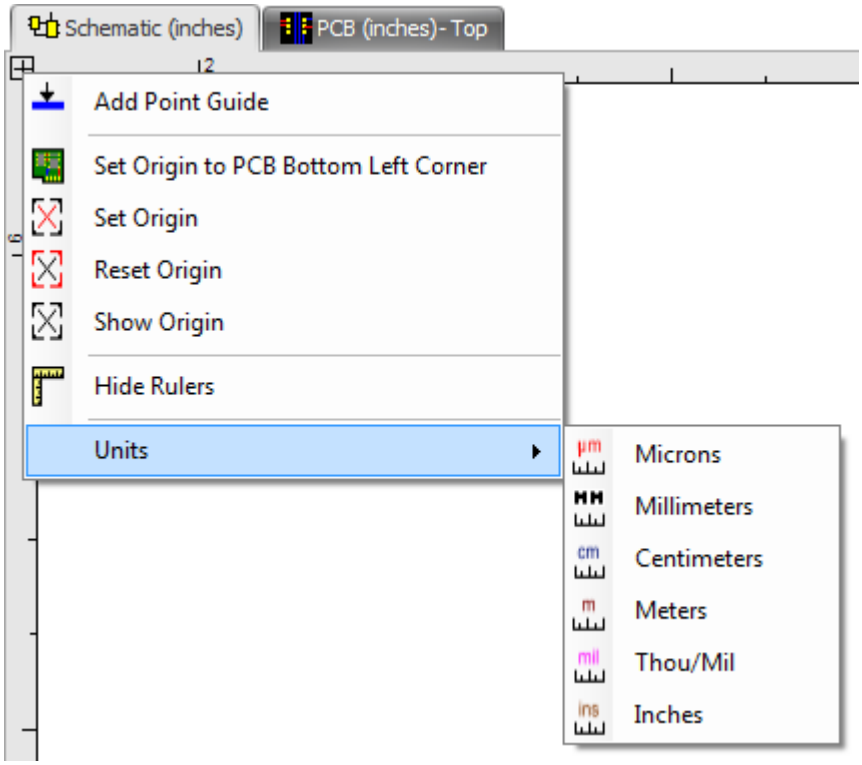
1.3.4.5 The Origin Box

The origin box is located at the top left of the viewport where the two rulers meet as shown below.



The origin box

If you right click the mouse over the origin box, the context menu shown below will appear. The context menu is very similar to the context menu you see with the rulers and contains many useful commands. Click on any of the commands to run them.



The origin boxes context menu

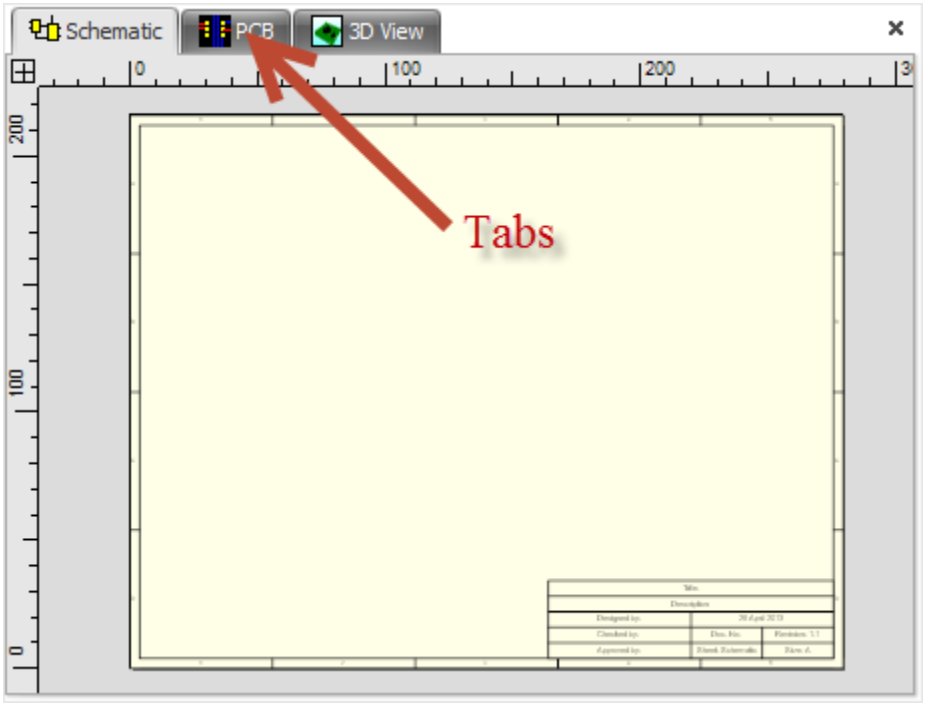
See the rulers topic to find out how to add a point guide, set the origin or change the units.

1.3.4.6 Tabbed Viewports

You can arrange the viewports in a tabbed control.

Click the Panels→Window→ 

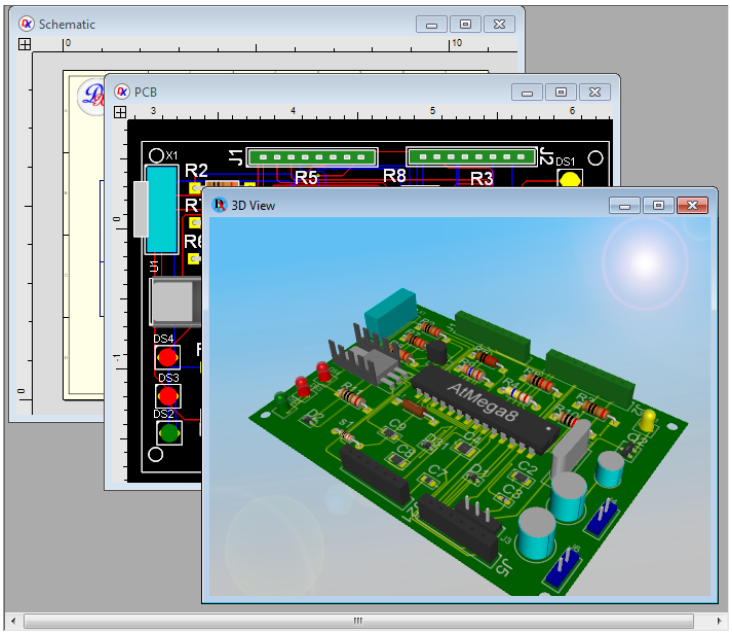
button to arrange the viewports in a tabbed control.



Tabbed Viewports

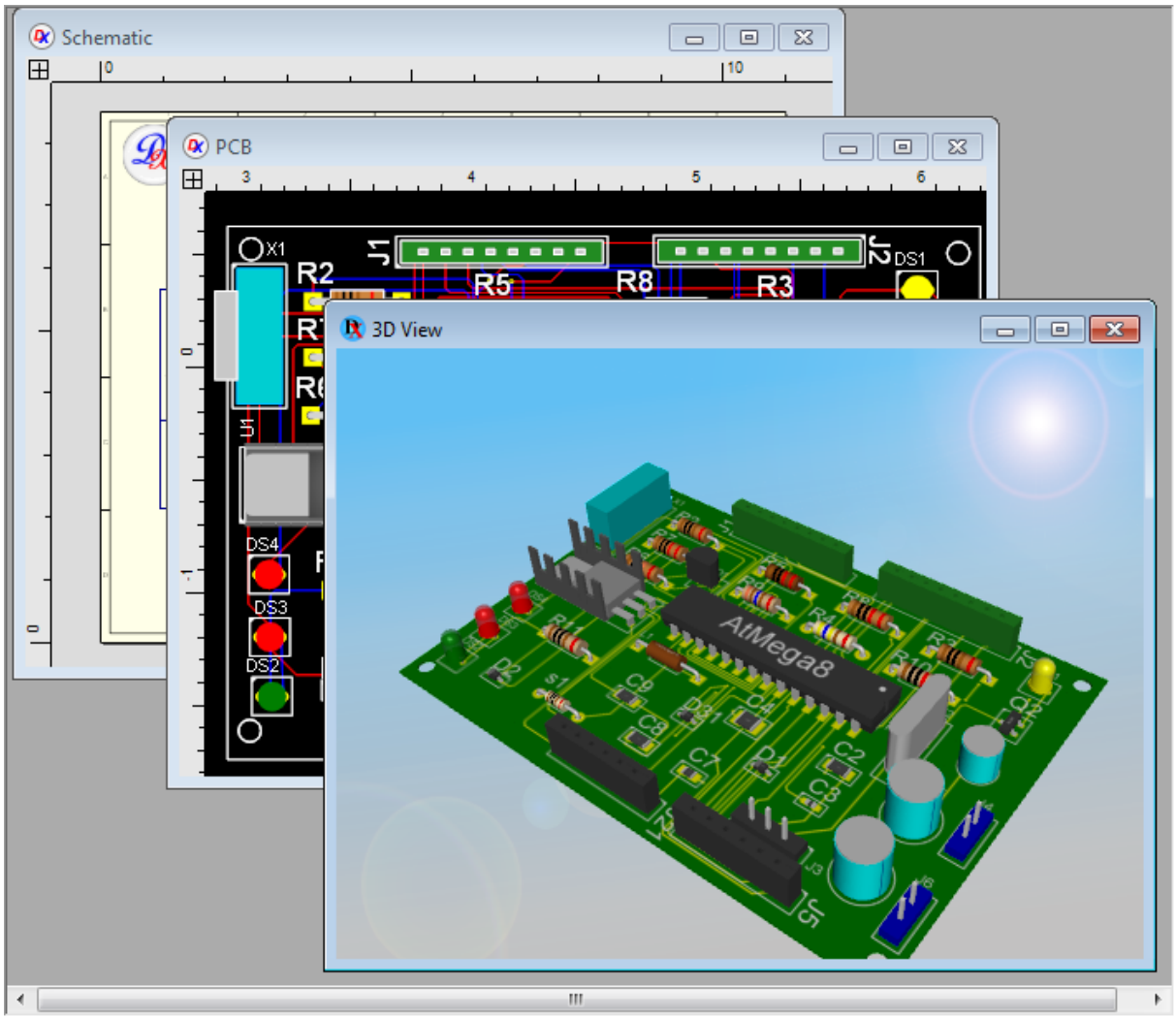
1.3.4.7 Tiled and Cascaded Viewports

Click the Panel→Windows→ Cascade button to arrange the viewports cascaded.



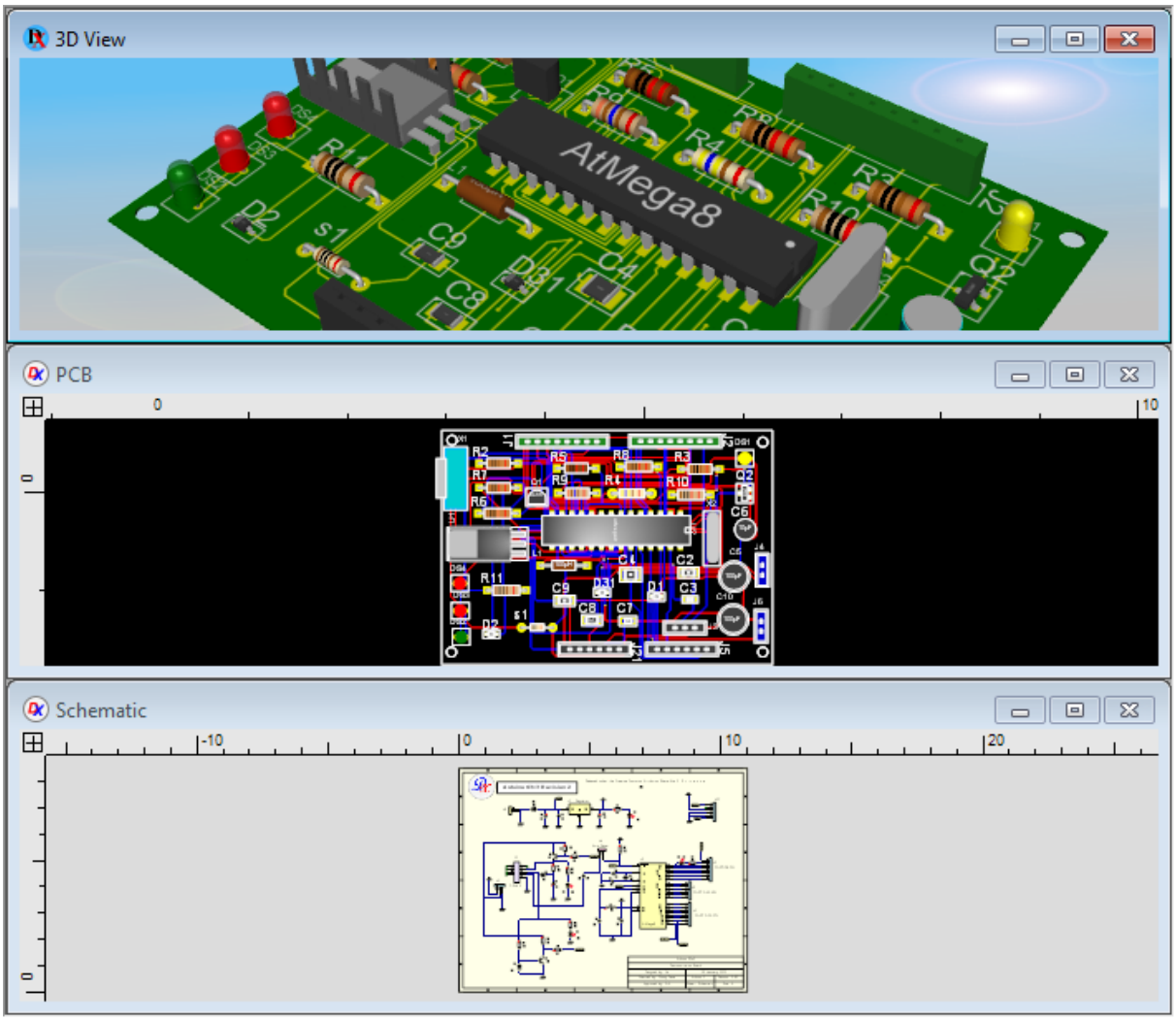
Cascaded Viewports

Click the Panel→Windows→ Cascade button to arrange the viewports cascaded.



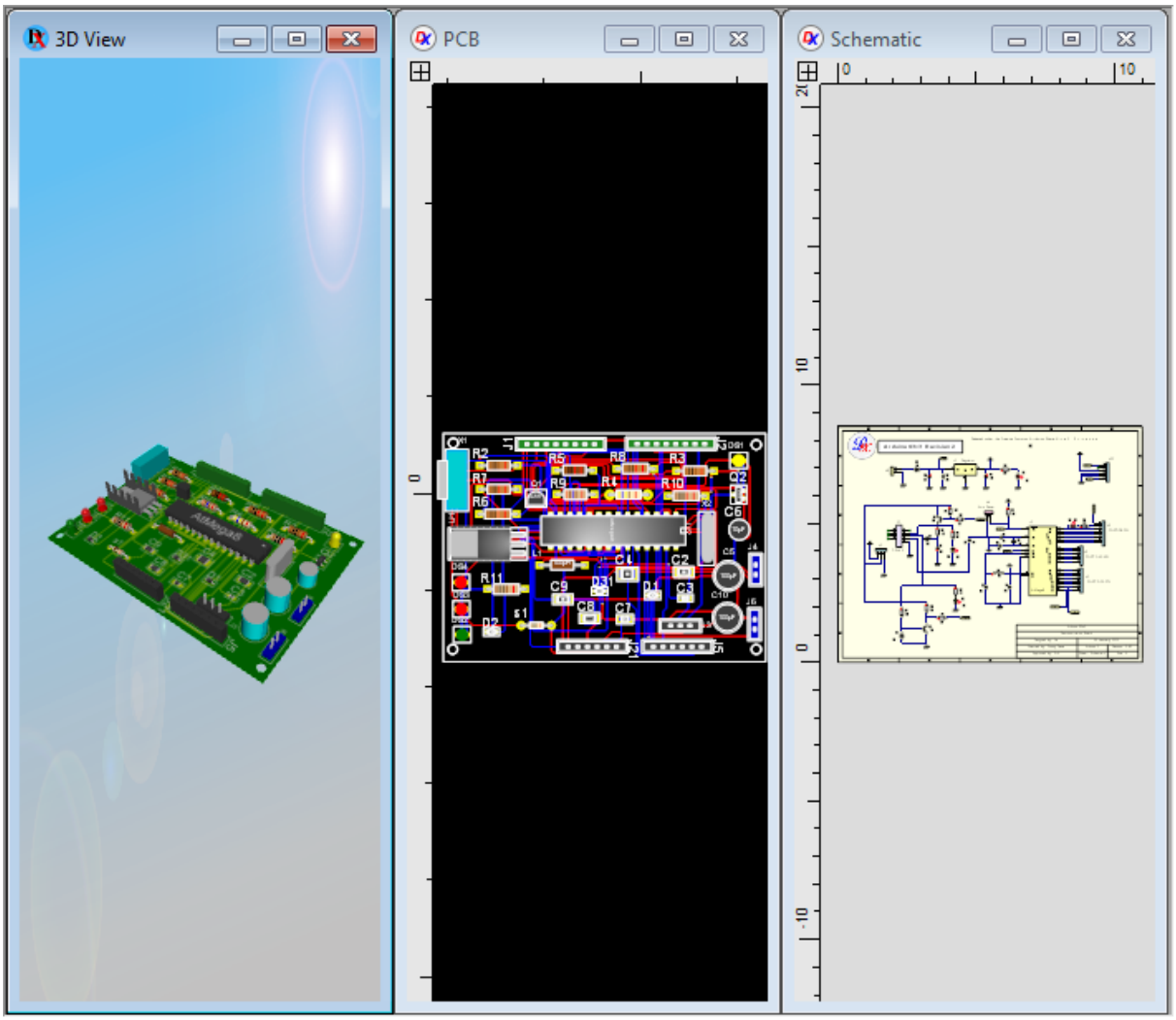
Cascaded Viewports

Click the Panel→Window→ Horizontal button to tile the viewports horizontally.



Horizontally Tiled Viewports

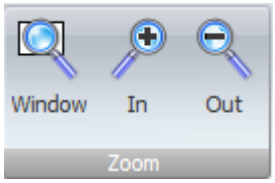
Click the Panel→Windows→ Vertical button to tile the viewports vertically.



Vertically Tiled Viewports

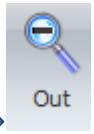
1.3.4.8 Zooming In and Out

Buttons to zoom and pan the viewport can be found in the View/Snap tab.



Click the View/Snap→Zoom→ 

button to zoom in or press the **PgUp** key.

Click the View/Snap→Zoom→ 

button to zoom out or press the **PgDn** key.

Click the View/Snap→Zoom→ 

button to zoom to a window. After clicking, move the mouse to a corner of the area to wish to zoom in on. Next, hold down the left mouse button and drag the cursor to define the area, releasing the left mouse button adjusts the view.

Zooming with the mouse

You can also use the **mouse scroll wheel.** Scroll it and the viewpoint will zoom in/out depending on the direction that you rotate the wheel. You can often use the mouse wheel and middle mouse button for panning and zooming while doing other things such as adding and editing objects.

Panning with the mouse

If you drag the mouse while holding down the **mouse scroll wheel** the view will pan.

Panning with the keyboard

You can pan the viewport using the 4 arrow keys on the numeric keyboard to pan.

Home

Press the Home key to view the entire sheet or all the contents of your sheet if the sheet border is not visible.

1.3.4.9 Smart Panning

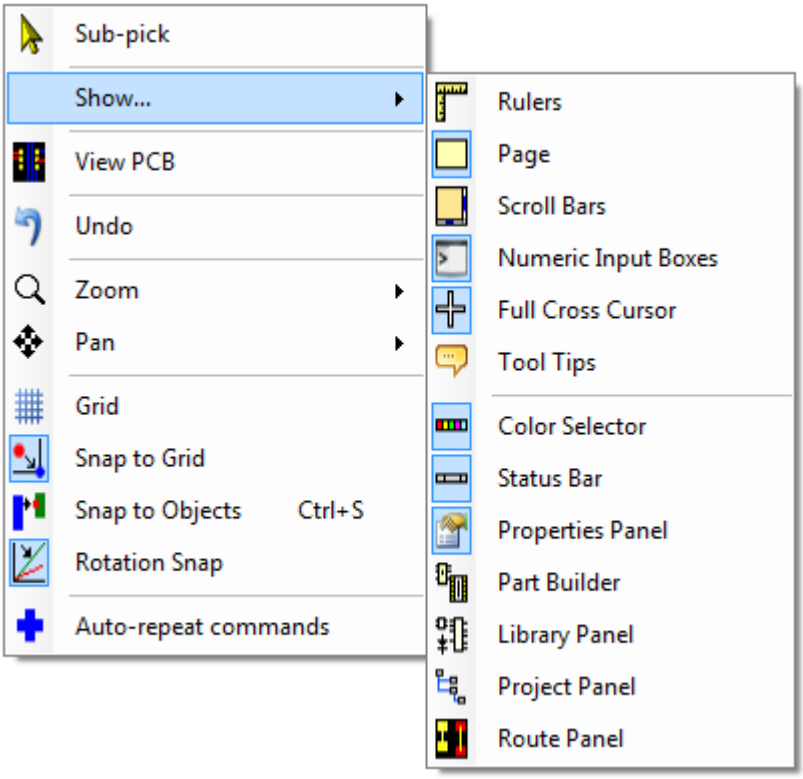
If smart panning is enabled, as you move selected items in a viewport, the viewport will automatically pan to always show the items. It remembers the original position and contents of the viewport before panning and will try and restore that view if at all possible.

You can always pan by using one of the direction arrows. You can also pan by holding down the middle mouse button and dragging the mouse. This will work when you're doing most things. Additionally, if you rotate the middle mouse wheel the view will zoom in/out. By rotating the middle mouse button and holding the middle mouse button down while dragging the mouse you can quickly change the view.

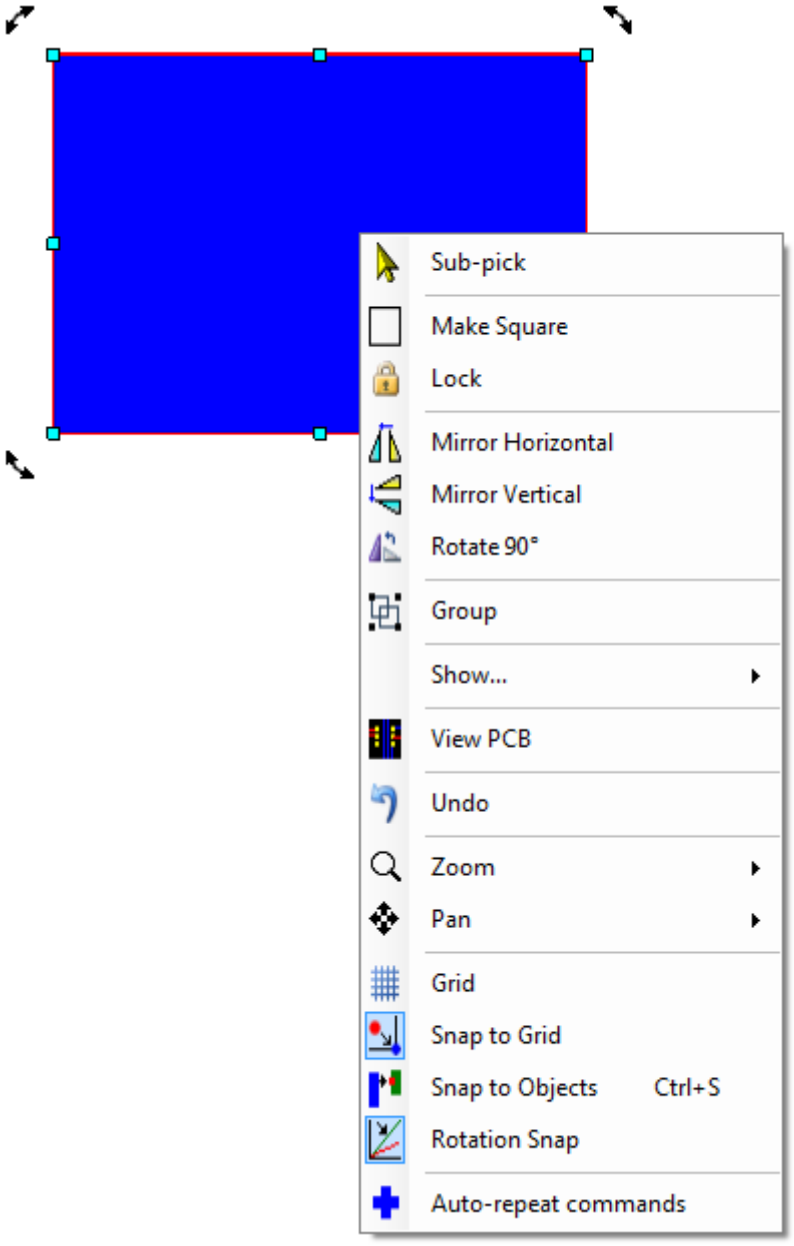
You can turn smart panning on/off by click the  button in the status bar.

1.3.4.10 The Viewport Context Menu

All viewports have a context menu. To view the context menu press the right mouse button. A context menu similar to the one shown below will appear. The items in the context menu depend on whether an object is selected. If an object is selected then the context menu will contain items suitable for editing the selected object.



The schematic context menu with nothing selected



The context menu for a rectangle

1.3.4.11 Switch Between Schematic and PCB Views

Press the TAB key to switch between schematic and PCB views. This only works in picking mode.

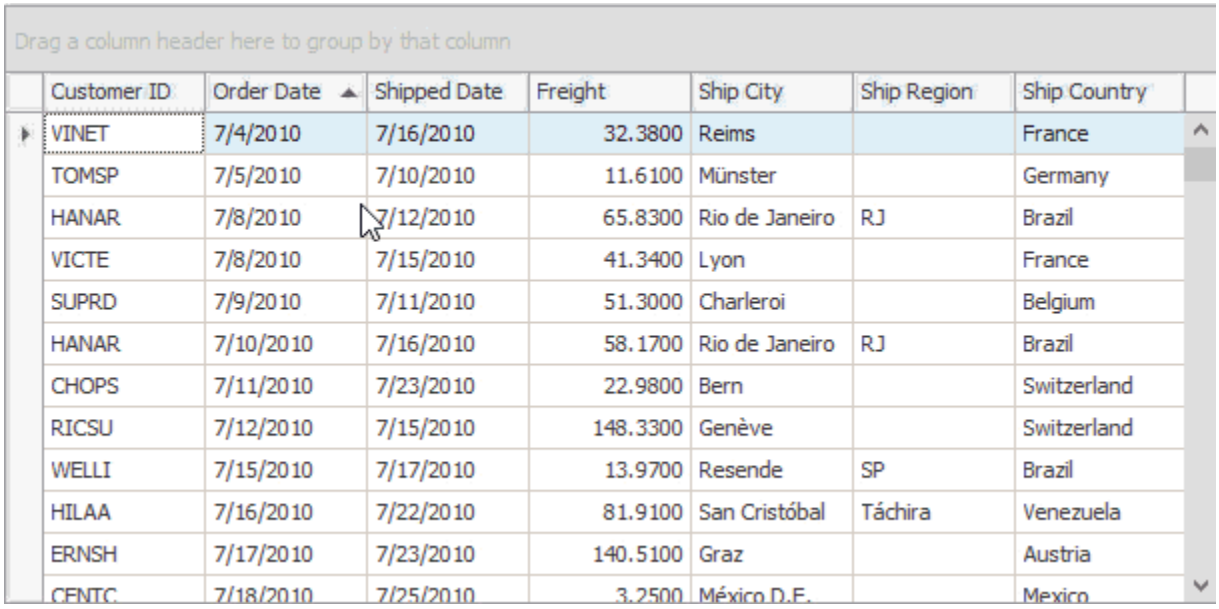
1.3.5 Dialog/Control Grids Views

Grid views provides rich capabilities for displaying, shaping and editing data from any data source.

Data shaping capabilities include, but are not limited to, sorting, grouping, summary calculation, cell merging, data editing, master-detail and split presentations, as well as a rich set of filtering and data searching options such as built-in column filters and Find Panel.

1.3.5.1 Grouping

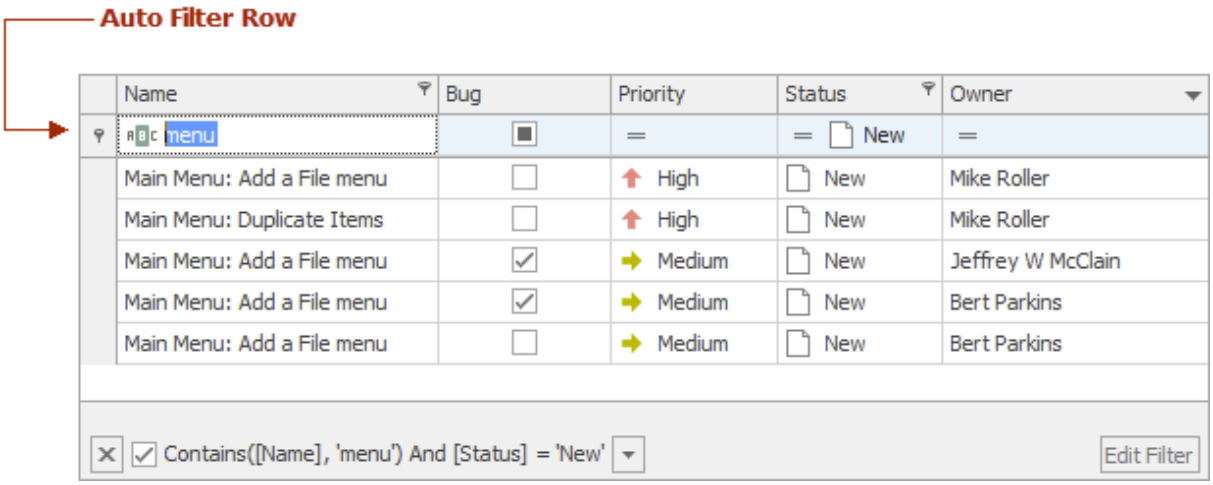
Data grouping is enabled in the Data Grid by default. To group data by a column, drag a column header into the group panel. Another option is to right-click a column header and select "*Group By This Column*".



By default, when you group data by columns, these columns automatically hide from the View, and all groups collapse.

1.3.5.2 Auto Filter Row

The **automatic filtering row** allows data to be filtered on the fly by typing text into that row. When you type text into this filtering row, a filter condition is automatically created based on the entered value and then applied to the focused column. The automatic filtering row is displayed at the top of the Grid View.

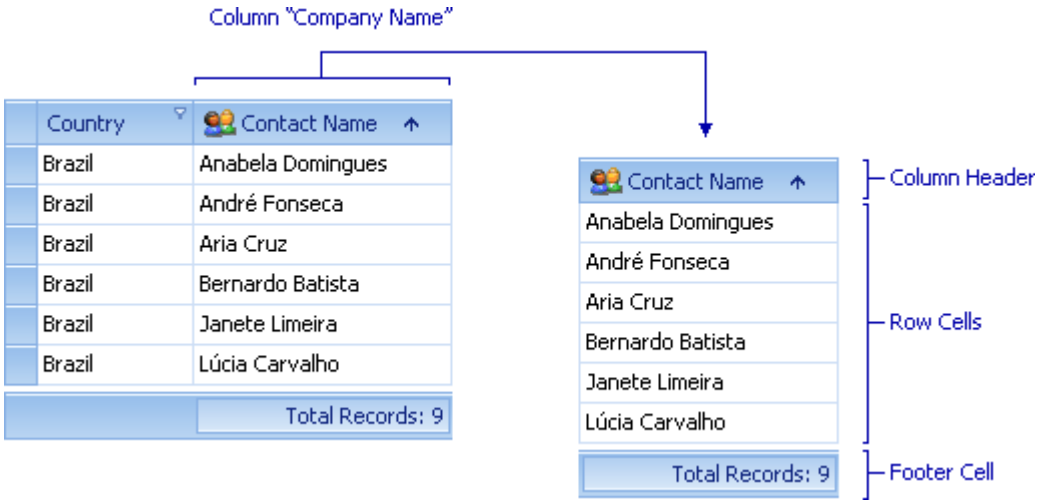


When the top data row is focused, you can move focus to the auto filter row by pressing the CTRL + UP ARROW keys.

1.3.5.3 Columns

Columns represent data source fields. A column consists of the following elements:

- a Column Header that displays the column caption and enables you to drag and resize the column, sort and filter column values, etc
- row cells displaying field values and editing facilities
- a footer cell that displays a total summary result

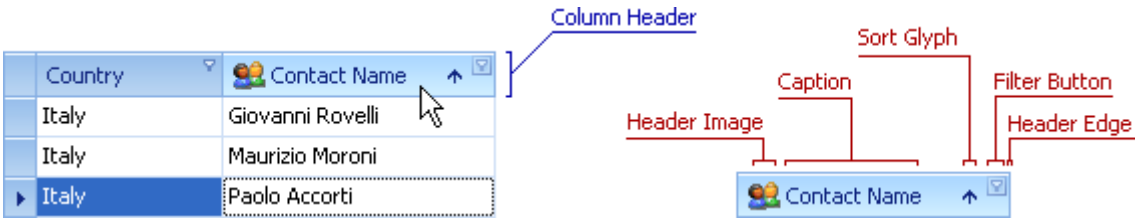


1.3.5.4 Column Header

Column headers identify columns in Grid Views. A column header contains:

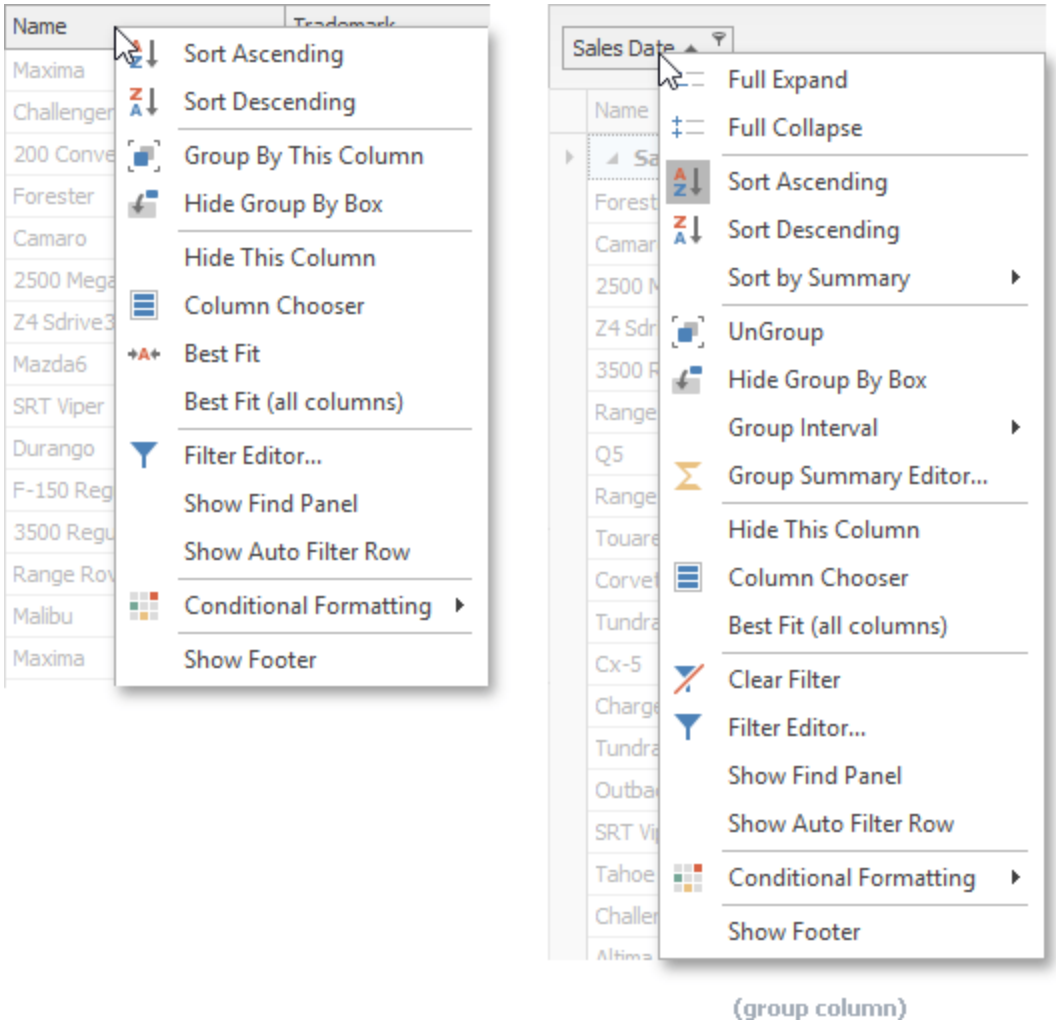
- a caption string identifying column content;
- a column header image providing graphical information about column content;
- a sort glyph identifying the data sort order (if any) applied to column values;
- a filter button enabling you to filter column values.

The column header image contains the headers of all visible columns. Headers of grouped columns are displayed in the group panel . The Customization Form displays hidden columns' headers. Right-clicking a column header in the header panel or group panel activates the column header context menu.



1.3.5.5 Column Header Context Menu

The **column header context menu** enables you to manipulate a column (apply sorting, grouping, calculate the best column width, etc.). The menu is activated by right-clicking a Column Header displayed in a Column Header Panel or in a Group Panel.



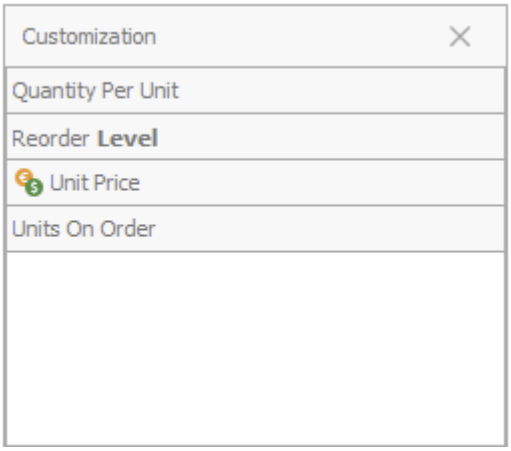
1.3.5.6 Column Header Panel

The column header panel displays headers of visible columns.

1.3.5.7 Customization Form

The **Customization Form** allows you to customize the layout of columns (and bands in Banded Grid Views). To invoke this form, select the *Column Chooser* (or the *Column/Band Chooser*) command from the Column Header Context Menu.

Default Customization Form

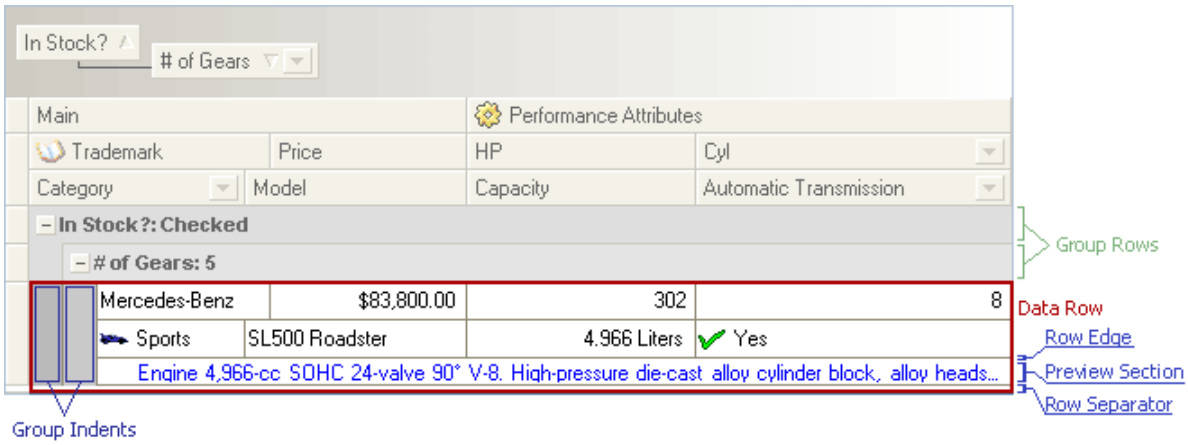


- The form displays the headers of hidden columns and bands.
- Users can show and hide columns/bands using drag-and-drop operations between the View and the Customization Form.
- An optional search box allows users to locate columns/bands by their captions.

1.3.5.8 Data Row

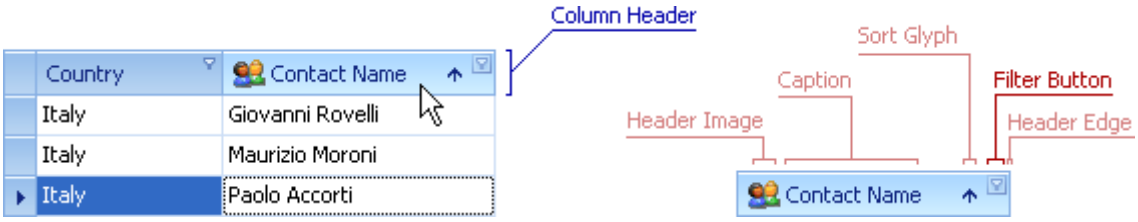
A data row represents an individual data source record. Rows contain row cells used for displaying and editing values of particular data fields. In Grids Views , row cells are arranged in a single line.

A row edge is a single line displayed below row cells. You can drag the row edge in order to change row height.



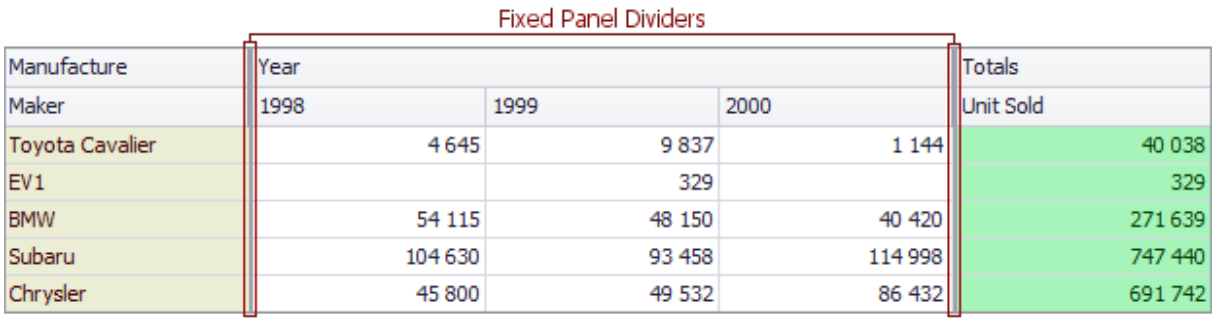
1.3.5.9 Filter Button

Filter buttons are displayed within column headers and can be clicked to activate filter dropdown lists. Such lists enable you to specify data filtering conditions.



1.3.5.10 Fixed Panel Divider

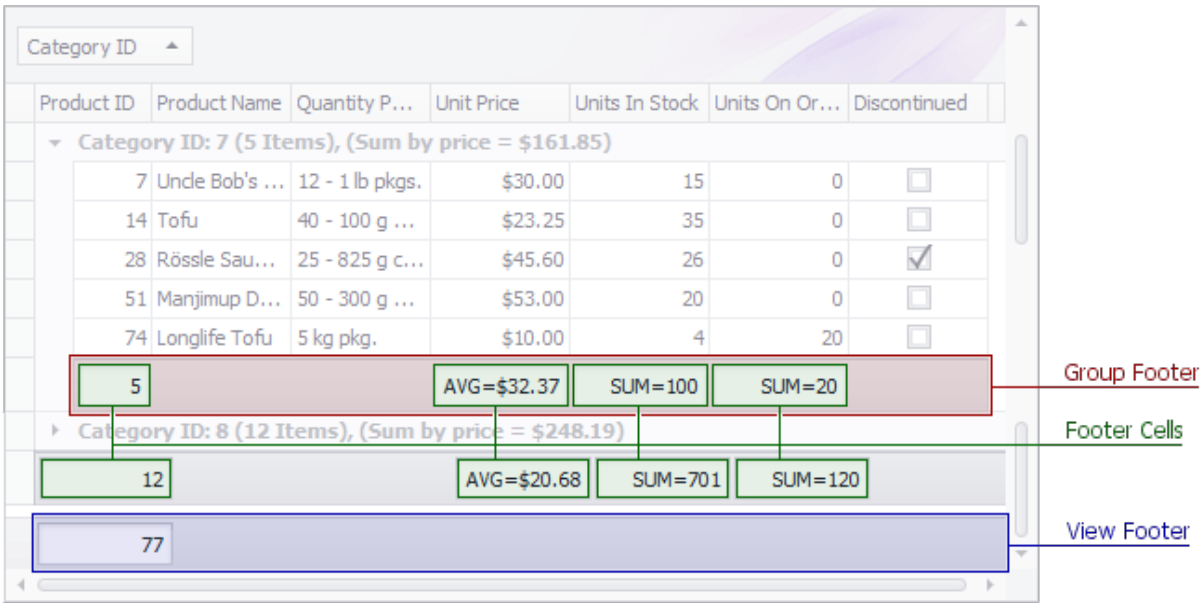
The **fixed panel divider** separates fixed columns or bands from a View's scrollable area. See Columns.



Manufacture	Year			Totals
Maker	1998	1999	2000	Unit Sold
Toyota Cavalier	4 645	9 837	1 144	40 038
EV1		329		329
BMW	54 115	48 150	40 420	271 639
Subaru	104 630	93 458	114 998	747 440
Chrysler	45 800	49 532	86 432	691 742

1.3.5.11 Footer Cell

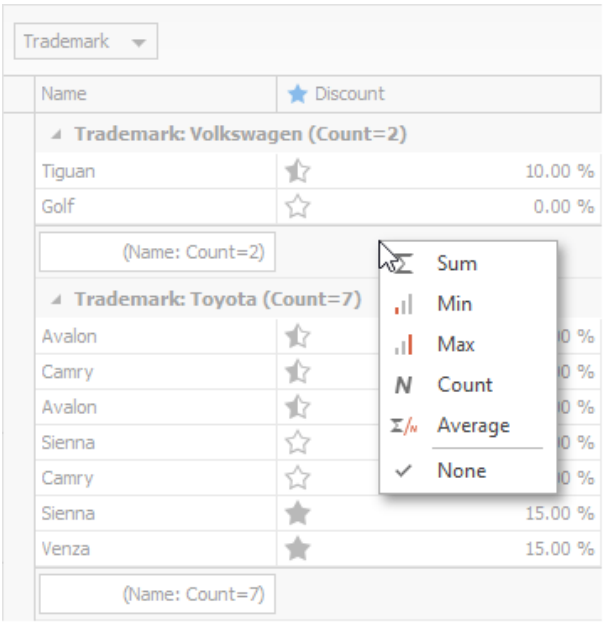
Footer cells are elements of the View Footer and Group Footer panels, and they display summary values. View footer cells present total summaries ,while group footer cells present group summaries. Right-clicking a footer cell activates the Footer Context Menu that enables you to set or change the summary type.

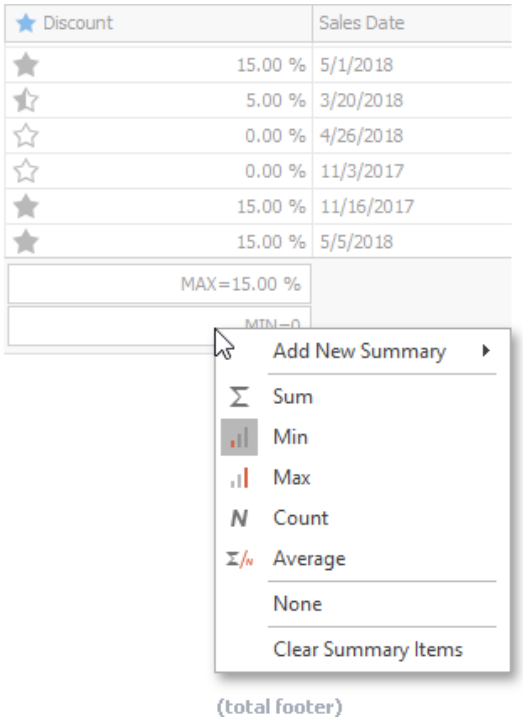


1.3.5.12 Footer Context Menu

Footer context menus enable you to specify or change the type of summary calculated for a column. The menu can be invoked by clicking a Footer Cell in the

View Footer or Group Footer panels. Only items suitable for the column are enabled in the menu.



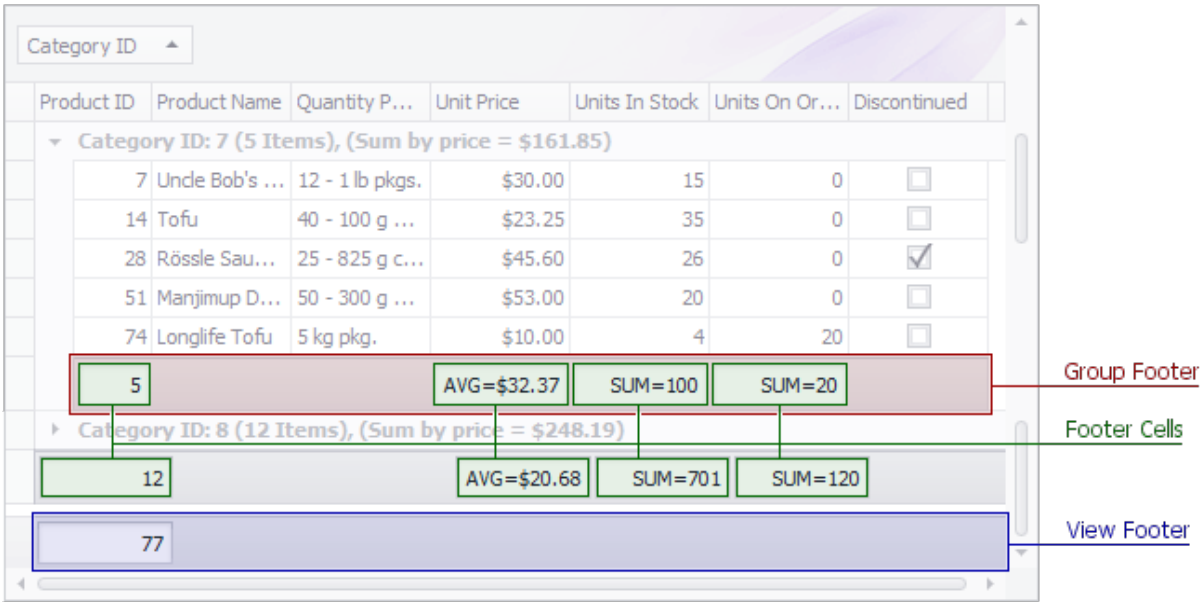


1.3.5.13 Group Expand Button

Group expand buttons enable you to expand or collapse group rows.

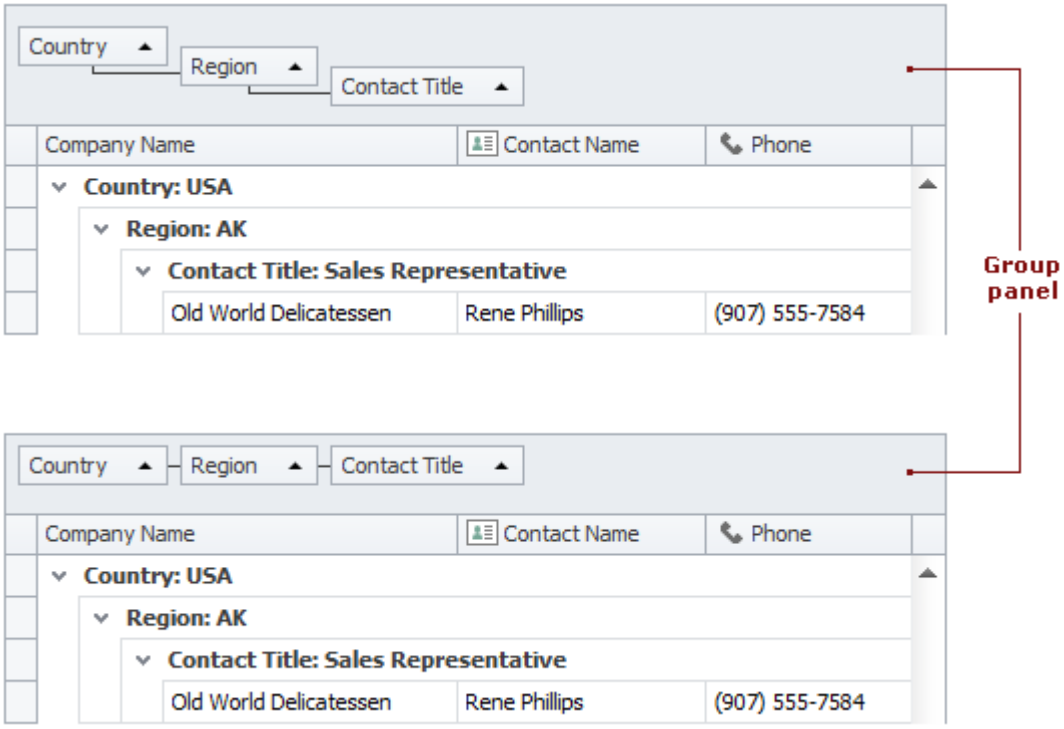
1.3.5.14 Group Footer

Group footers display group summaries, i.e. summaries calculated for data rows belonging to the group. Group footers contain footer cells, each corresponding to a column. Footer cells display formatted summary values. These cells can be clicked to invoke the Footer Context Menu.



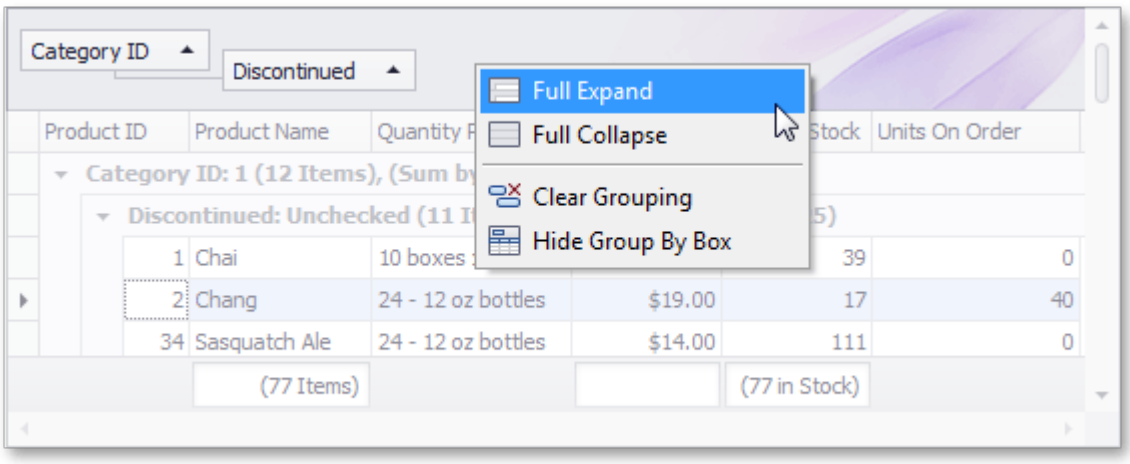
1.3.5.15 Group Panel

Group panels display headers of columns involved in data grouping. They also serve as drag-and-drop targets allowing you to drag column headers onto it. Right-clicking the group panel invokes the group panel context menu, which provides grouping related options.



1.3.5.16 Group Panel Context Menu

The group panel context menu can be invoked by right-clicking the group panel. It provides grouping related options.



1.3.5.17 Group Row

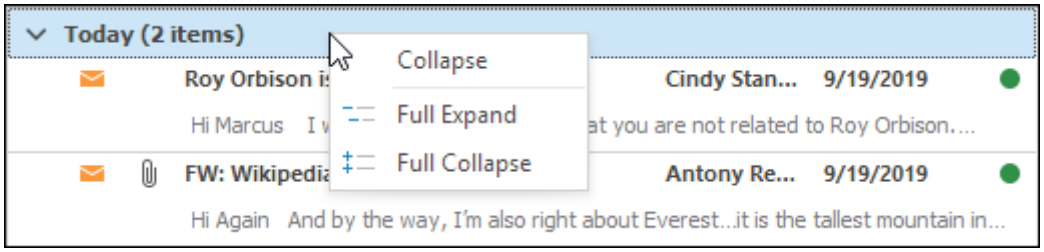
Group rows are used to organize data rows into a tree when data grouping is applied. A group row contains a Group Expand Button that enables you to expand and collapse a group row, and thus show or hide its child rows. Group rows can also display group summary values.

1.3.5.18 Group Row Check Box Selector

The **Group Row Check Box Selector** allows you to select/deselect all grid rows of a group simultaneously. This visual element is supported if the Multiple Row Selection feature is enabled.

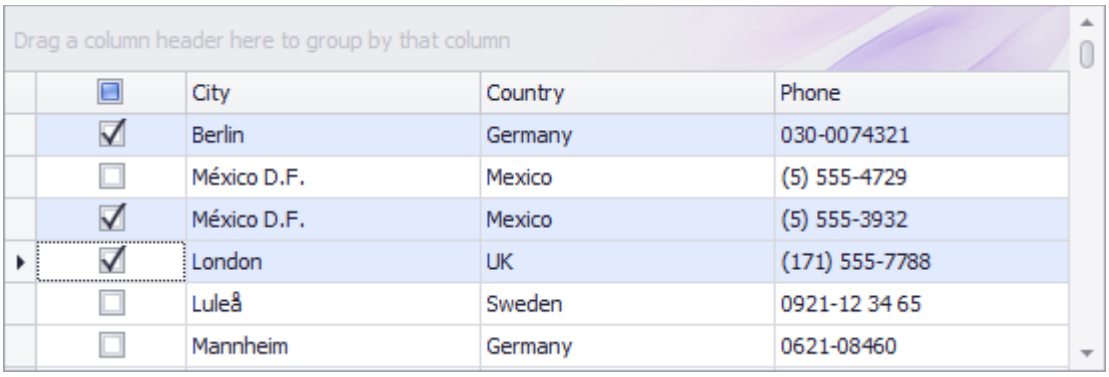
1.3.5.19 Group Row Context Menu

Users can right-click a Group Row to invoke a context menu that expands / collapses the group.



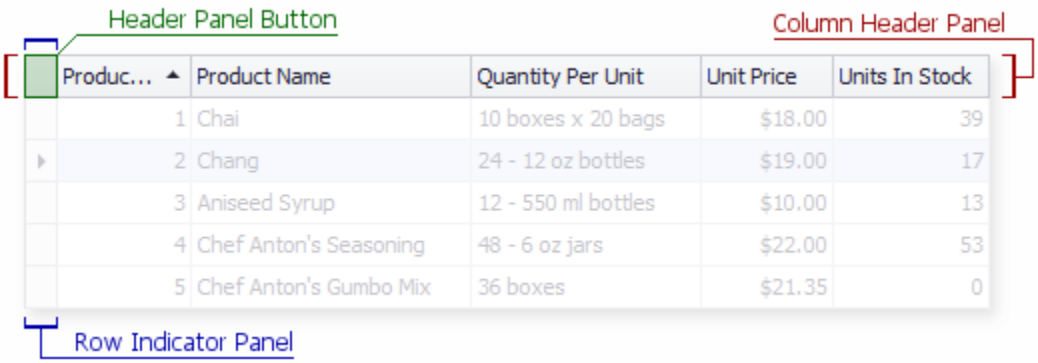
1.3.5.20 Multiple Row Selection

The Grids Views support multiple row selection via a built-in Check column. When this feature is enabled you can use checkboxes to toggle the selection state of certain rows, all rows or data group rows.



1.3.5.21 Header Panel Button

The **header panel button** belongs to both the column header and Row Indicator Panel elements. By default, the header panel button has no special functionality. When a View is used to represent detail data, the header panel button serves as a zoom button.



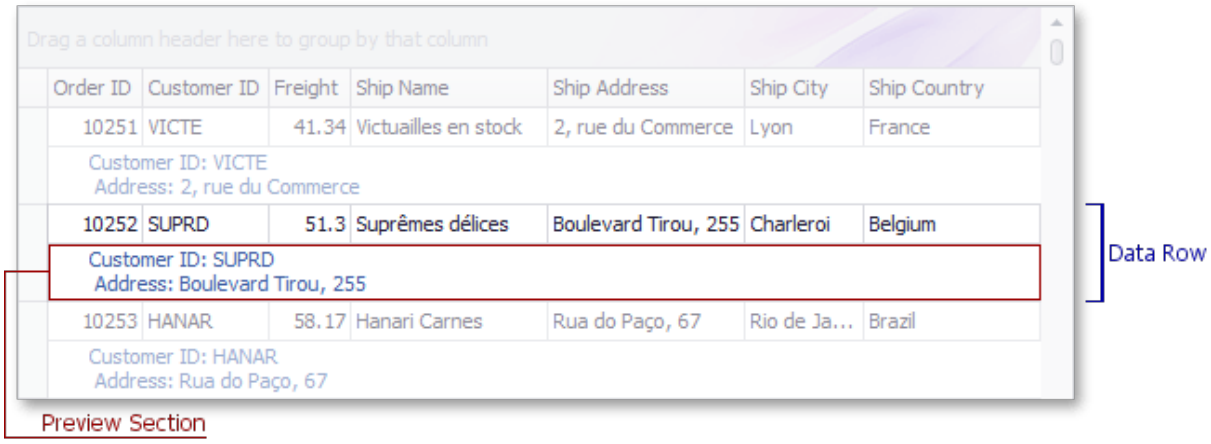
1.3.5.22 Zoom Button

The **zoom button** enables you to maximize the detail View, so that it occupies an entire grid control's region. For Grid Views and their descendants, a Zoom button represents an intersection of a detail View's Column Header and Row Indicator Panel.

Clicking the zoom button maximizes the detail View while hiding all the other Views including the master. The button's image now displays an **'X'** sign to indicate the View is maximized.

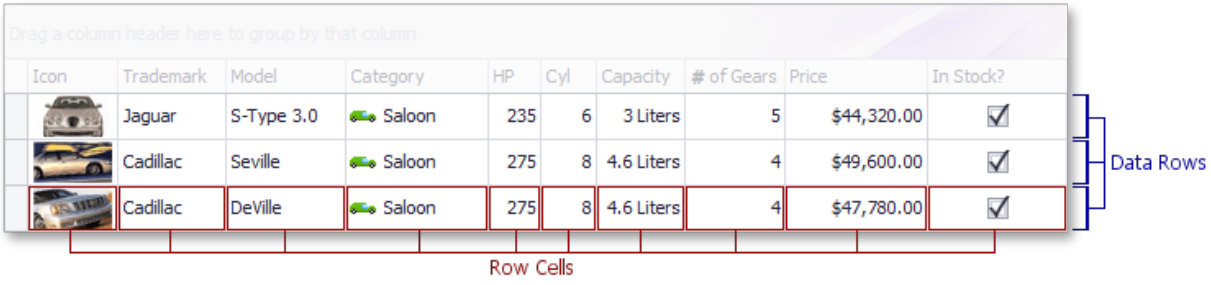
1.3.5.23 Preview Section

Preview sections are non-editable regions in data rows that allow large memo fields to be displayed across all View columns, thus the preview section's width is equal to the total column width.



1.3.5.24 Row Cell

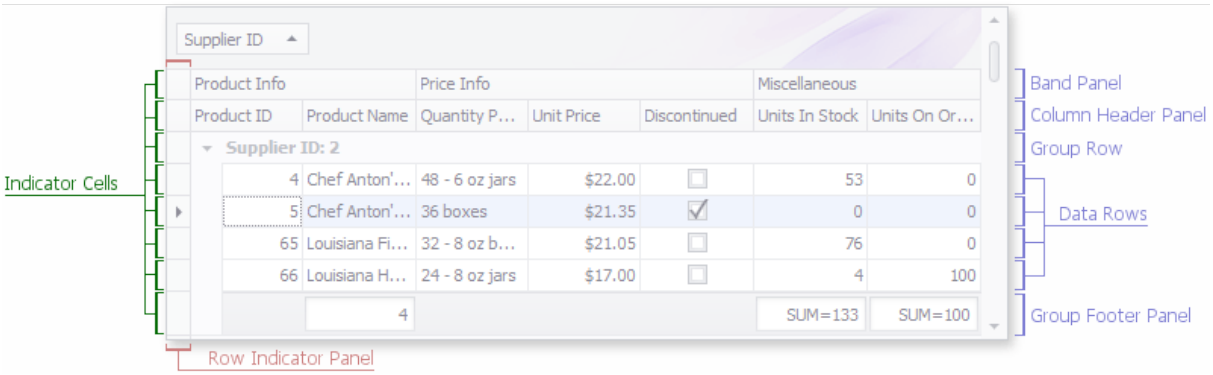
Row cells display field values within data rows. Each cell provides an in-place editor that can be invoked by you to edit a cell's value.



1.3.5.25 Row Indicator Panel

The **row indicator panel** represents a region displayed at the left edge of the View. The panel contains row indicator cells corresponding to different View sections (the

band header panel, the column header panel, data rows, the View footer, etc). The cell relating to the focused row indicates the row state (whether it is in edit mode or has been modified, etc.). When multiple selection is enabled, you can click cells and then drag the mouse cursor to another cell to select a range of rows.

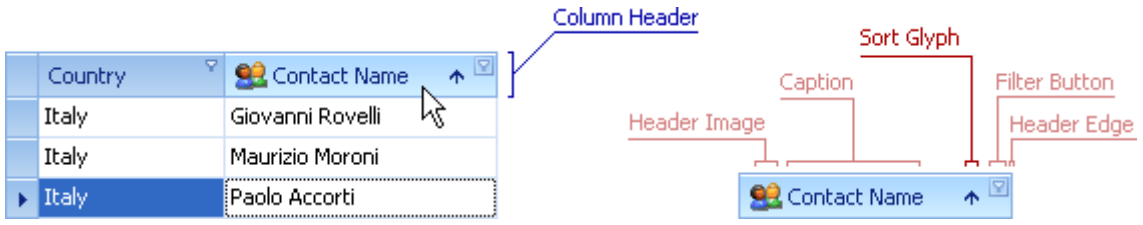


1.3.5.26 Row Separator

Represents a region that separates adjacent data rows.

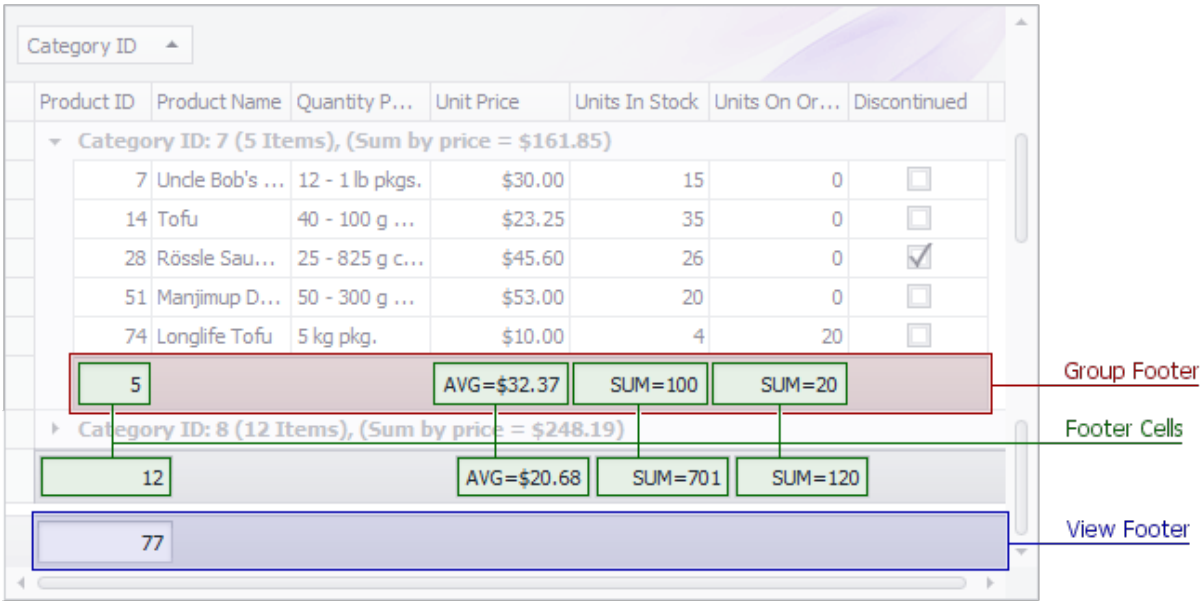
1.3.5.27 Sort Glyph

Sort glyphs are displayed within column headers when sorting is applied by column values. These glyphs indicate the applied sort order.



1.3.5.28 View Footer

The **view footer** is designed to display total sumaries, i.e. summaries calculated for all data rows in a View. The footer contains Footer Cells displaying summary values for particular columns. Clicking a footer cell invokes the Footer Context Menu.



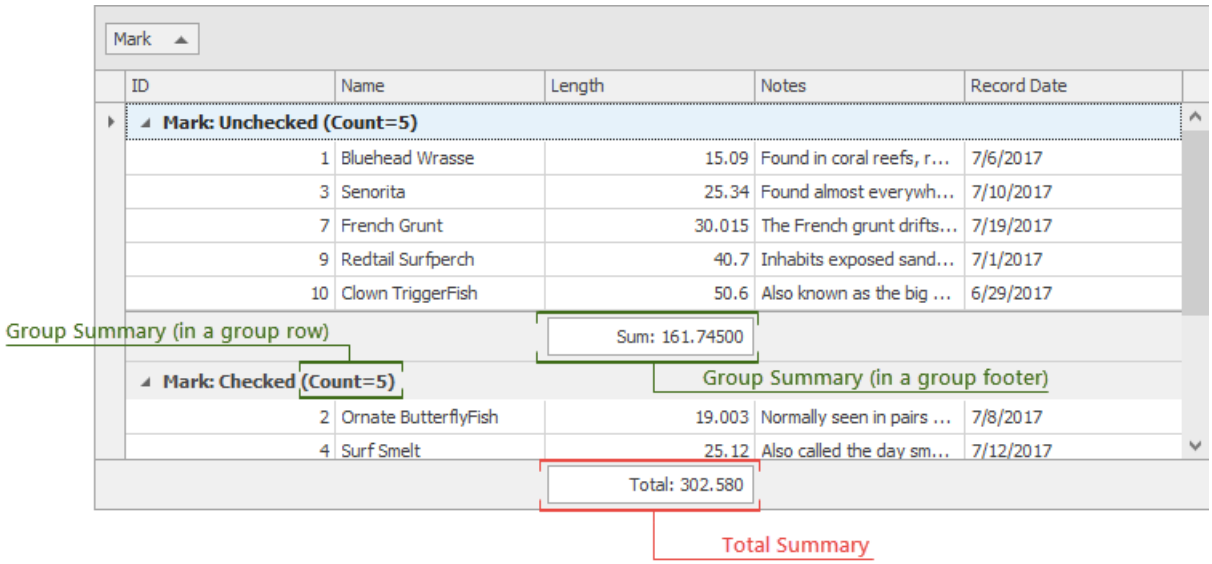
1.3.5.29 Summaries

Summaries are grid items that utilize aggregate functions to display summary information about displayed data: total record count, minimum values, etc.

Summary Types

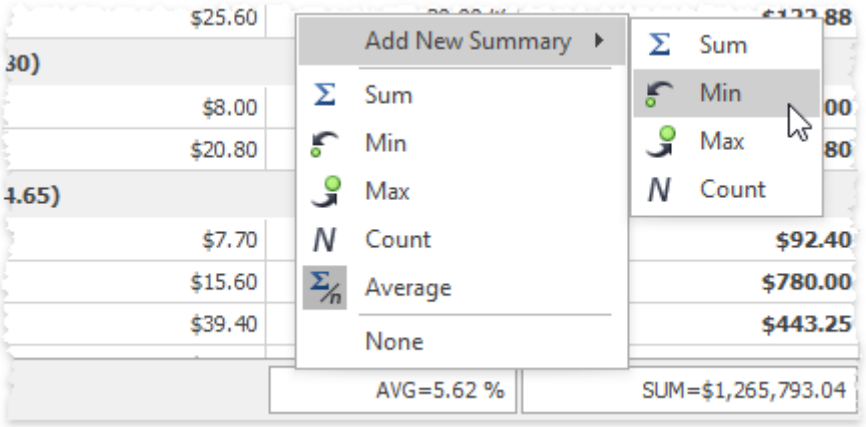
Total summaries Grid summary items are calculated over all Data Grid records and displayed within a view footer.

Group summaries Grid summary items are calculated over individual groups and displayed either in a group footer area or within group rows.



Total Summaries

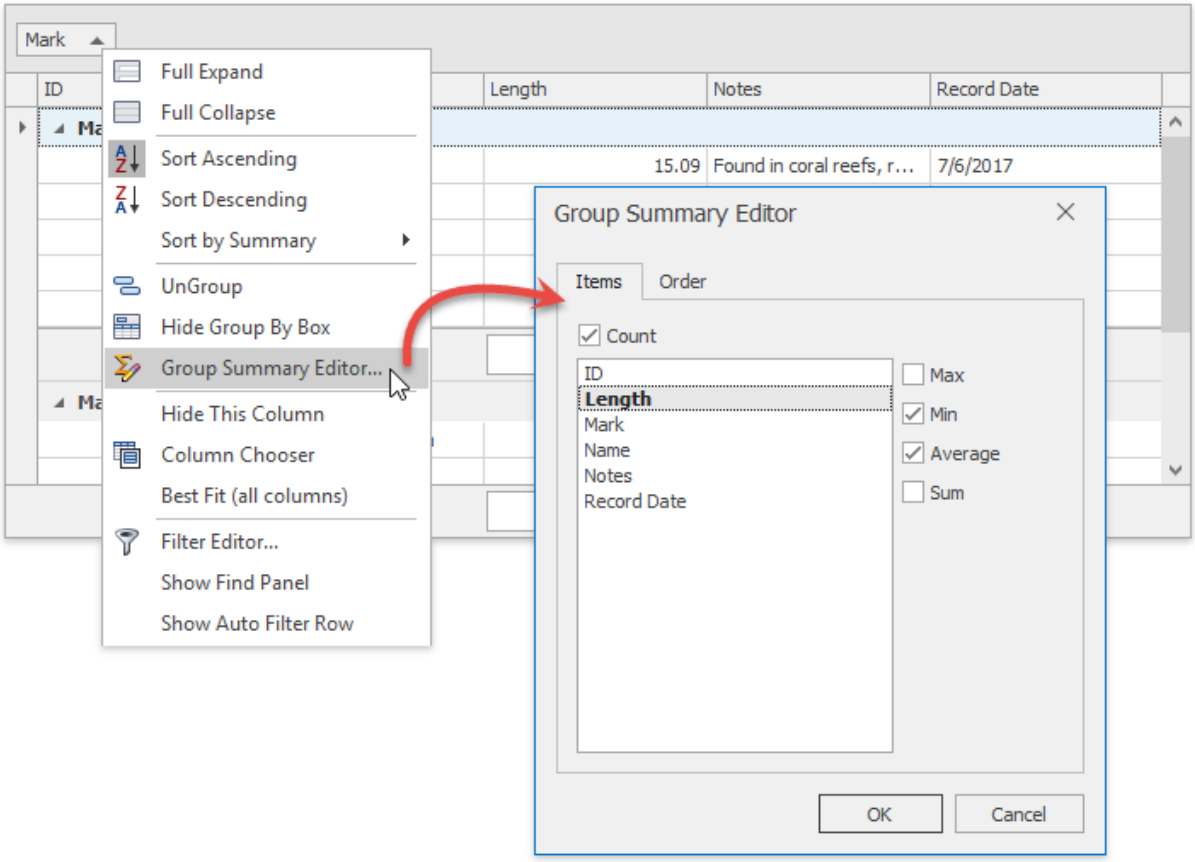
When the view footer is visible, you can right-click it and use a context menu to add and remove summary items. It is also possible to right-click existing summaries to change their aggregate functions.



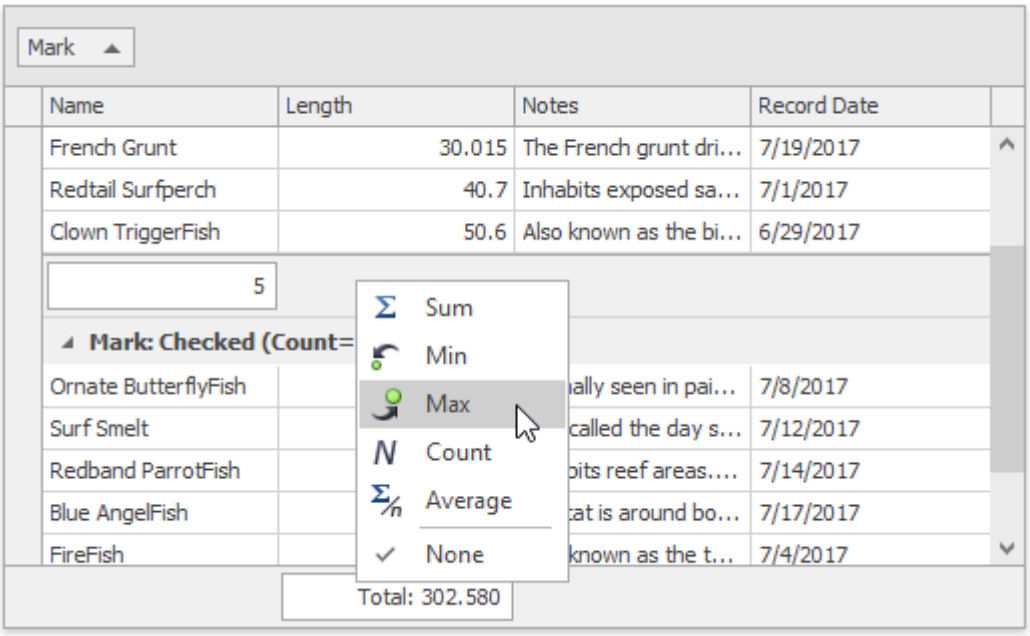
Group Summaries

You can manually add and modify group summaries in two cases:

- You can right-click a grouped column header to invoke a Group Summary Editor dialog.

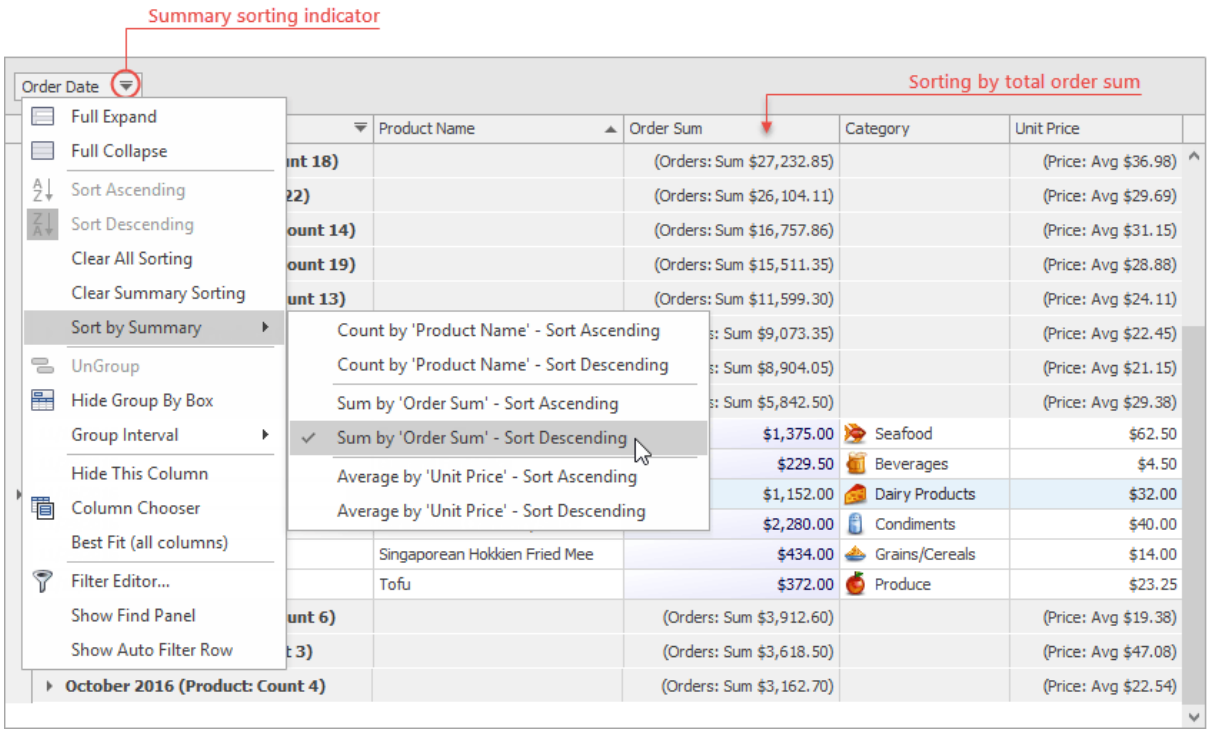


- when the group footer is visible you can right-click this footer below the required column and select a summary item type.



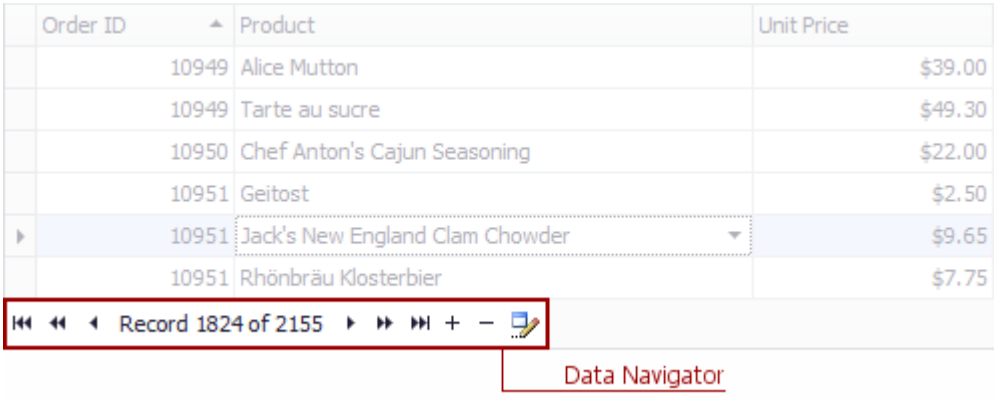
Sort Groups by Summary Values

When the Data Grid displays group summaries, you can right-click a column header in a group panel and select "*Sort by Summary*". By doing so, you can sort groups by any group summary item, associated with any Data Grid column. When sorting by summaries is active, a dash is painted above the regular ascending/descending sort indicator in column headers.



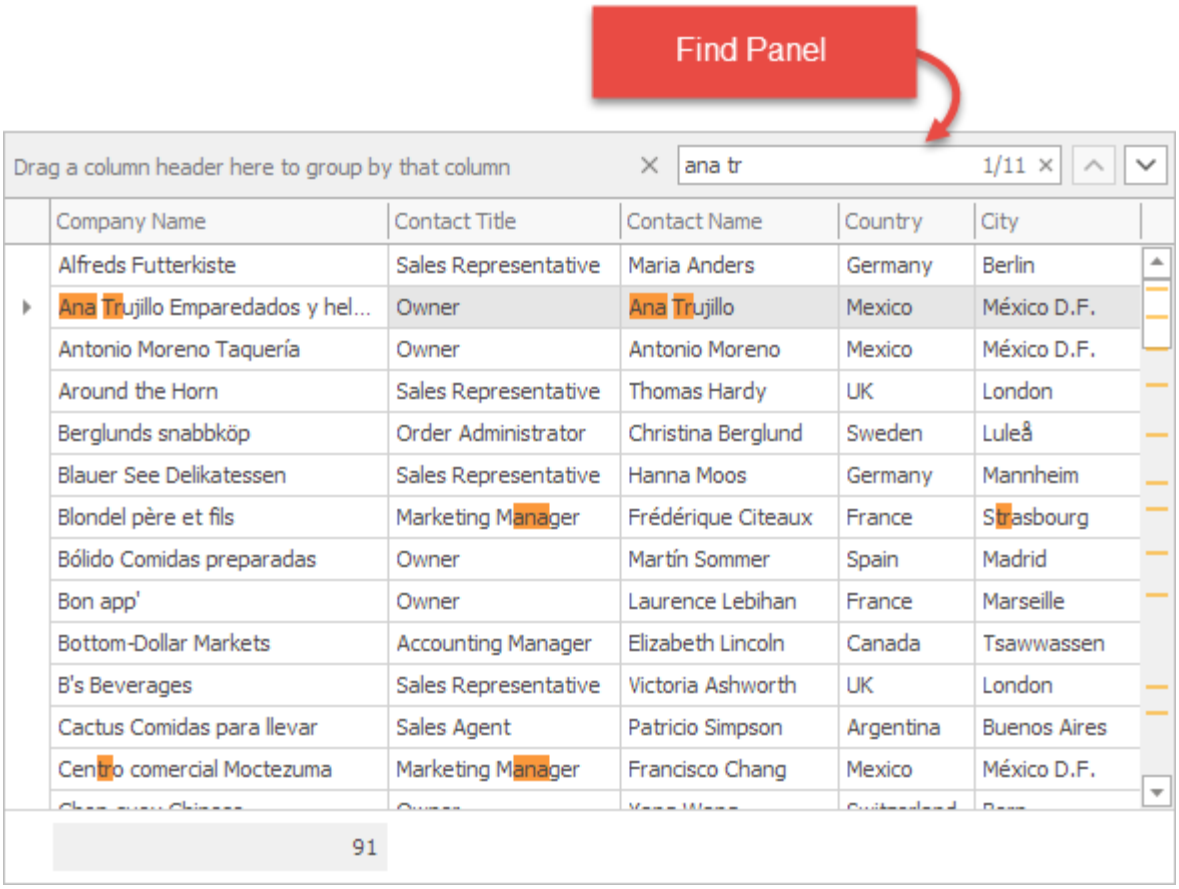
1.3.5.30 Data Navigator

The **data navigator** is an embedded control displayed at the bottom of a grid that enables the you to navigate and edit data. Note: it only handles visible rows. To access rows hidden within collapsed groups, you need to expand group rows first.



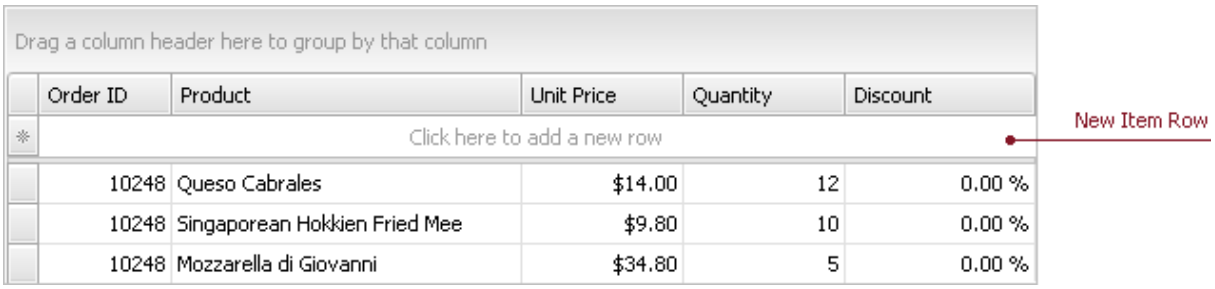
1.3.5.31 Find Panel

The grid can display a *find panel* that enables you to search for keywords in visible columns.



1.3.5.32 New Item Row/Card

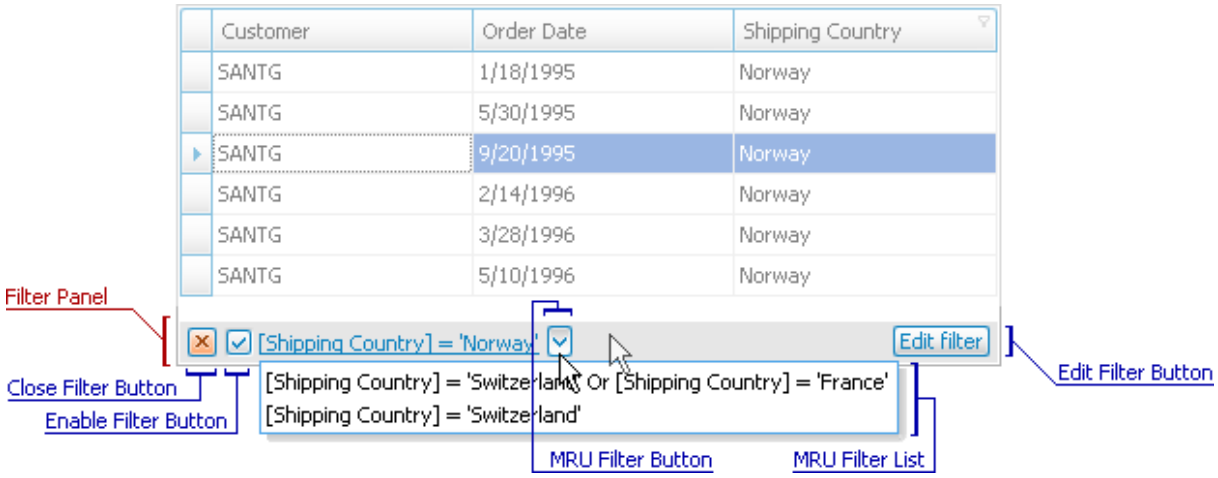
The **new item row/card** is used to enter new records. In Grid Views, the **new item row** can be displayed above or below all data and group rows. In Card Views and Layout Views the **new item card** is displayed as an empty card below all the cards.



1.3.5.33 Filter Panel

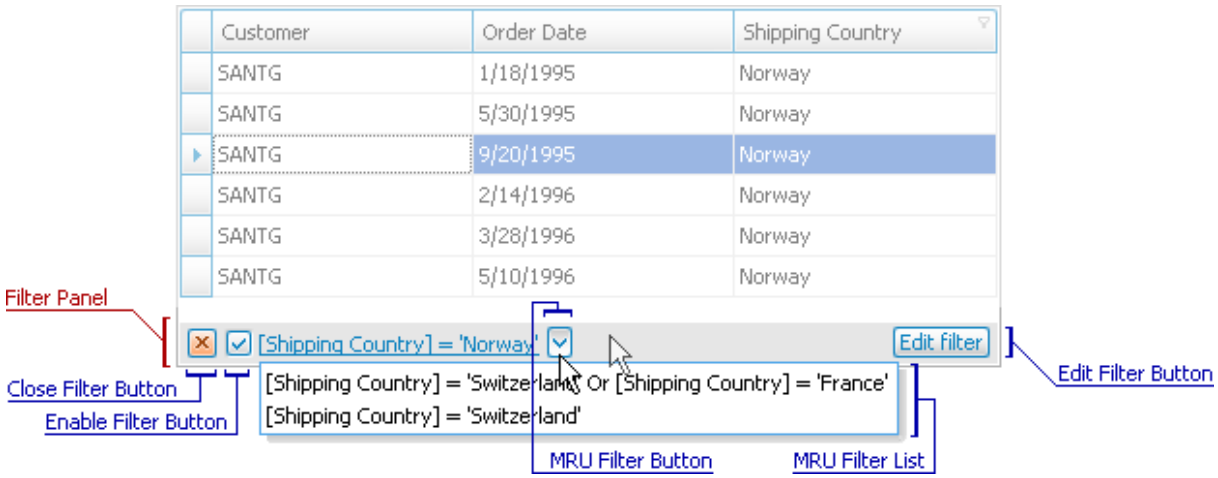
The **filter panel** displays a string representation of the filter criteria applied to a View. It can contain the following buttons:

- Close Filter Button - closes the filter panel clearing the filter;
- Enable Filter Button - can be used to temporarily disable and enable the current filter;
- MRU Filter Button - provides access to the most recently used filters;
- Edit Filter Button - opens the Filter Editor, allowing you to build complex filter criteria.



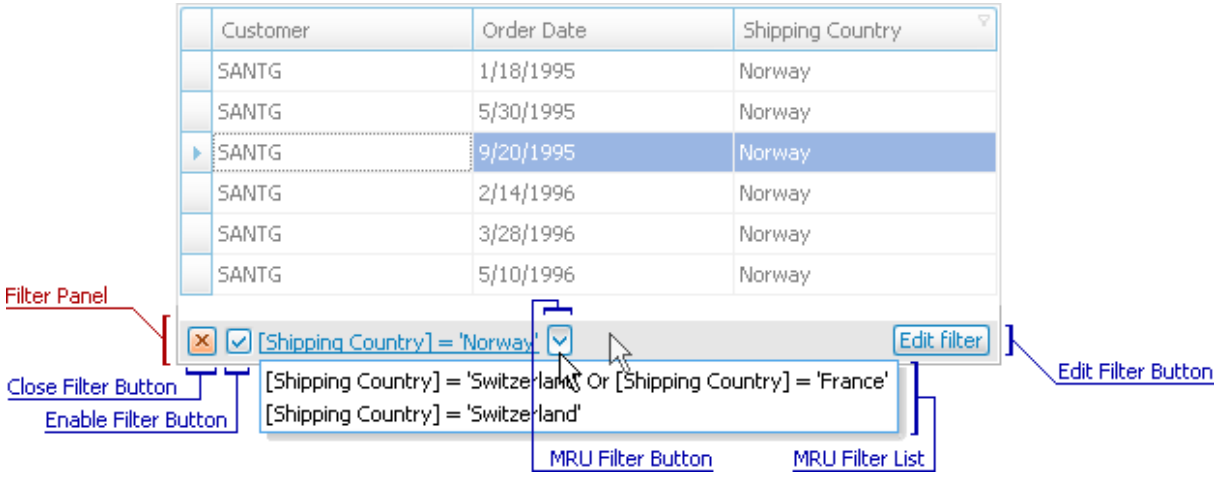
1.3.5.34 Filter Panel Close Button

The **close filter button** ("**x**") closes the Filter Panel and clears the current filter. The button is displayed on the left edge of the Filter Panel.



1.3.5.35 View's MRU (Most Recently Used) Filter List

The **MRU filter dropdown list** displays the most recently used filters in a View. This dropdown can be invoked by clicking the MRU filter button or the filter text within the Filter Panel .

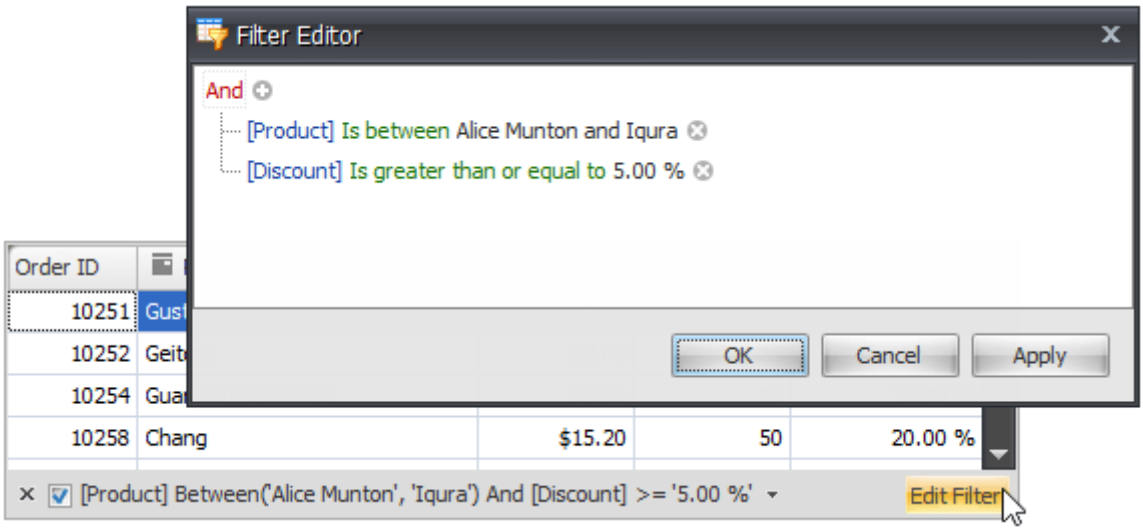


1.3.5.36 Edit Filter Button

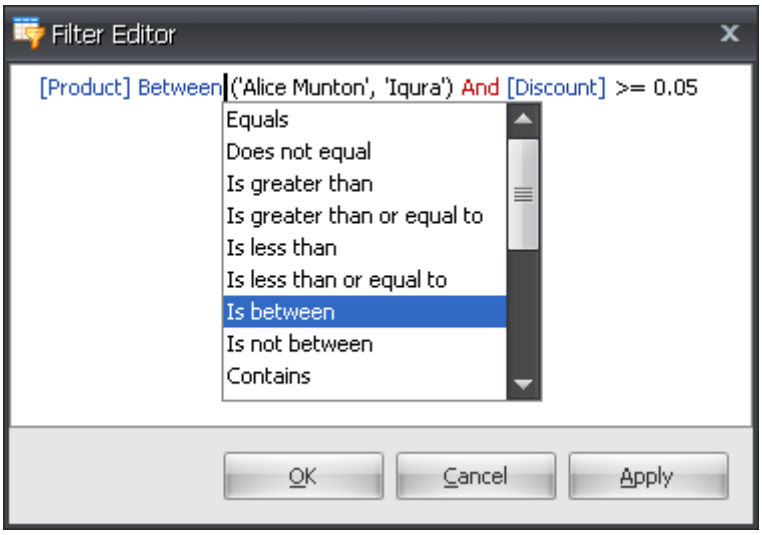
The **Edit Filter Button** enables you to invoke the Filter Editor. This button is displayed at the right edge of the Filter Panel.

1.3.5.37 Filter Editor

The **Filter Editor** allows you to build filter criteria in the Tree-like and/or IntelliSense text-based form.



Tree-like Filter Editor

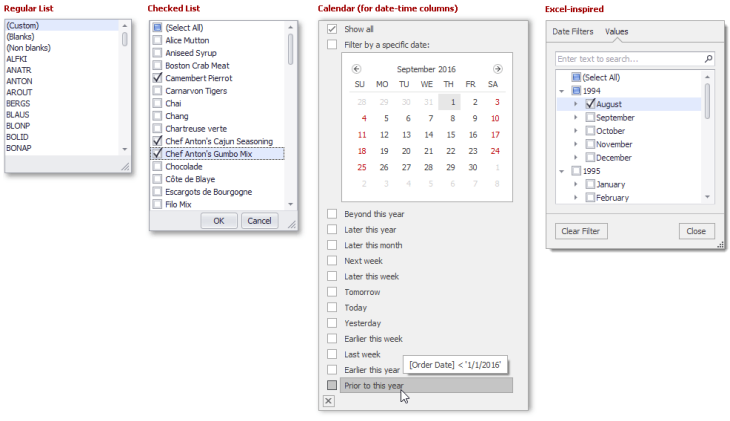


IntelliSense text-based Filter Editor

1.3.5.38 Column's Filter DropDown

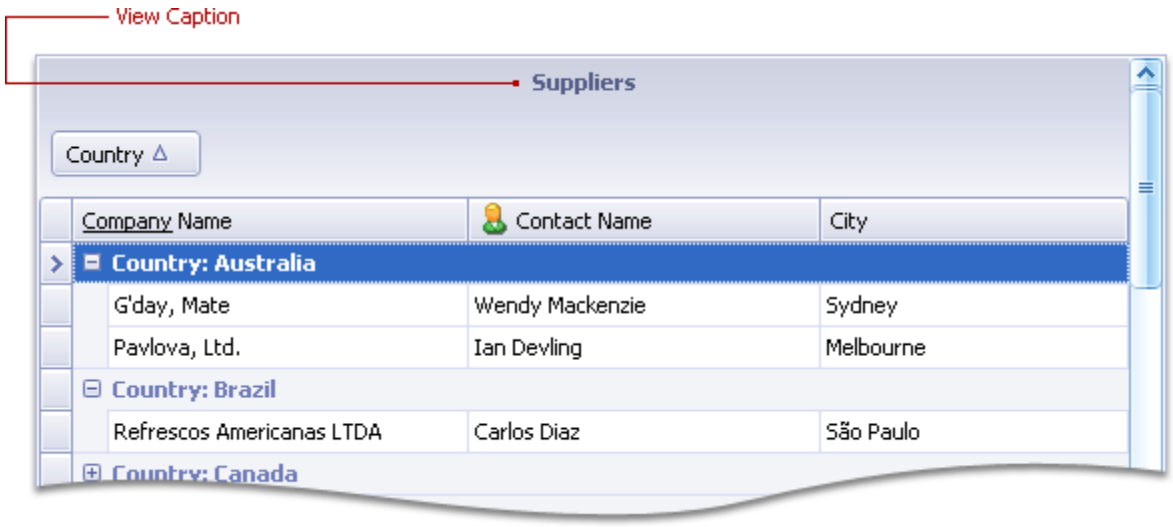
Filter dropdowns provide a UI for applying filter criteria against columns in the .Grids View

You can choose between four filter dropdown presentation styles.



1.3.5.39 View Caption

View captions represent titles for Views displayed at the top edge.

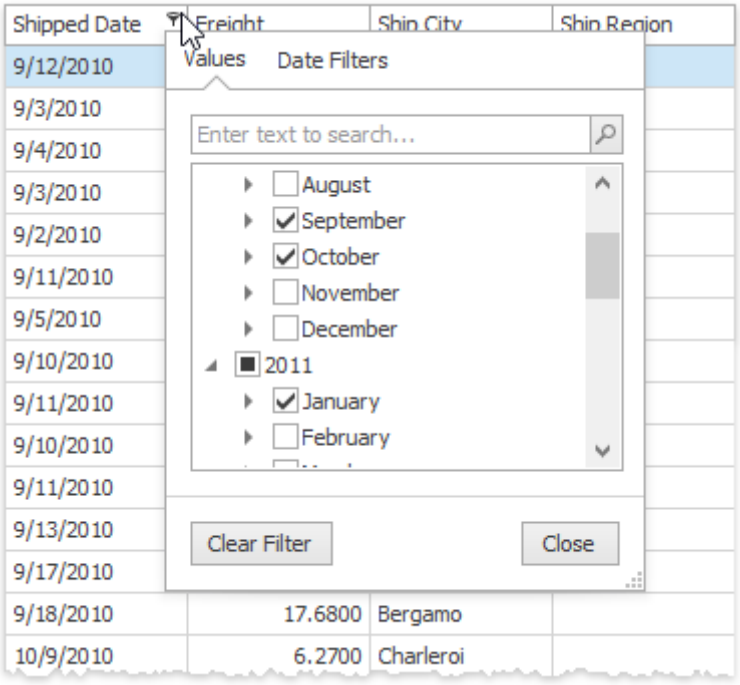


1.3.5.40 Filter and Search

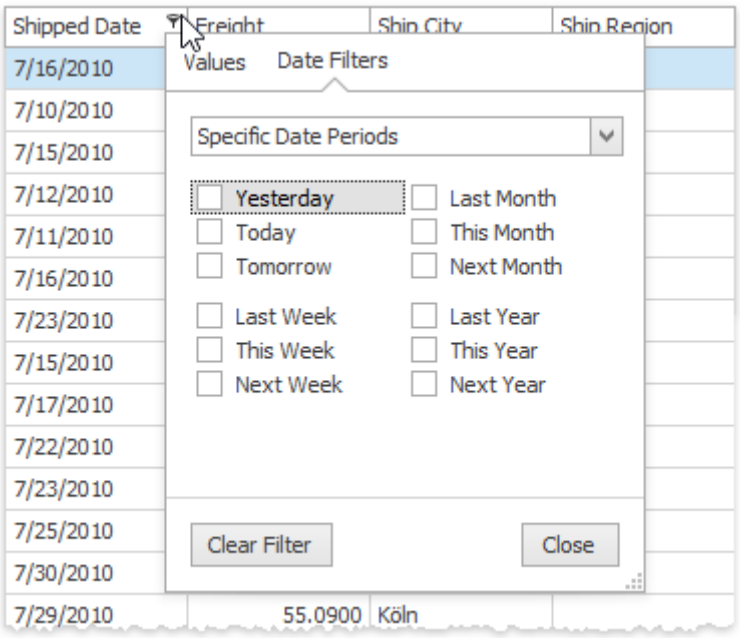
Filtering Dropdown Menus

When a filter is applied, the View displays only those records that meet the current filter criteria. You can filter data against one or multiple columns.

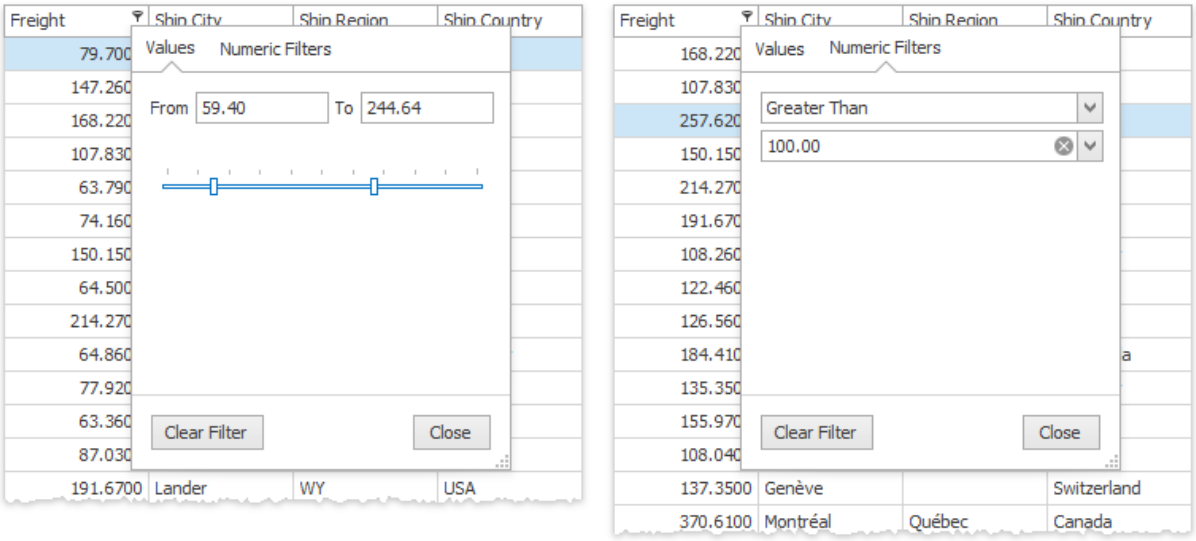
To invoke a filtering dropdown menu for a column, click the filter icon within the column header. In the "*Values*" tab, you can select specific cell values from those that are currently displayed by the Data Grid.



The "*Filters*" tab gives you a wider pool of filtering options. For example, when filtering by dates, you can only show those records that correspond to the previous week.



The content of a filtering dropdown menu depends on the type of data displayed by the related grid column. For instance, the figure below illustrates what this menu looks like when filtering by a numeric column.



1.3.5.41 Advanced Filter and Search Concepts

Filter Expressions

A filter expression is a formula (or a set of formulas) that specifies how data should be filtered. Each expression contains three parts:

- a data field whose values should be filtered;
- a filtering value that should be compared to records stored in the data field;
- an operator that compares data field values with a filtering value.

For example, the following expression selects all the "Count" data field values that are greater than 5 but less than 20:

```
[Count] Between ('5', '20')
```

Filter Expression Syntax

The table below enumerates most frequently used operators.

Operator	Description	Example
=	Equals. Selects data field values that equal the entered filtering value.	[OrderDate] = #2016-01-01#
<>	Does not equal. Selects data field values that are not equal to the entered filtering value.	[OrderDate] <> #2016-01-01#
>	Is greater than. Selects data field values that are greater than the entered filtering value.	[OrderDate] > #2016-01-01#
>=	Is greater than or equal to. Selects data field values that are greater than the entered filtering value or equal to it.	[OrderDate] >= #2016-01-01#
<=	Is less than or equal to. Selects data field values that are less than the entered filtering value or equal to it.	[OrderDate] >= #2016-01-01#
<	Is less than. Selects data field values that are less than the entered filtering value.	[OrderDate] >= #2016-01-01#
Between	Is between. Selects data field values that belong to the specific value interval.	[CustomerID] Between ('1', '100')
Not Between	Is not between. Selects data field values that lie outside the specific value interval.	Not [CustomerID] Between ('1', '100')
Contains	Contains. Selects data field values that contain the filtering value.	Contains([ShipCountry], 'land')

Not Contains	Does not contain. Selects data field values that do not contain the filtering value.	Not Contains([ShipCountry], 'land')
Starts with	Begins with. Selects data field values that start with the filtering value.	StartsWith([ShipCountry], 'G')
Ends with	Ends with. Selects data field values that ends with the filtering value.	Ends with([ShipCountry], 'ia')
In	Is any of. Selects data field values that equal any of the entered filtering values.	[ShipCountry] In ('Germany', 'Italy', 'USA')
Not In	Is none of. Selects data field values that do not equal any of the entered filtering values.	Not [ShipCountry] In ('Germany', 'Italy', 'USA')
Like	Is like. Selects data field values that contain the filtering value. Accepts wildcards: '_' to replace a single character, '%' to replace any number of characters.	[OrderDate] Like '%2011' same as Contains ([OrderDate], '2011')
Not Like	Is not like. Selects data field values that do not contain the filtering value. Accepts wildcards: '_' to replace a single character, '%' to replace any number of characters.	Not [OrderID] Like '103__' same as Not [OrderID] Between ('10300', '10399')
Is Null	Is null. Selects null values.	[ShipRegion] Is Null
Is Not Null	Is not null. Excludes null values.	[ShipRegion] Is Not Null

- **String** values must be enclosed within single quote characters. If a single quote character needs to be included as a literal to a filter, it must be doubled (e.g., [ProductID] LIKE 'Uncle Bob''s%');

- **Date-time** values must be wrapped with the '#' characters and represented using a culture-independent (invariant) format. The invariant culture is based on the English culture, but some of the idiosyncratic English formats have been replaced by more globally-accepted formats. Below are some of the culture-independent formats for representing date-time values.

  MM/dd/yyyy — 07/30/2008

  dd MMM yyyy — 30 JUL 2008

  yyyy-MM-dd — 2008-07-30

  yyyy-MM-ddTHH:mm:ss — 2008-07-30T22:59:59

  yyyy-MM-dd HH:mm:ssZ — 2008-07-30 15:59:59Z

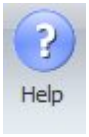
1.3.6 Themes

Setting the Theme

1.4 Getting Support

There are several ways to get help.

The first obvious place for help is this help file which is available by clicking on



button from the **Help tab** of the ribbon menu.

Next you can use the Support site.

You can also send Feedback or use the Wish List site.

Tutorial Videos

Getting Daily Tips

Find Out About Your Version

Find Out About Your Version

What's New

Getting the Latest Version

Subscribing to the Newsletter

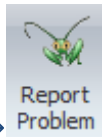
Reporting Problems

Adding to Your Wish-list

Sending Feedback

The AutoTRAX DEX Website

1.4.1 Reporting Problems

If you have a problem with AutoTRAX DEX and you can run AutoTRAX DEX, click the Help→Support→ 

button. This is the preferred method of reporting problems.

If you are unable to run AutoTRAX DEX, you can use the Support site to report problems.

Every effort possible is made to ensure that your problem gets resolved in a timely manner. Your patronage is appreciated and valued. You continued usage of AutoTRAX DEX, problem reporting, and suggestions help us to improve the product to your benefit.

1.4.2 Contacting Us

We love hearing from our customers as their feedback helps us improve AutoTRAX DEX, so please feel free to contact us.

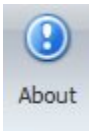
Email

You can contact us by email using the Send us email web page at https://pcbdex.com/Support/SendUsAnEmail

Location

You can find out more information about AutoTRAX DEX at https://pcbdex.com/Company or by sending us an email.

1.4.3 Find Out About Your Version

Information about AutoTRAX DEX can be obtained by using the 

button from the **Home tab** of the ribbon menu.

This will display the About AutoTRAX DEX Design Express dialog below

About AutoTRAX DEX Design Express

The build version is displayed towards the top right of the dialog. In this case AutoTRAX DEX was built on the 3rd. February, 2012 at 9.45 GMT.

View Installation Directory - Displays the file location where AutoTRAX DEX has been installed. This includes the program itself, all supporting assemblies and this help file.

View Data Directory - Displays the file location of the local database of parts.

View Settings Directory - Displays the file location of the settings files for AutoTRAX DEX

Clicking **OK** closes the dialog box and clicking on the **Help button** shows this help topic.

1.4.4 What's New

To find out What's New in AutoTRAX DEX click on the 

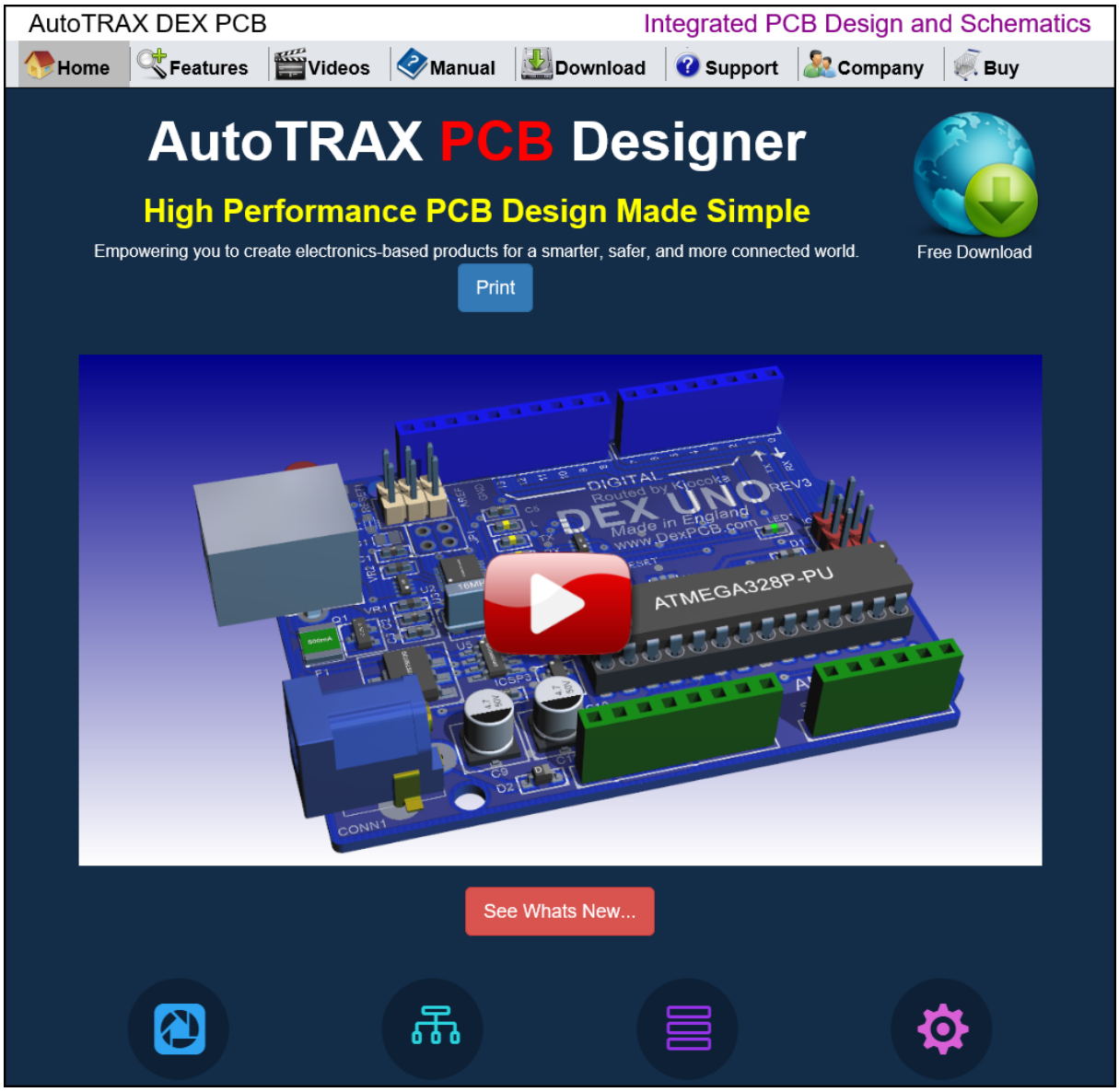
button in the **Home→Account** ribbon menu button group.

1.4.5 Subscribing to the NewsLetter

You can subscribe to the AutoTRAX DEX newsletter at
https://pcbdex.com/newsletter/

1.4.6 The AutoTRAX Website

The AutoTRAX DEX website is at https://pcbdex.com



On the web site you fill find many useful features including:

- Download Details
- Download Archives
- Tutorial Videos
- Screen Shots and Slide Shows
- Support Topics
- Free Gerber Viewers
- Running AutoTRAX DEX on Apple Macs.

- Purchase Details
- Renewal Details and how to get updates for another 12 months.
- Contacting Us
- Company Details

1.4.7 Getting the Latest Version

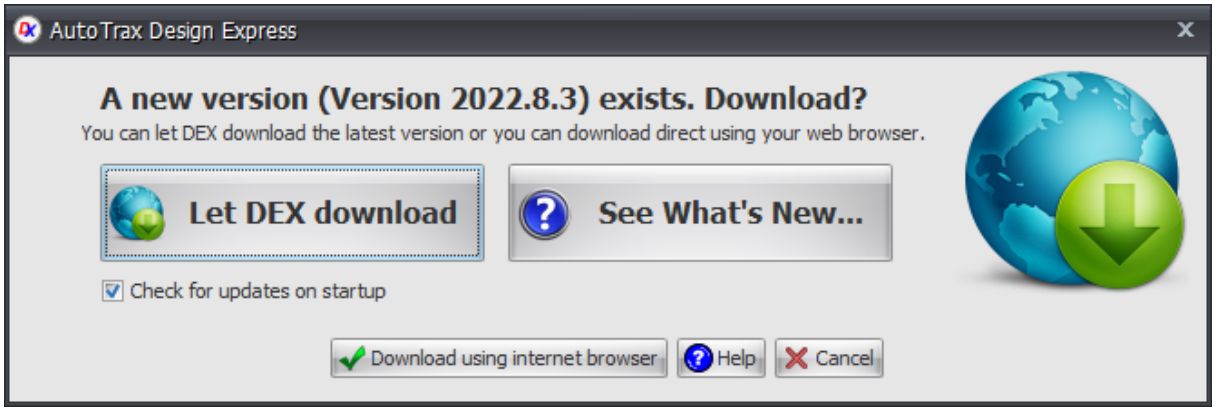
You can manually download the latest version of AutoTRAX DEX at https://pcbdex.com/download/

You can see an archive of older versions of AutoTRAX DEX at https://pcbdex.com/Download/Archive

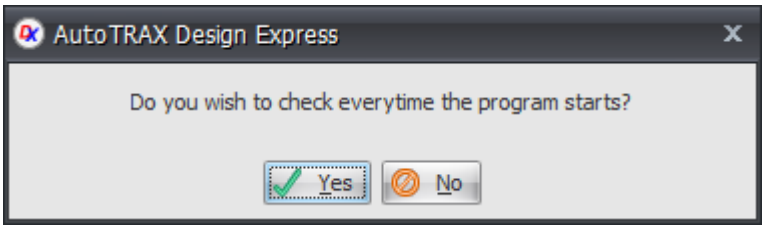
1.4.8 Downloading Older Versions

An archive of older versions of AutoTRAX DEX is available at https://pcbdex.com/Download/Archive

You can install more than one version of AutoTRAX DEX on the same machine. They will share a common set of configuration files. However you will need to disable auto-updating otherwise whenever you run an old version it will prompt you to download the latest. You can disable auto-updating when AutoTRAX DEX asks you if you wish to auto-update, see dialog below:



Click the cancel button to cancel auto-updating. AutoTRAX DEX will then ask you if you wish to check again next time the program runs. Click the No button to disable auto-updatings.



To re-enable auto-update click on the Get Latest Version button 

in the Home→Account menu

1.4.9 Sending Feedback

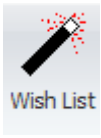
We would love to have feedback from you on improving both this manual and AutoTRAX DEX so that your design experience is improved.

To send feedback please go to the AutoTRAX DEX website and use the send feedback page at https://pcbdex.com/Support/Feedback

If you wish to send feedback on AutoTRAX DEX, click on the 

button in the **Help→Support** ribbon menu button group.

1.4.10 Adding to Your Wish List

If you have a useful feature you would like to have in AutoTRAX DEX, please select the 

button from the **Help→Support** ribbon menu button group. This will take you to the AutoTRAX DEX Software Wish-list web page.

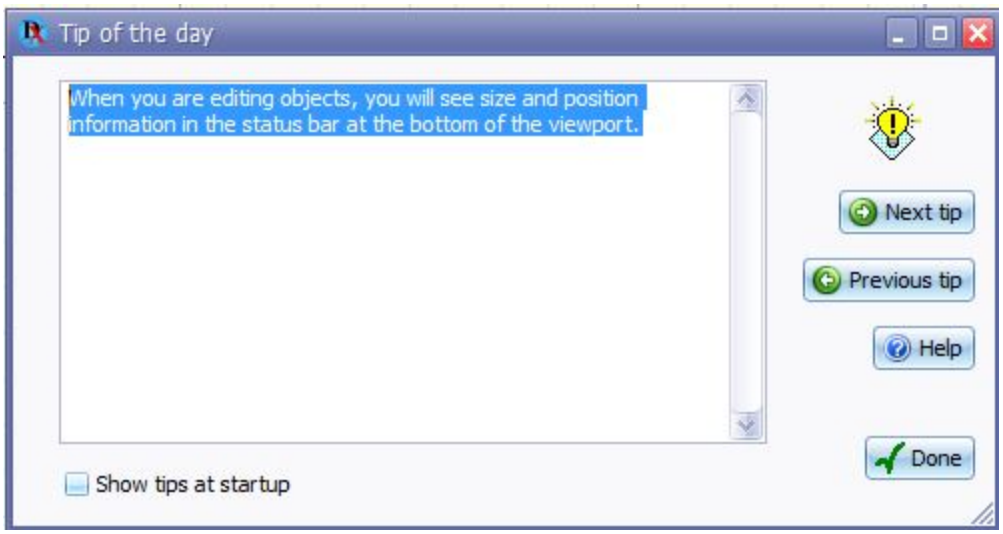
1.4.11 Getting Daily Tips

AutoTRAX DEX has a helpful **Tip of the Day** dialog that you can view on demand or optionally have appear every time you start AutoTRAX DEX.

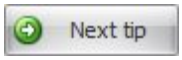
To display the **Tip of the Day** dialog use the 

button from **Help tab** of the ribbon menu.

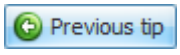
The dialog is shown below and contains handy tips on using AutoTRAX DEX.



Tip of the Day dialog box



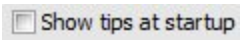
Shows the next 'tip of the day'.



Shows the previous 'tip of the day'.

Displays this help topic.

Closes the 'tip of the day' dialog.



If checked, then the 'tip of the day' will be displayed every time AutoTRAX DEX starts; otherwise, you will need to use the 

button from **Help tab** of the ribbon menu.

1.4.12 The Users Forum

The User Forum is a available on the Internet.

You can view it by clicking on 

in the Help ribbon page.

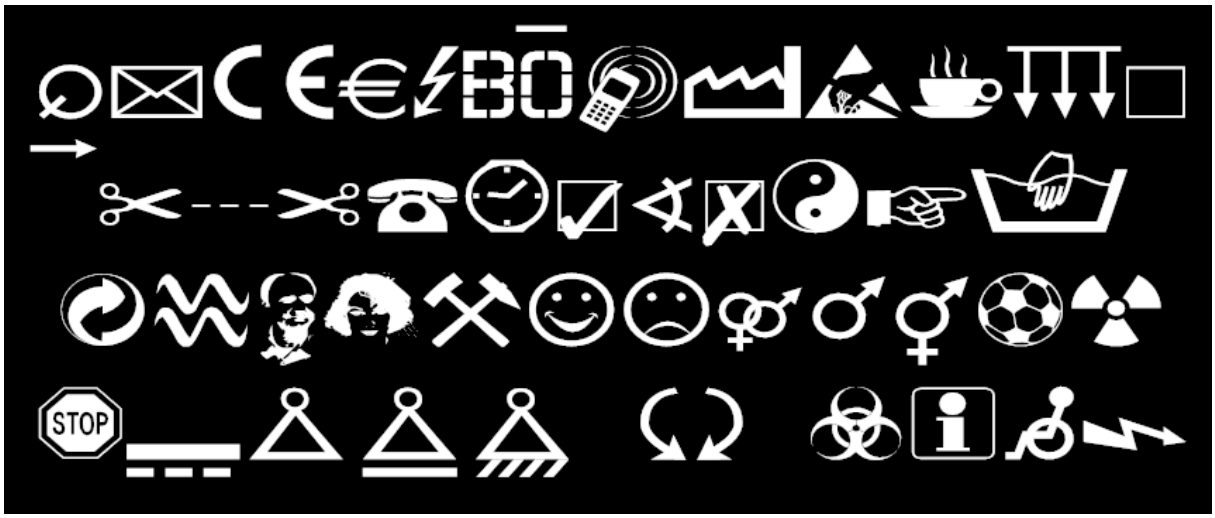
The Users Forum

1.4.13 Using Graphic Symbols Fonts

The Windows True Type Symbol Font MARVOSYM.TTF is a valuable addition for AutoTRAX DEX. It contains often-needed but rarely available symbols as the CE Symbol, Fax Machine, Answering Machine, Cellular Phone, Steel Profile Symbols, the original Euro Currency Symbol and lots more.

These are ideal for symbols to place on your PCB silkscreens or copper layers as well as in your schematic. These symbols will output to Gerber files.

NOTE: You must install this font on all machines you wish to view designs using this font. There are not needed once you generate Gerber files as the symbols are converted to Gerber commands.



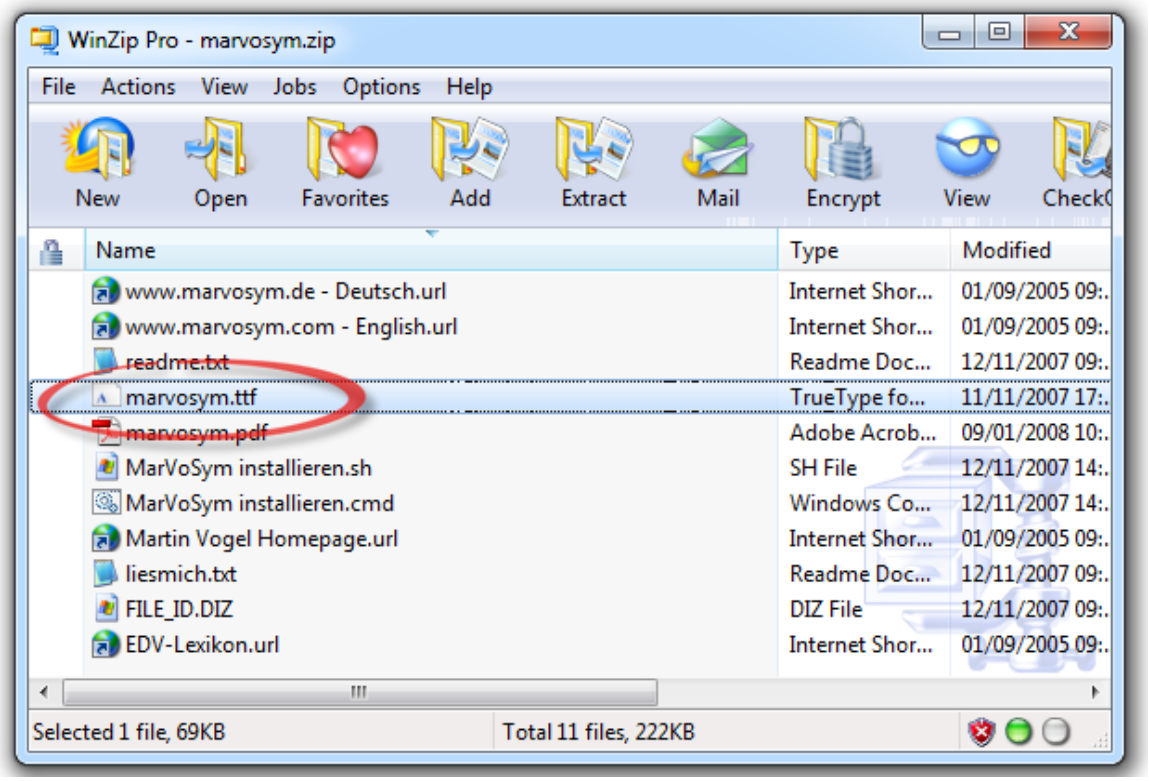
Sample Symbols

View the PDF documentation https://pcbdex.com/DownloadFiles/marvosym.pdf

Installing and Using The Font

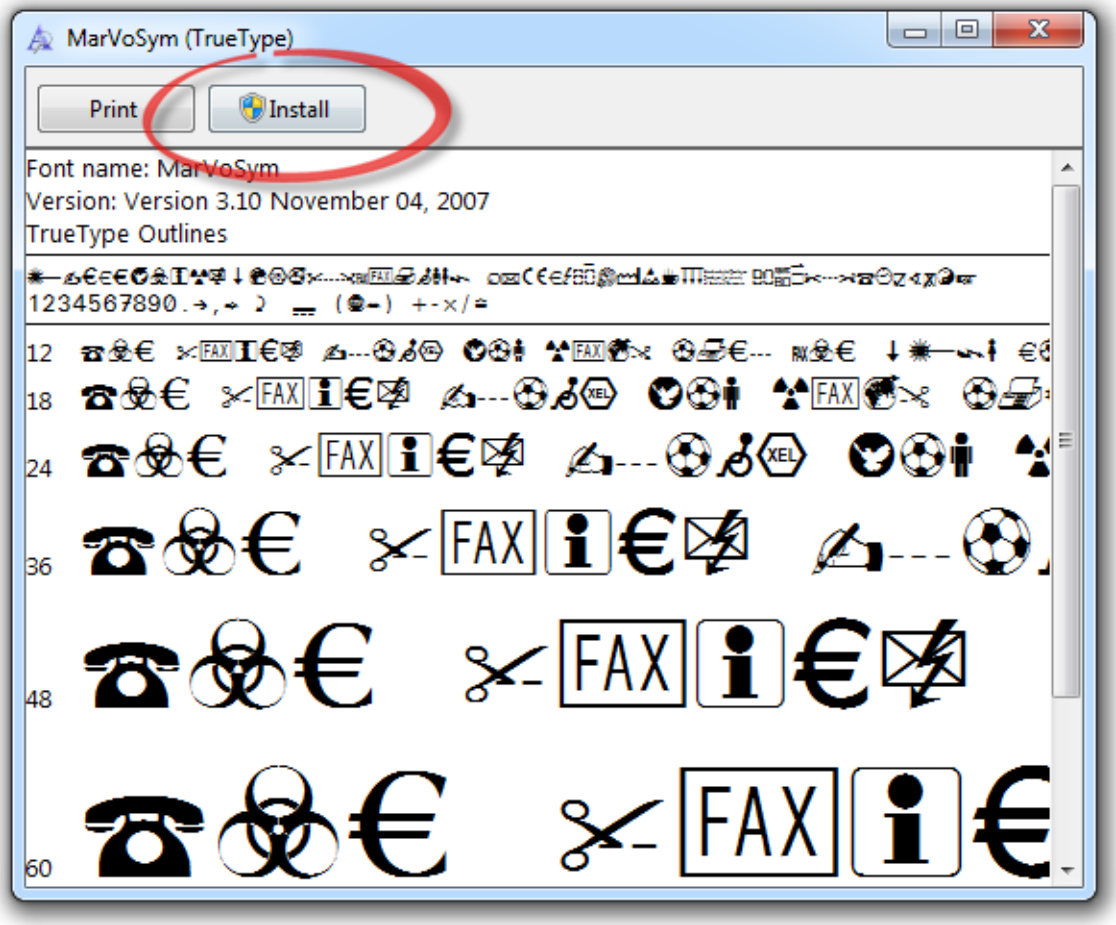
Download the font from https://pcbdex.com/DownloadFiles/marvosym.zip

Open the zip file and double click on marvosym.ttf



This shows the font dialog below. Click the 

button to install it.



To use it, add text to your schematic/PCB and set the font to MARVOSYM and set the font or scale the text.

1.5 Appendix

1.5.1 The Software License

License Agreement

THIS AutoTRAX DEX SOFTWARE LICENSE AGREEMENT ("LICENSE AGREEMENT") IS A LEGAL AGREEMENT BETWEEN YOU AND AUTOCRAT SOFTWARE, FOR THE SOFTWARE PRODUCT. BY USING THE SOFTWARE PRODUCT, YOU ARE AGREEING UNCONDITIONALLY TO BE BOUND BY THE TERMS OF THIS LICENSE AGREEMENT, EVEN IF THIS LICENSE AGREEMENT IS DEEMED A MODIFICATION OF ANY PREVIOUS ARRANGEMENT OR CONTRACT. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENSE AGREEMENT, DO NOT USE THE SOFTWARE PRODUCT. INSTEAD, YOU MAY RETURN THE SOFTWARE PRODUCT TO THE PLACE YOU OBTAINED IT FOR A FULL REFUND (IF APPLICABLE).

Software Product License

The Software Product is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The Software Product is licensed, not sold.

For purposes of this License Agreement, "Software Product" refers to the computer software and associated media, printed materials, and "online" or electronic documentation, including without limitation any and all executable files, addons, stencils, templates, databases, tutorials, help files and other files, that accompany the AutoTRAX DEX Software product identified above or in the accompanying documentation; "Use" means storing, loading (whether into temporary memory (i.e., RAM) or into permanent memory (e.g., hard disk, CD-ROM or other storage device)), installing, executing or displaying the Software Product; and "You" means the company, entity or individual whose funds are used to pay the license fee or who has otherwise acquired the Software Product.

1. Grant of License

Software Product. AutoTRAX DEX Software grants You the non-exclusive, non-sublicensable, limited license to Use one copy of the Software Product on a single computer, subject to the terms and conditions of this License Agreement.

Second Copy. The primary user of a computer on which a copy of the Software Product is installed may make a second copy for his or her exclusive Use on either a home or portable computer, but not both.

Storage/Network Use. You may also store or install a copy of the Software Product on a storage device, such as a network server, for the purpose of Using the Software Product on Your computers only over an internal network; however, You must acquire and dedicate a license for each separate computer on which the Software Product is Used from the storage device. A license for the Software Product may not be shared or Used concurrently on different computers. You shall inform all users of the Software Product of the terms and conditions of this License Agreement.

License Pack. If You have acquired this License Agreement in an AutoTRAX DEX Software License Pack, You may make the number of additional copies of the computer software portion of the Software Product authorized on the printed copy of this License Agreement, and You may Use each copy only in the manner specified in this License Agreement. You are also entitled to make a corresponding number of second copies for home or portable computer Use only in the manner specified in this License Agreement.

2. Ownership

Title, ownership rights and intellectual property rights in and to the Software Product shall remain in AutoTRAX DEX Software and its suppliers and are protected by US and international copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The Software Product is licensed, not sold. There is no transfer to You of any title to or ownership of the Software Product and the license granted under this License Agreement should not be construed as a sale of any right in the Software Product. You must treat the Software Product like any other copyrighted materials (e.g., a book or musical recording) except that You may make one copy of the Software Product solely for backup or archival purposes (with the inclusion of all copyright and other proprietary notices), only as long as You otherwise comply with the terms and conditions of this License Agreement. You may not make additional copies of the Software Product. You may not copy the printed materials accompanying the Software Product. All rights not specifically granted under this License Agreement are reserved by AutoTRAX DEX Software.

3. Description of Other Rights and Limitations

Limitations on Reverse-Engineering, Decompilation and Disassembly; Other Restrictions. You acknowledge that the Software Product in source code form remains a confidential trade secret of AutoTRAX DEX Software and therefore You agree not to reverse-engineer, decompile or disassemble the Software Product, or make any attempt to discover the source

code to the Software Product, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation. Except as expressly permitted in this License Agreement, the Software Product may not be Used, copied, translated, redistributed, retransmitted, published, sold, rented, leased, marketed, sub licensed, pledged, assigned, disposed of, encumbered, transferred, altered, modified or enhanced, whether in whole or in part, nor may You create derivative works from or based on the Software Product. You may not remove any proprietary notices, marks or labels from the Software Product.

Separation of Components. The Software Product is licensed as a single product and its component parts may not be separated for Use on more than one computer.

Transfer of Software Product. You may transfer all Your rights under this License Agreement on a permanent basis only, provided You retain no copies, You transfer the License Agreement, the corresponding serial number (if applicable) and all the Software Product (including without limitation all component parts, media and printed materials, and any upgrades) and the recipient agrees to all the terms and conditions of this License Agreement. If the Software Product is an upgrade product, any transfer must include the latest release, all prior versions and any prior products used to obtain the Software Product.

Termination. This License Agreement is in effect until terminated. You may terminate it at any time by destroying the Software Product and all copies You have made. Unauthorized copying or duplication of the Software Product will result in automatic termination of this License Agreement. Without prejudice to any other rights, AutoTRAX DEX Software may terminate this License Agreement if You fail to comply with any term or condition of this License Agreement. Upon termination of this License Agreement, You agree to destroy the Software Product and all copies You have made.

Academic Software Product. If the Software Product is identified as "Academic", You must be a Qualified Educational User (as defined by AutoTRAX DEX Software) to Use the Software Product. If You are not a Qualified Educational User, You have no rights under this License Agreement. To determine whether You are a Qualified Educational User, please contact AutoTRAX DEX Software - Academic Sales at the applicable address set forth below.

Not For Resale ("NFR") Software Product. If the Software Product is identified as "Not For Resale" or "NFR", then, notwithstanding any other terms and conditions of this License Agreement, You may not, for value or other consideration, distribute, resell, dispose of, or transfer the Software Product.

Support Services. AutoTRAX DEX Software may provide You with customer and technical support services related to the Software Product ("Support Services"). Use of Support Services is governed by the AutoTRAX DEX Software policies and programs described in the user manual, in "online" or electronic documentation, and/or in other AutoTRAX DEX Software-provided materials. Any supplemental software code provided to You as part of the Support Services shall be considered part of the Software Product and subject to the terms and

conditions of this License Agreement. With respect to any technical information You provide to AutoTRAX DEX Software as part of the Support Services, AutoTRAX DEX Software may use such information for its business purposes, including product support and development. AutoTRAX DEX Software will not utilize such technical information in a manner that personally identifies You.

4. Upgrades

If the Software Product is identified as an "upgrade", You must be properly licensed to Use a product, whether from AutoTRAX DEX Software or another supplier, that is specified by AutoTRAX DEX Software as being eligible for the upgrade in order to Use the Software Product. A Software Product identified as an upgrade replaces or supplements the product that formed the basis for Your eligibility for the upgrade. Notwithstanding any other terms and conditions of this License Agreement, You may transfer the Software Product only in conjunction with the product that formed the basis for Your eligibility for the upgrade, unless You destroy such product. If the Software Product is an upgrade from a component of a package of software programs which You licensed as a single product, the Software Product may be Used or transferred only as part of that single product package and may not be separated for Use on more than one computer.

5. Dual-Media Software

You may receive the Software Product in more than one medium; for example, in disk media, on a CD-ROM or in both media. Regardless of the type or size of medium You receive, You may Use only one medium that is appropriate for Your computer. You may not Use the other medium on another computer. You may not distribute, transmit, sell, loan, rent, lease or otherwise transfer the other medium to another user, except as part of the permanent transfer of the Software Product pursuant to the terms and conditions of this License Agreement.

6. Reports and Audit Rights

You shall institute reasonable measures to ensure compliance with the terms and conditions of this License Agreement. Upon AutoTRAX DEX Software's reasonable request, You agree to provide reports relating to Your Use of the Software Product as necessary to demonstrate Your compliance with the terms and conditions of this License Agreement. You further agree that AutoTRAX DEX Software has the right, upon reasonable prior notice, to audit Your records and inspect Your facilities to verify Your compliance with the terms and conditions of this License Agreement.

7. US Government Restricted Rights

The Software Product and documentation are provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the Government is subject to restrictions as set forth in sub-paragraph (c)(1)(ii) of The Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or sub-paragraphs (c)(1) and (2) of the Commercial Computer Software-

Restricted Rights at 48 CFR 52.227-19, as applicable. The contractor/manufacturer is AutoTRAX DEX Software, AutoTRAX DEX Software. Fair Oak, Skeath Lane, Sandon Bank, Stafford ST18 9TD, England.

8. Export Restrictions

You may not export or reexport the Software Product or any underlying information or technology except in full compliance with all United States and other applicable laws and regulations. In particular, but without limitation, none of the Software Product or underlying information or technology may be exported or reexported (a) into (or to a national or resident of) Cuba, Haiti, Iran, Iraq, Libya, Serbia, Montenegro, North Korea, Sudan or Syria or (b) to anyone on the US Treasury Department's list of Specially Designated Nationals or the US Commerce Department's Table of Deny Orders, as such countries, lists and orders may be amended or modified from time to time. By Using the Software Product, You are specifically agreeing to the foregoing and You are representing and warranting that You are not located in, under the control of, or a national or resident of any such country or on any such list.

9. Entire Agreement; Governing Law

This License Agreement constitutes the entire agreement between AutoTRAX DEX Software and You with regard to the subject matter hereof and supersedes any and all prior agreements, understandings and representations, whether written or oral, concerning the subject matter of this License Agreement. This License Agreement shall not be modify except by a written agreement executed by a duly authorized representative of each of AutoTRAX DEX Software and You.

Nothing in this License Agreement is intended to exclude, modify or restrict the operation of any applicable statute or other law, the provisions of which cannot lawfully be excluded, modified or restricted. If any court of competent jurisdiction determines that a provision of this License Agreement is illegal, invalid or unenforceable in any jurisdiction, then such provision shall be deemed modified to the minimum extent necessary to make it comply with the applicable statute or law of such jurisdiction, and the remaining provisions of this License Agreement shall continue in full force and effect. Any such modification shall not effect any provisions of this License Agreement in any other jurisdiction where this License Agreement governs the Use of the Software Product.

If You acquired the Software Product in Canada, You agree to the following: Each of AutoTRAX DEX Software and You hereby confirm, desire and agree that this License Agreement has been and shall be written solely in the English Language.

IF YOU ACQUIRED THE SOFTWARE PRODUCT IN THE UNITED STATES, THIS LICENSE AGREEMENT IS GOVERNED BY THE LAWS OF THE STATE OF WASHINGTON, USA, EXCEPT FOR THAT BODY OF LAW DEALING WITH CONFLICTS OF LAW, WITHOUT REFERENCE TO THE 1980 UNITED NATIONS CONVENTION ON THE INTERNATIONAL SALE OF GOODS.

IF YOU ACQUIRED THE SOFTWARE PRODUCT OUTSIDE THE UNITED STATES, THIS LICENSE AGREEMENT IS GOVERNED BY THE LAWS OF THE REPUBLIC OF IRELAND.

Software Product Limited Warranty

To the original customer only, AutoTRAX DEX Software provides the following warranties:

Limited Warranty. AutoTRAX DEX SOFTWARE WARRANTS THAT (a) FOR A PERIOD OF SIXTY (60) DAYS FROM THE DATE OF ORIGINAL PURCHASE ("WARRANTY PERIOD") AS EVIDENCED BY YOUR RECEIPT OR OTHER PROOF OF PURCHASE (i) THE SOFTWARE PRODUCT, UNLESS MODIFIED OR OTHERWISE ALTERED BY YOU, WILL PERFORM SUBSTANTIALLY IN ACCORDANCE WITH THE ACCOMPANYING WRITTEN MATERIALS, AND (ii) THE MEDIA ON WHICH THE SOFTWARE PRODUCT IS FURNISHED WILL BE FREE FROM DEFECTS IN MATERIALS AND WORKMANSHIP UNDER NORMAL USE; AND (b) ANY SUPPORT SERVICES PROVIDED BY AutoTRAX DEX SOFTWARE SHALL BE SUBSTANTIALLY AS DESCRIBED IN APPLICABLE WRITTEN MATERIALS PROVIDED TO YOU BY AutoTRAX DEX SOFTWARE. AutoTRAX DEX Software does not warrant that the Software Product will meet Your requirements or that Use of the Software Product will be uninterrupted or error-free. AutoTRAX DEX Software is not responsible for problems caused by changes in the operating characteristics of computer hardware or computer operating systems which are made after the release of the Software Product, nor for problems in the interaction of the Software Product with non-AutoTRAX DEX Software software products. Some jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to You. The Limited Warranty gives You specific legal rights. You may have others, which vary by jurisdiction.

Exclusive Remedy. AutoTRAX DEX Software's entire liability, and Your exclusive remedy, shall be, at AutoTRAX DEX Software's option, either (a) replacement of the defective media, (b) repair or replacement of the Software Product that does not meet AutoTRAX DEX Software's Limited Warranty, or (c) return of the price paid, if any, and termination of this License Agreement. This remedy is subject to return of the Software Product to AutoTRAX DEX Software with a copy of Your receipt within the Warranty Period or, solely for Software Product that was obtained electronically via "electronic software distribution", to delivery to AutoTRAX DEX Software of an AutoTRAX DEX Software-approved "certification of destruction" together with proof of purchase within the Warranty Period. This Limited Warranty is void if failure of the Software Product has resulted from accident, abuse or misapplication. Any replacement Software Product will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. Outside of the United States, neither these remedies nor any Support Services are available without proof of purchase from an authorized source.

No Other Warranties. THE ABOVE WARRANTIES ARE EXCLUSIVE. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, AutoTRAX DEX SOFTWARE AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NON-INFRINGEMENT, AND THOSE ARISING OUT OF USAGE OF TRADE OR COURSE OF DEALING, CONCERNING THE SOFTWARE PRODUCT, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. NO ORAL OR WRITTEN INFORMATION OR ADVICE GIVEN BY AutoTRAX DEX SOFTWARE, ITS AGENTS, DEALERS, DISTRIBUTORS OR EMPLOYEES SHALL INCREASE THE SCOPE OF THE ABOVE WARRANTIES OR CREATE ANY OTHER WARRANTIES.

No Liability for Damages. REGARDLESS OF WHETHER ANY REMEDY SET FORTH HEREIN FAILS OF ITS ESSENTIAL PURPOSE, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL AutoTRAX DEX SOFTWARE OR ITS SUPPLIERS (OR THEIR RESPECTIVE AGENTS, DIRECTORS, EMPLOYEES OR REPRESENTATIVES) BE LIABLE FOR ANY DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, CONSEQUENTIAL, INCIDENTAL, INDIRECT, SPECIAL, ECONOMIC, PUNITIVE OR SIMILAR DAMAGES, OR DAMAGES FOR LOSS OF BUSINESS PROFITS, LOSS OF GOODWILL, BUSINESS INTERRUPTION, COMPUTER FAILURE OR MALFUNCTION, LOSS OF BUSINESS INFORMATION OR ANY AND ALL OTHER COMMERCIAL OR PECUNIARY DAMAGES OR LOSSES) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, HOWEVER CAUSED AND ON ANY LEGAL THEORY OF LIABILITY (WHETHER IN TORT, CONTRACT OR OTHERWISE), EVEN IF AutoTRAX DEX SOFTWARE HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY. YOU ACKNOWLEDGE THAT THE LICENSE FEE REFLECTS THIS ALLOCATION OF RISK. In any event, if any statute implies warranties or conditions not stated in this License Agreement, AutoTRAX DEX Software's entire liability under any provision of this License Agreement shall be limited to the greater of the amount actually paid by You to license the Software Product and Five United States Dollars (US$5.00), or, in the case of Support Services, providing such Support Services again or refunding the cost thereof. Because some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to You.

2633 Lincoln Blvd., #310, Santa Monica, California 90405, USA

1.5.1.1 Refund Policy

The products available for purchase on our web site are downloadable, fully-functional, and try-before-you-buy. We provide free trial periods to let you fully evaluate our products before you make a purchase decision.

Please use the trial period to make sure that the software meets your needs before purchasing a license. All of our software is fully-functional during the trial period. None of our software requires registration to fully enable it.

If you purchase one of our products, after your payment has cleared you will receive an email with the purchase code to activate the software. Once this information is emailed to you, no refunds will be given. We have this policy since it would be impossible for you to return your registered version of our software.

During your trial period, our support staff is available,via the website https://pcbdex.com/Support/, to assist with installation and configuration. We strongly recommend that all customers download, install, and test the trial version of any product prior to making a purchase.

In rare instances and only within 30 days of purchase, if due to technical difficulties or platform incompatibilities the software will not function, we may, at our discretion, issue a refund. In such instances, we require that you provide enough information for us to positively identify your purchase transaction (e.g., order number, your company name, date of transaction, purchase code, number of licenses purchased, etc.). If we are able to positively identify your order, and if your request is made within 30 days of purchase, you must submit to us a letter of destruction of software on your company letterhead before we will process the refund. AutoTRAX DEX Software is not responsible for lost, delayed, or misdirected mail or email, delays for downloading, or other communication system delays.

Acceptance of this Refund Policy

It is your responsibility to familiarize yourself with this refund policy. By placing an order for any of our products, you indicate that you have read this refund policy and that you agree with and fully accept the terms of this refund policy.

If you do not agree with or fully accept the terms of this refund policy, we ask that you do not place an order with us.

1.5.2 File Associations

AutoTRAX DEX associated the following file extensions.

.project - used for project files.

.part - used for part files.

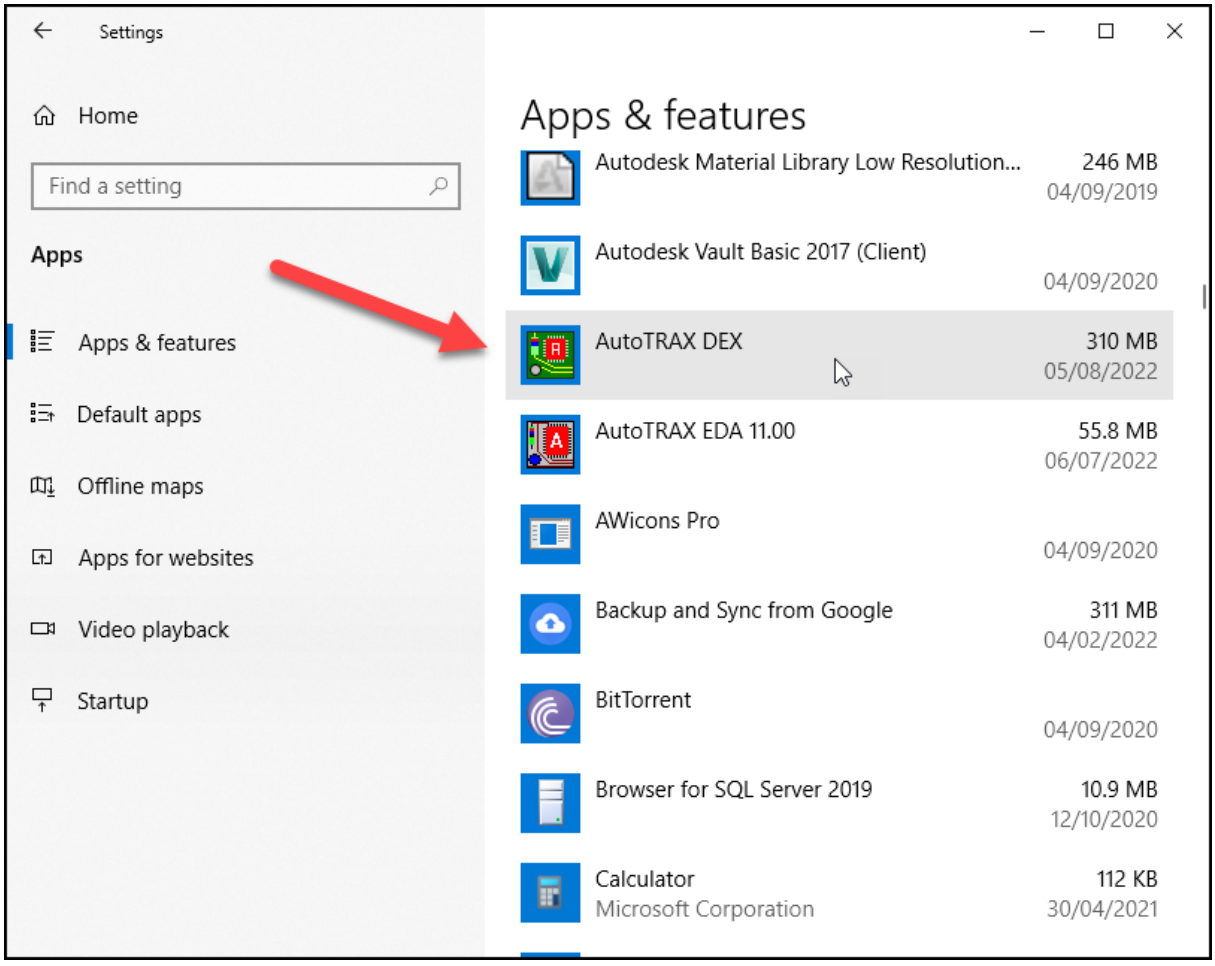
.art - used for art files.

A default program is the program that Windows uses when you open a particular type of file, such as a music file, an image, or a web-page. For example, if you have more than one web browser installed on your computer, you can choose one of them to be the default browser.

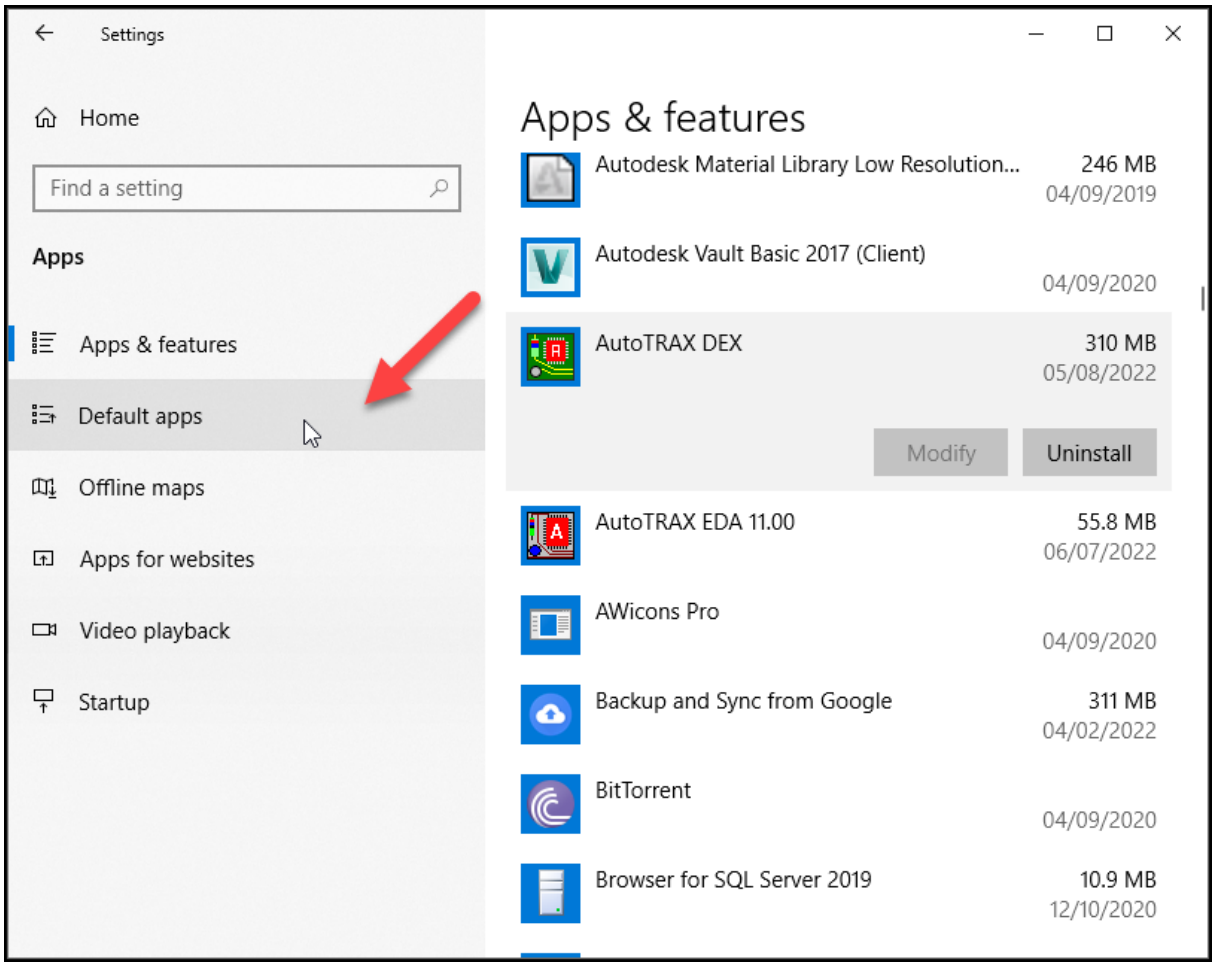
How to Change File Associations in Windows 10

Right-click the Start button (or use the WIN+X hotkey) and choose Settings.

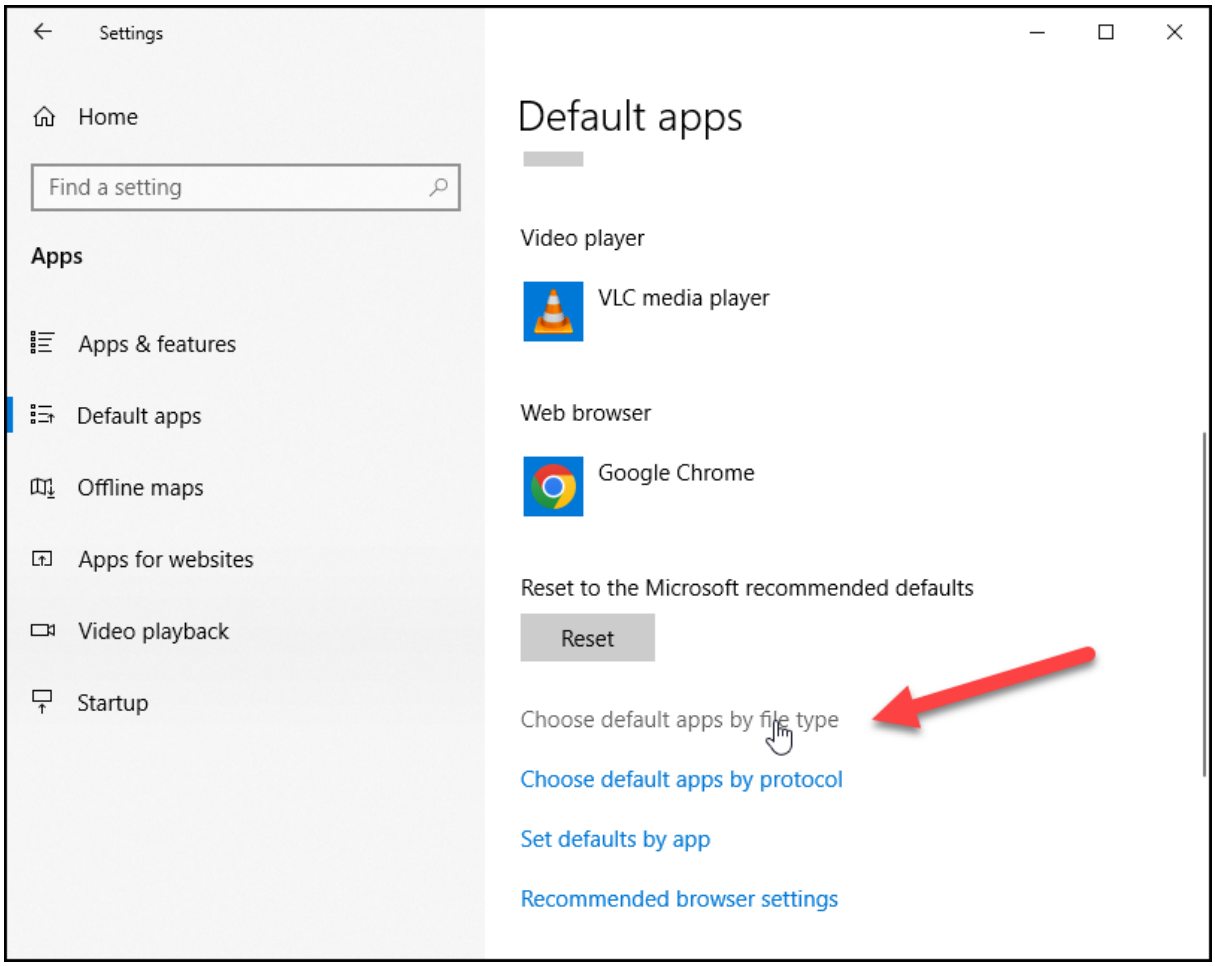
Select AutoTRAX DEX from the list



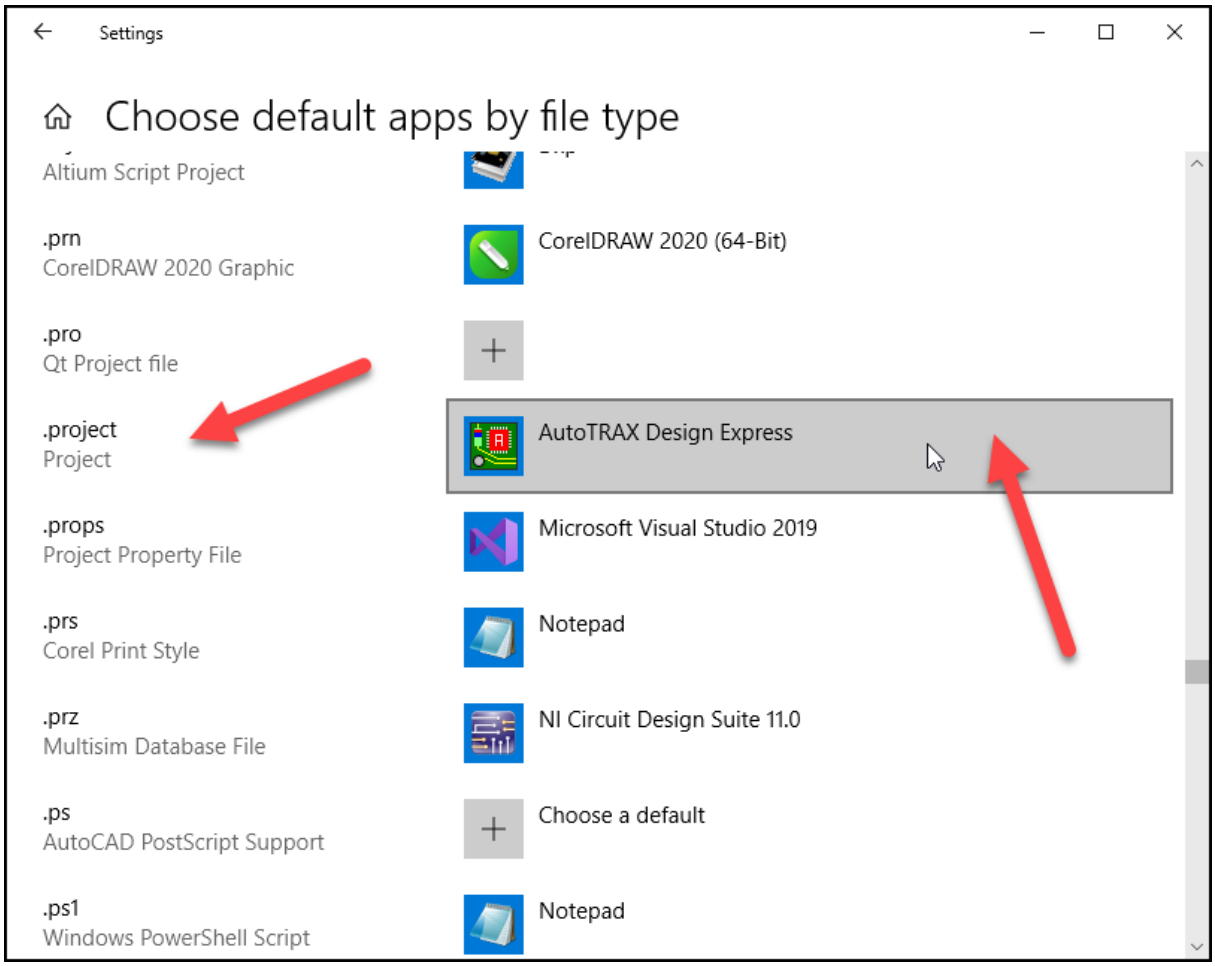
Choose Default apps on the left



Scroll down a little and select Choose default apps by file type



Locate the file extension for which you want to change the default program



1.5.3 The IPC

IPC, the Association Connecting Electronics Industries, is a trade association whose aim is to standardize the assembly and production requirements of electronic equipment and assemblies. It was founded in 1957 as the Institute for Printed Circuits. Its name was later changed to the Institute for Interconnecting and Packaging Electronic Circuits to highlight the expansion from bare boards to packaging and electronic assemblies. In 1999, the organization formally changed its name to IPC with the accompanying tagline, Association Connecting Electronics Industries.

IPC is accredited by the American National Standards Institute (ANSI) as a standards developing organization and is known globally for its standards. It publishes the most widely used acceptability standards in the electronics industry.

IPC standards are used by the electronics manufacturing industry. IPC-A-610, Acceptability of Electronic Assemblies, is used worldwide by original equipment manufacturers and EMS companies. There are more than 3600 trainers worldwide

who are certified to train and test on the standard. Standards are created by committees of industry volunteers. Task groups have been formed in China, the United States, and Denmark

View website...

1.5.4 Through-Hole Mounting (THM)

Through-hole mounting is the process by which component leads are placed into drilled holes on a bare PCB. The process was standard practice until the rise of surface mount technology (SMT) in the 1980s, at which time it was expected to completely phase out through-hole. Yet, despite a severe drop in popularity over the years, through-hole technology has proven resilient in the age of SMT, offering a number of advantages and niche applications: namely, reliability.

Through-hole components are best used for high-reliability products that require stronger connections between layers. Whereas SMT components are secured only by solder on the surface of the board, through-hole component leads run through the board, allowing the components to withstand more environmental stress. This is why through-hole technology is commonly used in military and aerospace products that may experience extreme accelerations, collisions, or high temperatures. Through-hole technology is also useful in test and prototyping applications that sometimes require manual adjustments and replacements.

Overall, through-hole's complete disappearance from PCB assembly is a wide misconception. Barring the above uses for through-hole technology, one should always keep in mind the factors of availability and cost. Not all components are available as SMD packages, and some through-hole components are less expensive.

However, that doesn't negate that fact that, in a modern assembly facility, through-hole is considered a secondary operation.

Through-hole technology (also spelled "thru-hole"), refers to the mounting scheme used for electronic components that involves the use of leads on the components that are inserted into holes drilled in printed circuit boards (PCB) and soldered to pads on the opposite side either by manual assembly (hand placement) or by the use of automated insertion mount machines.

History

Through-hole technology almost completely replaced earlier electronics assembly techniques such as point-to-point construction. From the second generation of computers in the 1950s until surface-mount technology (SMT) became popular in the late 1980s, every component on a typical PCB was a through-hole component. PCBs initially had tracks printed on one side only, later both sides, then multi-layer boards were in use. Through holes became plated-through holes (PTH) in order for the components to make contact with the required conductive layers. Plated-through holes are no longer required with SMT boards for making the component

connections, but are still used for making interconnections between the layers and in this role are more usually called vias.

Axial and radial leads

Components with wire leads are generally used on through-hole boards. Axial leads protrude from each end of a typically cylindrical or elongated box-shaped component, on the geometrical axis of symmetry. Axial-leaded components resemble wire jumpers in shape, and can be used to span short distances on a board, or even otherwise unsupported through an open space in point-to-point wiring. Axial components do not protrude much above the surface of a board, producing a low-profile or flat configuration when placed "lying down" or parallel to the board.

Radial leads project more or less in parallel from the same surface or aspect of a component package, rather than from opposite ends of the package. Originally, radial leads were defined as more-or-less following a radius of a cylindrical component (such as a ceramic disk capacitor). Over time, this definition was generalized in contrast to axial leads, and took on its current form. When placed on a board, radial components "stand up" perpendicular, occupying a smaller footprint on sometimes-scarce "board real estate", making them useful in many high-density designs. The parallel leads projecting from a single mounting surface gives radial components an overall "plugin nature", facilitating their use in high-speed automated component insertion ("board-stuffing") machines.

When needed, an axial component can be effectively converted into a radial component, by bending one of its leads into a "U" shape so that it ends up close to and parallel with the other lead. Extra insulation with heat-shrink tubing may be used to prevent shorting out on nearby components. Conversely, a radial component can be pressed into service as an axial component by separating its leads as far as possible, and extending them into an overall length-spanning shape. These improvisations are often seen in breadboard or prototype construction, but are deprecated for mass production designs. This is because of difficulties in use with automated component placement machinery, and poorer reliability because of reduced vibration and mechanical shock resistance in the completed assembly.

Multiple lead devices

For electronic components with two or more leads, for example diodes, transistors, ICs or resistor packs, a range of standard-sized semiconductor packages are used, either directly onto the PCB or via a socket.

Characteristics

While through-hole mounting provides strong mechanical bonds when compared to SMT techniques, the additional drilling required makes the boards more expensive to produce. They also limit the available routing area for signal traces on layers immediately below the top layer on multilayer boards since the holes must pass through all layers to the opposite side. To that end, through-hole mounting

techniques are now usually reserved for bulkier or heavier components such as electrolytic capacitors or semiconductors in larger packages such as the TO-220 that require the additional mounting strength, or for components such as plug connectors or electromechanical relays that require great strength in support.

Design engineers often prefer the larger through-hole rather than surface mount parts when prototyping, because they can be easily used with breadboard sockets. However, high-speed or high-frequency designs may require SMT technology to minimize stray inductance and capacitance in wire leads, which would impair circuit function. Ultra-compact designs may also dictate SMT construction, even in the prototype phase of design.

1.5.5 Surface Mount Technology (SMT)

SMT the process by which components are mounted directly onto the surface of the PCB. Known originally as “planar mounting,” the method was developed in the 1960s and has grown increasingly popular since the 1980s. Nowadays, virtually all electronic hardware is manufactured using SMT. It has become essential to PCB design and manufacturing, having improved the quality and performance of PCBs overall, and has reduced the costs of processing and handling greatly.

The key differences between SMT and through-hole mounting are (a) SMT does not require holes to be drilled through a PCB, (b) SMT components are much smaller, and (c) SMT components can be mounted on both side of the board. The ability to fit a high number of small components on a PCB has allowed for much denser, higher performing, and smaller PCBs.

Through-hole component leads, which run through the board and connect a board’s layers, have been replaced by "vias" -- small components which allow a conductive connection between the different layers of a PCB, and which essentially act as through-hole leads. Some surface mount components like BGAs are higher performing components with shorter leads and more interconnection pins that allow for higher speeds.

Surface-mount technology (SMT) is a method for producing electronic circuits in which the components are mounted or placed directly onto the surface of printed circuit boards (PCBs). An electronic device so made is called a surface-mount device (**SMD**). In industry, it has largely replaced the through-hole technology construction method of fitting components with wire leads into holes in the circuit board. Both technologies can be used on the same board, with the through-hole technology used for components not suitable for surface mounting such as large transformers and heat-sinked power semiconductors.

By employing SMT, the production process speeds up, but the risk of defects also increases due to component miniaturization and to the denser packing of boards. In those conditions, detection of failures has become critical for any SMT manufacturing process.

An SMT component is usually smaller than its through-hole counterpart because it has either smaller leads or no leads at all. It may have short pins or leads of various styles, flat contacts, a matrix of solder balls (BGAs), or terminations on the body of the component.

Surface mounting was originally called "planar mounting".

Surface-mount technology was developed in the 1960s and became widely used in the mid 1980s. By the late 1990s, the great majority of high-tech electronic printed circuit assemblies were dominated by surface mount devices. Much of the pioneering work in this technology was done by IBM. The design approach first demonstrated by IBM in 1960 in a small-scale computer was later applied in the Launch Vehicle Digital Computer used in the Instrument Unit that guided all Saturn IB and Saturn V vehicles. Components were mechanically redesigned to have small metal tabs or end caps that could be directly soldered to the surface of the PCB. Components became much smaller and component placement on both sides of a board became far more common with surface mounting than through-hole mounting, allowing much higher circuit densities and smaller circuit boards and, in turn, machines or subassemblies containing the boards.

History

Often only the solder joints hold the parts to the board; in rare cases parts on the bottom or "second" side of the board may be secured with a dot of adhesive to keep components from dropping off inside reflow ovens if the part has a large size or weight. Adhesive is sometimes used to hold SMT components on the bottom side of a board if a wave soldering process is used to solder both SMT and through-hole components simultaneously. Alternatively, SMT and through-hole components can be soldered on the same side of a board without adhesive if the SMT parts are first reflow-soldered, then a selective solder mask is used to prevent the solder holding those parts in place from reflowing and the parts floating away during wave soldering. Surface mounting lends itself well to a high degree of automation, reducing labor cost and greatly increasing production rates.

Conversely, SMT does not lend itself well to manual or low-automation fabrication, which is more economical and faster for one-off prototyping and small-scale production, and this is one reason why many through-hole components are still manufactured. Some SMDs can be soldered with a temperature-controlled manual soldering iron, but unfortunately, those that are very small or have too fine a lead pitch are impossible to manually solder without expensive hot-air solder reflow equipment. SMDs can be one-quarter to one-tenth the size and weight, and one-half to one-quarter the cost of equivalent through-hole parts, but on the other hand, the costs of a certain SMT part and of an equivalent through-hole part may be quite similar, though rarely is the SMT part more expensive.

Assembly techniques

Where components are to be placed, the printed circuit board normally has flat, usually tin-lead, silver, or gold plated copper pads without holes, called solder pads.

Solder paste, a sticky mixture of flux and tiny solder particles, is first applied to all the solder pads with a stainless steel or nickel stencil using a screen printing process. It can also be applied by a jet-printing mechanism, similar to an inkjet printer. After pasting, the boards then proceed to the pick-and-place machines, where they are placed on a conveyor belt. The components to be placed on the boards are usually delivered to the production line in either paper/plastic tapes wound on reels or plastic tubes. Some large integrated circuits are delivered in static-free trays. Numerical control pick-and-place machines remove the parts from the tapes, tubes or trays and place them on the PCB.

The boards are then conveyed into the reflow soldering oven. They first enter a pre-heat zone, where the temperature of the board and all the components is gradually, uniformly raised. The boards then enter a zone where the temperature is high enough to melt the solder particles in the solder paste, bonding the component leads to the pads on the circuit board. The surface tension of the molten solder helps keep the components in place, and if the solder pad geometries are correctly designed, surface tension automatically aligns the components on their pads.

There are a number of techniques for reflowing solder. One is to use infrared lamps; this is called infrared reflow. Another is to use a hot gas convection. Another technology which is becoming popular again is special fluorocarbon liquids with high boiling points which use a method called vapor phase reflow. Due to environmental concerns, this method was falling out of favor until lead-free legislation was introduced which requires tighter controls on soldering. At the end of 2008, convection soldering was the most popular reflow technology using either standard air or nitrogen gas. Each method has its advantages and disadvantages. With infrared reflow, the board designer must lay the board out so that short components don't fall into the shadows of tall components. Component location is less restricted if the designer knows that vapor phase reflow or convection soldering will be used in production. Following reflow soldering, certain irregular or heat-sensitive components may be installed and soldered by hand, or in large-scale automation, by focused infrared beam (FIB) or localized convection equipment.

If the circuit board is double-sided then this printing, placement, reflow process may be repeated using either solder paste or glue to hold the components in place. If a wave soldering process is used, then the parts must be glued to the board prior to processing to prevent them from floating off when the solder paste holding them in place is melted.

After soldering, the boards may be washed to remove flux residues and any stray solder balls that could short out closely spaced component leads. Rosin flux is removed with fluorocarbon solvents, high flash point hydrocarbon solvents, or low flash solvents e.g. limonene (derived from orange peels) which require extra rinsing or drying cycles. Water-soluble fluxes are removed with deionized water and detergent, followed by an air blast to quickly remove residual water. However, most electronic assemblies are made using a "No-Clean" process where the flux residues are designed to be left on the circuit board, since they are considered harmless. This saves the cost of cleaning, speeds up the manufacturing process, and reduces

waste. However, it is generally suggested to wash the assembly, even when a "No-Clean" process is used, when the application uses very high frequency clock signals (in excess of 1 GHz). Another reason to remove no-clean residues is to improve adhesion of conformal coatings and underfill materials. Regardless of cleaning or not those PCBs, current industry trend suggests to carefully review a PCB assembly process where "No-Clean" is applied, since flux residues trapped under components and RF shields may affect surface insulation resistance (SIR), especially on high component density boards.

Certain manufacturing standards, such as those written by the IPC - Association Connecting Electronics Industries require cleaning regardless of the solder flux type used to ensure a thoroughly clean board. Proper cleaning removes all traces of solder flux, as well as dirt and other contaminants that may be invisible to the naked eye. No-Clean or other soldering processes may leave "white residues" that, according to IPC, are acceptable "provided that these residues have been qualified and documented as benign". However, while shops conforming to IPC standard are expected to adhere to the Association's rules on board condition, not all manufacturing facilities apply IPC standard, nor are they required to do so. Additionally, in some applications, such as low-end electronics, such stringent manufacturing methods are excessive both in expense and time required.

Finally, the boards are visually inspected for missing or misaligned components and solder bridging. If needed, they are sent to a rework station where a human operator repairs any errors. They are then usually sent to the testing stations (in-circuit testing and/or functional testing) to verify that they operate correctly. Automated Optical Inspection (AOI) systems are commonly used in PCB manufacturing. This technology has proven highly efficient for process improvements and quality achievements.

Common abbreviations

Different terms describe the components, technique, and machines used in manufacturing. These terms are listed in the following table:

SMD Surface-mount devices (active, passive and electromechanical components)

SMT Surface-mount technology (assembling and mounting technology)

SMA Surface-mount assembly (module assembled with SMT)

SMC Surface-mount components (components for SMT)

SMP Surface-mount packages (SMD case forms)

SME Surface-mount equipment (SMT assembling machines)

Advantages

The main advantages of SMT over the older through-hole technique are:

- Smaller components.

- Much higher component density (components per unit area) and many more connections per component.
- Components can be placed on both sides of the circuit board.
- Higher density of connections because holes do not block routing space on inner layers, nor on back-side layers if components are mounted on only one side of the PCB.
- Small errors in component placement are corrected automatically as the surface tension of molten solder pulls components into alignment with solder pads. (On the other hand, through-hole components cannot be slightly misaligned, because once the leads are through the holes, the components are fully aligned and cannot move laterally out of alignment.)
- Better mechanical performance under shock and vibration conditions (partly due to lower mass, and partly due to less cantilevering)
- Lower resistance and inductance at the connection; consequently, fewer unwanted RF signal effects and better and more predictable high-frequency performance.
- Better EMC performance (lower radiated emissions) due to the smaller radiation loop area (because of the smaller package) and the lesser lead inductance.
- Fewer holes need to be drilled. (Drilling PCBs is time-consuming and expensive.)
- Lower initial cost and time of setting up for mass production, using automated equipment.
- Simpler and faster automated assembly. Some placement machines are capable of placing more than 136,000 components per hour.
- Many SMT parts cost less than equivalent through-hole parts.
- A surface mount package is favored where a low profile package is required or the space available to mount the package is limited. As electronic devices become more complex and available space is reduced, the desirability of a surface mount package increases. Concurrently, as the device complexity increases, the heat generated by operation increases. If the heat is not removed, the temperature of the device rises shortening the operational life. It is therefore highly desirable to develop surface mount packages having high thermal conductivity.

Disadvantages

- SMT is unsuitable for large, high-power, or high-voltage parts, for example in power circuitry. It is common to combine SMT and through-hole construction, with transformers, heat-sinked power semiconductors, physically large capacitors, fuses, connectors, and so on mounted on one side of the PCB through holes.

- SMT is unsuitable as the sole attachment method for components that are subject to frequent mechanical stress, such as connectors that are used to interface with external devices that are frequently attached and detached.
- SMDs' solder connections may be damaged by potting compounds going through thermal cycling.
- Manual prototype assembly or component-level repair is more difficult and requires skilled operators and more expensive tools, due to the small sizes and lead spacings of many SMDs. Handling of small SMT components can be difficult, requiring tweezers, unlike nearly all through-hole components. Whereas through-hole components will stay in place (under gravitational force) once inserted and can be mechanically secured prior to soldering by bending out two leads on the solder side of the board, SMDs are easily moved out of place by a touch of a soldering iron. Without expert skill, when manually soldering or desoldering a component, it is easy to accidentally reflow the solder of an adjacent SMT component and unintentionally displace it, something that is almost impossible to do with through-hole components.
- Many types of SMT component packages cannot be installed in sockets, which provide for easy installation or exchange of components to modify a circuit and easy replacement of failed components. (Virtually all through-hole components can be socketed.)
- SMDs cannot be used directly with plug-in breadboards (a quick snap-and-play prototyping tool), requiring either a custom PCB for every prototype or the mounting of the SMD upon a pin-leaded carrier. For prototyping around a specific SMD component, a less-expensive breakout board may be used. Additionally, stripboard style protoboards can be used, some of which include pads for standard sized SMD components. For prototyping, "dead bug" breadboarding can be used.
- Solder joint dimensions in SMT quickly become much smaller as advances are made toward ultra-fine pitch technology. The reliability of solder joints becomes more of a concern, as less and less solder is allowed for each joint. Voiding is a fault commonly associated with solder joints, especially when reflowing a solder paste in the SMT application. The presence of voids can deteriorate the joint strength and eventually lead to joint failure.
- SMDs, usually being smaller than equivalent through-hole components, have less surface area for marking, requiring marked part ID codes or component values to be more cryptic and smaller, often requiring magnification to be read, whereas a larger through-hole component could be read and identified by the unaided eye. This is a disadvantage for prototyping, repair, or rework, and possibly for production set-up.

Packages

Surface-mount components are usually smaller than their counterparts with leads, and are designed to be handled by machines rather than by humans. The electronics industry has standardized package shapes and sizes (the leading standardisation body is JEDEC).

Small outline diode (SOD)

SOD-923 0.8 × 0.6 × 0.4 mm

SOD-723 1.4 × 0.6 × 0.59 mm

SOD-523 (SC-79) 1.25 × 0.85 × 0.65 mm

SOD-323 (SC-90) 1.7 × 1.25 × 0.95 mm

SOD-128 5 × 2.7 × 1.1 mm

SOD-123 3.68 × 1.17 × 1.60 mm

SOD-80C 3.50 × □ 1.50 mm

1.5.6 Through-Hole vs. Surface Mount

Through-Hole Mounting (THM)

Through-hole mounting is the process by which component leads are placed into drilled holes on a bare PCB. The process was standard practice until the rise of surface mount technology (SMT) in the 1980s, at which time it was expected to completely phase out through-hole. Yet, despite a severe drop in popularity over the years, through-hole technology has proven resilient in the age of SMT, offering a number of advantages and niche applications: namely, reliability.

Through-hole components are best used for high-reliability products that require stronger connections between layers. Whereas SMT components are secured only by solder on the surface of the board, through-hole component leads run through the board, allowing the components to withstand more environmental stress. This is why through-hole technology is commonly used in military and aerospace products that may experience extreme accelerations, collisions, or high temperatures. Through-hole technology is also useful in test and prototyping applications that sometimes require manual adjustments and replacements.

Overall, through-hole's complete disappearance from PCB assembly is a wide misconception. Barring the above uses for through-hole technology, one should always keep in mind the factors of availability and cost. Not all components are available as SMD packages, and some through-hole components are less expensive.

However, that doesn't negate that fact that, in a modern assembly facility, through-hole is considered a secondary operation.

Axial vs. Radial Lead Components

There are two types of through-hole components: axial and radial lead components. Axial leads run through a component in a straight line ("axially"), with each end of the lead wire exiting the component on either end. Both ends are then placed through two separate holes in the board, allowing the component to fit closer, flatter fit. Radial lead components, on the other hand, protrude from the board, as its leads are located on one side of the component.

Both through-hole component types are "twin" lead components, and both have their distinct advantages. While axial lead components are used for their snugness to the board, radial leads occupy less surface area, making them better for high density boards. Generally, axial lead configuration may come in the form of carbon resistors, electrolytic capacitors, fuses, and light-emitting diodes (LEDs). Radial lead components are available as ceramic disk capacitors.

Advantages: THM provides stronger mechanical bonds than SMT, making through-hole ideal for components that might undergo mechanical stress, such as connectors or transformers. Good for test and prototyping.

Disadvantages: On the bare PCB side, THM requires the drilling holes, which is expensive and time consuming. THM also limits the available routing area on any multilayer boards, because the drilled holes must pass through all the PCB's layers. On the assembly side, component placement rates for THM are a fraction of surface mount placement rates, making THM prohibitively expensive. Further, THM requires the use of wave, selective, or hand-soldering techniques, which are much less reliable and repeatable than reflow ovens used for surface mount. Most of all, through-hole technology requires soldering on both sides of the board, as opposed to surface-mounts, which only -- for the most part -- require attention to one side of the board.

Surface Mount Technology (SMT)

SMT the process by which components are mounted directly onto the surface of the PCB. Known originally as "planar mounting," the method was developed in the 1960s and has grown increasingly popular since the 1980s. Nowadays, virtually all electronic hardware is manufactured using SMT. It has become essential to PCB design and manufacturing, having improved the quality and performance of PCBs overall, and has reduced the costs of processing and handling greatly.

The key differences between SMT and through-hole mounting are (a) SMT does not require holes to be drilled through a PCB, (b) SMT components are much smaller, and (c) SMT components can be mounted on both side of the board. The ability to fit a high number of small components on a PCB has allowed for much denser, higher performing, and smaller PCBs.

Through-hole component leads, which run through the board and connect a board's layers, have been replaced by "vias" -- small components which allow a conductive connection between the different layers of a PCB, and which essentially act as

through-hole leads. Some surface mount components like BGAs are higher performing components with shorter leads and more interconnection pins that allow for higher speeds.

Nomenclature

There are perhaps too many terms that describe different aspects of surface mount technology. Here's what they mean:

SMA (surface-mount assembly) – a build or module assembled using SMT.

SMC (surface-mount components) – components for SMT.

SMD (surface-mount devices) – active, passive, and electromechanical components.

SME (surface-mount equipment) – machines used for SMT.

SMP (surface mount packages) – SMD case forms.

SMT (surface-technology) – the act and method of assembling and mounting electronic technology.

1.5.7 Device Package Types

SOIC

Small outline Integrated Circuit – These are good SMT alternatives to the duel in-line package (DIP), due to their dramatically reduced size. In general, they take up 30 – 50% less space and 70% less thickness than an average DIP.

TSOP

Thin Small Outline Package – TSOPs are low profile packages with fine-pitch leads. TSOPs are typically meant to accommodate large silicon chips in high density packages (RAM or flash memory ICs), largely because of their low volume/high pin count.

QFNs

Quad Flat Pack – QFNs are high lead count packages (44 – 304). Its leads are typically gull wing. There are many kinds of QFNs, and they are one of the most common surface-mount ICs.

PLCC

Plastic Leaded Chip Carrier - Connections are made on all four edges of a square package with a relatively high pin count. PLCCs can have roughly 18 – 100 leads (usually J-leads). Many of them can fit into IC sockets and can be easily replaced in the field. PLCCs have long been a popular option.

LCC

Lead-less Chip Carrier – Not to be confused with PLCC, LCCs have no leads. Rather, LCCs are soldered directly onto PCBs by their (castellation) solder pads. These are usually designed for Mil Spec because, with no leads to damage, they're quite "rugged." LCCs are great for high temperature and aerospace applications.

PGA

Pin Grid Array – PGAs are typically square or rectangular, with pins arranged underneath the package. They're design was highly influential on the now ubiquitous BGA.

Flip Chip

Flip chips are bare die packages, with small bottom-side solder bumps that act as leads. They are soldered directly onto the PCB.

BGA

Ball Grid Array – BGAs are perhaps one of the best performing SMT packages in use today, due to their high densities. The BGA is a descendent of the PGA, yet instead of pins, it has solder balls that can be placed directly onto the PCB. Because of their high density, BGAs are typically used to house microprocessors.

Ball Grid Array BGA uses the underside of the package to place pads with balls of solder in grid pattern as connections to PCB.

FBGA

Fine-pitch ball-grid array. A square or rectangular array of solder balls on one surface

LBGA

Low-profile ball-grid array. Also known as laminate ball-grid array

TEPBGA

Thermally-enhanced plastic ball-grid array

CBGA

Ceramic ball-grid array

OBGA

Organic ball-grid array

TFBGA

Thin fine-pitch ball-grid array

PBGA

Plastic ball-grid array

MAP-BGA

Mold array process - ball-grid array

UCSP

Micro (µ) chip-scale package. Similar to a BGA (A Maxim trademark example)

µBGA

Micro ball-grid array Ball spacing less than 1 mm

LFBGA

Low-profile fine-pitch ball-grid array

TBGA

Thin ball-grid array

SBGA

Super ball-grid array. Above 500 balls

UFBGA

Ultra-fine ball-grid array

1.5.8 Packages

BCC - Bump Chip Carrier

BGA - Ball Grid Array

BGAs are perhaps one of the best performing SMT packages in use today, due to their high densities. The BGA is a descendent of the PGA, yet instead of pins, it has solder balls that can be placed directly onto the PCB. Because of their high density, BGAs are typically used to house microprocessors. Ball Grid Array BGA uses the underside of the package to place pads with balls of solder in grid pattern as connections to PCB.

The BGA is descended from the pin grid array (PGA), which is a package with one face covered (or partly covered) with pins in a grid pattern which, in operation, conduct electrical signals between the integrated circuit and the printed circuit board (PCB) on which it is placed. In a BGA the pins are replaced by pads on the bottom

of the package, each initially with a tiny solder ball stuck to it. These solder spheres can be placed manually or by automated equipment, and are held in place with a tacky flux. The device is placed on a PCB with copper pads in a pattern that matches the solder balls. The assembly is then heated, either in a reflow oven or by an infrared heater, melting the balls. Surface tension causes the molten solder to hold the package in alignment with the circuit board, at the correct separation distance, while the solder cools and solidifies, forming soldered connections between the device and the PCB.

In more advanced technologies, solder balls may be used on both the PCB and the package. Also, in stacked multi-chip modules, solder balls are used to connect two packages.

Advantages

High density

The BGA is a solution to the problem of producing a miniature package for an integrated circuit with many hundreds of pins. Pin grid arrays and dual-in-line surface mount (SOIC) packages were being produced with more and more pins, and with decreasing spacing between the pins, but this was causing difficulties for the soldering process. As package pins got closer together, the danger of accidentally bridging adjacent pins with solder grew.

Heat conduction

A further advantage of BGA packages over packages with discrete leads (i.e. packages with legs) is the lower thermal resistance between the package and the PCB. This allows heat generated by the integrated circuit inside the package to flow more easily to the PCB, preventing the chip from overheating.

Low-inductance leads

The shorter an electrical conductor, the lower its unwanted inductance, a property which causes unwanted distortion of signals in high-speed electronic circuits. BGAs, with their very short distance between the package and the PCB, have low lead inductances, giving them superior electrical performance to pinned devices

Disadvantages

Noncompliant connections

A disadvantage of BGAs is that the solder balls cannot flex in the way that longer leads can, so they are not mechanically compliant. As with all surface mount devices, bending due to a difference in coefficient of thermal expansion between PCB substrate and BGA (thermal stress) or flexing and vibration (mechanical stress) can cause the solder joints to fracture.

Thermal expansion issues can be overcome by matching the mechanical and thermal characteristics of the PCB to those of the package. Typically, plastic BGA devices more closely match PCB thermal characteristics than ceramic devices.

The predominant use of RoHS compliant lead-free solder alloy assemblies has presented some further challenges to BGAs including "head in pillow" soldering phenomenon, "pad cratering" problems as well as their decreased reliability versus lead-based solder BGAs in extreme operating conditions such as high temperature, high thermal shock and high gravitational force environments, in part due to lower ductility of RoHS-compliant solders.

Mechanical stress issues can be overcome by bonding the devices to the board through a process called "underfilling", which injects an epoxy mixture under the device after it is soldered to the PCB, effectively gluing the BGA device to the PCB. There are several types of underfill materials in use with differing properties relative to workability and thermal transfer. An additional advantage of underfill is that it limits tin whisker growth.

Another solution to non-compliant connections is to put a "compliant layer" in the package that allows the balls to physically move in relation to the package. This technique has become standard for packaging DRAMs in BGA packages.

Other techniques for increasing the board-level reliability of packages include use of low-expansion PCBs for ceramic BGA (CBGA) packages, interposers between the package and PCB, and re-packaging a device.

Difficulty of inspection

Once the package is soldered into place, it is difficult to find soldering faults. X-ray machines, industrial CT scanning machines special microscopes, and endoscopes to look underneath the soldered package have been developed to overcome this problem. If a BGA is found to be badly soldered, it can be removed in a rework station, which is a jig fitted with infrared lamp (or hot air), a thermocouple and a vacuum device for lifting the package. The BGA can be replaced with a new one, or it can be refurbished (or reballed) and re-installed on the circuit board. Pre-configured solder balls matching the array pattern can be used to reball BGAs when only one or a few need to be reworked.

Due to the cost of visual X-ray BGA inspection, electrical testing is very often used instead. Very common is boundary scan testing using an IEEE 1149.1 JTAG port.

A cheaper and easier inspection method, albeit destructive, is becoming increasingly popular because it does not require special equipment. Commonly referred to as dye and pry, the process includes immersing the entire PCB or just the BGA attached module into a dye, and after drying, the module is pried off and the broken joins are inspected. If a solder location contains the dye, then it indicates that the connection was imperfect.

Difficulties during circuit development

During development it is not practical to solder BGAs into place, and sockets are used instead, but tend to be unreliable. There are two common types of socket: the more reliable type has spring pins that push up under the balls, although it does not allow using BGAs with the balls removed as the spring pins may be too short.

The less reliable type is a ZIF socket, with spring pinchers that grab the balls. This does not work well, especially if the balls are small.

Cost of equipment

Expensive equipment is required to reliably solder BGA packages; hand-soldering BGA packages is very difficult and unreliable, usable only for the smallest packages in the smallest quantities. However, as more ICs have become available only in leadless (e.g. quad-flat no-leads package) or BGA packages, various DIY reflow methods have been developed using inexpensive heat sources such as heat guns, and domestic toaster ovens and electric skillets.

BQFP - Bumpered Quad Flat Pack

CABGA/SSBGA - Chip Array/Small Scale Ball Grid Array

CBGA - Ceramic Ball Grid Array

CCGA - Ceramic Column Grid Array

CERPACK - Ceramic Package

CFP - Ceramic Flat Pack

CGA - Column Grid Array

CLCC - Ceramic Leadless Chip Carrier Packages

CLGA - Ceramic Land Grid Array

CQFP - Ceramic Quad Flat Pack,

CQGP - Ceramic Quad

CSBGA - Cavity Down BGA

CSOP - Ceramic Small Outline Package

CSP BGA - Chip Scale Package BGA

TBD - Ceramic Lead-Less Chip Carrier

DBS - DIL Bent SIL

DFN - Dual Flat Pack, No Lead

DLCC - Dual Lead-Less Chip Carrier (Ceramic) DLCC Graphic

DMP - Dual In-line Mini Molded Package

DQFN - Depopulated Quad Flat-pack; No-leads

EPTSSOP - Thin Shrink Small Outline Exposed Pad Plastic Packages

ETQFP - Extra Thin Quad Flat Package

FBGA - Fine-pitch Ball Grid Array

FCBGA - Flipchip BGA

FCPBGA - Flip Chip Plastic BGA

FFP - Flip-chip Fine Package

FleXBGA - Flexible Ball Grid Array

FLP - Flat Lead Package

fpBGA - Fine Pitch Ball Grid Array

HBCC - Heatsink Bottom Chip Carrier

HBGA - High Performance Ball Grid Array

HDIP - Heat-dissipating Dual In-line Package

HSBGA - Heat Slug Ball Grid Array

HSOP - Heatsink Small Outline Package

HTSSOP - Heatsink Thin Shrink Small Outline Package

HUQFN - Heatsink Ultra-thin Quad Flat-pack; No-leads

HVQFN - Heatsink Very-thin Quad Flat-pack; No-leads

HVSON - Heatsink Very-thin Small Outline; No-leads

HWQFN - Heatsink Very-Very-thin Quad Flat-pack; No-leads

HWSON - Heatsink Very-Very-thin Small Outline package; No leads

HXQFN - Heatsink eXtremely-thin Quad Flat-pack; No-leads

HXSON - Heatsink eXtremely Small Outline Package; No leads

JDIP - J-Leaded Dual In-Line J-Lead DIP Picture

JLCC - J-Leaded Chip Carrier (Ceramic) J-Lead Picture

LBGA - Low-Profile Ball Grid Array

LCC - Leaded Chip Carrier LCC Graphic

LCC - Leaded Chip Carrier Un-formed LCC Graphic

LCCC - Leaded Ceramic Chip Carrier

LCGA - Low-Profile Ball Grid Array

LFBGA - Low-Profile, Fine-Pitch Ball Grid Array

The Low-Profile Fine Pitch Ball Grid Array, or LFPBGA, is a smaller version of the ball grid array (BGA) package. It is basically an FBGA package that has a package height ranging from 1.2 mm and 1.7 mm. It is therefore thicker than the TFBGA and the VFBGA.

Typical LFBGA's have ball counts that range from 48 to 865 solder balls. The typical LFBGA ball pitch is 0.50 mm to 0.8 mm. A typical LFBGA is about 1.3 mm to 1.7 mm thick.

LGA - Land Grid Array LGA Graphic

LLCC - Leadless Chip Carrier

A leadless chip carrier (LCC or LLCC) is an integrated circuit package that has no pins/leads for contact. This surface-mount device makes use of metal pads at the outer edges to establish connection with the circuit board. Leadless chip carriers are popular, as they are light in weight, adaptable to a wide range of applications and are considered ideal for surface-mount applications.

A leadless chip carrier is usually square or rectangular in shape. Unlike other integrated circuit packaging, leadless chip carriers do not establish connection to devices by means of pins, but by means studs or metal pads provided around the periphery of the package. Due to reduction in weight, area and volume, leadless chip carriers are more durable and could withstand more vibration and shock compared to dual-in-line packages.

One of the salient features of a leadless chip carrier is its easy and convenient direct insertion to its socket mounted on the circuit board. It can also easily mount directly on the board. It is a low-cost solution for surface mounting, as it is lightweight and has no metallic external legs or leads. Unlike dual-inline packages, no holes are required for leadless chip carriers.

There are a few drawbacks associated with leadless chip carriers. The system is not considered homogeneous when leadless chip carriers are mounted to printed circuit

boards. Failures of these systems have been common after limited stressing or thermal expansion. Most of these issues can be resolved by selecting the proper printed circuit board material.

LQFP - Low-profile Quad Flat pack

The Low Profile Quad Flat Pack, or LQFP, is a surface-mount IC package with leads extending from all four sides of the package body.

MCMBGA - Multi Chip Module Ball Grid Array

MCMCABGA - Multi Chip Module-Chip Array Ball Grid Array

MLCC - Micro Leadframe Chip Carrier

MLP - Micro Lead-frame Package MLP graphic

MQFP - Metric Quad Flat Pack (high pin count QFP)

MSOP - Mini Small Outline Plastic Packages

OBGA - Organic Ball Grid Array

ODFN - Optical Dual Flat No-Lead Plastic Package

PBGA - Plastic Ball Grid Array, PBGA graphic

PLCC - Plastic Leaded Chip Carrier

A plastic-leaded chip carrier (PLCC) has a rectangular plastic housing. It is a reduced cost evolution of the ceramic leadless chip carrier (CLCC).

A premolded PLCC was originally released in 1976, but did not see much market adoption. Texas Instruments later released a postmolded variant that was soon adopted by most major semiconductor companies. The JEDEC trade group started a task force in 1981 to categorize PLCCs, with the MO-047 standard released in 1984 for square packages and the MO-052 standard released in 1985 for rectangular packages. The PLCC utilizes a "J"-lead with pin spacings of 0.05" (1.27 mm). The metal strip forming the lead is wrapped around and under the edge of the package, resembling the letter J in cross-section. Lead counts range from 20 to 84. PLCC packages can be square or rectangular. Body widths range from 0.35" to 1.15". The PLCC “J” Lead configuration requires less board space versus equivalent gull leaded components, which have flat leads that extend out perpendicularly to the narrow edge of the package. The PLCC is preferred over DIP style chip carriers when lead counts exceed 40 pins due to the PLCC's more efficient use of board surface area.

POS Package on Substrate

PQFD Plastic Quad Flat --

PQFP Plastic Quad Flat Pack

PSOP Plastic Small-Outline Package PSOP graphic

QFN Quad Flat No-Lead

QFP Quad Flat pack QFP Graphics

QSOP Quarter Size Outline Package

SBGA Super BGA - above 500 Pin count

SDMP Shrink Dual In-line Mini Molded Package

SO Flat Pack - Small Outline Flat Pack IC

SOIC - Small Outline IC

SOJ - Small-Outline Package (J-Lead), SOJ

SOLIC - Small Outline Large Integrated Circuit (Gull-Wing Lead Wide Body)

SON - Small-Outline No-leads (leadless package)

SOP - Small Out-line Package

SSOP - Shrink Small-Outline Package

SOT - Small Outline Transistor Plastic Package

TBGA - Tape Ball Grid Array

TBGA - Thin Ball Grid Array

TDFN - Thin Dual Flat No-Lead Plastic Package

TEPBGA - Thermally Enhanced Plastic Ball Grid Array

TFBGA - Thin profile Fine-pitch Ball Grid Array

TQFN - Thin Quad Flat No-Lead Plastic Package

TQFP - Thin Quad Flat Pack TQFP Graphic

TSOP - Thin Small-Outline Package

TSSOP - Thin Shrink Small-Outline Package

TSOT - Thin Small Outline Transistor Plastic Package

TVSOP - Thin Very Small-Outline Package

TVSP - Thin Very Small Package

UFBGA - Ultra Fine-Line BGA

UTDFN - Ultra Thin Dual Flat No-Lead Plastic Package

VFBGA - Very thin Fine-pitch Ball Grid Array

VQFB - Very-thin Quad Flat Pack

VSO - Very Small Outline

VSSOP - Very thin Shrink Small Outline Package

VSP - Very Small Package

XQFN - eXtremely thin Quad Flat package; No leads

XSON - eXtremely thin Small Outline package; No leads

1.5.9 Solder

Solder is a fusible metal alloy used to create a permanent bond between metal workpieces. The word solder comes from the Middle English word soudur, via Old French soldure and soulder, from the Latin solidare, meaning "to make solid". In fact, solder must first be melted in order to adhere to and connect the pieces together after cooling, which requires that an alloy suitable for use as solder have a lower melting point than the pieces being joined. The solder should also be resistant to oxidative and corrosive effects that would degrade the joint over time. Solder used in making electrical connections also needs to have favorable electrical characteristics.

Soft solder typically has a melting point range of 90 to 450 °C , and is commonly used in electronics. Alloys that melt between 180 and 190 °C are the most commonly used. Soldering performed using alloys with a melting point above 450 °C is called "hard soldering", "silver soldering", or brazing.

In specific proportions, some alloys can become eutectic — that is, the alloy's melting point is lower than that of either component. Non-eutectic alloys have markedly different solidus and liquidus temperatures, and within that range they exist as a paste of solid particles in a melt of the lower-melting phase. In electrical work, if the joint is disturbed in the pasty state before it has solidified totally, a poor electrical connection may result; use of eutectic solder reduces this problem. The pasty state of a non-eutectic solder can be exploited in plumbing, as it allows molding of the solder during cooling, e.g. for ensuring watertight joint of pipes, resulting in a so-called "wiped joint".

For electrical and electronics work, solder wire is available in a range of thicknesses for hand-soldering (manual soldering is performed using a soldering iron or soldering gun), and with cores containing flux. It is also available as a paste, as a preformed foil shaped to match the workpiece, more suitable for mechanized mass-production, or in small "tabs" that can be wrapped around the joint and melted with a flame, for field repairs where an iron isn't usable or available. Alloys of lead and tin were commonly used in the past and are still available; they are particularly convenient for hand-soldering. Lead-free solders have been increasing in use due to regulatory requirements plus the health and environmental benefits of avoiding lead-based electronic components. They are almost exclusively used today in consumer electronics.

Lead-free solder

On July 1, 2006 the European Union Waste Electrical and Electronic Equipment Directive (WEEE) and Restriction of Hazardous Substances Directive (RoHS) came into effect, restricting the inclusion of lead in most consumer electronics sold in the EU, and having a broad effect on consumer electronics sold worldwide. In the US, manufacturers may receive tax benefits by reducing the use of lead-based solder. Lead-free solders in commercial use may contain tin, copper, silver, bismuth, indium, zinc, antimony, and traces of other metals. Most lead-free replacements for conventional 60/40 and 63/37 Sn-Pb solder have melting points from 50 to 200 °C higher, though there are also solders with much lower melting points.

It may be desirable to use minor modification of the solder pots (e.g., titanium liners or impellers) used in wave-soldering, to reduce maintenance cost due to increased tin-scavenging of high-tin solder.

Lead-free solder may be less desirable for critical applications, such as aerospace and medical projects, because its properties are less thoroughly known.

Tin-silver-copper (Sn-Ag-Cu, or "SAC") solders are used by two-thirds of Japanese manufacturers for reflow and wave soldering, and by about 75% of companies for hand soldering. The widespread use of this popular lead-free solder alloy family is

based on the reduced melting point of the Sn-Ag-Cu ternary eutectic behavior (217 ° C), which is below the 22/78 Sn-Ag (wt.%) eutectic of 221 °C and the 59/41 Sn-Cu eutectic of 227 °C. The ternary eutectic behavior of Sn-Ag-Cu and its application for electronics assembly was discovered (and patented) by a team of researchers from Ames Laboratory, Iowa State University, and from Sandia National Laboratories-Albuquerque.

Much recent research has focused on selection of 4th element additions to Sn-Ag-Cu to provide compatibility for the reduced cooling rate of solder sphere reflow for assembly of ball grid arrays, e.g., 18/64/14/4 tin-silver-copper-zinc (Sn-Ag-Cu-Zn) (melting range 217–220 °C) and 18/64/16/2 tin-silver-copper-manganese (Sn-Ag-Cu-Mn) (melting range of 211–215 °C).

Tin-based solders readily dissolve gold, forming brittle intermetallics; for Sn-Pb alloys the critical concentration of gold to embrittle the joint is about 4%. Indium-rich solders (usually indium-lead) are more suitable for soldering thicker gold layer as the dissolution rate of gold in indium is much slower. Tin-rich solders also readily dissolve silver; for soldering silver metallization or surfaces, alloys with addition of silvers are suitable; tin-free alloys are also a choice, though their wettability is poorer. If the soldering time is long enough to form the intermetallics, the tin surface of a joint soldered to gold is very dull.

Flux

Flux is a reducing agent designed to help reduce (return oxidized metals to their metallic state) metal oxides at the points of contact to improve the electrical connection and mechanical strength. The two principal types of flux are acid flux (sometimes called "active flux"), containing strong acids, used for metal mending and plumbing, and rosin flux (sometimes called "passive flux"), used in electronics. Rosin flux comes in a variety of "activities", corresponding roughly to the speed and effectiveness of the organic acid components of the rosin in dissolving metallic surface oxides, and consequently the corrosiveness of the flux residue.

Due to concerns over atmospheric pollution and hazardous waste disposal, the electronics industry has been gradually shifting from rosin flux to water-soluble flux, which can be removed with deionized water and detergent, instead of hydrocarbon solvents.

In contrast to using traditional bars or coiled wires of all-metal solder and manually applying flux to the parts being joined, much hand soldering since the mid-20th century has used flux-core solder. This is manufactured as a coiled wire of solder, with one or more continuous bodies of inorganic acid or rosin flux embedded lengthwise inside it. As the solder melts onto the joint, it frees the flux and releases that on it as well.

1.5.10 Disaster Recovery

Sadly it is sometimes the nature of Windows that AutoTRAX DEX and other programs do not work.

If AutoTRAX DEX does not start, a quick fix can be to delete the settings.

First click 

in the Ribbon menu Panels->Panels command group. This will restore all the panel settings.

If this does not work:

Try deleting...

C:\Users\XXX\AppData\Roaming\AutoTRAX DEX Software\AutoTRAX DEX\Settings

where **XXX** is you.

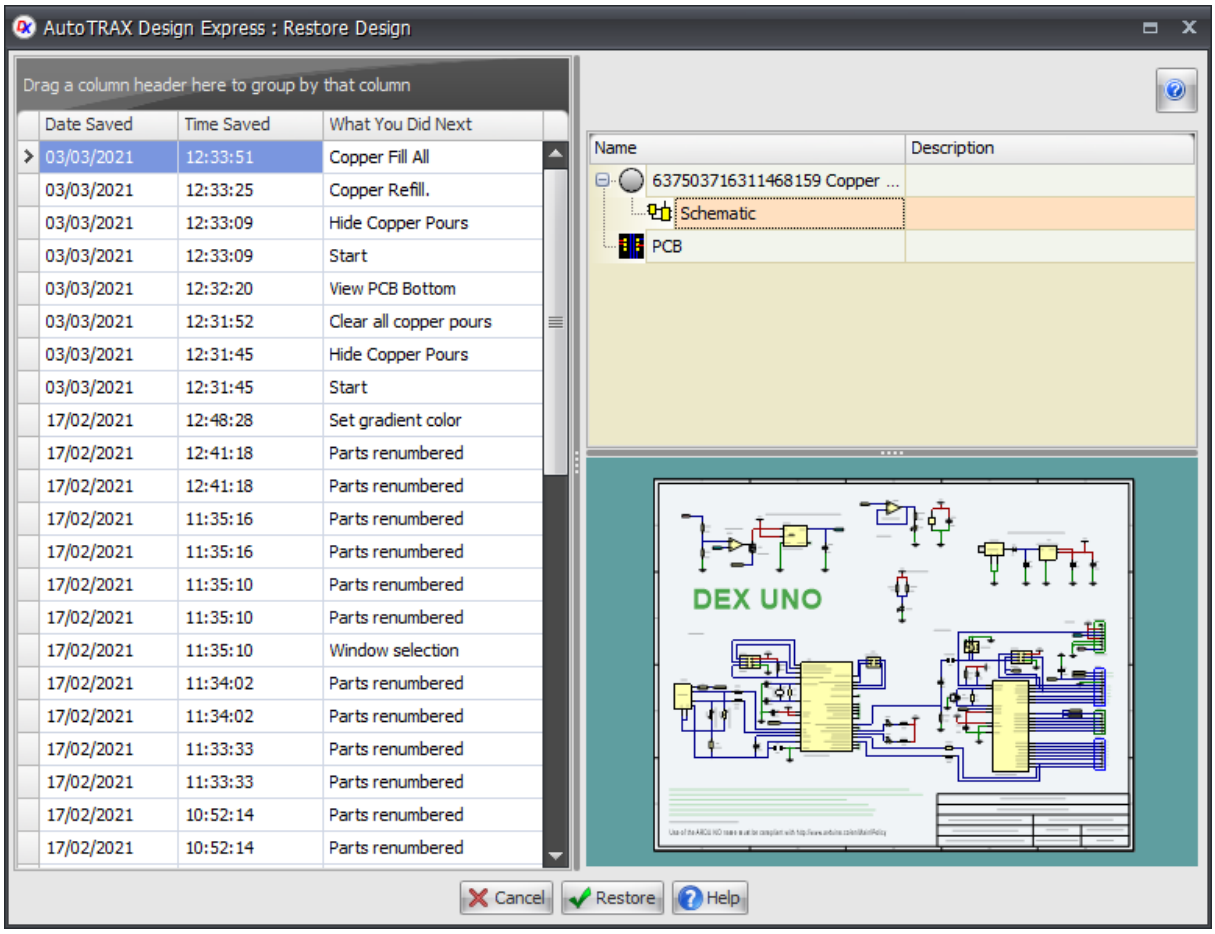
1. Next, make sure you have the latest graphics card driver. See Update the Graphics Card Driver
2. If it still is not working Manually Removing AutoTRAX DEX and then install it from https://pcbdex.com/Download
3. If it still is not working Reinstall .NET
4. If it is still not working use Restore Points
5. If things are still not working I suggest trying a different graphics card.
6. Finally if things are still not working, backup all your data and repair Windows using the Windows DVD that came with your machine.

1.5.10.1 Restoring Previous Designs

By default AutoTRAX DEX saves a copy of EVERY change you make to your project. These are saves in a directory called XXX.backup where XXX is the full path name of your project.

To restore to any of the click the Home→Open→ 

button. The Restore Design dialog show below will appear.



The Restore Design dialog

Click on any of the rows in the left grid to view the project. The schematic (first page) and the PCB will be displayed in the viewports on the right.

Date Saved The date the design was saved

. Time Saved The time the design was saved.

What You Did Next The action that occurred after the design was saved.

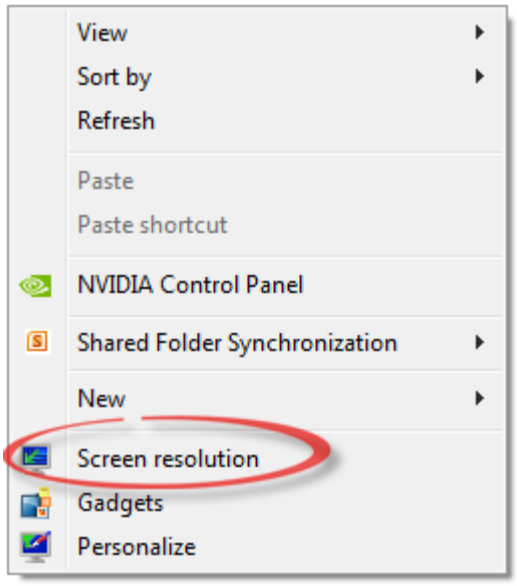
Click Cancel to cancel/close the dialog.

Click Restore to restore the selected design state.

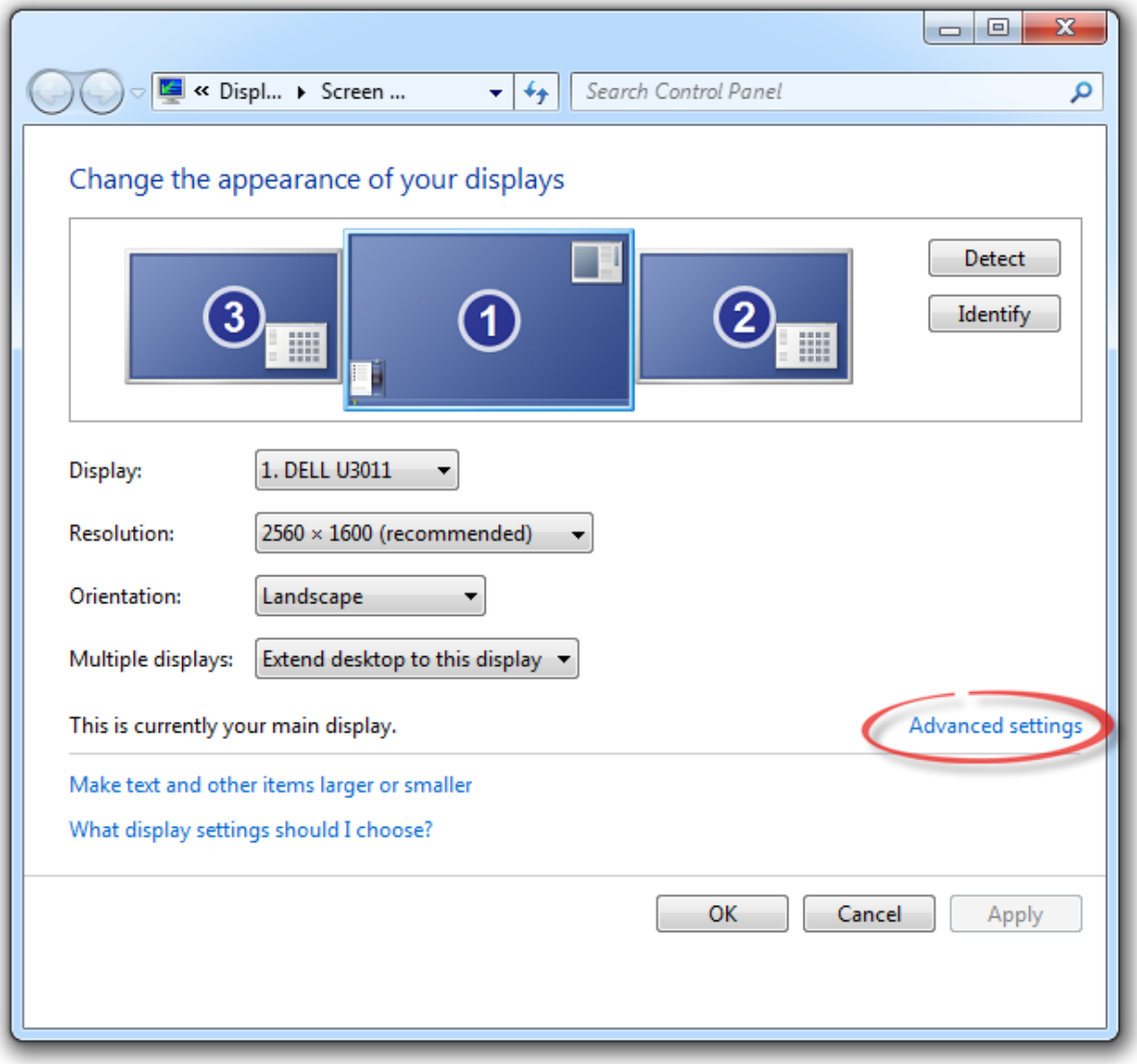
Click Help to display this help topic.

1.5.10.2 Update the Graphics Card Driver

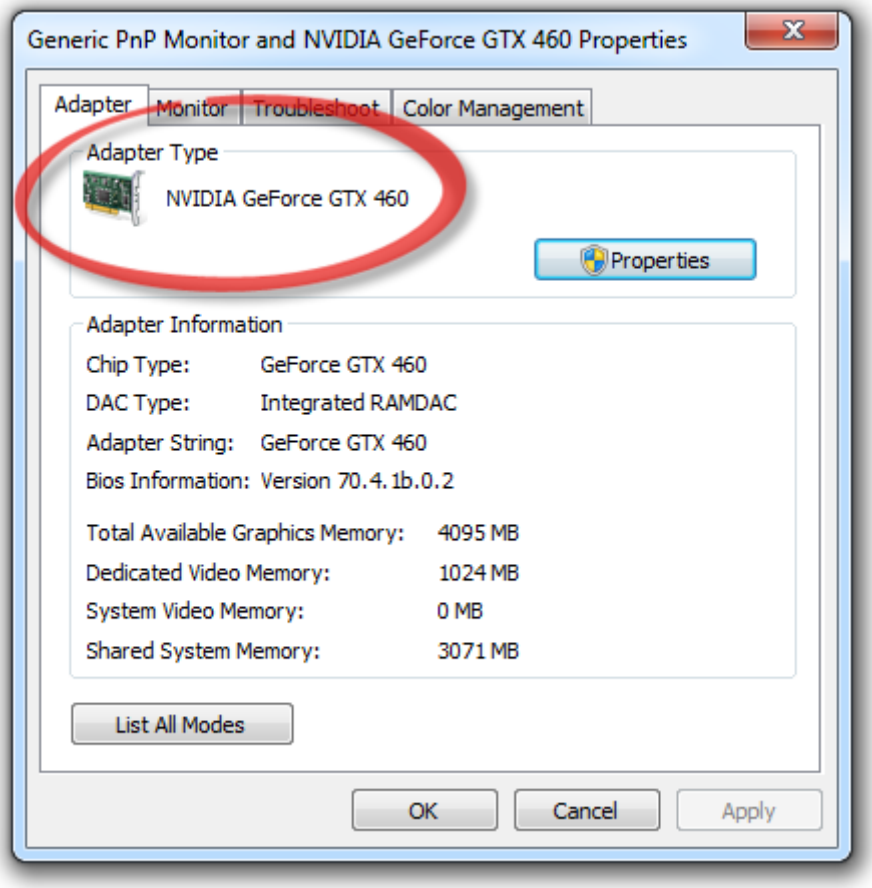
To discover what your graphics card, right click on a blank area of the desktop.



Click the Screen Resolution menu item.



Click the Advanced Settings. You should see the device type as shown below.

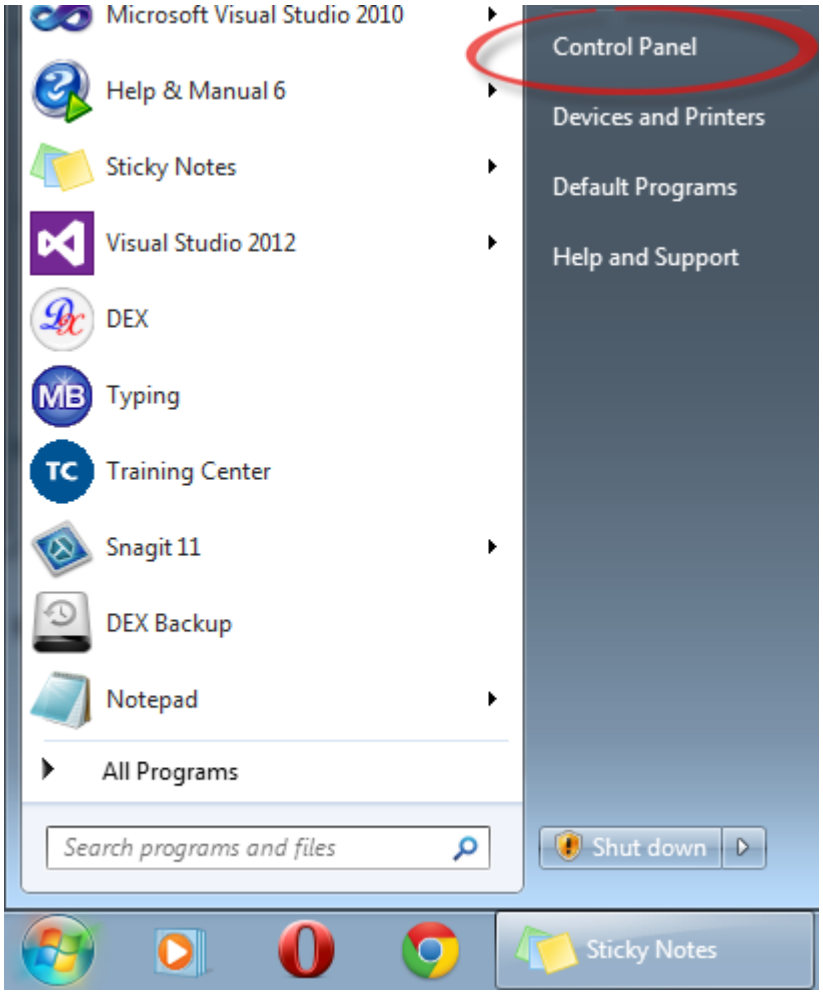


Now Google for the device adding the word driver. Download and install the driver. Hopefully this will fix the problem.

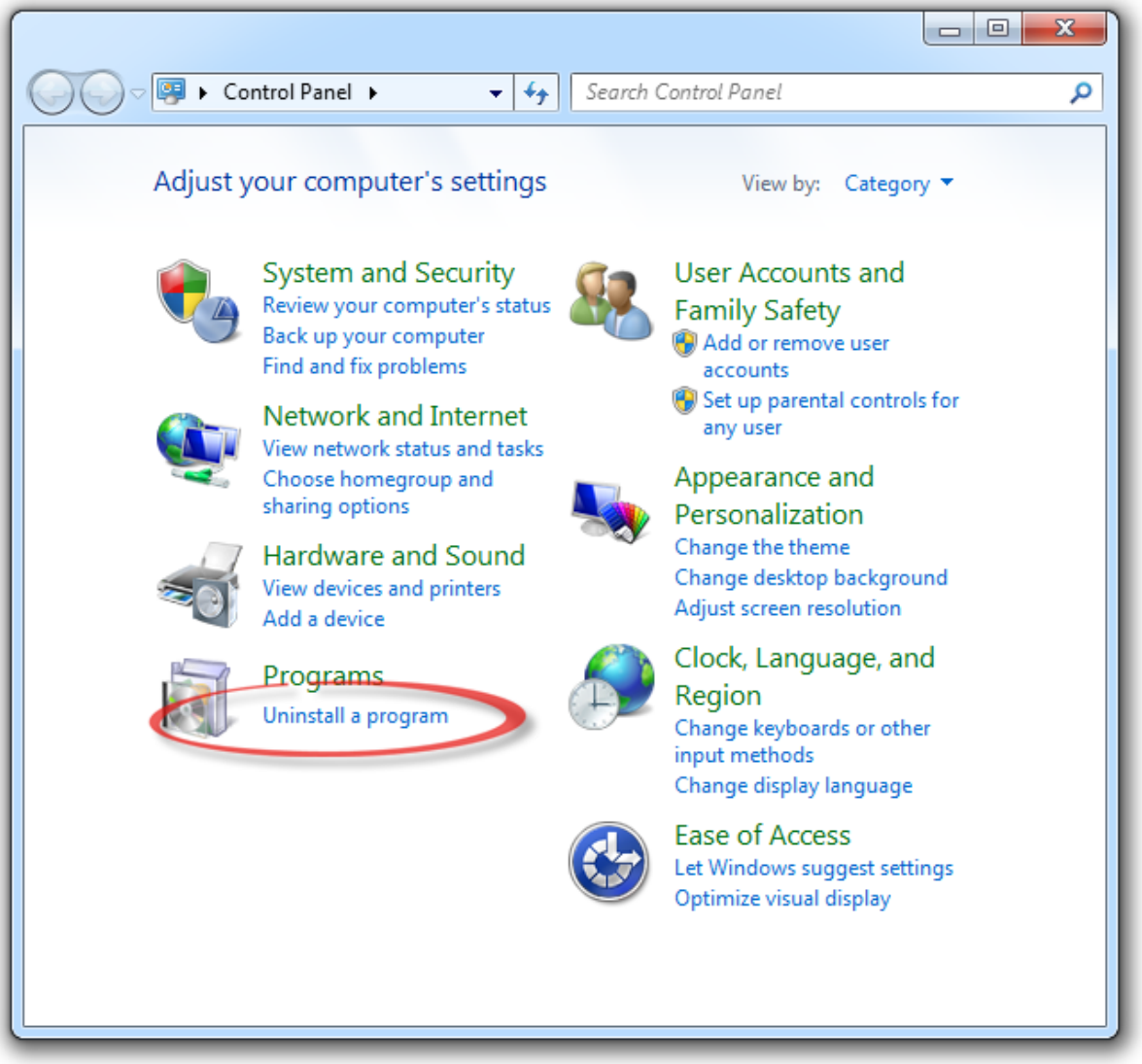
1.5.10.3 Reinstall .NET

First uninstall .NET

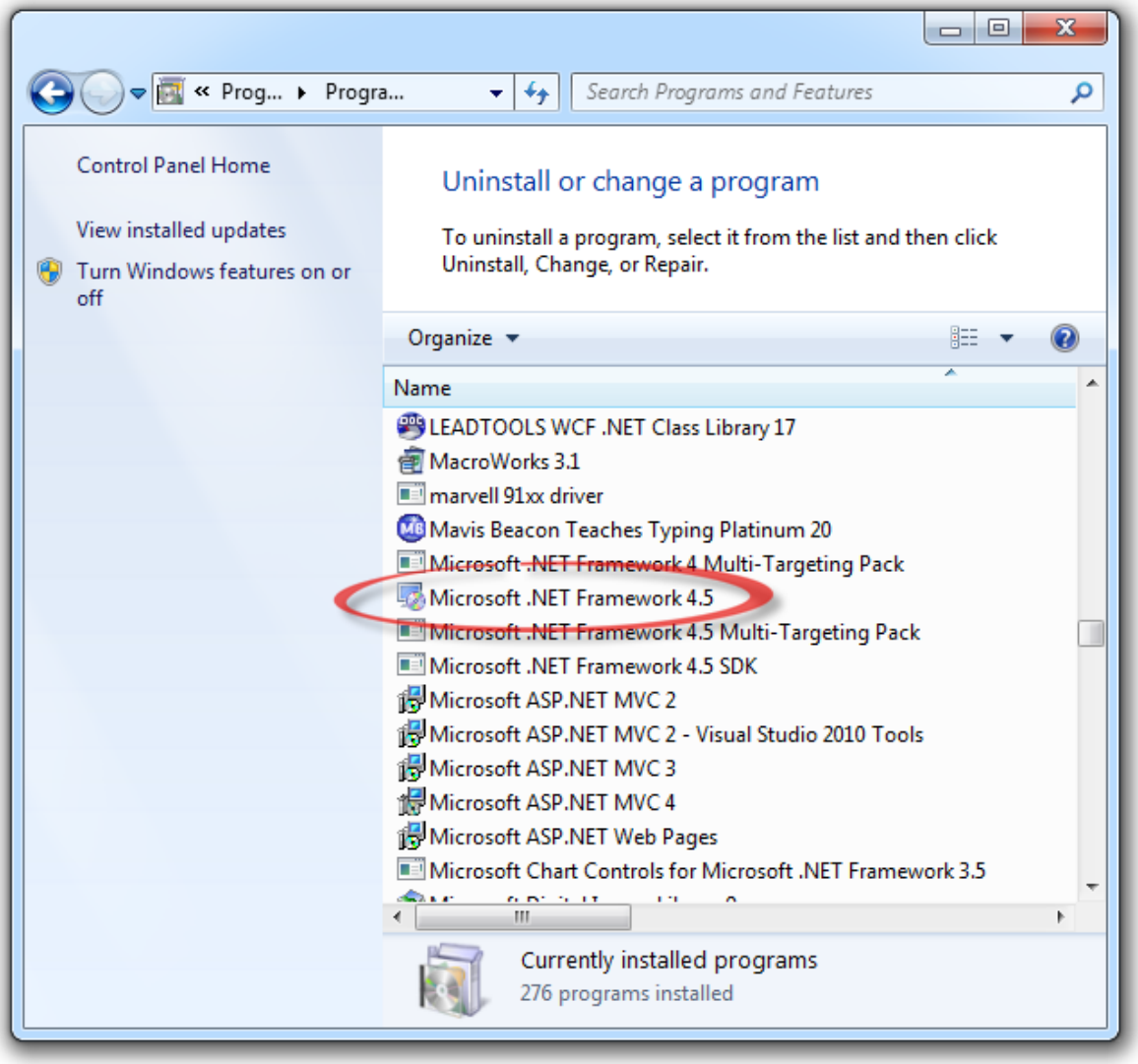
Click on the Control Panel in the Start Menu.



Click on Uninstall a program



Click on the .NET version to remove.



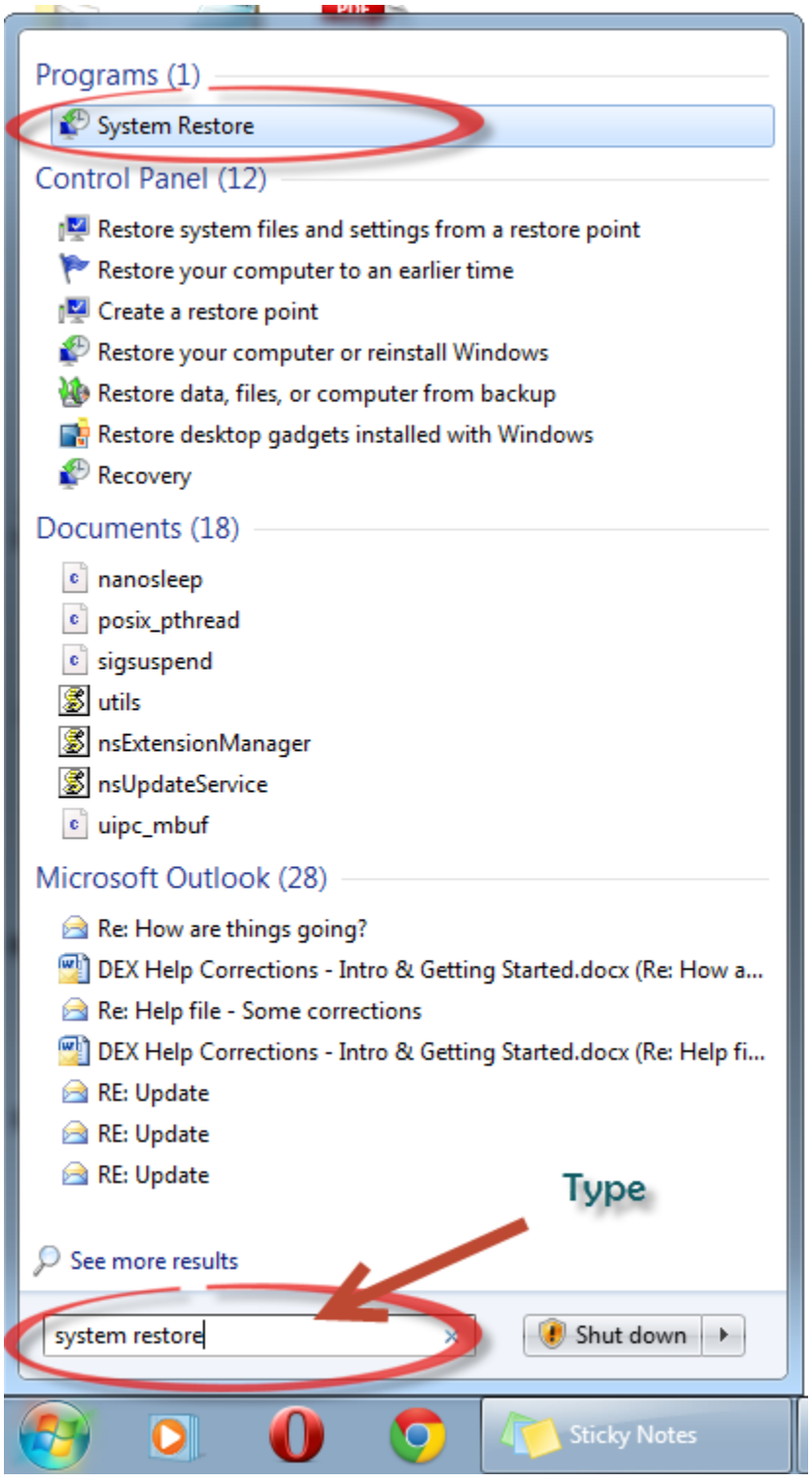
Install the latest .NET from https://msdn.microsoft.com/en-us/library/5a4x27ek.aspx

Hopefully this will fix the problem.

1.5.10.4 Restore Points

Sometimes installing a program or driver can make Windows run slowly or unpredictably. System Restore can return your PC's system files and programs to a time when everything was working fine, potentially preventing hours of troubleshooting headaches. It won't affect your documents, pictures, or other data. Read more https://windows.microsoft.com/en-gb/windows7/products/features/system-restore

Type in 'system restore' in the start menu and click on **System Restore.**



1.5.10.5 Preventing Disasters

AutoTRAX DEX saves every change you make to you design from the moment you first save it. See Restoring Previous Designs

In addition you should backup your Parts Library.

I strongly advise you to use third party software to automatically backup your .backup directory and all your .project files. See Free Backup Software

1.5.11 Scripting

You can write embedded Python scripts to extend the functionality of AutoTRAX

AutoTRAX DEX has it's own scripting engine based on the Python programming language. (IronPython).

With embedded Python scripts in AutoTRAX, you can automate tasks, manipulate designs, and access external data sources. Some common tasks that can be automated include component placement, routing, and design rule checking. Python scripts can also be used to interface with external software or databases to import or export data.

To use Python scripts in AutoTRAX, you can create a new Python script file or open an existing one using the built-in Python editor. You can then write your code in the editor and execute it from within AutoTRAX.

It's important to note that using Python scripts in AutoTRAX requires some programming knowledge and understanding of the software's scripting API. You may want to consult the AutoTRAX documentation or seek help from the software's community forums to get started.

See the language reference.

You can create your own command plug-ins that can be linked to a menu button.

A command plug-in consists of:

IronPython Script

This is a Python program that is run on demand by clicking the commands button in the menu or by clicking on the Run in the scripting panel.

AutoTRAX DEX exposes a large API that you can call from your IronPython program.

In addition IronPython exposes the complete Microsoft .NET API for you to use.

Button Icons

You can optionally add 2 buttons icons to be displayed in the menu button. You can set a 16x16 pixel small icon and a 32x32 pixel large icon.

Button Caption

You can optionally add a text caption for the button.

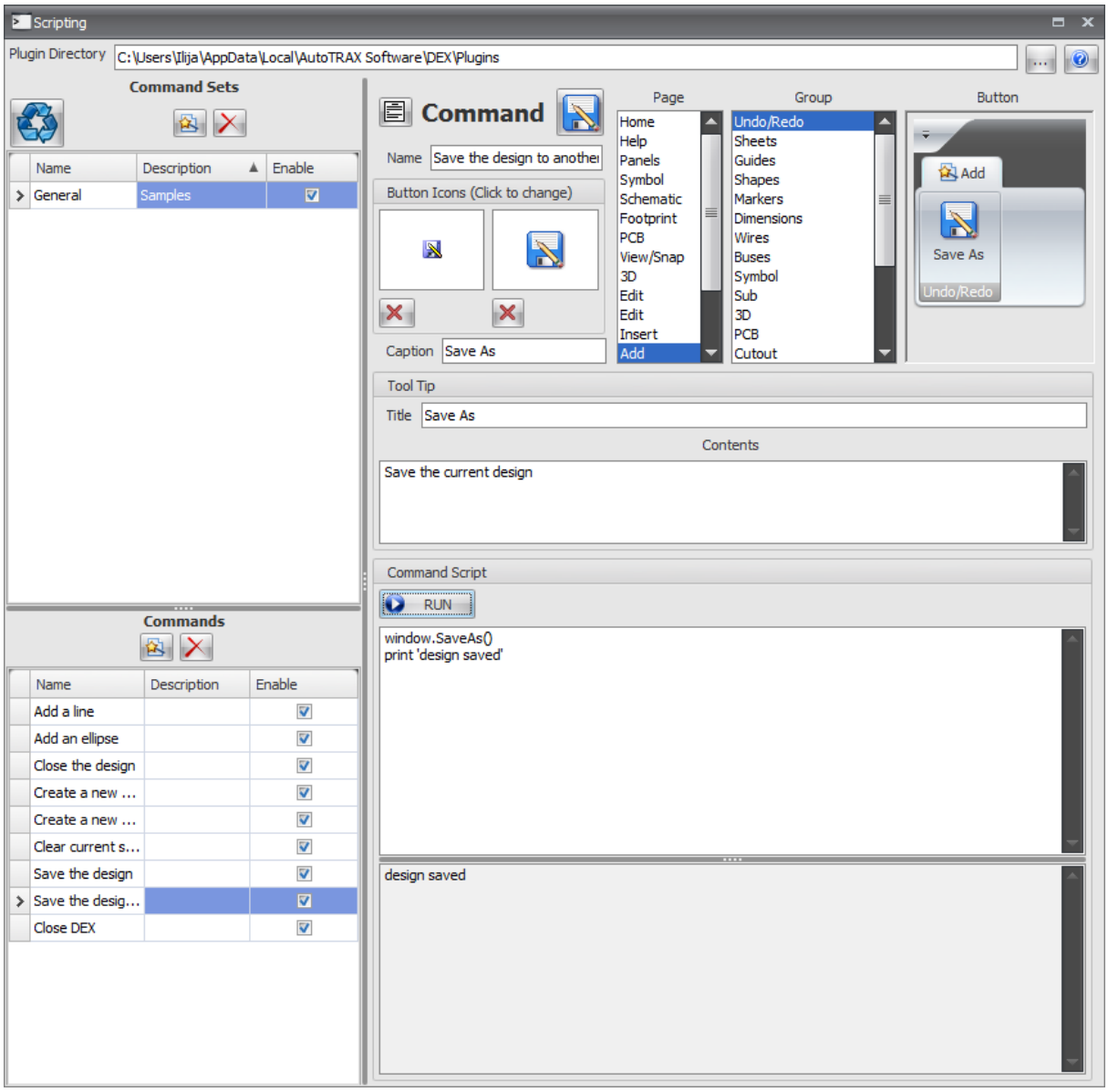
ToolTip

You can optionally add a toolTip to be displayed when the mouse is moved over the button. This gives you information on the command. The toolTip has a title and text content.

If you enable a command plug-in, then it will be added to the menu as a button.

1.5.11.1 The Scripting Panel

The scripting panel allows you to create command and command buttons to add to the AutoTRAX DEX menu. It is shown below.

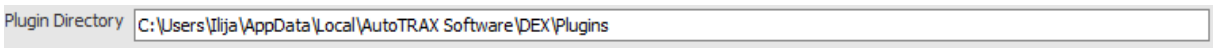


The Scripting Panel

Help

Click to see this help topic.

Plug-in Directory



This is the directory that the command plug-in collection files are kept. Each file is an XML text file with a .xml file extension. This allows you to view and edit them in third part XML programs.

 Click to browse to set a directory.

Command Sets

Command plug-ins are contained in a plug-in collection files. You can have any number of plug-in collection files in the plug-ins directory. When AutoTRAX DEX starts, it loads all plug-in collection files found and adds menu items for each command plug-in the file if the collection is enabled.

 Click to update the menu with the commands.

 Click to create a new empty command plug-in collection file.

 Click to delete the selected command plug-in collection file. It will be moved to the recycle bin.

The grid displays all collection files loaded on startup.

Name is the name of the collection file (less the .xml extension)

Description is a brief description of what the collection contains.

Enable - if checked then the commands in the collection file will be added to the menu as buttons subject to the individual commands being enabled (see below).

Commands

This lists the command plug-ins found in the selected collection file.

 Click to create a new empty command plug-in.

 Click to delete the selected command plug-in. Once deleted, it s gone forever. You will be prompted before deletion occurs. So please keep a backup.

Name is the name of the command plug-in.

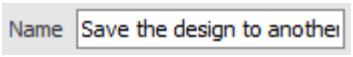
Description is a brief description of the command..

Enable - if checked then the command will be added to the menu as a buttons subject to it's collection file being enable (see above).

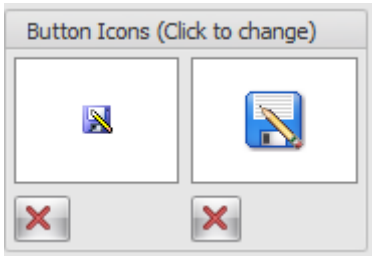
Command

 Click to view the AutoTRAX DEX API.

 Click to save the command. This will only be displayed if you have made any changes and the command has not been saved.

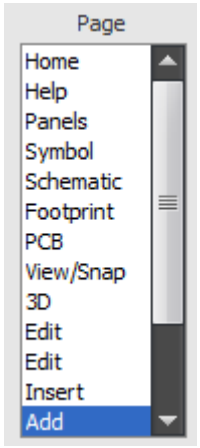


This is the name of the command.

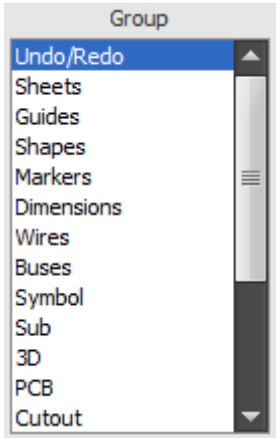


Click either of the icon areas to set the small or large icon. The small icon but be 16x16 pixels and the large icon must be 32x32 pixels.

 Click to remove the icon. You can have buttons with no icons and only the text will be displayed.



Selects the Ribbon Page/Tab that the button will be added to.



Selects the button group that the button will be added to.

Tooltip

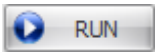
This is the title for the tooltip.

This is the main tooltip text.

This is a preview of what the button will look like. Hover the mouse over it to see the tooltip.

Command Script

This is the IronPython program script for the command. See the API for more details.



Click to run the program script.

```
window.SaveAs()
print 'design saved'
```

This is the program script.

This is the output after running the script. It will also list any errors found.

1.5.11.2 The DEX API

AutoTRAX DEX create the following global reset variables for you.

window

This is a Window object that lets you access functionality on the main window of AutoTRAX DEX.

design

This is the Design being edited.

app

Various useful methods.

1.5.11.2.1 The Window Variable

The window variable give you access to the AutoTRAX DEX main window commands.

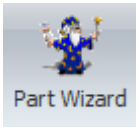
Useage

Call method on the window variable e.g.

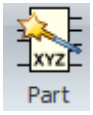
```
window.CreateNewPart()
```

Methods (Loads more to come in the next few days)

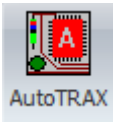
CreateNewProject - create a new project



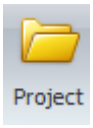
ShowPartWizard - Start the part wizard.



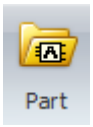
CreateNewPart - create a new part



ImportDEXFile - read an AutoTRAX DEX EDA file.

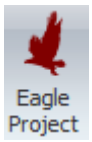


OpenProject - open an existing project

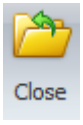


OpenPart - open an existing part

RecoverDesign - display the recover design dialog



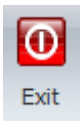
OpenEagle - open an existing Eagle schematic/board file, combing them both.



Close - close the current design.

Save - save the current design

SaveAs - save the design to a different file name



Exit - close AutoTRAX DEX

1.5.11.2.2 The Design Variable

The design variable give you access to the current design data.

Properties (Loads more to come in the next few days)

bool	**IsProject**	True if the design is a project. (read only)
bool	**IsPart**	True if the design is a part. (read only)
bool	**IsArt**	True if the design is artwork. (read only)
bool	**Modified**	True if the design has been modified but not saved. (read only)
bool	**NeverBeenSaved**	True if the design has never been saved (A new design). (read only)
string	**`DocumentName`**	Name of design document. Includes full path and extension
string	**`ShortName`**	The name of the design document without path and with the extension

Useage

```
if  design.IsProject:
     print 'The design is a project'
else:
     print 'The design is not a project'
```

Methods (Loads more to come in the next few days)

none	**Save**	Save the design.
none	**AddLine (startX, startY,edX,endY)**	Add a line
List<Symbol>	**Symbols()**	List of symbols in the design

Useage

Call method on the window variable e.g.

```
design.AddLine(1,2,5,6 )
```

1.5.11.2.2.1 Design Structure

Coming soon...

Coming soon...

Symbols

Nodes

The Symbol Reference

The Symbol Value

Coming soon...

Coming soon...

Coming soon...

Built in variable PCB

Properties

Type	Name	Description
IEnumerable<Point> Border	`Border`	The PCB border
float	Thickness	The thickness of the PCB

The Footprint Reference

The Footprint Value

Coming soon...

Coming soon...

Coming soon...

Lines

Rectangles

Ellipses and Circles

Text

Images

Coming soon...

Coming soon...

Coming soon...

Coming soon...

Coming soon...

Coming soon...

1.5.11.2.3 The ui Variable

The **ui** variable give you some useful user interface functions.

Methods (Loads more to come in the next few days)

none	**ShowFile**	Display the contents of a file

Useage

Call method on the window variable e.g.

```
ui.ShowFile( '\myFileName.txt' )
```

1.5.11.2.4 Built-in variables

Coming soon...

1.5.11.2.4.1 pcb

To create a Rectangular PCB

```
pcb.MakeRectangular( centerX = 1, centerY = 2, width = 3, height = 4 )
```

To create an Elliptical PCB

```
pcb.MakeElliptical( centerX = 1, centerY = 2, width = 3, height = 4 )
```

To create a Polygonal PCB

```
width = 5
height = 4
profile = [
        Point( 0, 0 ),
        Point( 0, height ),
        Point( width/2.0, height*.15 ),
        Point( width, height ),
        Point( width,0 ) ]
pcb.MakePolygonal( profile )
```

Setting the Color of the PCB

```
pcb.Color = Color.DarkGreen
```

```
pcb.Color = Color.FromArgb( red,green,blue ) #  0 <= red,green,blue <= 25
```

pcb.Color = Color.FromArgb(alpha, red,green,blue) # 0 <= alpha, red,green,blue <= 255. alpha will give you transparency 255 is opaque

Setting the Thickness of the PCB

To set the thickness of the PCB.

```
pcb.Thickness = 0.01
```

Removing All Holes and Cutouts

To set the thickness of the PCB.

```
pcb.RemoveAllHolesAndCutouts()
```

Adding Automatic Mounting Holes (Projects only)

```
pcb.AddMountingHoles = True
```

Setting the Size of Automatic Mounting Holes (Projects only)

```
pcb.MountingHoleDiameter = 0.1
```

To add a Rectangular Cutout

```
pcb.AddRectangularCutout( centerX = 1, centerY = 2, width = 3, height = 4
```

To add an Elliptical Cutout

```
pcb.AddEllipticalCutout( centerX = 1, centerY = 2, width = 3, height = 4
```

To add a Polygonal Cutout

```
width = 5
height = 4
profile = [
        Point( 0, 0 ),
        Point( 0, height ),
        Point( width/2.0, height*.15 ),
        Point( width, height ),
        Point( width,0 ) ]
pcb.AddPolygonalCutout( profile )
```

To create a Regular Polygonal PCB

```
pcb.MakePolygonal( geom.Polygon( sides = 8, centerX = 4, centerY = 4, radi
```

1.5.11.3 Python

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The one new syntax component in Python is the colon character (:). All Python compound statements—statements that have other statements nested inside them—follow the same general pattern of a header line terminated in a colon, followed by a nested block of code usually indented underneath the header line

Header line:

    Nested statement block

The colon at the end of the header line is required.

You don't need to terminate statements with semicolons in Python the way you do in C-like languages. The general rule is that the end of a line automatically terminates the statement that appears on that line.

You do not type anything explicit in your code to mark the beginning and end of a nested block of code. Instead, you need to consistently indent all the statements in a given single nested block the same distance to the right, and Python uses the statements' physical indentation to determine where the block starts and stops. e.g.

```
if x > y:
     x = 1
     y = 2

```

Python doesn't care how you indent (you may use either spaces or tabs), or how much you indent (you may use any number of spaces or tabs).

```

```

Control Statement

1.5.11.3.1 An Informal Introduction to Python

In the following examples, input and output are distinguished by the presence or absence of prompts (>>> and ...): to repeat the example, you must type everything after the prompt, when the prompt appears; lines that do not begin with a prompt are output from the interpreter. Note that a secondary prompt on a line by itself in an example means you must type a blank line; this is used to end a multi-line command.

Many of the examples in this manual, even those entered at the interactive prompt, include comments. Comments in Python start with the hash character, #, and extend

to the end of the physical line. A comment may appear at the start of a line or following whitespace or code, but not within a string literal. A hash character within a string literal is just a hash character. Since comments are to clarify code and are not interpreted by Python, they may be omitted when typing in examples.

Some examples:

```
# this is the first comment
spam = 1  # and this is the second comment
          # ... and now a third!
text = "# This is not a comment because it's inside quotes."
```

Using Python as a Calculator

Let's try some simple Python commands. Start the interpreter and wait for the primary prompt, >>>. (It shouldn't take long.)

Numbers

The interpreter acts as a simple calculator: you can type an expression at it and it will write the value. Expression syntax is straightforward: the operators +, -, * and / work just like in most other languages (for example, Pascal or C); parentheses (()) can be used for grouping. For example:

```
>>>
>>> 2 + 2
4
>>> 50 - 5*6
20
>>> (50 - 5.0*6) / 4
5.0
>>> 8 / 5.0
1.6
```

The integer numbers (e.g. 2, 4, 20) have type int, the ones with a fractional part (e.g. 5.0, 1.6) have type float. We will see more about numeric types later in the tutorial.

The return type of a division (/) operation depends on its operands. If both operands are of type int, floor division is performed and an int is returned. If either operand is a float, classic division is performed and a float is returned. The // operator is also provided for doing floor division no matter what the operands are. The remainder can be calculated with the % operator:

```
>>>
>>> 17 / 3  # int / int -> int
5
>>> 17 / 3.0  # int / float -> float
5.666666666666667
>>> 17 // 3.0  # explicit floor division discards the fractional part
5.0
>>> 17 % 3  # the % operator returns the remainder of the division
2
>>> 5 * 3 + 2  # result * divisor + remainder
17
```

With Python, it is possible to use the ** operator to calculate powers [1]:

1.5.11.3.2 Comments

Comments begin with the # symbol.

hash-mark comments are the most basic way to document your code. Python simply ignores all the text following a # (as long as it's not inside a string literal), so you can follow this character with any words and descriptions meaningful to you.

comments are best limited to smaller code documentation (e.g., "make sure file exists before opening it") and are best limited in scope to a statement or small group of statements within a script or function.

1.5.11.3.3 Control Statement

The If Statement

The For Statement

The While Statement

1.5.11.3.3.1 The If Statement

Conditional Execution

The **if**, **else**, and **elif** statements control conditional code execution.The general format of a conditional statement is as follows:

```
if design.IsProject:
     print 'The design is a project'
elif design.IsPart:
     print 'The design is a part
elif design.IsArt:
     print 'The design is an artwork file'
else:
```

```
     print 'The design type is unknown'
```

There can be zero or more **elif** parts, and the **else** part is optional. The keyword 'elif' is short for 'else if', and is useful to avoid excessive indentation.

An if ... elif ... elif ... sequence is a substitute for the switch or case statements found in other languages.

1.5.11.3.3.2 Loops and Iterations

You implement loops using the for and while statements.

The For Statement

The While Statement

1.5.11.3.3.3 The For Statement

The for statement iterates over all the elements of s until no more elements are available.The for statement works with any object that supports iteration.

```
for part in design.Parts:
     statements
```

1.5.11.3.3.4 The While Statement

The while statement executes statements until the associated expression evaluates to false.

```
i = 10
while i > 0:
     print i
     i = i-1
```

1.5.11.3.4 Functions

A function is a device that groups a set of statements so they can be run more than once in a program. Coding an operation as a function makes it a generally useful tool, which we can use in a variety of contexts.

```
def name(arg1, arg2,... argN):
     statements
```

Functions can have an optional return statement.

```
def name(arg1, arg2,... argN):
     ...
```

```
     return value
```

Example

Defining the function

```
# Get file name to save to
def GetFileFile():
     dialog = SaveFileDialog()
     filter = 'Text files (*.txt)|*.txt|All files (*.*)|*.*'
     dialog.Filter = filter
     dialog.Title = 'Save Parts'
     dialog.FileName = design.ShortName
     result = dialog.ShowDialog()
     if result == DialogResult.OK:
          return dialog.FileName
     else:
          return None
```

Calling the function

```
fileName = GetFileFile()
```

1.5.11.4 CookBook

Here are some sample IronPython programs.

Save Parts List

saves the parts details to a file.

1.5.11.4.1 Save Parts List

This program saves the parts details to a file.

```
# write all symbol references and values to a file

from System.Windows.Forms import SaveFileDialog, DialogResult
from System.IO import StreamWriter, File

# Get file name to save to
def GetFileFile():
     dialog = SaveFileDialog()
     filter = 'Text files (*.txt)|*.txt|All files (*.*)|*.*'
     dialog.Filter = filter
     dialog.Title = 'Save Parts'
     dialog.FileName = design.ShortName
     result = dialog.ShowDialog()
```

```
     if result == DialogResult.OK:
          return dialog.FileName
     else:
          return None

# Save part details to a file
def SaveParts( fileName ):
     writer = File.CreateText( fileName )
     writer.WriteLine( 'Parts List' )
     writer.WriteLine()
     for part in design.Symbols():
          writer.WriteLine( part.SymbolReference.TextValue +  ' ' +  part
     writer.Close()

# Program

fileName = GetFileFile()
if fileName is not None:
     SaveParts( fileName )
     # display the output
     app.ViewFile( fileName )
```

1.5.11.5 .NET Integration

IronPython aims to be a fully compatible implementation of the Python language. At the same time, the value of a separate implementation than CPython is to make available the .NET ecosystem of libraries. IronPython does this by exposing .NET concepts as Python entities. Existing Python syntax and new Python libraries (like CLR) are used to make .NET features available to IronPython code.

Loading .NET assemblies

The smallest unit of distribution of functionality in .NET is an assembly which usually corresponds to a single file with the .dll file extension. The assembly is available either in the installation folder of the application, or in the GAC (Global assembly cache). Assemblies can be loaded by using the methods of the CLR module. The following code will load the System.Xml.dll assembly which is part of the standard .NET implementation, and installed in the GAC:

```
>>> import clr
>>> clr.AddReference("System.Xml")
```

The full list of assemblies loaded by IronPython is available in CLR.References:

```
>>> "System.Xml" in [assembly.GetName().Name for assembly in clr.Referenc
```

True

All .NET assemblies have a unique version number which allows using a specific version of a given assembly. The following code will load the version of System.Xml.dll that ships with .NET 2.0 and .NET 3.5:

```
>>> import clr
>>> clr.AddReference("System.Xml, Version=2.0.0.0, Culture=neutral, Publi
```

You can load assemblies that are neither in the GAC nor in the appbase (typically, the folder of ipy.exe or your host application executable) either by using clr.AddReferenceToFileAndPath or by setting sys.path. See clr.AddReference-methods for details.

Note

IronPython only knows about assemblies that have been loaded using one of clr.AddReference-methods. It is possible for other assemblies to already be loaded before IronPython is loaded, or for other assemblies to be loaded by other parts of the application by calling System.Reflection.Assembly.Load, but IronPython will not be aware of these.

Assemblies loaded by default

When you use ipy.exe, mscorlib.dll and System.dll are automatically loaded. This enables you to start using these assemblies (which IronPython itself is dependent on) without having to call clr.AddReference-methods.

When IronPython code is embedded in an application, the application controls which assemblies are loaded by default.

Using .NET types

Once an assembly is loaded, the namespaces and types contained in the assembly can be accessed from IronPython code.

Importing .NET namespaces

.NET namespaces and sub-namespaces of loaded assemblies are exposed as Python modules:

>>> import System

```
>>> System #doctest: +ELLIPSIS
<module 'System' (CLS module, ... assemblies loaded)>
>>> System.Collections #doctest: +ELLIPSIS
<module 'Collections' (CLS module, ... assemblies loaded)>
```

The types in the namespaces are exposed as Python types, and are accessed as attributes of the namespace. The following code accesses the System.Environment class from mscorlib.dll:

```
>>> import System
>>> System.Environment
<type 'Environment'>
```

Just like with normal Python modules, you can also use all the other forms of import as well:

```
>>> from System import Environment
>>> Environment
<type 'Environment'>
>>> from System import *
>>> Environment
<type 'Environment'>
```

Warning

Using from <namespace> import * can cause Python builtins (elements of __builtins__) to be hidden by .NET types or sub-namespaces. Specifically, after doing from System import *, Exception will access the System.Exception .NET type, not Python's Exception type.

The root namespaces are stored as modules in sys.modules:

```
>>> import System
>>> import sys
>>> sys.modules["System"] #doctest: +ELLIPSIS
<module 'System' (CLS module, ... assemblies loaded)>
```

When new assemblies are loaded, they can add attributes to existing namespace module objects.

Import precedence relative to Python modules

import gives precedence to .py files. For example, if a file called System.py exists in the path, it will get imported instead of the System namespace:

```
>>> # create System.py in the current folder
>>> f = open("System.py", "w")
>>> f.write('print "Loading System.py"')
>>> f.close()
>>>
>>> # unload the System namespace if it has been loaded
>>> import sys
>>> if sys.modules.has_key("System"):
...     sys.modules.pop("System") #doctest: +ELLIPSIS
<module 'System' (CLS module, ... assemblies loaded)>
>>>
>>> import System
Loading System.py
>>> System #doctest: +ELLIPSIS
<module 'System' from '...System.py'>
```

Note

Do make sure to delete System.py:

```
>>> import os
>>> os.remove("System.py")
>>> sys.modules.pop("System") #doctest: +ELLIPSIS
<module 'System' from '...System.py'>
>>> import System
>>> System #doctest: +ELLIPSIS
```

<module 'System' (CLS module, ... assemblies loaded)>

Accessing generic types

.NET supports generic types which allow the same code to support multiple type parameters which retaining the advantages of types safety. Collection types (like lists, vectors, etc) are the canonical example where generic types are useful. .NET has a number of generic collection types in the System.Collections.Generic namespace.

IronPython exposes generic types as a special type object which supports indexing with type object(s) as the index (or indices):

>>> from System.Collections.Generic import List, Dictionary

>>> int_list = List[int]()

>>> str_float_dict = Dictionary[str, float]()

Note that there might exist a non-generic type as well as one or more generic types with the same name [1]. In this case, the name can be used without any indexing to access the non-generic type, and it can be indexed with different number of types to access the generic type with the corresponding number of type parameters. The code below accesses System.EventHandler and also System.EventHandler<TEventArgs>

>>> from System import EventHandler, EventArgs

>>> EventHandler # this is the combo type object

<types 'EventHandler', 'EventHandler[TEventArgs]'>

>>> # Access the non-generic type

>>> dir(EventHandler) #doctest: +ELLIPSIS

['BeginInvoke', 'Clone', 'DynamicInvoke', 'EndInvoke', ...

>>> # Access the generic type with 1 type paramter

>>> dir(EventHandler[EventArgs]) #doctest: +ELLIPSIS

['BeginInvoke', 'Call', 'Clone', 'Combine', ...

[1]

This refers to the user-friendly name. Under the hoods, the .NET type name includes the number of type parameters:

```
>>> clr.GetClrType(EventHandler[EventArgs]).Name
'EventHandler`1'
```

Accessing nested types

Nested types are exposed as attributes of the outer class:

```
>>> from System.Environment import SpecialFolder
>>> SpecialFolder
<type 'SpecialFolder'>
```

Importing .NET members from a type

.NET types are exposed as Python classes. Like Python classes, you usually cannot import all the attributes of .NET types using from <name> import *:

```
>>> from System.Guid import *
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: no module named Guid
```

You can import specific members, both static and instance:

```
>>> from System.Guid import NewGuid, ToByteArray
>>> g = NewGuid()
>>> ToByteArray(g) #doctest: +ELLIPSIS
Array[Byte](...
```

Note that if you import a static property, you will import the value when the import executes, not a named object to be evaluated on every use as you might mistakenly expect:

```
>>> from System.DateTime import Now
>>> Now #doctest: +ELLIPSIS
```

<System.DateTime object at ...>

>>> # Let's make it even more obvious that "Now" is evaluated only once

>>> a_second_ago = Now

>>> import time

>>> time.sleep(1)

>>> a_second_ago is Now

True

>>> a_second_ago is System.DateTime.Now

False

Importing all .NET members from a static type

Some .NET types only have static methods, and are comparable to namespaces. C# refers to them as static classes , and requires such classes to have only static methods. IronPython allows you to import all the static methods of such static classes. System.Environment is an example of a static class:

>>> from System.Environment import *

>>> Exit is System.Environment.Exit

True

Nested types are also imported:

>>> SpecialFolder is System.Environment.SpecialFolder

True

However, properties are not imported:

>>> OSVersion

Traceback (most recent call last):

  File "<stdin>", line 1, in <module>

NameError: name 'OSVersion' is not defined

>>> System.Environment.OSVersion #doctest: +ELLIPSIS

<System.OperatingSystem object at ...>

Type-system unification (type and System.Type)

.NET represents types using System.Type. However, when you access a .NET type in Python code, you get a Python type object [2]:

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5)
>>> isinstance(type(ba), type)
True
```

This allows a unified (Pythonic) view of both Python and .NET types. For example, isinstance works with .NET types as well:

```
>>> from System.Collections import BitArray
>>> isinstance(ba, BitArray)
True
```

If need to get the System.Type instance for the .NET type, you need to use clr.GetClrType. Conversely, you can use clr.GetPythonType to get a type object corresponding to a System.Type object.

The unification also extends to other type system entities like methods. .NET methods are exposed as instances of method:

```
>>> type(BitArray.Xor)
<type 'method_descriptor'>
>>> type(ba.Xor)
<type 'builtin_function_or_method'>
```

[2]

Note that the Python type corresponding to a .NET type is a sub-type of type:

```
>>> isinstance(type(ba), type)
True
>>> type(ba) is type
```

False

This is an implementation detail.

Similarity with builtin types

.NET types behave like builtin types (like list), and are immutable. i.e. you cannot add or delete descriptors from .NET types:

```
>>> del list.append
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: cannot delete attribute 'append' of builtin type 'list'
>>>
>>> import System
>>> del System.DateTime.ToByteArray
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can't set attributes of built-in/extension type 'DateTime'
```

Instantiating .NET types

.NET types are exposed as Python classes, and you can do many of the same operations on .NET types as with Python classes. In either cases, you create an instance by calling the type:

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5) # Creates a bit array of size 5
```

IronPython also supports inline initializing of the attributes of the instance. Consider the following two lines:

```
>>> ba = BitArray(5)
>>> ba.Length = 10
```

The above two lines are equivalent to this single line:

```
>>> ba = BitArray(5, Length = 10)
```

You can also call the __new__ method to create an instance:

```
>> ba = BitArray.__new__(BitArray, 5)
```

Invoking .NET methods

.NET methods are exposed as Python methods. Invoking .NET methods works just like invoking Python methods.

Invoking .NET instance methods

Invoking .NET instance methods works just like invoking methods on a Python object using the attribute notation:

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5)
>>> ba.Set(0, True) # call the Set method
>>> ba[0]
True
```

IronPython also supports named arguments:

```
>>> ba.Set(index = 1, value = True)
>>> ba[1]
True
```

IronPython also supports dict arguments:

```
>>> args = [2, True] # list of arguments
>>> ba.Set(*args)
>>> ba[2]
True
```

IronPython also supports keyword arguments:

```
>>> args = { "index" : 3, "value" : True }
>>> ba.Set(**args)
>>> ba[3]
True
```

Argument conversions

When the argument type does not exactly match the parameter type expected by the .NET method, IronPython tries to convert the argument. IronPython uses conventional .NET conversion rules like conversion operators , as well as IronPython-specific rules. This snippet shows how arguments are converted when calling the Set(System.Int32, System.Boolean) method:

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5)
>>> ba.Set(0, "hello") # converts the second argument to True.
>>> ba[0]
True
>>> ba.Set(1, None) # converts the second argument to False.
>>> ba[1]
False
```

See appendix-type-conversion-rules for the detailed conversion rules. Note that some Python types are implemented as .NET types and no conversion is required in such cases. See builtin-type-mapping for the mapping.

Some of the conversions supported are:

Python argument type	.NET method parameter type
int	System.Int8, System.Int16
float	System.Float
tuple with only elements of type T	System.Collections.Generic.IEnumerable<T>
function, method	System.Delegate and any of its sub-classes

Method overloads

.NET supports overloading methods by both number of arguments and type of arguments. When IronPython code calls an overloaded method, IronPython tries to select one of the overloads at runtime based on the number and type of arguments passed to the method, and also names of any keyword arguments. In most cases, the expected overload gets selected. Selecting an overload is easy when the argument types are an exact match with one of the overload signatures:

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5) # calls __new__(System.Int32)
>>> ba = BitArray(5, True) # calls __new__(System.Int32, System.Boolean)
>>> ba = BitArray(ba) # calls __new__(System.Collections.BitArray)
```

The argument types do not have be an exact match with the method signature. IronPython will try to convert the arguments if an unamibguous conversion exists to one of the overload signatures. The following code calls __new__(System.Int32) even though there are two constructors which take one argument, and neither of them accept a float as an argument:

```
>>> ba = BitArray(5.0)
```

However, note that IronPython will raise a TypeError if there are conversions to more than one of the overloads:

```
>>> BitArray((1, 2, 3))
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: Multiple targets could match: BitArray(Array[Byte]), BitArray(Array[bool]), BitArray(Array[int])
```

If you want to control the exact overload that gets called, you can use the Overloads method on method objects:

```
>>> int_bool_new = BitArray.__new__.Overloads[int, type(True)]
>>> ba = int_bool_new(BitArray, 5, True) # calls __new__(System.Int32, System.Boolean)
>>> ba = int_bool_new(BitArray, 5, "hello") # converts "hello" to a System.Boolan
```

```
>>> ba = int_bool_new(BitArray, 5)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: __new__() takes exactly 2 arguments (1 given)
```

Using unbound class instance methods

It is sometimes desirable to invoke an instance method using the unbound class instance method and passing an explicit self object as the first argument. For example, .NET allows a class to declare an instance method with the same name as a method in a base type, but without overriding the base method. See System.Reflection.MethodAttributes.NewSlot for more information. In such cases, using the unbound class instance method syntax allows you chose precisely which slot you wish to call:

```
>>> import System
>>> System.ICloneable.Clone("hello") # same as : "hello".Clone()
'hello'
```

The unbound class instance method syntax results in a virtual call, and calls the most derived implementation of the virtual method slot:

```
>>> s = "hello"
>>> System.Object.GetHashCode(s) == System.String.GetHashCode(s)
True
>>> from System.Runtime.CompilerServices import RuntimeHelpers
>>> RuntimeHelpers.GetHashCode(s) == System.String.GetHashCode(s)
False
```

Calling explicitly-implemented interface methods

.NET allows a method with a different name to override a base method implementation or interface method slot. This is useful if a type implements two interfaces with methods with the same name. This is known as explicity implemented interface methods. For example, Microsoft.Win32.RegistryKey implements System.IDisposable.Dispose explicitly:

```
>>> from Microsoft.Win32 import RegistryKey
>>> clr.GetClrType(RegistryKey).GetMethod("Flush") #doctest: +ELLIPSIS
<System.Reflection.RuntimeMethodInfo object at ... [Void Flush()]>
>>> clr.GetClrType(RegistryKey).GetMethod("Dispose")
>>>
```

In such cases, IronPython tries to expose the method using its simple name - if there is no ambiguity:

```
>>> from Microsoft.Win32 import Registry
>>> rkey = Registry.CurrentUser.OpenSubKey("Software")
>>> rkey.Dispose()
```

However, it is possible that the type has another method with the same name. In that case, the explicitly implemented method is not accessible as an attribute. However, it can still be called by using the unbound class instance method syntax:

```
>>> rkey = Registry.CurrentUser.OpenSubKey("Software")
>>> System.IDisposable.Dispose(rkey)
```

Invoking static .NET methods

Invoking static .NET methods is similar to invoking Python static methods:

```
>>> System.GC.Collect()
```

Like Python static methods, the .NET static method can be accessed as an attribute of sub-types as well:

```
>>> System.Object.ReferenceEquals is System.GC.ReferenceEquals
True
```

Invoking generic methods

Generic methods are exposed as attributes which can be indexed with type objects. The following code calls System.Activator.CreateInstance<T>

```
>>> from System import Activator, Guid
>>> guid = Activator.CreateInstance[Guid]()
```

Type parameter inference while invoking generic methods

In many cases, the type parameter can be inferred based on the arguments passed to the method call. Consider the following use of a generic method [3]:

```
>>> from System.Collections.Generic import IEnumerable, List
>>> list = List[int]([1, 2, 3])
>>> import clr
>>> clr.AddReference("System.Core")
>>> from System.Linq import Enumerable
>>> Enumerable.Any[int](list, lambda x : x < 2)
True
```

With generic type parameter inference, the last statement can also be written as:

```
>>> Enumerable.Any(list, lambda x : x < 2)
True
```

See appendix for the detailed rules.

[3] System.Core.dll is part of .NET 3.0 and higher.

ref and out parameters

The Python language passes all arguments by-value. There is no syntax to indicate that an argument should be passed by-reference like there is in .NET languages like C# and VB.NET via the ref and out keywords. IronPython supports two ways of passing ref or out arguments to a method, an implicit way and an explicit way.

In the implicit way, an argument is passed normally to the method call, and its (potentially) updated value is returned from the method call along with the normal return value (if any). This composes well with the Python feature of multiple return values. System.Collections.Generic.Dictionary has a method bool TryGetValue(K key, out value). It can be called from IronPython with just one argument, and the call returns a tuple where the first element is a boolean and the second element is the value (or the default value of 0.0 if the first element is False):

```
>>> d = { "a":100.1, "b":200.2, "c":300.3 }
>>> from System.Collections.Generic import Dictionary
>>> d = Dictionary[str, float](d)
>>> d.TryGetValue("b")
(True, 200.2)
>>> d.TryGetValue("z")
(False, 0.0)
```

In the explicit way, you can pass an instance of clr.Reference[T] for the ref or out argument, and its Value field will get set by the call. The explicit way is useful if there are multiple overloads with ref parameters:

```
>>> import clr
>>> r = clr.Reference[float]()
>>> d.TryGetValue("b", r)
True
>>> r.Value
200.2
```

Extension methods

Extension methods are currently not natively supported by IronPython. Hence, they cannot be invoked like instance methods. Instead, they have to be invoked like static methods.

Accessing .NET indexers

.NET indexers are exposed as __getitem__ and __setitem__. Thus, the Python indexing syntax can be used to index .NET collections (and any type with an indexer):

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5)
>>> ba[0]
False
>>> ba[0] = True
>>> ba[0]
True
```

The indexer can be called using the unbound class instance method syntax using __getitem__ and __setitem__. This is useful if the indexer is virtual and is implemented as an explicitly-implemented interface method:

```
>>> BitArray.__getitem__(ba, 0)
True
```

Non-default .NET indexers

Note that a default indexer is just a property (typically called Item) with one argument. It is considered as an indexer if the declaraing type uses DefaultMemberAttribute to declare the property as the default member.

See property-with-parameters for information on non-default indexers.

Accessing .NET properties

.NET properties are exposed similar to Python attributes. Under the hood, .NET properties are implemented as a pair of methods to get and set the property, and IronPython calls the appropriate method depending on whether you are reading or writing to the properity:

```
>>> from System.Collections import BitArray
```

```
>>> ba = BitArray(5)
>>> ba.Length # calls "BitArray.get_Length()"
5
>>> ba.Length = 10 # calls "BitArray.set_Length()"
```

To call the get or set method using the unbound class instance method syntax, IronPython exposes methods called GetValue and SetValue on the property descriptor. The code above is equivalent to the following:

```
>>> ba = BitArray(5)
>>> BitArray.Length.GetValue(ba)
5
>>> BitArray.Length.SetValue(ba, 10)
```

Properties with parameters

COM and VB.NET support properties with paramters. They are also known as non-default indexers. C# does not support declaring or using properties with parameters.

IronPython does support properties with parameters. For example, the default indexer above can also be accessed using the non-default format as such:

```
>>> ba.Item[0]
False
```

Accessing .NET events

.NET events are exposed as objects with __iadd__ and __isub__ methods which allows using += and -= to subscribe and unsubscribe from the event. The following code shows how to subscribe a Python function to an event using +=, and unsubscribe using -=

```
>>> from System.IO import FileSystemWatcher
>>> watcher = FileSystemWatcher(".")
>>> def callback(sender, event_args):
...     print event_args.ChangeType, event_args.Name
```

```
>>> watcher.Created += callback
>>> watcher.EnableRaisingEvents = True
>>> import time
>>> f = open("test.txt", "w+"); time.sleep(1)
Created test.txt
>>> watcher.Created -= callback
>>>
>>> # cleanup
>>> import os
>>> f.close(); os.remove("test.txt")
```

You can also subscribe using a bound method:

```
>>> watcher = FileSystemWatcher(".")
>>> class MyClass(object):
...     def callback(self, sender, event_args):
...         print event_args.ChangeType, event_args.Name
>>> o = MyClass()
>>> watcher.Created += o.callback
>>> watcher.EnableRaisingEvents = True
>>> f = open("test.txt", "w+"); time.sleep(1)
Created test.txt
>>> watcher.Created -= o.callback
>>>
>>> # cleanup
>>> f.close(); os.remove("test.txt")
```

You can also explicitly create a delegate instance to subscribe to the event. Otherwise, IronPython automatically does it for you. [4]:

```
>>> watcher = FileSystemWatcher(".")
>>> def callback(sender, event_args):
```

```
...     print event_args.ChangeType, event_args.Name
>>> from System.IO import FileSystemEventHandler
>>> delegate = FileSystemEventHandler(callback)
>>> watcher.Created += delegate
>>> watcher.EnableRaisingEvents = True
>>> import time
>>> f = open("test.txt", "w+"); time.sleep(1)
Created test.txt
>>> watcher.Created -= delegate
>>>
>>> # cleanup
>>> f.close(); os.remove("test.txt")
```

[4] The only advantage to creating an explicit delegate is that it is uses less memory. You should consider it if you subscribe to lots of events, and notice excessive System.WeakReference objects.

Special .NET types

.NET arrays

IronPython supports indexing of System.Array with a type object to access one-dimensional strongly-typed arrays:

```
>>> System.Array[int]
<type 'Array[int]'>
```

IronPython also adds a __new__ method that accepts a IList<T> to initialize the array. This allows using a Python list literal to initialize a .NET array:

```
>>> a = System.Array[int]([1, 2, 3])
```

Further, IronPython exposes __getitem__ and __setitem__ allowing the array objects to be indexed using the Python indexing syntax:

```
>>> a[2]
```

3

Note that the indexing syntax yields Python semantics. If you index with a negative value, it results in indexing from the end of the array, whereas .NET indexing (demonstrated by calling GetValue below) raises a System.IndexOutOfRangeException exception:

```
>>> a.GetValue(-1)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: Index was outside the bounds of the array.
>>> a[-1]
3
```

Similarly, slicing is also supported:

```
>>> a[1:3]
Array[int]((2, 3))
```

Multi-dimensional arrays

.NET Exceptions

raise can raise both Python exceptions as well as .NET exceptions:

```
>>> raise ZeroDivisionError()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError
>>> import System
>>> raise System.DivideByZeroException()
Traceback (most recent call last):
```

```
  File "<stdin>", line 1, in <module>
ZeroDivisionError: Attempted to divide by zero.
```

The except keyword can catch both Python exceptions as well as .NET exceptions:

```
>>> try:
...     import System
...     raise System.DivideByZeroException()
... except System.DivideByZeroException:
...     print "This line will get printed..."
...
This line will get printed...
>>>
```

The underlying .NET exception object

IronPython implements the Python exception mechanism on top of the .NET exception mechanism. This allows Python exception thrown from Python code to be caught by non-Python code, and vice versa. However, Python exception objects need to behave like Python user objects, not builtin types. For example, Python code can set arbitrary attributes on Python exception objects, but not on .NET exception objects:

```
>>> e = ZeroDivisionError()
>>> e.foo = 1 # this works
>>> e = System.DivideByZeroException()
>>> e.foo = 1
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'DivideByZeroException' object has no attribute 'foo'
```

To support these two different views, IronPython creates a pair of objects, a Python exception object and a .NET exception object, where the Python type and the .NET exception type have a unique one-to-one mapping as defined in the table below. Both objects know about each other. The .NET exception object is the one that actually gets thrown by the IronPython runtime when Python code executes a raise statement. When Python code uses the except keyword to catch the Python

exception, the Python exception object is used. However, if the exception is caught by C# (for example) code that called the Python code, then the C# code naturally catches the .NET exception object.

The .NET exception object corresponding to a Python exception object can be accessed by using the clsException attribute (if the module has excecuted import clr):

```
>>> import clr
>>> try:
...     1/0
... except ZeroDivisionError as e:
...     pass
>>> type(e)
<type 'exceptions.ZeroDivisionError'>
>>> type(e.clsException)
<type 'DivideByZeroException'>
```

IronPython is also able to access the Python exception object corresponding to a .NET exception object [5], thought this is not exposed to the user [6].

[5]

The Python exception object corresponding to a .NET exception object is accessible (to the IronPython runtime) via the System.Exception.Data property. Note that this is an implementation detail and subject to change:

```
>>> e.clsException.Data["PythonExceptionInfo"] #doctest: +ELLIPSIS
<IronPython.Runtime.Exceptions.PythonExceptions+ExceptionDataWrapper object at ...>
```

[6] ... except via the DLR Hosting API ScriptEngine.GetService<ExceptionOperations>().GetExceptionMessage

Python exception .NET exception

Exception System.Exception

SystemExit IP.O.SystemExit

StopIteration System.InvalidOperationException subtype

StandardError System.SystemException

KeyboardInterrupt IP.O.KeyboardInterruptException

ImportError IP.O.PythonImportError

EnvironmentError IP.O.PythonEnvironmentError

IOError System.IO.IOException

OSError S.R.InteropServices.ExternalException

WindowsError System.ComponentModel.Win32Exception

EOFError System.IO.EndOfStreamException

RuntimeErrorIP.O.RuntimeException

NotImplementedError System.NotImplementedException

NameError IP.O.NameException

UnboundLocalError IP.O.UnboundLocalException

AttributeErrorSystem.MissingMemberException

SyntaxError IP.O.SyntaxErrorException (System.Data has something close)

IndentationError IP.O.IndentationErrorException

TabError IP.O.TabErrorException

TypeError Microsoft.Scripting.ArgumentTypeException

AssertionError IP.O.AssertionException

LookupError IP.O.LookupException

IndexError System.IndexOutOfRangeException

KeyError S.C.G.KeyNotFoundException

ArithmeticError System.ArithmeticException

OverflowError System.OverflowException

ZeroDivisionError System.DivideByZeroException

FloatingPointError IP.O.PythonFloatingPointError

ValueError ArgumentException

UnicodeError IP.O.UnicodeException

UnicodeEncodeError System.Text.EncoderFallbackException

UnicodeDecodeError System.Text.DecoderFallbackException

UnicodeTranslateError IP.O.UnicodeTranslateException

ReferenceError IP.O.ReferenceException

SystemError IP.O.PythonSystemError

MemoryError System.OutOfMemoryException

Warning System.ComponentModel.WarningException

UserWarning IP.O.PythonUserWarning

DeprecationWarning IP.O.PythonDeprecationWarning

PendingDeprecationWarning IP.O.PythonPendingDeprecationWarning

SyntaxWarning IP.O.PythonSyntaxWarning

OverflowWarning IP.O.PythonOverflowWarning

RuntimeWarning IP.O.PythonRuntimeWarning

FutureWarning IP.O.PythonFutureWarning

Revisiting the rescue keyword

Given that raise results in the creation of both a Python exception object and a .NET exception object, and given that rescue can catch both Python exceptions and .NET exceptions, a question arises of which of the exception objects will be used by the rescue keyword. The answer is that it is the type used in the rescue clause. i.e. if the rescue clause uses the Python exception, then the Python exception object will be used. If the rescue clause uses the .NET exception, then the .NET exception object will be used.

The following example shows how 1/0 results in the creation of two objects, and how they are linked to each other. The exception is first caught as a .NET exception. The .NET exception is raised again, but is then caught as a Python exception:

```
>>> import System
>>> try:
...     try:
...         1/0
...     except System.DivideByZeroException as e1:
...         raise e1
```

```
... except ZeroDivisionError as e2:
...     pass
>>> type(e1)
<type 'DivideByZeroException'>
>>> type(e2)
<type 'exceptions.ZeroDivisionError'>
>>> e2.clsException is e1
True
```

User-defined exceptions

Python user-defined exceptions get mapped to System.Exception. If non-Python code catches a Python user-defined exception, it will be an instance of System.Exception, and will not be able to access the exception details:

```
>>> # since "Exception" might be System.Exception after "from System import *"
>>> if "Exception" in globals(): del Exception
>>> class MyException(Exception):
...     def __init__(self, value):
...         self.value = value
...     def __str__(self):
...         return repr(self.value)
>>> try:
...     raise MyException("some message")
... except System.Exception as e:
...     pass
>>> clr.GetClrType(type(e)).FullName
'System.Exception'
>>> e.Message
'Python Exception: MyException'
```

In this case, the non-Python code can use the ScriptEngine.GetService<ExceptionOperations>().GetExceptionMessage DLR Hosting API to get the exception message.

Enumerations

.NET enumeration types are sub-types of System.Enum. The enumeration values of an enumeration type are exposed as class attributes:

print System.AttributeTargets.All # access the value "All"

IronPython also supports using the bit-wise operators with the enumeration values:

>>> import System

>>> System.AttributeTargets.Class | System.AttributeTargets.Method

<enum System.AttributeTargets: Class, Method>

Value types

Python expects all mutable values to be represented as a reference type. .NET, on the other hand, introduces the concept of value types which are mostly copied instead of referenced. In particular .NET methods and properties returning a value type will always return a copy.

This can be confusing from a Python programmer’s perspective since a subsequent update to a field of such a value type will occur on the local copy, not within whatever enclosing object originally provided the value type.

While most .NET value types are designed to be immutable, and the .NET design guidelines recommend value tyeps be immutable, this is not enforced by .NET, and so there do exist some .NET valuetype that are mutable

For example, take the following C# definitions:

```
struct Point {
    # Poorly defined struct - structs should be immutable
    public int x;
    public int y;
```

```
}

class Line {
    public Point start;
    public Point end;

    public Point Start { get { return start; } }
    public Point End { get { return end; } }
}
```

If line is an instance of the reference type Line, then a Python programmer may well expect "line.Start.x = 1" to set the x coordinate of the start of that line. In fact the property Start returned a copy of the Point value type and it's to that copy the update is made:

```
print line.Start.x    # prints '0'
line.Start.x = 1
print line.Start.x    # still prints '0'
```

This behavior is subtle and confusing enough that C# produces a compile-time error if similar code is written (an attempt to modify a field of a value type just returned from a property invocation).

Even worse, when an attempt is made to modify the value type directly via the start field exposed by Line (i.e. "`line.start.x = 1`"), IronPython will still update a local copy of the Point structure. That's because Python is structured so that "foo.bar" will always produce a useable value: in the case above "line.start" needs to return a full value type which in turn implies a copy.

C#, on the other hand, interprets the entirety of the "`line.start.x = 1`" statement and actually yields a value type reference for the "line.start" part which in turn can be used to set the "x" field in place.

This highlights a difference in semantics between the two languages. In Python "line.start.x = 1" and "foo = line.start; foo.x = 1" are semantically equivalent. In C# that is not necessarily so.

So in summary: a Python programmer making updates to a value type embedded in an object will silently have those updates lost where the same syntax would yield the expected semantics in C#. An update to a value type returned from a .NET property will also appear to succeed will updating a local copy and will not cause an error as it does in the C# world. These two issues could easily become the source of subtle, hard to trace bugs within a large application.

In an effort to prevent the unintended update of local value type copies and at the same time preserve as pythonic and consistent a view of the world as possible, direct updates to value type fields are not allowed by IronPython, and raise a ValueError:

```
>>> line.start.x = 1 #doctest: +SKIP
Traceback (most recent call last):
  File , line 0, in input##7
ValueError Attempt to update field x on value type Point; value type fields can not be directly modified
```

This renders value types “mostly” immutable; updates are still possible via instance methods on the value type itself.

Proxy types

IronPython cannot directly use System.MarshalByRefObject instances. IronPython uses reflection at runtime to determine how to access an object. However, System.MarshalByRefObject instances do not support reflection.

You can use unbound-class-instance-method syntax to call methods on such proxy objects.

Delegates

Python functions and bound instance methods can be converted to delegates:

```
>>> from System import EventHandler, EventArgs
>>> def foo(sender, event_args):
```

```
...     print event_args
>>> d = EventHandler(foo)
>>> d(None, EventArgs()) #doctest: +ELLIPSIS
<System.EventArgs object at ... [System.EventArgs]>
```

Variance

IronPython also allows the signature of the Python function or method to be different (though compatible) with the delegate signature. For example, the Python function can use keyword arguments:

```
>>> def foo(*args):
...     print args
>>> d = EventHandler(foo)
>>> d(None, EventArgs()) #doctest: +ELLIPSIS
(None, <System.EventArgs object at ... [System.EventArgs]>)
```

If the return type of the delegate is void, IronPython also allows the Python function to return any type of return value, and just ignores the return value:

```
>>> def foo(*args):
...     return 100 # this return value will get ignored
>>> d = EventHandler(foo)
>>> d(None, EventArgs())
```

If the return value is different, IronPython will try to convert it:

```
>>> def foo(str1, str2):
...     return 100.1 # this return value will get converted to an int
>>> d = System.Comparison[str](foo)
>>> d("hello", "there")
100
```

Subclassing .NET types

Sub-classing of .NET types and interfaces is supported using class. .NET types and interfaces can be used as one of the sub-types in the class construct:

```
>>> class MyClass(System.Attribute, System.ICloneable, System.IComparable):
...     pass
```

.NET does not support multiple inheritance while Python does. IronPython allows using multiple Python classes as subtypes, and also multiple .NET interfaces, but there can only be one .NET class (other than System.Object) in the set of subtypes:

```
>>> class MyPythonClass1(object): pass
>>> class MyPythonClass2(object): pass
>>> class MyMixedClass(MyPythonClass1, MyPythonClass2, System.Attribute):
...     pass
```

Instances of the class do actually inherit from the specified .NET base type. This is important because this means that statically-typed .NET code can access the object using the .NET type. The following snippet uses Reflection to show that the object can be cast to the .NET sub-class:

```
>>> class MyClass(System.ICloneable):
...     pass
>>> o = MyClass()
>>> import clr
>>> clr.GetClrType(System.ICloneable).IsAssignableFrom(o.GetType())
True
```

Note that the Python class does not really inherit from the .NET sub-class. See type-mapping.

Overriding methods

Base type methods can be overriden by defining a Python method with the same name:

```
>>> class MyClass(System.ICloneable):
...     def Clone(self):
...         return MyClass()
>>> o = MyClass()
>>> o.Clone() #doctest: +ELLIPSIS
<MyClass object at ...>
```

IronPython does require you to provide implementations of interface methods in the class declaration. The method lookup is done dynamically when the method is accessed. Here we see that AttributeError is raised if the method is not defined:

```
>>> class MyClass(System.ICloneable): pass
>>> o = MyClass()
>>> o.Clone()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'MyClass' object has no attribute 'Clone'
```

Methods with multiple overloads

Python does not support method overloading. A class can have only one method with a given name. As a result, you cannot override specific method overloads of a .NET sub-type. Instead, you need to use define the function accepting an arbitrary argument list (see _tut-arbitraryargs), and then determine the method overload that was invoked by inspecting the types of the arguments:

```
>>> import clr
>>> import System
>>> StringComparer = System.Collections.Generic.IEqualityComparer[str]
>>>
>>> class MyComparer(StringComparer):
...     def GetHashCode(self, *args):
...         if len(args) == 0:
```

```
...             # Object.GetHashCode() called
...             return 100
...
...         if len(args) == 1 and type(args[0]) == str:
...             # StringComparer.GetHashCode() called
...             return 200
...
...         assert("Should never get here")
...
>>> comparer = MyComparer()
>>> getHashCode1 = clr.GetClrType(System.Object).GetMethod("GetHashCode")
>>> args = System.Array[object](["another string"])
>>> getHashCode2 = clr.GetClrType(StringComparer).GetMethod("GetHashCode")
>>>
>>> # Use Reflection to simulate a call to the different overloads
>>> # from another .NET language
>>> getHashCode1.Invoke(comparer, None)
100
>>> getHashCode2.Invoke(comparer, args)
200
```

Note

Determining the exact overload that was invoked may not be possible, for example, if None is passed in as an argument.

Methods with ref or out parameters

Python does not have syntax for specifying whether a method paramter is passed by-reference since arguments are always passed by-value. When overriding a .NET method with ref or out parameters, the ref or out paramter is received as a clr.Reference[T] instance. The incoming argument value is accessed by reading the Value property, and the resulting value is specified by setting the Value property:

```
>>> import clr
```

```
>>> import System
>>> StrFloatDictionary = System.Collections.Generic.IDictionary[str, float]
>>>
>>> class MyDictionary(StrFloatDictionary):
...     def TryGetValue(self, key, value):
...         if key == "yes":
...             value.Value = 100.1 # set the *out* parameter
...             return True
...         else:
...             value.Value = 0.0  # set the *out* parameter
...             return False
...     # Other methods of IDictionary not overriden for brevity
...
>>> d = MyDictionary()
>>> # Use Reflection to simulate a call from another .NET language
>>> tryGetValue = clr.GetClrType(StrFloatDictionary).GetMethod("TryGetValue")
>>> args = System.Array[object](["yes", 0.0])
>>> tryGetValue.Invoke(d, args)
True
>>> args[1]
100.1
```

Generic methods

When you override a generic method, the type parameters get passed in as arguments. Consider the following generic method declaration:

```
// csc /t:library /out:convert.dll convert.cs
public interface IMyConvertible {
   T1 Convert<T1, T2>(T2 arg);
}
```

The following code overrides the generic method Convert:

```
>>> import clr
>>> clr.AddReference("convert.dll")
>>> import System
>>> import IMyConvertible
>>>
>>> class MyConvertible(IMyConvertible):
...     def Convert(self, t2, T1, T2):
...         return T1(t2)
>>>
>>> o = MyConvertible()
>>> # Use Reflection to simulate a call from another .NET language
>>> type_params = System.Array[System.Type]([str, float])
>>> convert = clr.GetClrType(IMyConvertible).GetMethod("Convert")
>>> convert_of_str_float = convert.MakeGenericMethod(type_params)
>>> args = System.Array[object]([100.1])
>>> convert_of_str_float.Invoke(o, args)
'100.1'
```

Note

Generic method receive information about the method signature being invoked, whereas normal method overloads do not. The reason is that .NET does not allow normal method overloads to differ by the return type, and it is usually possible to determine the argument types based on the argument values. However, with generic methods, one of the type parameters may only be used as the return type. In that case, there is no way to determine the type paramter.

Calling from Python

When you call a method from Python, and the method overrides a .NET method from a base type, the call is performed as a regular Python call. The arguments do not undergo conversion, and neither are they modified in any way like being wrapped with clr.Reference. Thus, the call may need to be written differently than if the method was overriden by another language. For example, trying to call TryGetValue on the MyDictionary type from the overriding-ref-args section as shown

below results in a TypeError, whereas a similar call works with System.Collections.Generic.Dictionary[str, float]:

```
>>> result, value = d.TryGetValue("yes")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: TryGetValue() takes exactly 3 arguments (2 given)
```

Overriding properties

.NET properties are backed by a pair of .NET methods for reading and writing the property. The C# compiler automatically names them as get_<PropertyName> and set_<PropertyName>. However, .NET itself does not require any specific naming pattern for these methods, and the names are stored in the the metadata associated with the property definition. The names can be accessed using the GetGetMethod and GetSetMethods of the System.Reflection.PropertyInfo class:

```
>>> import clr
>>> import System
>>> StringCollection = System.Collections.Generic.ICollection[str]
>>> prop_info = clr.GetClrType(StringCollection).GetProperty("Count")
>>> prop_info.GetGetMethod().Name
'get_Count'
>>> prop_info.GetSetMethod() # None because this is a read-only property
>>>
```

Overriding a virtual property requires defining a Python method with the same names as the underlying getter or setter .NET method:

```
>>>
>>> class MyCollection(StringCollection):
...     def get_Count(self):
...         return 100
...     # Other methods of ICollection not overriden for brevity
>>>
```

```
>>> c = MyCollection()
>>> # Use Reflection to simulate a call from another .NET language
>>> prop_info.GetGetMethod().Invoke(c, None)
100
```

Overiding events

Events have underlying methods which can be obtained using EventInfo.GetAddMethod and EventInfo.GetRemoveMethod

```
>>> from System.ComponentModel import IComponent
>>> import clr
>>> event_info = clr.GetClrType(IComponent).GetEvent("Disposed")
>>> event_info.GetAddMethod().Name
'add_Disposed'
>>> event_info.GetRemoveMethod().Name
'remove_Disposed'
```

To override events, you need to define methods with the name of the underlying methods:

```
>>> class MyComponent(IComponent):
...     def __init__(self):
...         self.dispose_handlers = []
...     def Dispose(self):
...         for handler in self.dispose_handlers:
...             handler(self, EventArgs())
...
...     def add_Disposed(self, value):
...         self.dispose_handlers.append(value)
...     def remove_Disposed(self, value):
...         self.dispose_handlers.remove(value)
...     # Other methods of IComponent not implemented for brevity
```

```
>>>
>>> c = MyComponent()
>>> def callback(sender, event_args):
...     print event_args
>>> args = System.Array[object]((System.EventHandler(callback),))
>>> # Use Reflection to simulate a call from another .NET language
>>> event_info.GetAddMethod().Invoke(c, args)
>>>
>>> c.Dispose() #doctest: +ELLIPSIS
<System.EventArgs object at ... [System.EventArgs]>
Calling base constructor
```

.NET constructors can be overloaded. To call a specific base type constructor overload, you need to define a __new__ method (not __init__) and call __new__ on the .NET base type. The following example shows how a sub-type of System.Exception choses the base constructor overload to call based on the arguments it receives:

```
>>> import System
>>> class MyException(System.Exception):
...     def __new__(cls, *args):
...         # This could be implemented as:
...         #     return System.Exception.__new__(cls, *args)
...         # but is more verbose just to make a point
...         if len(args) == 0:
...             e = System.Exception.__new__(cls)
...         elif len(args) == 1:
...             message = args[0]
...             e = System.Exception.__new__(cls, message)
...         elif len(args) == 2:
...             message, inner_exception = args
...             if hasattr(inner_exception, "clsException"):
```

```
...             inner_exception = inner_exception.clsException
...         e = System.Exception.__new__(cls, message, inner_exception)
...     return e
>>> e = MyException("some message", IOError())
```

Accessing protected members of base types

Normally, IronPython does not allow access to protected members (unless you are using private-binding). For example, accessing MemberwiseClone causes a TypeError since it is a protected method:

```
>>> import clr
>>> import System
>>> o = System.Object()
>>> o.MemberwiseClone()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot access protected member MemberwiseClone without a python subclass of object
```

IronPython does allow Python sub-types to access protected members of .NET base types. However, Python does not enforce any accessibility rules. Also, methods can be added and removed dynamically from a class. Hence, IronPython does not attempt to guard access to protected members of .NET sub-types. Instead, it always makes the protected members available just like public members:

```
>>> class MyClass(System.Object):
...     pass
>>> o = MyClass()
>>> o.MemberwiseClone() #doctest: +ELLIPSIS
<MyClass object at ...>
```

Declaring .NET types

Relationship of classes in Python code and normal .NET types

A class definition in Python does not map directly to a unique .NET type. This is because the semantics of classes is different between Python and .NET. For example, in Python it is possible to change the base types just by assigning to the __bases__ attribute on the type object. However, the same is not possible with .NET types. Hence, IronPython implements Python classes without mapping them directly to .NET types. IronPython does use some .NET type for the objects, but its members do not match the Python attributes at all. Instead, the Python class is stored in a .NET field called .class, and Python instance attributes are stored in a dictionary that is stored in a .NET field called .dict [7]

```
>>> import clr
>>> class MyClass(object):
...     pass
>>> o = MyClass()
>>> o.GetType().FullName #doctest: +ELLIPSIS
'IronPython.NewTypes.System.Object_...'
>>> [field.Name for field in o.GetType().GetFields()]
['.class', '.dict', '.slots_and_weakref']
>>> o.GetType().GetField(".class").GetValue(o) == MyClass
True
>>> class MyClass2(MyClass):
...     pass
>>> o2 = MyClass2()
>>> o.GetType() == o2.GetType()
True
```

Also see Type-system unification (type and System.Type)

[7] These field names are implementation details, and could change.

__clrtype__

It is sometimes required to have control over the .NET type generated for the Python class. This is because some .NET APIs expect the user to define a .NET type with certain attributes and members. For example, to define a pinvoke method, the user is required to define a .NET type with a .NET method marked with DllImportAttribute

, and where the signature of the .NET method exactly describes the target platform method.

Starting with IronPython 2.6, IronPython supports a low-level hook which allows customization of the .NET type corresponding to a Python class. If the metaclass of a Python class has an attribute called __clrtype__, the attribute is called to generate a .NET type. This allows the user to control the the details of the generated .NET type. However, this is a low-level hook, and the user is expected to build on top of it.

The ClrType sample available in the IronPython website shows how to build on top of the __clrtype__ hook.

Accessing Python code from other .NET code

Statically-typed languages like C# and VB.Net can be compiled into an assembly that can then be used by other .NET code. However, IronPython code is executed dynamically using ipy.exe. If you want to run Python code from other .NET code, there are a number of ways of doing it.

Using the DLR Hosting APIs

The DLR Hosting APIs allow a .NET application to embed DLR languages like IronPython and IronRuby, load and execute Python and Ruby code, and access objects created by the Python or Ruby code.

Compiling Python code into an assembly

The pyc sample can be used to compile IronPython code into an assembly. The sample builds on top of clr-CompileModules. The assembly can then be loaded and executed using Python-ImportModule. However, note that the MSIL in the assembly is not CLS-compliant and cannot be directly accessed from other .NET languages.

dynamic

Starting with .NET 4.0, C# and VB.Net support access to IronPython objects using the dynamic keyword. This enables cleaner access to IronPython objects. Note that you need to use the hosting-apis to load IronPython code and get the root object out of it.

Integration of Python and .NET features

Type system integration.

See "Type-system unification (type and System.Type)"

Also see extensions-to-python-types and extensions-to-dotnet-types

List comprehension works with any .NET type that implements IList

with works with with any System.Collections.IEnumerable or System.Collections.Generic.IEnumerable<T>

pickle and ISerializable

__doc__ on .NET types and members:

__doc__ uses XML comments if available. XML comment files are installed if . As a result, help can be used:

```
>>> help(System.Collections.BitArray.Set) #doctest: +NORMALIZE_WHITESPACE
Help on method_descriptor:
Set(...)
    Set(self, int index, bool value)
                    Sets the bit at a specific
     position in the System.Collections.BitArray to
     the specified value.
<BLANKLINE>
    index:
```

The zero-based index of the

bit to set.

<BLANKLINE>

value:

The Boolean value to assign

to the bit.

If XML comment files are not available, IronPython generates documentation by reflecting on the type or member:

>>> help(System.Collections.Generic.List.Enumerator.Current) #doctest: +NORMALIZE_WHITESPACE

Help on getset descriptor System.Collections.Generic in mscorlib, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089.Enumerator.Current:

<BLANKLINE>

Current

Get: T Current(self)

Extensions to Python types

import clr exposes extra functionality on some Python types to make .NET features accessible:

method objects of any builtin or .NET types:

instance method

Overloads(t1 [, t2...])

type objects

instance method

__getitem__(t1 [, t2...]) - creates a generic instantiation

Extensions to .NET types

IronPython also adds extensions to .NET types to make them more Pythonic. The following instance methods are exposed on .NET objects (and .NET classes where explicitly mentioned):

Types with op_Implicit

Types with op_Explicit

Types inheriting from a .NET class or interface

.NET base-type

Synthesized Python method(s)

System.Object

all methods of object eg. __class__, __str__, __hash__, __setattr__

System.IDisposable

__enter__, __exit__

System.Collections.IEnumerator

next

System.Collections.ICollection System.Collections.Generic.ICollection<T>

__len__

System.Collections.IEnumerable System.Collections.Generic.IEnumerable<T>
System.Collections.IEnumerator System.Collections.Generic.IEnumerator<T>

__iter__

System.IFormattable

__format__

System.Collections.IDictionary System.Collections.Generic.IDictionary<TKey,
TValue> System.Collections.Generic.ICollection<T>
System.Collections.Generic.IList<T> System.Collections.IEnumerable
System.Collections.Generic.IEnumerable<T> System.Collections.IEnumerator
System.Collections.Generic.IEnumerator<T>

__contains__

System.Array

Class methods:

Indexing of the type object with a type object to access a specific array type

__new__(l) where l is IList<T> (or supports __getitem__?)

__getitem__, __setitem__, __slice__

System.Delegate

Class method : __new__(type, function_or_bound_method)

__call__

System.Enum

__or__ ?

Types with a .NET operator method name

.NET operator method

Synthesized Python method

op_Addition, Add

__add__

Compare

__cmp__

get_<Name> [8]

__getitem__

set_<Name> [9]

__setitem__

[8] where the type also has a property <Name>, and a DefaultMemberAttribute for <Name>

[9] where the type also has a property <Name>, and a DefaultMemberAttribute for <Name>

Equality and hashing

- This is currently just copied from IronRuby, and is known to be incorrect

Object equality and hashing are fundamental properties of objects. The Python API for comparing and hashing objects is __eq__ (and __ne__) and __hash__ respectively. The CLR APIs are System.Object.Equals and System.Object.GetHashCode respectively. IronPython does an automatic mapping

between the two concepts so that Python objects can be compared and hashed from non-Python .NET code, and __eq__ and __hash__ are available in Python code for non-Python objects as well.

When Python code calls __eq__ and __hash__

If the object is a Python object, the default implementations of __eq__ and __hash__ get called. The default implementations call System.Object.ReferenceEquals and System.Runtime.CompileServices.RuntimeHelpers.GetHashCode respectively.

If the object is a CLR object, System.Object.Equals and System.Object.GetHashCode respectively get called on the .NET object.

If the object is a Python subclass object inheriting from a CLR class, the CLR's class's implementation of System.Object.Equals and System.Object.GetHashCode will get called if the Python subclass does not define __eq__ and __hash__. If the Python subclass defines __eq__ and __hash__, those will be called instead.

When static MSIL code calls System.Object.Equals and System.Object.GetHashCode

If the object is a Python objects, the Python object will direct the call to __eq__ and __hash__. If the Python object has implementations for these methods, they will be called. Otherwise, the default implementation mentioned above gets called.

If the object is a Python subclass object inheriting from a CLR class, the CLR's class's implementation of System.Object.Equals and System.Object.GetHashCode will get called if the Python subclass does not define __eq__ and __hash__. If the Python subclass defines __eq__ and __hash__, those will be called instead.

Hashing of mutable objects

The CLR expects that System.Object.GetHashCode always returns the same value for a given object. If this invariant is not maintained, using the object as a key in a System.Collections.Generic.Dictionary<K,V> will misbehave. Python allows __hash__ to return different results, and relies on the user to deal with the scenario of using the object as a key in a Hash. The mapping above between the Python and CLR concepts of equality and hashing means that CLR code that deals with Python objects has to be aware of the issue. If static MSIL code uses a Python object as a the key in a Dictionary<K,V>, unexpected behavior might happen.

To reduce the chances of this happenning when using common Python types, IronPython does not map __hash__ to GetHashCode for Array and Hash. For other Python classes, the user can provide separate implementations for __eq__ and Equals, and __hash__ and GetHashCode if the Python class is mutable but also needs to be usable as a key in a Dictionary<K,V>.

System.Object.ToString, __repr__ and __str__

ToString on Python objects

Calling ToString on Python objects calls the default System.Object.ToString implementation, even if the Python type defines __str__:

```
>>> class MyClass(object):
...     def __str__(self):
...         return "__str__ result"
>>> o = MyClass()
>>> # Use Reflection to simulate a call from another .NET language
>>> o.GetType().GetMethod("ToString").Invoke(o, None) #doctest: +ELLIPSIS
'IronPython.NewTypes.System.Object_...'
```

__repr__/__str__ on .NET objects

All Python user types have __repr__ and __str__:

```
>>> class MyClass(object):
...     pass
>>> o = MyClass()
>>> o.__repr__() #doctest: +ELLIPSIS
'<MyClass object at ...>'
>>> o.__str__() #doctest: +ELLIPSIS
'IronPython.NewTypes.System.Object_...'
>>> str(o) #doctest: +ELLIPSIS
```

'<MyClass object at ...>'

For .NET types which do not override ToString, IronPython provides __repr__ and __str__ methods which behave similar to those of Python user types [10]:

```
>>> from System.Collections import BitArray
>>> ba = BitArray(5)
>>> ba.ToString() # BitArray inherts System.Object.ToString()
'System.Collections.BitArray'
>>> ba.__repr__() #doctest: +ELLIPSIS
'<System.Collections.BitArray object at ... [System.Collections.BitArray]>'
>>> ba.__str__() #doctest: +ELLIPSIS
'<System.Collections.BitArray object at ... [System.Collections.BitArray]>'
```

For .NET types which do override ToString, IronPython includes the result of ToString in __repr__, and maps ToString directly to __str__:

```
>>> e = System.Exception()
>>> e.ToString()
"System.Exception: Exception of type 'System.Exception' was thrown."
>>> e.__repr__() #doctest: +ELLIPSIS
"<System.Exception object at ... [System.Exception: Exception of type
'System.Exception' was thrown.]>"
>>> e.__str__() #doctest:
"System.Exception: Exception of type 'System.Exception' was thrown."
```

For Python types that override ToString, __str__ is mapped to the ToString override:

```
>>> class MyClass(object):
...     def ToString(self):
...         return "ToString implemented in Python"
>>> o = MyClass()
>>> o.__repr__() #doctest: +ELLIPSIS
'<MyClass object at ...>'
```

```
>>> o.__str__()
'ToString implemented in Python'
>>> str(o) #doctest: +ELLIPSIS
'<MyClass object at ...>'
```

[10] There is some inconsistency in handling of __str__ that is tracked by https://ironpython.codeplex.com/WorkItem/View.aspx?WorkItemId=24973

OleAutomation and COM interop

IronPython supports accessing OleAutomation objects (COM objects which support dispinterfaces).

IronPython does not support the win32ole library, but Python code using win32ole can run on IronPython with just a few modifications.

Creating a COM object

Different languages have different ways to create a COM object. VBScript and VBA have a method called CreateObject to create an OleAut object. JScript has a method called . There are multiple ways of doing the same in IronPython.

The first approach is to use System.Type.GetTypeFromProgID and System.Activator.CreateInstance . This method works with any registered COM object:

```
>>> import System
>>> t = System.Type.GetTypeFromProgID("Excel.Application")
>>> excel = System.Activator.CreateInstance(t)
>>> wb = excel.Workbooks.Add()
>>> excel.Quit()
```

The second approach is to use clr.AddReferenceToTypeLibrary to load the type library (if it is available) of the COM object. The advantage is that you can use the type library to access other named values like constants:

```
>>> import System
>>> excelTypeLibGuid = System.Guid("00020813-0000-0000-C000-000000000046")
>>> import clr
>>> clr.AddReferenceToTypeLibrary(excelTypeLibGuid)
>>> from Excel import Application
>>> excel = Application()
>>> wb = excel.Workbooks.Add()
>>> excel.Quit()
```

Finally, you can also use the interop assembly. This can be generated using the tlbimp.exe tool. The only advantage of this approach was that this was the approach recommeded for IronPython 1. If you have code using this approach that you developed for IronPython 1, it will continue to work:

```
>>> import clr
>>> clr.AddReference("Microsoft.Office.Interop.Excel")
>>> from Microsoft.Office.Interop.Excel import ApplicationClass
>>> excel = ApplicationClass()
>>> wb = excel.Workbooks.Add()
>>> excel.Quit()
```

Using COM objects

One you have access to a COM object, it can be used like any other objects. Properties, methods, default indexers and events all work as expected.

Properties

There is one important detail worth pointing out. IronPython tries to use the type library of the OleAut object if it can be found, in order to do name resolution while accessing methods or properties. The reason for this is that the IDispatch interface does not make much of a distinction between properties and method calls. This is because of Visual Basic 6 semantics where "excel.Quit" and "excel.Quit()" have the exact same semantics. However, IronPython has a strong distinction between properties and methods, and methods are first class objects. For IronPython to know whether "excel.Quit" should invoke the method Quit, or just return a callable object,

it needs to inspect the typelib. If a typelib is not available, IronPython assumes that it is a method. So if a OleAut object has a property called "prop" but it has no typelib, you would need to write "p = obj.prop()" in IronPython to read the property value.

Methods with out parameters

Calling a method with "out" (or in-out) parameters requires explicitly passing in an instance of "clr.Reference", if you want to get the updated value from the method call. Note that COM methods with out parameters are not considered Automation-friendly [11]. JScript does not support out parameters at all. If you do run into a COM component which has out parameters, having to use "clr.Reference" is a reasonable workaround:

```
>>> import clr
>>> from System import Type, Activator
>>> command_type = Type.GetTypeFromProgID("ADODB.Command")
>>> command = Activator.CreateInstance(command_type)
>>> records_affected = clr.Reference[int]()
>>> command.Execute(records_affected, None, None) #doctest: +SKIP
>>> records_affected.Value
0
```

Another workaround is to leverage the inteorp assembly by using the unbound class instance method syntax of "outParamAsReturnValue = InteropAssemblyNamespace.IComInterface(comObject)".

[11] Note that the Office APIs in particular do have "VARIANT*" parameters, but these methods do not update the value of the VARIANT. The only reason they were defined with "VARIANT*" parameters was for performance since passing a pointer to a VARIANT is faster than pushing all the 4 DWORDs of the VARIANT onto the stack. So you can just treat such parameters as "in" parameters.

Accessing the type library

The type library has names of constants. You can use clr.AddReferenceToTypeLibrary to load the type library.

Non-automation COM objects

IronPython does not fully support COM objects which do not support dispinterfaces since they appear likey proxy objects [12]. You can use the unbound class method syntax to access them.

[12] This was supported in IronPython 1, but the support was dropped in version 2.

Miscellaneous

Security model

When running Python code using ipy.exe, IronPython behaves like Python and does not do any sand-boxing. All scripts execute with the permissions of the user. As a result, running Python code downloaded from the Internet for example could be potentially be dangerous.

However, ipy.exe is just one manifiestation of IronPython. IronPython can also be used in other scenarios like in Silverlight or embedded in an application. All the IronPython assemblies are security-transparent. As a result, IronPython code can be run in a sand-box and the host can control the security priviledges to be granted to the Python code. This is one of the benefits of IronPython building on top of .NET. For example, when running in a web browser via the Silverlight plugin, Python code will not be able to write to the file system or make network connections to hosts other than the host where the web page orginites from. This security is enforced at the .NET level itself, and hence is very secure.

Execution model and call frames

IronPython code can be executed by any of the following techniques:

Interpretation

Compiling on the fly using DynamicMethod

Compiling on the fly using DynamicMethod

Ahead-of-time compilation to an assembly on disk using the pyc sample

A combination of the above - ie. a method might initially be interpreted, and can later be compiled once it has been called a number of times.

As a result, call frames of IronPython code are not like frames of statically typed langauges like C# and VB.Net. .NET code using APIs like those listed below need to think about how it will deal with IronPython code:

StackTrace.__new__

GetExecutingAssembly

Exception.ToString

Accessing non-public members

It is sometimes useful to access private members of an object. For example, while writing unit tests for .NET code in IronPython or when using the interactive command line to observe the innner workings of some object. ipy.exe supports this via the -X:PrivateBinding` command-line option. It can also be enabled in hosting scenarios via the property ; this requires IronPython to be executing with FullTrust.

Mapping between Python builtin types and .NET types

IronPython is an implementation of the Python language on top of .NET. As such, IronPython uses various .NET types to implement Python types. Usually, you do not have to think about this. However, you may sometimes have to know about it.

Python type .NET type

object System.Object

int System.Int32

long System.Numeric.BigInteger [13]

float System.Double

str, unicode System.String

bool System.Boolean

[13] This is true only in CLR 4. In previous versions of the CLR, long is implemented by IronPython itself.

import clr and builtin types

Since some Python builtin types are implemented as .NET types, the question arises whether the types work like Python types or like .NET types. The answer is that by default, the types work like Python types. However, if a module executes import clr, the types work like both Python types and like .NET types. For example, by default, object' does not have the System.Object method called GetHashCode:

```
>>> hasattr(object, "__hash__")
True
>>> # Note that this assumes that "import clr" has not yet been executed
>>> hasattr(object, "GetHashCode") #doctest: +SKIP
False
```

However, once you do import clr, object has both __hash__ as well as GetHashCode:

```
>>> import clr
>>> hasattr(object, "__hash__")
True
>>> hasattr(object, "GetHashCode")
True
```

LINQ

Language-integrated Query (LINQ) is a set of features that was added in .NET 3.5. Since it is a scenario rather than a specific feature, we will first compare which of the scenarios work with IronPython:

LINQ-to-objects

Python's list comprehension provides similar functionality, and is more Pythonic. Hence, it is recommended to use list comprehension itself.

DLinq - This is currently not supported.

Feature by feature comparison

LINQ consists of a number of language and .NET features, and IronPython has differing levels of support for the different features:

C# and VB.NET lambda function - Python supports lambda functions already.

Anonymous types - Python has tuples which can be used like anonymous types.

Extension methods - See

Generic method type parameter inference - See

Expression trees - This is not supported. This is the main reason DLinq does not work.

Appendix - Type conversion rules

Note that some Python types are implemented as .NET types and no conversion is required in such cases. See builtin-type-mapping for the mapping.

Python argument type .NET method parameter type

int System.Byte, System.SByte, System.UInt16, System.Int16

User object with __int__ method Same as int

str or unicode of size 1 System.Char

User object with __str__ method Same as str

float System.Float

tuple with T-typed elements System.Collections.Generic.IEnumerable<T> or System.Collections.Generic.IList<T>

function, method System.Delegate and any of its sub-classes

dict with K-typed keys and V-typed values System.Collections.Generic.IDictionary<K,V>

type System.Type

Appendix - Detailed method overload resolution rules

: This is old information

Roughly equivalent to VB 11.8.1 with additional level of preferred narrowing conversions

Start with the set of all accessible members

Keep only those members for which the argument types can be assigned to the parameter types by a widening conversion

If there is one or more member in the set find the best member

If there is one best member then call it

If there are multiple best members then throw ambiguous

Add in those members for which the argument types can be assigned to the parameter types by either a preferred narrowing or a widening conversion

If there is one applicable member then call it

If there is more than one applicable member then throw ambiguous

Add in those members for which the argument types can be assigned to the parameter types by any narrowing or a widening conversion

If there is one applicable member then call it

If there is more than one applicable member then throw ambiguous

Otherwise throw no match

Applicable Members By Number of Arguments – Phase 1

The number of arguments is identical to the number of parameters

The number of arguments is less than the number of parameters, but all parameters without an argument are optional – have a non-DbNull default value.

The method includes a parameter array and the params-expanded form of the method is applicable to the arguments

The params-expanded form is constructed by replacing the parameter array in the declaration with zero or more value parameters of the element type of the parameter array such that the number of arguments matches the number of parameters in the expanded form

The method includes byref parameters and the byref-reduced form of the method is applicable to the arguments

The byref-reduced form is constructed by removing all out parameters from the list and replacing all ref parameters with their target type. The return information for such a match will be provided in a tuple of return values.

Applicable Members By Type Of Arguments – Phase 2

If a conversion of the given type exists from the argument object to the type of the parameter for every argument then the method is applicable

For ref or out parameters, the argument must be an instance of the appropriate Reference class – unless the byref-reduced form of the method is being used

Better Member (same as C# 7.4.2.2)

Parameter Types : Given an argument list A with a set of types {A1, A1, ..., An} and type applicable parameter lists P and Q with types {P1, P2, ..., Pn} and {Q1, Q2, ..., Qn} P is a better member than Q if

For each argument, the conversion from Ax to Px is not worse than the conversion from Ax to Qx, and

For at least one argument, the conversion from Ax to Px is better than the conversion from Ax to Qx

Parameter Modifications : The method that uses the minimal conversions from the original method is considered the better match. The better member is the one that matches the earliest rule in the list of conversions for applicable methods. If both members use the same rules, then the method that converts the fewest of its parameters is considered best. For example, if multiple params methods have identical expanded forms, then the method with the most parameters prior to params-expanded form will be selected

Static vs. instance methods : When comparing a static method and an instance method that are both applicable, then the method that matches the calling convention is considered better. If the method is called unbound on the type object then the static method is preferred; however, if the method is called bound to an instance than the instance method will be preferred.

Explicitly implemented interface methods: Methods implemented as public methods on a class are considered better than methods that are private on the declaring class which explicitly implement an interface method.

Generic methods: Non-generic methods are considered better than generic methods.

Better Conversion (same as C# 7.4.2.3)

If T1 == T2 then neither conversion is better

If S is T1 then C1 is the better conversion (and vice-versa)

If a conversion from T1 to T2 exists, and no conversion from T2 to T1 exists, then C1 is the better conversion (and vice versa)

Conversion to a signed numeric type is preferred over conversion to a non-signed type of equal or greater size (this means that sbyte is preferred over byte)

Special conversion rule for ExtensibleFoo: An ExtensibleFoo has a conversion to a type whenever there is an appropriate conversion from Foo to that type.

Implicit Conversions

Implicit numeric conversions (C# 6.1.2)

Implicit reference conversions (C# 6.1.4) == Type.IsAssignableFrom

null -> Nullable<T>

COM object to any interface type

User-defined implicit conversions (C# 6.1.7)

Conversion from DynamicType -> Type

Narrowing Conversions (see VB 8.9 but much more restrictive for Python) are conversions that cannot be proved to always succeed, conversions that are known to possibly lose information, and conversions across domains of types sufficiently different to merit narrowing notation. The following conversions are classified as narrowing conversions:

Preferred Narrowing Conversions

BigInteger -> Int64 – because this is how Python represents numbers larger than 32 bits

IList<object> -> IList<T>

IEnumerator<object> -> IEnumerator<T>

IDictionary<object,object> -> IDictionary<K,V>

<Need to edit from here on down>

Narrowing Conversions

Bool -> int

Narrowing conversions of numeric types when overflow doesn't occur

String(length == 1) -> char and Char -> string(length == 1)

Generic Python protocols to CLS types

Callable (or anything?) -> Delegate

Object (iterable?) -> IEnumerator?

__int__ to int, __float__, __complex__

Troubling conversions planning to keep

Object -> bool (__nonzero__)

Double -> int – this is standard Python behavior, albeit deprecated behavior

Tuple -> Array<T>

All of the below will require explicit conversions

Enum to numeric type – require explicit conversions instead

From numeric types to char (excluded by C#)

Dict -> Hashtable

List -> Array<T>, List<T> and ArrayList

Tuple -> List<T> and ArrayList

Rules for going the other direction when C# methods are overridden by Python or delegates are implemented on the Python side:

This change alters our rules for how params and by ref parameters are handled for both overridden methods and delegates.

by ref (ref or out) parameters are always passed to Python as an instance of clr.Reference. The Value property on these can be used to get and set the underlying value and on return from the method this will be propogated back to the caller.

params parameters are ignored in these cases and the underlying array is passed to the Python function instead of splitting out all of the args.

The principle behind this change is to present the most direct reflection of the CLS signature to the Python programmer when they are doing something where the signature could be ambiguous. For calling methods with by ref parameters we support both explicit Reference objects and the implicit skipped parameters. When overriding we want to support the most direct signature to remove ambiguity. Similarly for params methods we support both calling the method with an explicit array of args or with n-args. To remove the ambiguity when overriding we only support the explicit array.

I'm quite happy with this principle in general. The one part that sucks for me is that these methods are now not callable from Python in the non-explict forms any more. For example, if I have a method void Foo(params object[] args) then I will override it with a Python method Foo(args) and not Foo(*args). This means that the CLS base type's method can be called as o.Foo(1,2,3) but the Python subclass will have to be called as o.Foo((1,2,3)). This is somewhat ugly, but I can't come up with any other relatively simple and clear option here and I think that because overriding overloaded methods can get quite complicated we should err on the side of simplicity.

1.5.11.6 The Python Tutorial

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well.

For a description of standard objects and modules, see The Python Standard Library. The Python Language Reference gives a more formal definition of the

language. To write extensions in C or C++, read IronPython .NET API Reference Manual and c-api-index. There are also several books covering Python in depth.

This tutorial does not attempt to be comprehensive and cover every single feature, or even every commonly used feature. Instead, it introduces many of Python's most noteworthy features, and will give you a good idea of the language's flavor and style. After reading it, you will be able to read and write Python modules and programs, and you will be ready to learn more about the various Python library modules described in The Python Standard Library.

1.5.12 Acknowledgments

Developing AutoTRAX DEX has been a long but extremely fun laden path.

We would also like to **thank all the users** who have given support and feedback over the years.

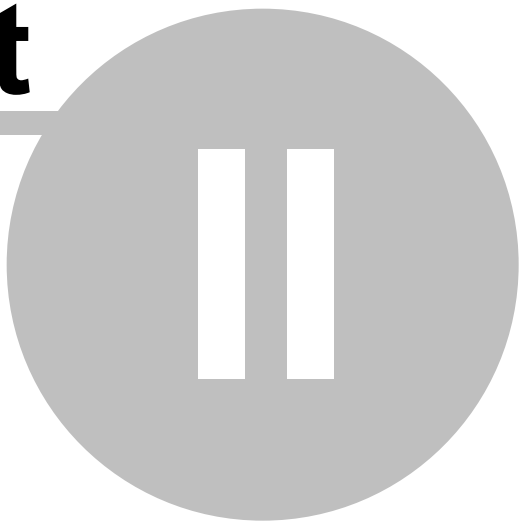
Many thanks go to the many members of the user forums.

Specific thanks go to:

- **Mick Gulovsen** for the PCB design manual.
- **Kjocoka** for manually routing the AutoTRAX DEX UNO sample project.

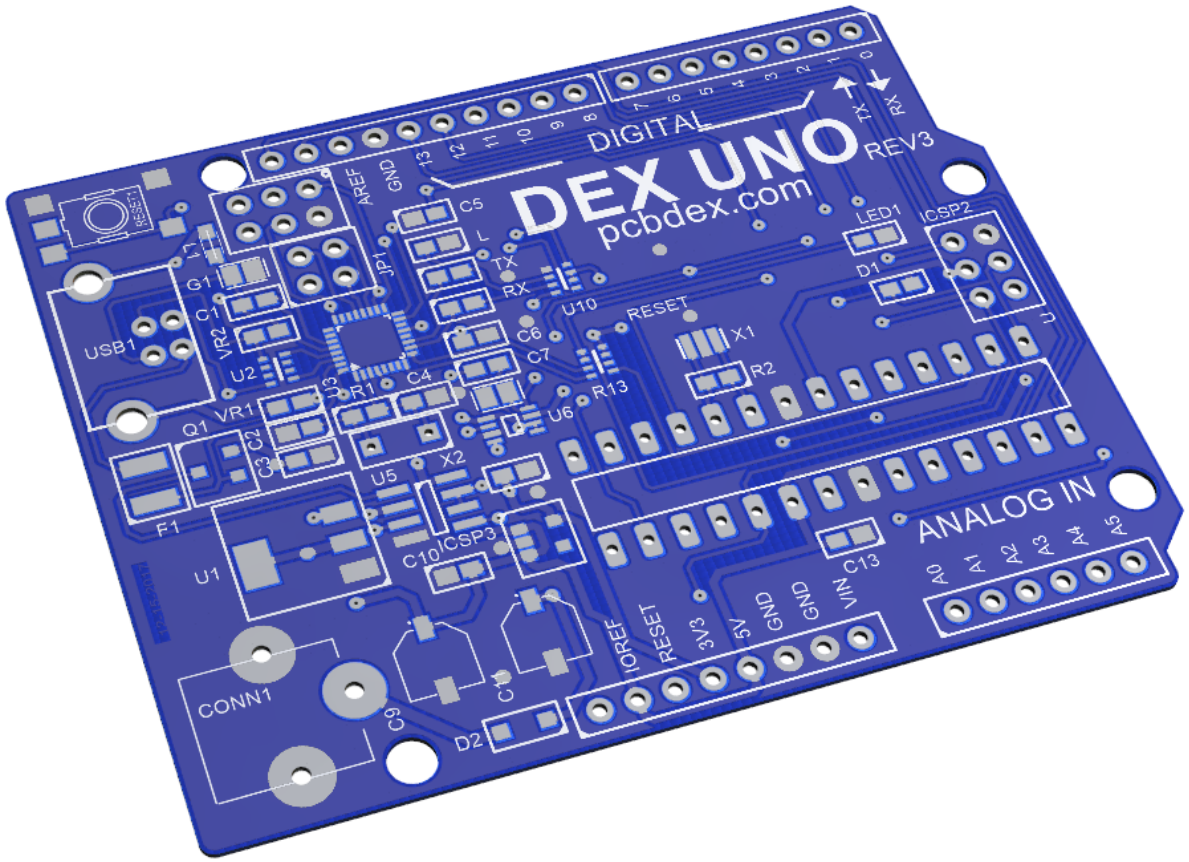
Thanks

Part



2 PCB Design

Published Tuesday, August 8, 2023 at 9:06:03 PM GMT



Typical 2-Sided PCB

Designing a printed circuit board (PCB) involves several steps. Here is a general overview of the process:

Measure Twice, Cut Once

"Measure twice, cut once" is a popular adage, particularly in the fields of construction, woodworking, and any craft where precise measurement is crucial. The saying encourages careful preparation and thorough checking to avoid costly or irreversible mistakes.

The underlying principle here is that thorough planning can help prevent mistakes and wasted resources. It's generally easier and less costly to take an extra moment to double-check your work than it is to correct a mistake after it's made. This adage can also be applied metaphorically in various aspects of life where careful planning and consideration are important, such as making financial decisions, planning a project, or making strategic choices.

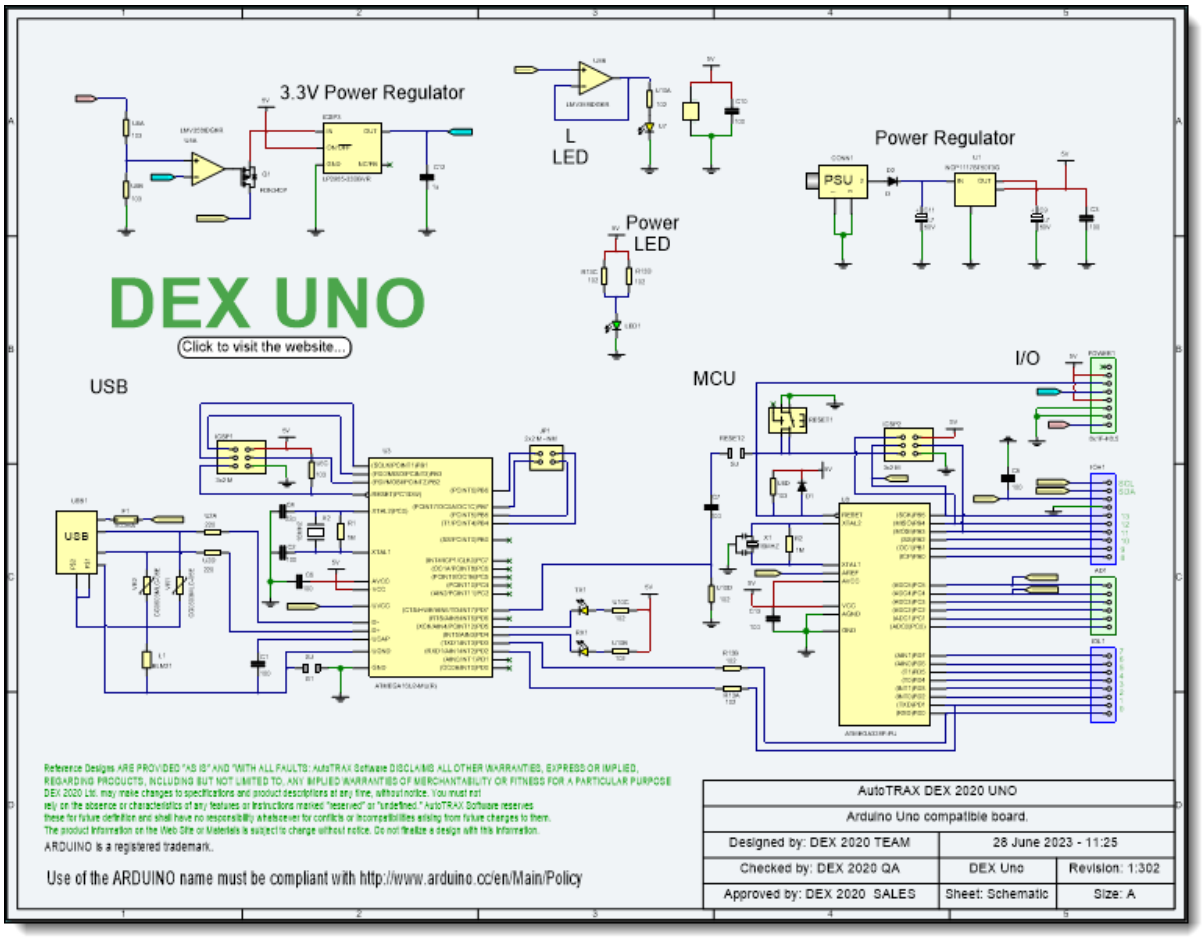
Define the Specifications

Determine the purpose, functionality, and requirements of your PCB. Consider factors such as the size, number of layers, power requirements, component placement, and signal integrity.

Define the Specifications

Schematic Design

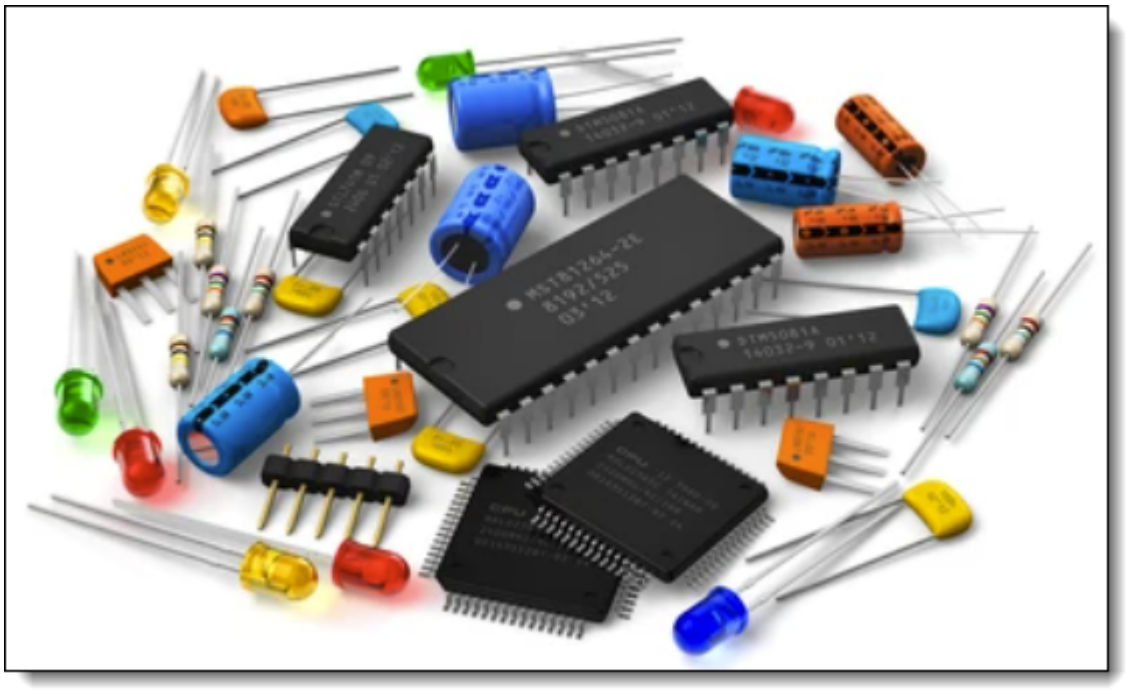
Create a schematic diagram of your circuit using a schematic capture tool. This diagram shows the interconnections between components, helping you visualize the circuit's functionality.



Schematic Design

Component Selection

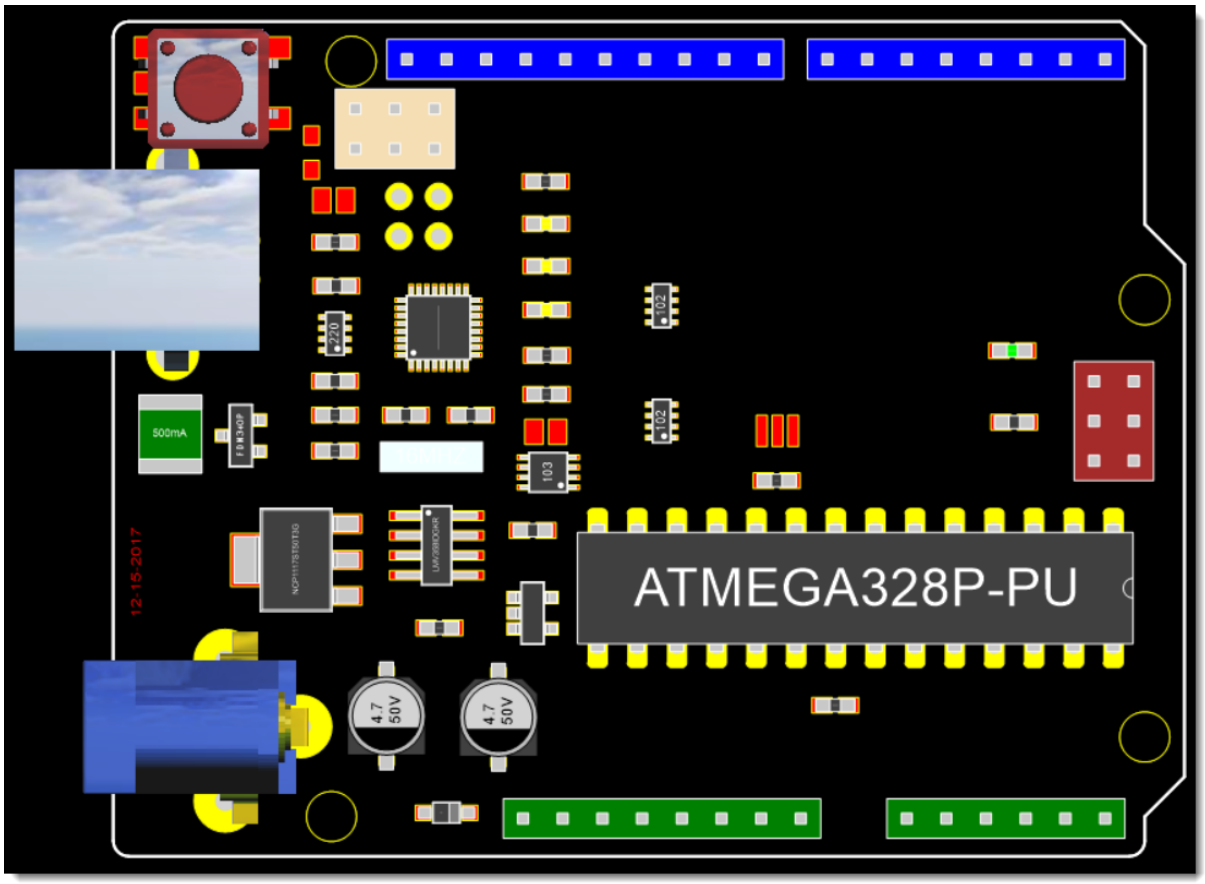
Choose the components needed for your circuit based on the schematic. Consider factors such as availability, cost, performance, and package size. Ensure that the components you select are suitable for PCB assembly and have appropriate footprints.



Component Selection

PCB Layout

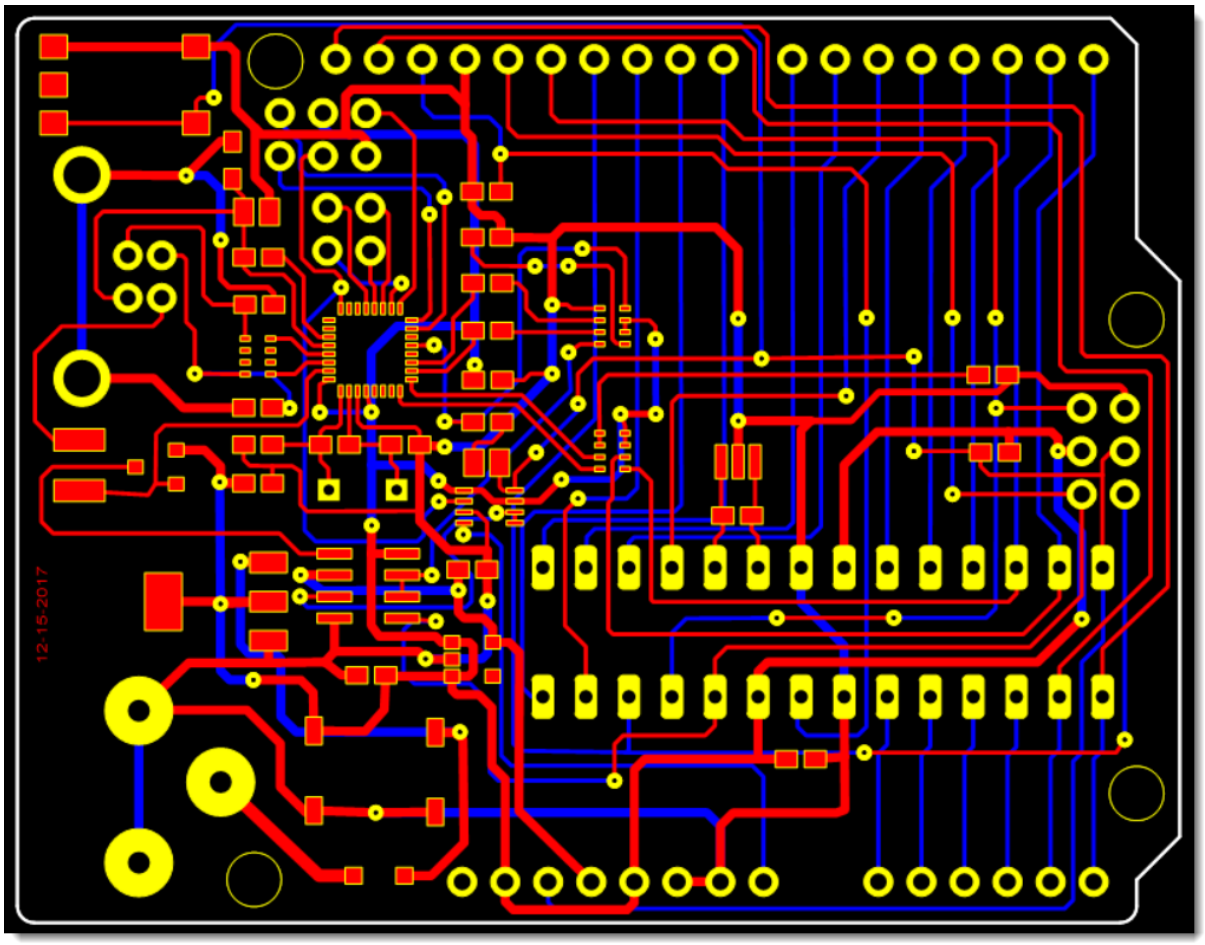
Transfer the schematic design to a PCB layout tool. Place the components on the PCB, taking into account factors like signal routing, power distribution, and thermal considerations. Arrange the components in a logical and efficient manner, considering the overall size and shape of the PCB.



PCB Layout

Routing

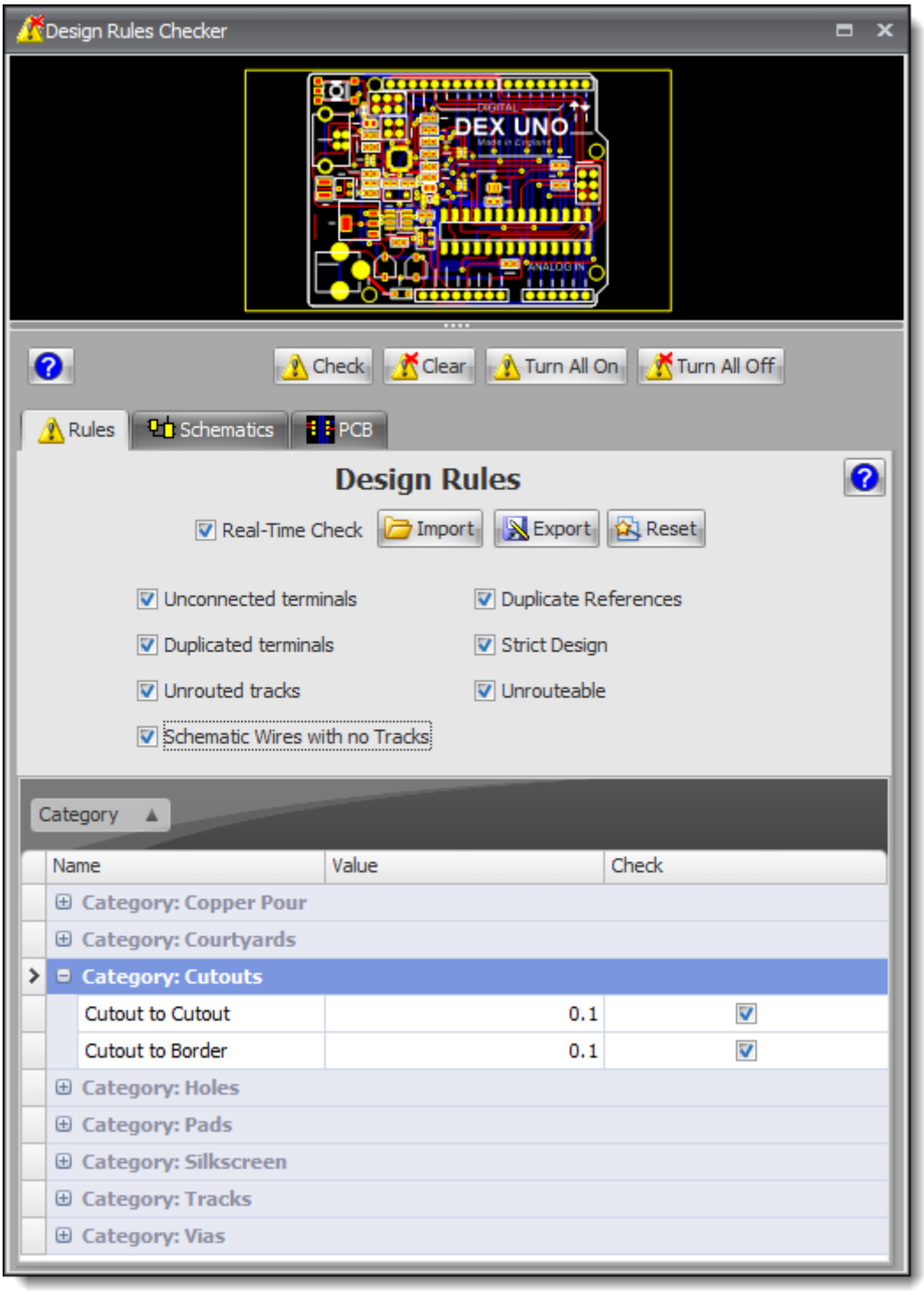
Connect the components using copper traces on the PCB. Pay attention to signal integrity, power delivery, and impedance matching. Ensure that the routing is optimized to minimize noise, crosstalk, and other potential issues. Use ground and power planes where necessary to provide stable reference planes.



PCB Routing

Design Rule Check (DRC)

Run a design rule check to verify that your PCB layout adheres to the manufacturing constraints, such as minimum trace width, spacing, and clearance requirements. The DRC helps identify potential errors or violations that may impact the functionality or manufacturability of the PCB.

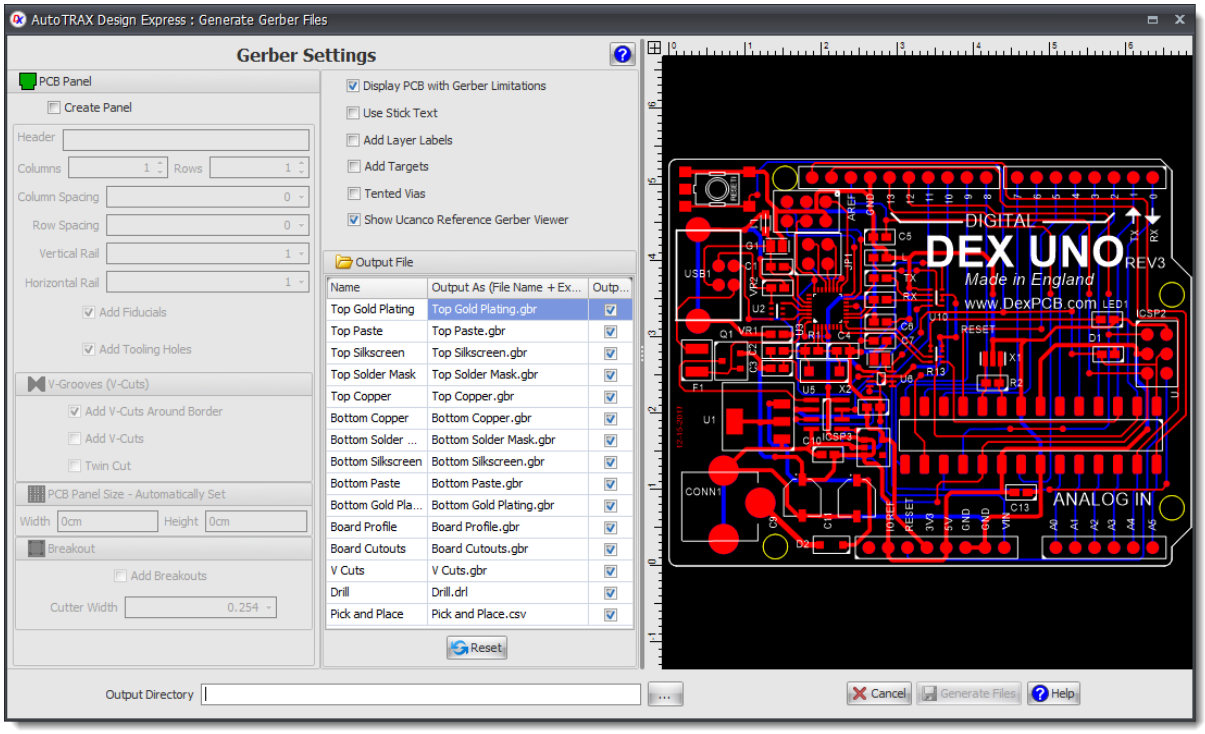


Design Rule Checker

Gerber Files Generation

Once your layout passes the DRC, generate the Gerber files, which are the standard format used by PCB manufacturers to fabricate the board. These files contain all the

necessary information about the copper traces, component placement, drill holes, and other features of the PCB.



Gerber Files Generation

Prototype and Testing

Order a prototype of your PCB from a manufacturer and assemble it with the chosen components. Test the functionality of the circuit, ensuring that it meets your requirements. If any issues are identified, make necessary revisions to the design.

Prototype and Testing

Manufacturing

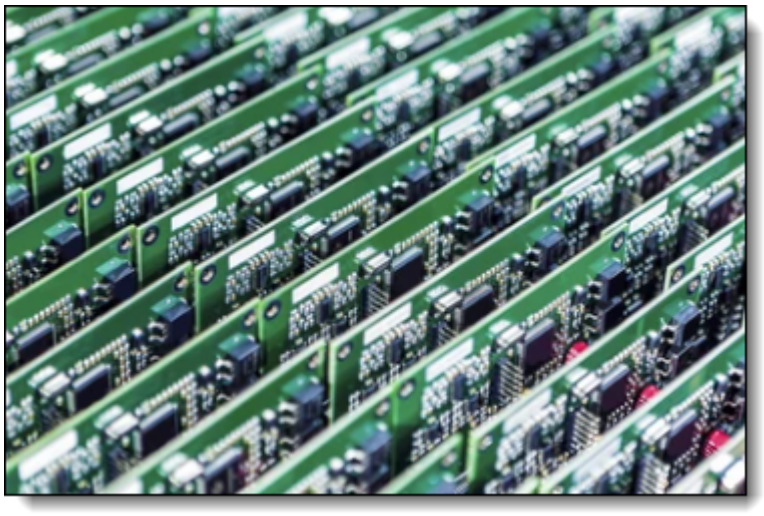
Once you are satisfied with the prototype, you can proceed with manufacturing the PCB in larger quantities. Choose a reliable PCB manufacturer who can meet your quality, cost, and timeline requirements.

PCB Manufacturing

Assembly and Testing

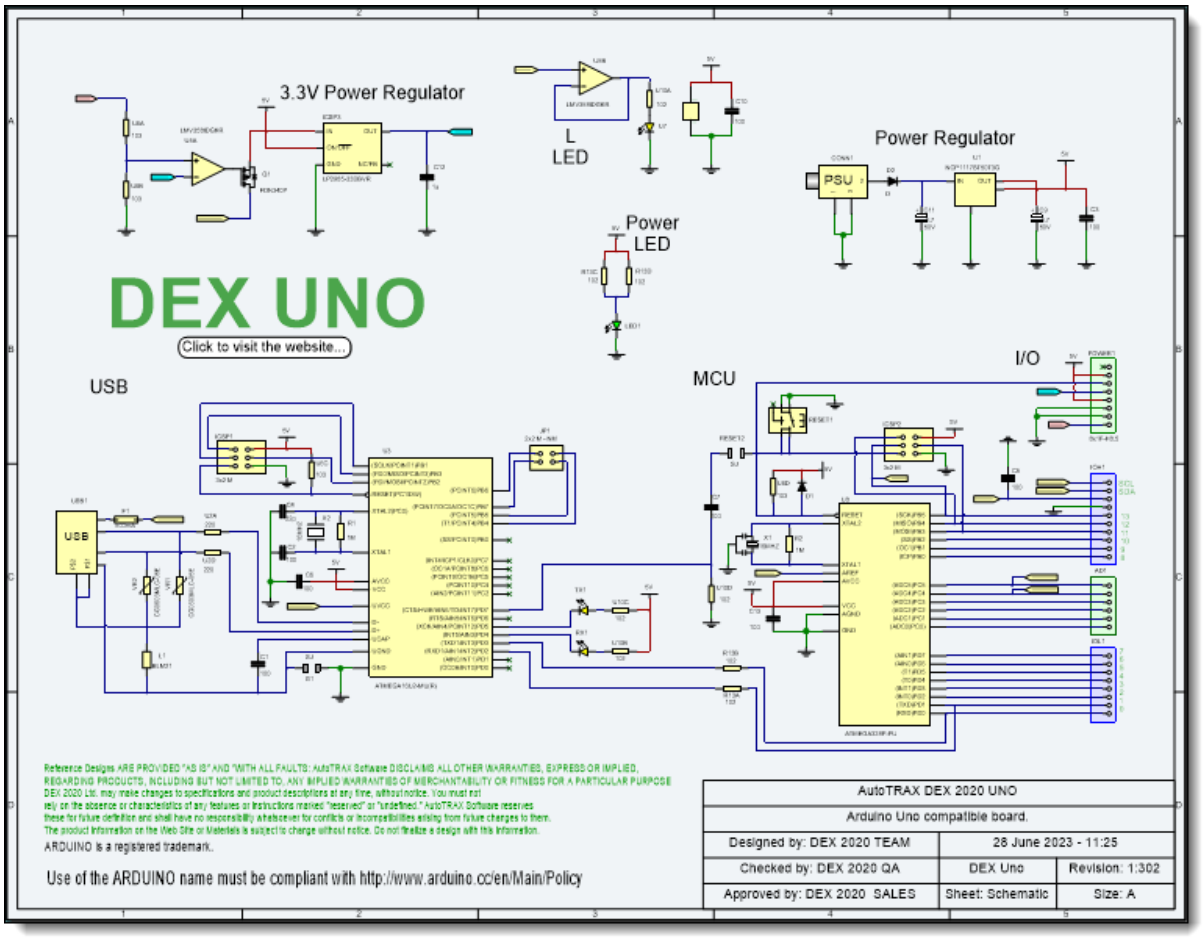
After receiving the manufactured PCBs, populate them with the required components using either manual or automated assembly methods. Perform thorough testing to ensure the final product functions as intended.

It's important to note that designing a PCB can be complex, and it often requires experience and expertise. Consider consulting with an experienced PCB designer or engineer, especially for more complex projects or if you are new to PCB design. Additionally, there are various PCB design software tools available that can assist you in the design process.



PCB Assembly and Testing

2.1 Schematic Design



A Typical Schematic Sheet

Printed design schematics are a type of visual representation used by designers and engineers in order to capture the details of their designs. A schematic typically includes components and how they are connected, as well as electrical connections between various parts. Schematics can be either hand-drawn or computer-generated. Hand-drawn schematics are generally more precise and accurate than those generated by computer software, though they require more time and effort to create. Computer-generated schematics are often easier to read due to the use of standardized symbols, but may lack detail which is necessary for complex designs. In either case, printed design schematics help to communicate complex information quickly and accurately.

Designers use printed design schematics for a variety of purposes. They can be used to plan out a system or device from start to finish, map out individual components or circuit boards, or troubleshoot existing systems. Many types of equipment such as computers and other electronic devices rely on these schematics

in order to operate correctly. When designing an item from scratch, a designer may lay out all the needed components before actually assembling them into one unit. This helps reduce errors that could occur during assembly if the designer had not planned each step carefully beforehand.

The use of printed design schematics also facilitates communication between different parties involved in the project such as engineers, technicians, fabricators and other people who need to understand the plans in order to perform their respective tasks properly. Having a clear diagram of what needs to be done makes it easier for everyone involved to cooperate efficiently and ensure that the end product meets all specifications required for its intended purpose.

Schematic diagrams also provide invaluable insights into how various devices function when assembled together by showing exactly which components interact with each other at a given point in time. Knowing these connections can assist technicians in diagnosing any issues with a system more easily without having to guess as much about how it works internally. Printed design schematics therefore serve as essential tools for both designers and technicians alike when it comes to creating new products and troubleshooting existing ones with efficiency and accuracy.

2.1.1 Schematic Symbols

A PCB (Printed Circuit Board) schematic symbol represents an electronic component in a schematic diagram of an electronic circuit. These symbols are used in circuit diagrams or schematics to depict the actual electronic components.

Here are some examples of commonly used PCB schematic symbols:

- **Resistor:** Usually represented by a zig-zag line.
- **Capacitor:** Shown as two parallel lines, often with one line shorter than the other to represent the polarity.
- **Inductor:** Looks like a series of curved bumps or a spring-like symbol.
- **Transistor:** Generally represented as a three-terminal device with two arrows indicating the direction of current flow.
- **Diode:** Depicted as a triangle pointing to a line, showing the direction of current flow.
- **Integrated Circuit (IC):** Usually represented as a rectangular box with multiple leads.
- **Battery:** Shown as a series of long and short parallel lines, representing the positive and negative terminals respectively.
- **Ground:** Looks like a series of lines that get progressively shorter, symbolizing a connection to the ground or earth.

Each symbol is designed to provide a visual representation of the component, making it easier for engineers and technicians to understand the workings of a circuit. It's important to note that these symbols can sometimes vary between different regions or standards, but the above descriptions should give you a basic understanding of the most common symbols.

In electronic design, a Printed Circuit Board (PCB) schematic symbol is a graphical representation of an electronic component. It helps engineers and designers understand the functionality and connections of the component within the circuit. Each electronic component has a specific schematic symbol that is standardized to ensure consistency across different designs and projects.

There are various standards for PCB schematic symbols, and the most commonly used one is the IEEE (Institute of Electrical and Electronics Engineers) standard, known as IEEE Std 315. This standard provides guidelines and conventions for creating schematic symbols for various electronic components.

When creating a schematic symbol, several important reference points and attributes are included. These references help identify the purpose and characteristics of the component. Some of the essential references found in a PCB schematic symbol include:

- **Component Name:** This reference is usually a text label that identifies the component, such as "Resistor," "Capacitor," "IC," etc.
- **Component Value:** For passive components like resistors, capacitors, and inductors, the component value (e.g., resistance, capacitance, or inductance) is often included as a text label.
- **Pin Designators:** These are unique labels or numbers assigned to each pin of the component. They help identify which pins are connected to other components on the PCB.
- **Pin Types:** Each pin is assigned a pin type, such as input, output, power, ground, etc. This information is essential for correct circuit connections.
- **Pin Names:** In addition to pin designators, pins are usually labeled with names or functional descriptions. For example, a microcontroller might have pins labeled as "VCC" for power supply, "GND" for ground, and specific I/O pins like "TX" and "RX."
- **Polarity Markings:** For components with polarity (e.g., diodes, electrolytic capacitors), polarity markings are included to indicate the correct orientation during PCB layout.
- **Reference Designator:** This is a unique identifier assigned to each component in the schematic. It helps cross-reference the component with its corresponding footprint in the PCB layout.
- **Manufacturer Part Number:** Sometimes, the manufacturer's part number for the component is included as additional information.

It's essential to follow the relevant standards and guidelines when creating PCB schematic symbols to ensure consistency and compatibility with various design tools and manufacturing processes.

Many electronic design automation (EDA) tools, such as the AutoTRAX DEX PCB Designer, come with libraries containing a wide range of pre-defined schematic symbols that comply with industry standards. Designers can use these libraries to build their circuits efficiently. Additionally, manufacturers and component suppliers often provide downloadable libraries with schematic symbols and footprints for their components, which can be used directly in the AutoTRAX DEX PCB Designer.

2.1.2 Schematic Virtual Parts

In the context of electronic design, a "schematic virtual part" typically refers to a representation of a component in a schematic diagram that does not correspond to a physical electronic component. Instead, it serves as a placeholder or a virtual representation of a part that will be defined later or has external dependencies.

Schematic virtual parts are used in various scenarios, including:

- **Future Component Selection:** When designing a circuit, there may be instances where the exact component specifications are not yet determined, or the availability of the specific component is uncertain. In such cases, designers can use a virtual part as a placeholder until they finalize the component selection.
- **Custom Components:** In some designs, custom-made or specialized components may be used, which do not have pre-defined schematic symbols or models in the library. Designers can create virtual parts with custom names and attributes to represent these components in the schematic.
- **External Components:** In complex systems, certain components might reside outside the schematic or even outside the entire design project. Virtual parts allow designers to represent connections or signals to external components while keeping the main schematic organized and concise.
- **Simulation Models:** Some components may require complex behavioral or SPICE simulation models that are not readily available in the standard library. Virtual parts can be used to represent these components while the actual simulation models are created or obtained from external sources.
- **Documentation and Annotations:** Virtual parts can also be used to annotate the schematic with additional information or to indicate specific design requirements, without introducing unnecessary clutter to the diagram.

When using schematic virtual parts, it's crucial to ensure that they are properly documented, and their purpose is clearly communicated with other team members or collaborators. They should be replaced with actual physical components or complete simulation models before finalizing the design or proceeding to the PCB layout phase.

The AutoTRAX DEX PCB Designer supports virtual parts and allow designers to define custom symbols or placeholders with specific attributes to represent components that will be completed or finalized later in the design process.

2.1.3 Schematic Wires and Nodes

In an electronic schematic, wires and nodes are used to represent the connections and junctions between different components. They play a fundamental role in illustrating how a circuit is wired together. Here's a little more detail on what they are:

Wires

Wires in a schematic represent the physical connections between components in an electronic circuit. They're usually drawn as straight lines that connect different components. They don't necessarily correspond to physical wires in the actual circuit; instead, they show how the various components are electrically connected, whether by wires, traces on a printed circuit board (PCB), or some other conductive path.

Nodes

A node, in the context of a circuit diagram or schematic, is a point in the circuit where two or more circuit elements are connected together. This could be a junction between three or more wires, a connection point in a circuit, or a pin on an integrated circuit or other component. In an electronic schematic, nodes are usually represented as a dot where the wires meet. A node is the point at which the potential voltage can be measured.

Note that if two wires cross each other on a schematic without a node being shown, this usually means that they are not electrically connected. However, it's always best to refer to the conventions used in the specific schematic or set of schematics you are working with, as different designers or companies might use slightly different conventions.

2.1.4 What is a Schematic Node?

A schematic node is a point in an electronic schematic diagram where two or more electronic components or connections intersect. It is represented as a dot or a junction point in the schematic.

Schematic diagrams are graphical representations of electronic circuits that use standardized symbols to depict the components and connections in the circuit. The schematic node is used to show that two or more connections or components are electrically connected, without indicating the exact physical layout or routing of the connections.

The schematic node is an important feature of schematic diagrams, as it allows engineers and designers to create complex electronic circuits with multiple

connections and components in a compact and easy-to-read format. The node also helps to identify potential points of failure or signal interference in the circuit, allowing for easier troubleshooting and optimization.

In summary, a schematic node is a point of electrical connection between two or more components or connections in an electronic schematic diagram.

2.1.5 Schematic Node Names

In electronic design, schematic node names, also known as net labels or node labels, are textual identifiers assigned to specific electrical connections or nets in a schematic diagram. Each node name represents an electrical node, which is a point in the circuit where two or more components or wires are connected together.

Schematic node names serve several important purposes in the design process:

Uniqueness: Each node name must be unique within the entire schematic diagram. This uniqueness ensures that each electrical connection is uniquely identified, making it easier to understand the circuit and avoid confusion.

Connectivity: Node names help establish electrical connections between different components and nets. By assigning the same node name to multiple points, those points are electrically linked, indicating that they are part of the same electrical net.

Cross-Referencing: Node names facilitate cross-referencing between different parts of the schematic and can be used to identify connections between different sheets in a multi-sheet design.

Design Rule Checking (DRC): Schematic node names are essential for design rule checking (DRC) in Electronic Design Automation (EDA) tools. DRC verifies that there are no errors, such as unconnected nets or floating wires, by ensuring that all components and connections are correctly labeled.

Signal Naming: Node names are often used to represent signals in the circuit, such as clock signals, data signals, control signals, etc.

Test Points: Node names can be used to identify test points or specific points in the circuit where measurements need to be taken during testing and troubleshooting.

Node names are typically represented as text labels attached to a wire or connection point in the schematic. They can be placed at the midpoint of a wire or near a component pin to indicate the electrical connection between them. In most schematic design tools, you can manually assign node names to connections or use automatic net naming features to generate node names based on predefined rules.

Properly labeled nodes help ensure accurate representation and understanding of the circuit, making the design process more efficient and reducing the likelihood of errors during layout and manufacturing.

2.1.6 What is a Schematic Off-page Connector?

In electronics design, a schematic off-page connector is a graphical symbol used to indicate a connection between different pages of a schematic diagram. It allows a designer to connect different parts of a circuit that are on separate schematic pages, without cluttering the individual pages.

An off-page connector is typically represented by a rectangle or other symbol with a name or identifier that indicates the destination page. The symbol is placed on the edge of the page where the connection originates, and a corresponding symbol with the same name or identifier is placed on the edge of the destination page where the connection terminates.

The use of off-page connectors allows a designer to break up a large, complex circuit into smaller, more manageable pages, while still maintaining a clear understanding of how the various parts of the circuit are connected. It also makes it easier to modify or update the circuit, as changes can be made to individual pages without affecting the rest of the circuit.

When using off-page connectors, it is important to ensure that the names or identifiers used for the connectors are unique and unambiguous. This helps to avoid confusion or errors when interpreting the schematic and can make the design process more efficient and reliable. Additionally, it is important to ensure that the connectors are placed in a logical and organized manner, to make it easier to understand the overall circuit topology.

Off-page connectors and nodes in schematic diagrams are symbols used to represent connections that go to other sheets of the schematic or other sections of the same sheet that aren't immediately adjacent. They are essentially a way to keep schematic diagrams clear and readable, especially for complex circuits that might otherwise be too crowded or confusing.

Off-Page Connectors

These are used in multi-sheet schematics to indicate that a wire or connection continues on another page. An off-page connector is typically represented as a box or a circle with a number or a name inside. The same connector symbol, with the same identifier, will be shown on the page where the connection continues. This way, you can easily trace where the signal is going or coming from by looking for the corresponding off-page connector on the other sheets.

Nodes

Nodes in a schematic represent points in the circuit where two or more components are connected. If the same node name appears in different parts of a schematic (even on different pages), this means these points are all connected together. This is particularly useful in large, complex schematics where it might not be practical to draw all the connections directly.

These conventions help maintain clarity in schematic diagrams, but it's worth noting that different tools or designers might use slightly different symbols or approaches. Always refer to any available key or legend, and when in doubt, ask the person who created the schematic.

2.1.7 Schematic Off-page Connectors

In electronic design, off-page connectors (also known as off-page connectors or off-sheet connectors) are symbols used in schematics to represent connections between different sheets or pages of a multi-sheet circuit design. They provide a way to indicate that a particular signal or net extends to another sheet where the rest of the circuit related to that signal is located.

Off-page connectors are essential for organizing complex designs that cannot fit on a single schematic sheet. They allow designers to break down a large circuit into more manageable sections, each represented on separate sheets. The off-page connectors provide a clear and concise way to show how signals are connected between different sheets.

Key characteristics of off-page connectors:

- **Symbol Representation:** Off-page connectors typically look like small circles or squares placed at the edge of a schematic sheet. Each connector has a unique reference designator or name, allowing it to be easily identified and referenced on other sheets.
- **Uniqueness:** Each off-page connector must have a unique name or reference designator within the entire project. This uniqueness ensures that the correct connection is made between sheets.
- **Connection Lines:** When a signal or net extends to another sheet, a line is drawn from the off-page connector on one sheet to the corresponding off-page connector on the other sheet. This line represents the continuation of the signal's path.
- **Directional Arrows:** To indicate the flow of the signal, directional arrows may be added to the connection lines. These arrows show the direction in which the signal flows from one sheet to another.
- **Sheet Numbers:** Often, sheet numbers or names are included near the off-page connectors to indicate the order of the sheets and aid in navigating the circuit.

Using off-page connectors, designers can maintain clarity and organization in their multi-sheet circuit designs. They make it easier to follow signal paths and understand the overall connectivity of the entire system.

In modern Electronic Design Automation (EDA) software tools, off-page connectors are often automatically generated and managed by the software when multiple sheets are used in a project. Designers can place the connectors, and the software takes care of the connectivity between the sheets, helping to reduce errors and improve design efficiency.

2.1.8 Schematic Buses

In electronics design, a schematic bus connection is a method of representing multiple signals or connections in a single line in a schematic diagram. It is a convenient way to simplify the schematic and reduce clutter, while still conveying all the necessary information about the connections.

A schematic bus connection is typically represented by a single line with multiple branches, each branch representing a separate connection or signal. The branches are usually labeled with unique identifiers or names, which indicate the specific signals or connections that they represent.

A bus connection can be used to represent multiple signals that are related or connected in some way, such as the address lines in a microprocessor bus or the data lines in a serial communication protocol. By using a bus connection, the schematic can be simplified and made easier to read, without sacrificing the necessary information about the connections.

When designing a schematic with bus connections, it is important to ensure that the labels or names used for the branches of the bus are clear and unambiguous. This helps to avoid confusion or errors when interpreting the schematic and can make the design process more efficient and reliable.

2.1.9 Graphics in Schematics

Graphics in schematics refer to non-electrical elements or visual aids used to enhance the clarity, readability, and understanding of the schematic diagram. While the primary focus of a schematic is to represent electrical connections and components, the inclusion of graphics can make the design more informative and accessible to designers, engineers, and other stakeholders.

Some common uses of graphics in schematics include:

Block Diagrams: Block diagrams are graphical representations of functional blocks within a circuit. They help provide a high-level overview of the circuit's architecture and organization, making it easier to understand the flow of signals and information between different sections.

Illustrations and Icons: Graphics can be used to represent physical components or system elements with icons or illustrations. For instance, an amplifier can be represented with an icon that resembles the physical shape of an amplifier.

Signal Flow Arrows: Arrows can be added to indicate the direction of signal flow within the circuit. This is especially useful for complex circuits where signal paths are not immediately apparent.

Callouts and Annotations: Callouts or annotations can be placed on specific parts of the schematic to provide additional information or explanations about certain components or connections.

Notes and Comments: Text boxes or graphical elements can be used to add notes or comments to the schematic, explaining design choices, providing instructions, or noting design considerations.

Color Coding: Using different colors for wires or specific components can help distinguish different sections of the circuit or highlight critical parts.

Backgrounds and Grids: Adding backgrounds or grids can improve the visual organization of the schematic and make it easier to align components and wires.

Title Blocks: Including a title block at the top or bottom of the schematic can provide essential project information, such as the title, author, revision, and date.

It's important to use graphics judiciously and avoid overloading the schematic with unnecessary visual elements that may clutter the design or obscure the primary electrical information. The primary goal of any schematic is to accurately represent the electrical connections and components in the circuit. Graphics should enhance the understanding of the design rather than detract from it.

The AutoTRAX DEX PCB DEesigner provides features to add and manage graphics in schematics, allowing designers to create professional and informative schematic diagrams for their electronic designs.

2.1.9.1 Line Styles

n schematic diagrams, line styles are used to differentiate between different types of connections or nets, making the circuit diagram more visually informative and easier to understand. Each line style has a specific meaning or purpose, and designers use them to represent various types of signals, power connections, or other special elements within the circuit.

Here are some common schematic graphics line styles:

Solid Lines: Solid lines are the default and most commonly used line style in schematic diagrams. They represent regular electrical connections between components and nets.

Dashed Lines: Dashed lines are often used to represent hidden or invisible connections that are not explicitly shown on the schematic but are understood to exist.

Dotted Lines: Dotted lines can be used to indicate non-electrical connections or to show relationships that are not part of the main circuit but are relevant to the design.

Bold Lines: Bold lines are used to highlight important or critical connections in the schematic, such as power supply lines or high-current paths.

Arrowed Lines: Arrow symbols can be added to lines to indicate signal direction, flow, or transmission. This is particularly useful in complex circuits to show the path of data or control signals.

Curved Lines: Curved lines are sometimes used to avoid overlapping with other components or to create a more aesthetically pleasing schematic layout.

Vertical and Horizontal Lines: In some cases, designers use vertical or horizontal lines to represent long buses or signal paths to avoid excessive crossing lines and improve readability.

Zigzag Lines: Zigzag lines can be used to represent specific components like varistors or voltage-dependent resistors (VDRs) in surge protection circuits.

It's essential to use line styles consistently throughout the schematic to maintain clarity and avoid confusion. Standardized line styles help make schematics more readable and reduce the chances of errors during the design process.

Modern Electronic Design Automation (EDA) software tools often provide a variety of line styles, and designers can choose the appropriate style for each type of connection or element in the schematic. Additionally, EDA software may have customizable line style settings, allowing designers to adjust line widths, colors, and other properties according to their preferences or specific design requirements.

2.1.9.2 Fill Styles

In schematic diagrams, fill styles are used to visually differentiate or highlight specific areas or components within the circuit. Unlike line styles, which affect the appearance of connections and nets, fill styles pertain to the interior regions of components or shapes on the schematic. Fill styles can help improve the clarity of the diagram, emphasize certain elements, or convey additional information about the circuit design.

Here are some common schematic graphics fill styles:

Solid Fill: Solid fill is the most basic and commonly used fill style. It involves filling the interior of a component or shape with a solid color, typically representing a standard electrical connection or component.

Hatching: Hatching involves filling the interior of a component or shape with diagonal or parallel lines. This fill style is often used to indicate components that are not yet defined or need further specification.

Patterned Fill: Patterned fills use various patterns or textures to fill the interior of components or shapes. These patterns can represent different materials, layers, or characteristics of the components.

Color Fill: Different colors can be used to fill components or shapes to distinguish between different functional blocks or signal paths. For example, power-related components may be filled with a specific color, while data-related components use a different color.

Gradient Fill: Gradient fill involves blending two or more colors together to create a smooth transition from one color to another. This style is often used for visual appeal or to indicate a gradual change in characteristics.

Transparent Fill: Sometimes, designers use transparent fills to make the interior of a shape partially transparent. This style can be used to show overlapping components without completely obscuring them.

Symbol Background Fill: In some cases, the background of symbols or icons representing specific components may have a fill style to differentiate them from regular components.

No Fill: The absence of any fill style is also a valid option, especially for components that don't require specific highlighting or differentiation.

Using fill styles effectively can improve the visual organization and understanding of complex schematic diagrams. However, it's important not to overuse or clutter the schematic with excessive fill styles, as it may lead to confusion or distract from the main purpose of the diagram.

Modern Electronic Design Automation (EDA) software tools often provide various fill style options, and designers can select the appropriate style for each component or shape on the schematic. Additionally, EDA software may allow customization of fill colors, patterns, and transparency to meet specific design requirements and visual preferences.

2.1.9.3 Text Fonts and Colors

In schematic diagrams, text fonts and colors are essential elements used to convey information and improve the readability and visual appeal of the circuit design. Properly chosen fonts and colors can make the schematic more understandable, organized, and easier to interpret. They help distinguish different elements and highlight critical information in the design.

Here are some considerations for schematic graphics text fonts and colors:

Text Fonts: Selecting an appropriate font is crucial for readability. Fonts should be clear, legible, and easy to read, even when the schematic is printed in smaller sizes. Commonly used fonts in schematics include Arial, Helvetica, Times New Roman, Verdana, and Calibri.

Text Sizes: Text sizes should be consistent and suitable for the specific application. Title texts and labels for important components may have larger font sizes, while annotations and less critical information may have smaller font sizes.

Bold and Italic: Using bold or italic text can help emphasize important information, titles, or section headings in the schematic.

Colors

Choosing appropriate colors can help organize the schematic and distinguish different types of elements. For instance:

- **Black:** Typically used for most text and component outlines.
- **Red:** Often used for power-related information, such as power supplies and voltage references.
- **Blue:** Commonly used for net names or signal labels.
- **Green:** Frequently used for ground connections and ground symbols.
- **Brown:** Used for component reference designators.
- **Gray:** Sometimes used for annotations or non-critical information.

- **Contrast:** Ensure sufficient contrast between the text and background colors to avoid readability issues.
- **Consistency:** Maintaining consistent font styles and colors throughout the schematic improves its visual coherence and readability.
- **Text Alignment:** Align text neatly to improve the appearance and organization of the schematic.
- **Annotations:** Use text annotations to provide additional information, notes, or explanations about specific parts of the design.
- **Text Orientation:** In some cases, text orientation may be adjusted to fit in tight spaces or for better visual alignment.

It's essential to use fonts and colors judiciously to avoid overwhelming the schematic with too many variations, which may lead to confusion. Strive for a balanced and visually appealing layout that clearly conveys the necessary information.

The AutoTRAX DEX PCB Designer offer a range of font styles, sizes, and color options, allowing designers to customize the appearance of text elements in their schematics. Additionally, software tools often include design rule checks (DRC) to ensure proper font sizing and readability in the final output.

2.1.10 Hierarchical Design

Enter topic text here.

2.1.11 Multiple Symbols for a Part

it is possible for a single electronic component to have multiple symbols, especially when the component has different functions or configurations. The use of multiple symbols for a part allows designers to represent various operating modes, package types, or other variations of the same component.

Here are some scenarios where a single part may have multiple symbols:

Multi-Functional Components: Some components can perform different functions based on how they are used in the circuit. For example, an operational amplifier (op-amp) can be used as an inverting amplifier, non-inverting amplifier, integrator, differentiator, etc. Each configuration may have a unique symbol to represent its specific functionality.

Variable Pin Configurations: Some ICs or components come in packages with variable pin configurations. For instance, a microcontroller may be available in different package types, such as Dual Inline Package (DIP), Surface Mount Device (SMD), or Quad Flat Package (QFP), and each package may have a different pin arrangement. Each package type would require a specific symbol to represent its pin layout.

Polarized Components: Polarized components like diodes, electrolytic capacitors, and LEDs have distinct symbols to indicate their polarity. Depending on how they are oriented in the circuit, the symbols may differ.

Multi-Purpose ICs: Integrated circuits (ICs) that offer multiple functions in one chip (e.g., programmable logic devices, microcontrollers, or system-on-chip solutions) may have different symbols to represent each function or mode of operation.

Switches and Relays: Switches and relays can be represented differently depending on whether they are normally open, normally closed, single-throw, double-throw, etc.

Transistors: Different transistor types (e.g., NPN, PNP, enhancement mode, depletion mode, etc.) can have distinct symbols to differentiate their characteristics.

When using multiple symbols for a single part, it is crucial to maintain clarity and consistency within the design. Proper documentation and annotation should accompany the symbols to ensure that the intended functionality or configuration is understood by everyone involved in the design process.

Most Electronic Design Automation (EDA) software tools offer libraries with various symbols for common electronic components, including those with multiple configurations. Designers can select the appropriate symbol from the library and associate it with the specific part variant they are using in their design. This way, the

schematic accurately reflects the intended functionality and configuration of the components throughout the circuit.

2.1.12 Schematic Sub-systems

n electronic design, a schematic sub-system refers to a portion of a larger circuit or system that is represented as a separate schematic diagram. Schematic sub-systems are commonly used to break down complex designs into manageable and modular sections, making the overall design process more organized and easier to understand. Each sub-system focuses on specific functionality or a group of related components.

Here are some key points to understand about schematic sub-systems:

- **Modularity:** Schematic sub-systems promote modularity in circuit design. By breaking down a complex design into smaller, interconnected sub-systems, engineers can focus on specific functions independently. This allows for easier testing, troubleshooting, and changes to individual sub-systems without affecting the entire circuit.
- **Functional Separation:** Sub-systems typically represent individual functional blocks of a larger system. For example, in an audio amplifier design, sub-systems might include the input stage, pre-amplification stage, power amplification stage, and output stage. Each sub-system focuses on its unique task and can be designed and tested separately.
- **Hierarchical Design:** Schematic sub-systems enable hierarchical design methodologies, where a large system is represented at different levels of abstraction. Each sub-system can contain further sub-systems, creating a hierarchy that simplifies complex designs and enhances readability.
- **Reuse and Collaboration:** Sub-systems can be reused in different projects and shared among design teams. Once a sub-system is thoroughly tested and validated, it can serve as a building block for future designs, saving time and effort in the design process.
- **Abstraction and Complexity Management:** By representing complex circuits as sub-systems, designers can focus on specific levels of abstraction. Higher-level schematics show the interconnections between sub-systems, while lower-level schematics delve into the internal details of each sub-system.
- **Documentation and Communication:** Schematic sub-systems aid in documentation and communication. Engineers can create separate documentation for each sub-system, making it easier for team members and stakeholders to understand specific aspects of the design.

- **Interface Definition:** Sub-systems have well-defined interfaces, including input and output connections, which allow seamless integration into the larger system. This reduces the chances of errors and compatibility issues during integration.

The AutoTRAX DEX PCB Designer tools support hierarchical schematic design, making it easy to create and manage sub-systems within a larger project.

It's essential to plan the organization of sub-systems carefully and consider the interactions between different blocks when designing a complex circuit. Good sub-system organization improves design efficiency, maintainability, and scalability in electronic projects.

2.1.13 Schematic Dangling Wires

In electronic design, "dangling wires" refer to wires or nets in a schematic that are not connected to any component or circuit element. They are also known as "floating wires" or "unconnected nets." Dangling wires can occur due to various reasons, and they can lead to confusion, errors, or incomplete circuit functionality during the PCB layout or simulation phases.

Here are some common reasons why dangling wires might appear in a schematic:

- **Incomplete Connections:** The designer may have forgotten to connect one or more ends of a wire to a component or another net. This can happen accidentally during the schematic creation process, especially in complex designs.
- **Misalignment or Overlapping:** Sometimes, wires may appear to be connected visually, but they are not electrically connected. This can happen if wires are not properly aligned or overlap without forming a proper electrical junction.
- **Misplaced or Missing Components:** Dangling wires can result from missing components that were intended to be connected but were not placed on the schematic. Similarly, if a component is placed incorrectly or accidentally removed, it can lead to unconnected nets.
- **Disconnection during Editing:** During schematic editing or modifications, wires may get disconnected accidentally if care is not taken to maintain proper connectivity.
- **Unused Pins:** Some components may have extra pins that are not used in the specific circuit, leading to apparent dangling wires.

It is essential to eliminate dangling wires from a schematic to ensure the design is accurate, easy to understand, and free of potential errors. Here are some steps to handle dangling wires:

- **Review and Edit:** Carefully review the schematic for any dangling wires and edit the connections where necessary. Ensure that each wire is appropriately connected to the relevant components or nets.

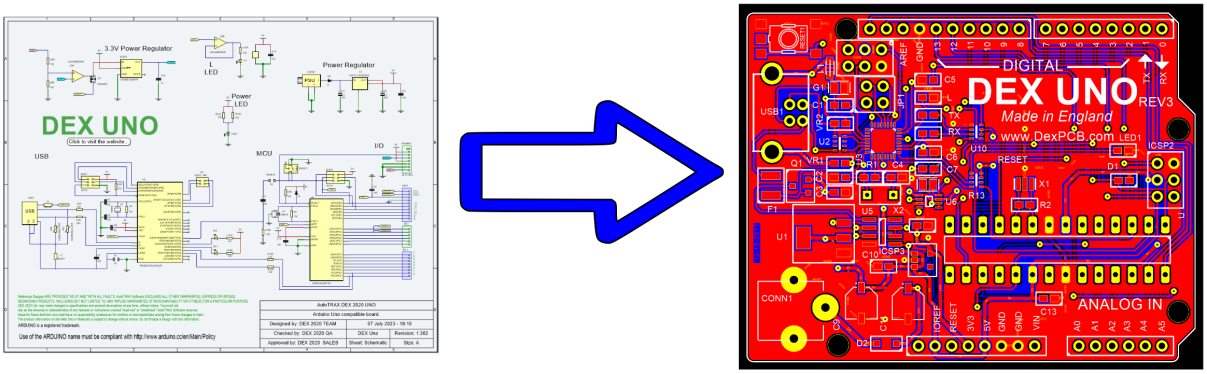
- **Use Net Labels:** Net labels help define electrical connections without the need for a continuous wire. Use net labels to connect different parts of a circuit that should be electrically linked but are not directly connected with a wire.
- **Avoid Overlapping:** Pay attention to the placement of components and wires to prevent accidental overlapping, which can lead to visually connected but unconnected nets.
- **Clean Up Unused Components:** If there are unused or redundant components, consider removing them from the schematic to eliminate any potential dangling wires.
- **Use Design Rule Checking (DRC):** Many schematic design tools include DRC features that can automatically detect and highlight unconnected nets or other connectivity issues.
- **Simulation and Validation:** After eliminating dangling wires, perform simulation and validation to ensure that the circuit functions as intended.

By taking care to eliminate dangling wires, designers can create more accurate and reliable schematic representations of their circuits, reducing the likelihood of errors during PCB layout and manufacturing.

In the AutoTRAX DEX PCB Designer is is impossible to have a dangling wire.

2.1.14 Foward Annotation

PCB forward annotation is a process in electronic design automation (EDA) software that involves sending design changes from the schematic capture stage to the PCB layout stage.



Schematic to PCB Forward Annotation

Here's a typical flow for it:

Design Capture

During this phase, an electrical engineer uses a software tool to draw the schematic of the circuit. This includes all the components (like resistors, capacitors, microcontrollers, etc.), their connections, and values/properties. Each component is also assigned a unique reference designator, like R1, C1, U1, etc.

Initial PCB Layout

After the schematic is completed, it is transferred to a PCB layout tool. This process is called forward annotation. The software tool transfers all components, their reference designators, and connections (netlist) to the PCB layout environment. The PCB designer uses this information to place components on the board and route traces to make the necessary electrical connections.

Design Changes

Often, changes will be made to the design after the initial PCB layout has begun. Maybe a component value needs to be changed, or a part needs to be swapped out for a different one. These changes are made in the schematic capture tool.

Forward Annotation

Once changes have been made to the schematic, they need to be updated in the PCB layout. This process is also called forward annotation. The schematic capture tool will generate a report of changes, and the PCB layout tool will apply these changes to the layout.

Back Annotation

Sometimes, changes may also be made in the PCB layout tool - for example, if a component needs to be moved for space constraints. This change then needs to be reflected back in the schematic, a process called back annotation. This ensures that the schematic and PCB layout remain consistent with each other.

Note that the terms forward annotation and back annotation come from the idea that you are "annotating" or "marking" changes to be transferred "forward" from schematic to layout, or "back" from layout to schematic. Not all EDA software uses the exact terms, but the concepts are fundamental to the process of electronic design.

2.2 Spice Simulation

Bipolar Junction Transistors

2.2.1 Devices

In SPICE (Simulation Program with Integrated Circuit Emphasis), device models are used to simulate the behavior of various electronic components and semiconductor devices in electronic circuits. These models describe how different components respond to applied voltages, currents, and other parameters under different operating conditions. Device models enable accurate circuit simulations and help designers predict circuit behavior before physical implementation.

SPICE provides a variety of device models for different types of components, including resistors, capacitors, diodes, transistors (MOSFETs, BJTs, JFETs, etc.), inductors, voltage sources, current sources, and more. These models are typically defined in SPICE netlists using specific syntax and parameters.

Resistors

Resistors

Inductors

Diodes

Bipolar Junction Transistors

JFET

MESFET

MOSFET

2.2.1.1 Resistors

In SPICE (Simulation Program with Integrated Circuit Emphasis), a resistor model is used to simulate the behavior of resistors in electronic circuits. While resistors are relatively simple components, their models in SPICE help accurately capture their behavior under various conditions, including DC and AC analyses.

There are different resistor models available in SPICE, with varying levels of complexity and accuracy. One of the commonly used models is the simple linear resistor model, which follows Ohm's law and represents the resistor as a linear relationship between voltage across the resistor.

It's important to note that SPICE resistor models are typically idealized representations of real-world resistors. If you need to model the effects of

temperature, manufacturing variations, or other non-ideal behaviors, you may need to use more advanced resistor models or include additional components in your simulation. Additionally, SPICE resistor models assume that the resistors are linear and do not account for non-linear behavior in certain types of resistive materials.

2.2.1.2 Inductors

In SPICE (Simulation Program with Integrated Circuit Emphasis), inductors are electronic components that model the behavior of real-world inductors in electronic circuits. Inductors store energy in a magnetic field when current flows through them and resist changes in current.

SPICE provides various inductor models with different levels of complexity to simulate the behavior of inductors accurately. The two primary types of inductor models in SPICE are ideal inductor models and more advanced models that consider parasitic effects and frequency-dependent behavior.

Ideal Inductor Model: The ideal inductor model is the simplest representation of an inductor in SPICE. It assumes that the inductor has no resistance, no magnetic saturation, and no parasitic capacitance. The ideal inductor model only considers the inductance value.

SPICE inductor models are typically used in simulations to analyze the behavior of inductors in various circuit configurations, such as inductors in filters, transformers, oscillators, and more. Depending on the level of accuracy required for your simulation and the complexity of the circuit, you can choose the appropriate inductor model that suits your design needs.

2.2.1.3 Diodes

In SPICE (Simulation Program with Integrated Circuit Emphasis), diodes are modeled to simulate the behavior of real-world diodes in electronic circuits. Diodes are semiconductor devices that allow current to flow in one direction while blocking it in the opposite direction. SPICE provides various diode models to accurately capture the behavior of diodes under different operating conditions.

The most common types of diode models in SPICE include ideal diode models and more advanced models that consider non-ideal characteristics, temperature effects, and more. Here are some key diode models in SPICE:

Ideal Diode Model

The ideal diode model is the simplest representation of a diode in SPICE. It assumes that the diode has an infinite reverse resistance (blocks all reverse current) and zero forward voltage drop (conductive when forward-biased). The ideal diode model does not consider reverse recovery time or other non-ideal effects.

Non-Ideal Diode Models

More advanced diode models in SPICE consider non-ideal characteristics, such as reverse recovery time, temperature effects, and series resistance. These models provide a more accurate representation of diode behavior over a wide range of conditions.

Some of the advanced diode models include:

Diode Level 1 Model: This model includes the diode's saturation current, reverse breakdown voltage, and parasitic series resistance. It is suitable for simple diode behavior analysis.

Diode Level 3 Model: This model is more comprehensive and includes additional parameters to account for temperature effects, high-frequency behavior, and more accurate reverse recovery characteristics.

Schottky Diode Model: Schottky diode models are used for Schottky barrier diodes, which have different characteristics compared to regular PN-junction diodes.

Diode models in SPICE are used to analyze diode behavior in various circuit applications, such as rectifiers, voltage clamps, voltage regulators, and more. Depending on the accuracy required for your simulation and the complexity of the circuit, you can choose the appropriate diode model that suits your design needs. Keep in mind that while ideal diode models are simple to use, they do not capture all the nuances of real-world diode behavior.

2.2.1.4 Bipolar Junction Transistors

In SPICE (Simulation Program with Integrated Circuit Emphasis), a BJT (Bipolar Junction Transistor) model is used to simulate the behavior of bipolar transistors, which are semiconductor devices commonly used in electronic circuits for amplification and switching. BJT models in SPICE describe the electrical characteristics of the transistor, including its current-voltage relationships, capacitances, and other parameters that affect its operation.

There are various BJT models available in SPICE, each with its level of complexity and accuracy. Some of the most commonly used BJT models in SPICE include:

Ebers-Moll Model

This is a basic and widely used model that represents the BJT as a current-controlled current source. It captures the fundamental characteristics of a bipolar transistor, including the relationships between base current, collector current, and emitter current.

The Ebers-Moll model is a simplified but fundamental model used to describe the behavior of bipolar junction transistors (BJTs) in electronics. It provides a basic representation of how current flows through the various regions of a BJT: the emitter, base, and collector. The Ebers-Moll model is widely used in SPICE simulations and other electronic design tools to analyze and predict the behavior of bipolar transistors.

The model is named after its developers, John S. Ebers and John L. Moll, who introduced it in the early 1950s. It focuses on the relationships between currents and voltages in a BJT and is based on the following key assumptions:

The BJT operates in the active region, where both the emitter-base junction and the collector-base junction are forward-biased.

The BJT operates in a small-signal AC regime, where variations in current and voltage are small around a quiescent (DC) operating point.

The Ebers-Moll model provides insights into the relationships between the voltages and currents in a BJT. It's a good starting point for understanding BJT behavior and can be useful for basic analysis, such as gain calculations and determining bias points. However, the Ebers-Moll model has limitations and does not capture all the complexities of real-world BJT behavior, such as high-frequency effects, parasitic capacitances, and other non-ideal characteristics.

For more accurate and detailed simulations, higher-level BJT models like the Gummel-Poon model or advanced models are used. These models take into account additional parameters and effects for more precise analysis of transistor behavior.

Gummel-Poon Model

The Gummel-Poon model is an extension of the Ebers-Moll model and is a more comprehensive and accurate representation of bipolar junction transistors (BJTs) in electronic circuit simulations. Named after its developers, Hermann Gummel and John Poon, this model includes additional parameters and effects to better describe the behavior of BJTs in various operating conditions.

The Gummel-Poon model enhances the Ebers-Moll model by accounting for several factors that affect BJT operation, such as:

- **High-Frequency Effects:** The Gummel-Poon model introduces capacitances and transit time effects, which are important in high-frequency operation. These effects are not captured by the basic Ebers-Moll model.
- **Early Voltage :** The Early voltage is included in the Gummel-Poon model to account for the Early effect, where an increase in collector current leads to a reduction in the effective base width. This effect affects the output characteristics of the transistor.

- **Base Resistance:** The Gummel-Poon model includes a base resistance term to account for the resistive nature of the base region. This resistance affects the input characteristics and voltage-divider action in the BJT.
- **Temperature Effects:** The Gummel-Poon model includes parameters that account for the temperature dependence of various transistor parameters, which is essential for accurate simulations over a range of temperatures.
- **Emitter Saturation Current :** The Gummel-Poon model introduces the emitter saturation current, which is more accurate than the Is parameter in the Ebers-Moll model.
- **Additional Model Parameters:** The Gummel-Poon model includes additional parameters to capture various non-ideal effects, making it suitable for a wider range of BJT applications.

The Gummel-Poon model provides a more realistic representation of BJT behavior and is particularly useful for analyzing BJT performance in various biasing and operational conditions, including high-frequency operation. However, it is also more complex and requires more parameter values to be accurately defined compared to the Ebers-Moll model.

Modified Gummel-Poon Model (MGP)

The Modified Gummel-Poon (MGP) model is an advanced extension of the Gummel-Poon model for simulating bipolar junction transistors (BJTs) in electronic circuit designs. Like the Gummel-Poon model, the MGP model includes additional parameters and effects to better capture the behavior of BJTs under various operating conditions. The MGP model further refines and enhances the accuracy of the simulation results compared to the Gummel-Poon model.

The key improvements introduced by the Modified Gummel-Poon model include:

- **Improved Temperature Dependencies:** The MGP model includes more accurate temperature dependencies for various parameters, making it suitable for simulations over a wide range of temperatures.
- **More Detailed Transit Time Modeling:** The transit time effects in the MGP model are modeled with increased accuracy, accounting for the transit time variations with collector current and voltage.
- **Advanced Base Resistance Modeling:** The MGP model introduces a more sophisticated representation of the base resistance and its impact on transistor performance.
- **Enhanced Early Voltage Model:** The Early voltage model in the MGP model is designed to provide better accuracy over a wider range of operating conditions.

- **Bias-Dependent Current Sources:** The MGP model includes bias-dependent current sources that consider the impact of operating conditions on transistor behavior.
- **Higher Accuracy at High Frequencies:** The MGP model provides improved accuracy in high-frequency simulations compared to the standard Gummel-Poon model.

The Modified Gummel-Poon model addresses some of the limitations and inaccuracies associated with the Gummel-Poon model, especially in scenarios involving temperature variations, high-frequency operation, and complex biasing conditions. It is a suitable choice for accurate simulations of BJTs in various electronic circuits, including analog amplifiers, RF circuits, and mixed-signal designs.

It's important to note that as the complexity of the model increases, so does the number of parameters that need to be accurately characterized for specific transistors. Therefore, careful attention to model parameter extraction and verification is crucial for obtaining reliable simulation results.

Designers often select the appropriate level of BJT model based on the complexity of their design, the accuracy required, and the level of detail necessary to capture the transistor's behavior under different conditions

Level-1, Level-2, Level-3, etc. Models

These models are increasingly complex and accurate, incorporating more parameters to represent various aspects of transistor behavior in greater detail. Higher-level models consider effects like parasitic capacitances, transit time effects, and temperature dependencies.

When using a BJT model in SPICE simulation, you typically need to provide parameters that define the transistor's characteristics, such as doping concentrations, junction area, and transit times. The accuracy of the simulation depends on the choice of the model and the accuracy of the provided parameters.

To simulate a circuit containing BJTs in SPICE, you generally need to:

Include the appropriate BJT model in your SPICE netlist.

Specify the model parameters based on the specific transistor you are using.

Connect the transistor in the circuit with proper biasing and connections.

Set up your simulation (DC analysis, AC analysis, transient analysis, etc.).

Run the simulation to observe the behavior of the BJT and the entire circuit.

Keep in mind that BJT models are part of a larger SPICE simulation setup, and the accuracy of the simulation depends on the quality of the model, the chosen simulation settings, and the accuracy of the provided transistor parameters. It's

important to refer to the documentation of your specific SPICE software for information on BJT models and their usage.

2.2.1.5 JFET

In SPICE (Simulation Program with Integrated Circuit Emphasis), Junction Field-Effect Transistors (JFETs) are modeled to simulate the behavior of these semiconductor devices in electronic circuits. JFETs are three-terminal devices that use an electric field to control the conductivity of a semiconductor channel, making them useful for switching and amplification applications.

SPICE provides various JFET models to accurately represent their behavior under different operating conditions. The most common types of JFET models in SPICE include ideal JFET models and more advanced models that consider non-ideal characteristics, temperature effects, and more.

Here are some key JFET models in SPICE:

Ideal JFET Model

The ideal JFET model is a simplified representation of a JFET that captures its basic behavior. It assumes that the JFET operates in its linear (ohmic) region when biased and provides a constant current when on.

Non-Ideal JFET Models

More advanced JFET models in SPICE consider non-ideal characteristics such as pinch-off voltage variations, channel resistance, temperature effects, and more.

Some of the advanced JFET models include:

- **JFET Level 1 Model:** This model includes parameters such as transconductance, pinch-off voltage, and channel resistance to provide a more accurate representation of JFET behavior.
- **JFET Level 3 Model:** This model is more comprehensive and includes additional parameters to account for temperature effects, parasitic capacitances, and channel length modulation.

JFET models in SPICE are used to analyze JFET behavior in various circuit applications, such as amplifiers, switches, and voltage-controlled resistors. Depending on the accuracy required for your simulation and the complexity of the circuit, you can choose the appropriate JFET model that suits your design needs. Keep in mind that while ideal JFET models are simple to use, they may not capture all the nuances of real-world JFET behavior.

2.2.1.6 MESFET

In SPICE (Simulation Program with Integrated Circuit Emphasis), MESFETs (Metal-Semiconductor Field-Effect Transistors) are modeled to simulate the behavior of these semiconductor devices in electronic circuits. MESFETs are a type of field-effect transistor that uses the modulation of a depletion layer within a semiconductor channel to control current flow.

SPICE provides various MESFET models to accurately represent their behavior under different operating conditions. MESFET models in SPICE include both idealized and more advanced models that consider non-ideal characteristics, temperature effects, and other relevant parameters.

Here are some key MESFET models in SPICE:

Ideal MESFET Model

The ideal MESFET model is a simplified representation of a MESFET that captures its basic behavior. It assumes that the MESFET operates in its linear (ohmic) region when biased and provides a constant current when on.

Non-Ideal MESFET Models

More advanced MESFET models in SPICE consider non-ideal characteristics such as channel length modulation, mobility variation, gate-source capacitance, and temperature effects.

Some of the advanced MESFET models include:

- **MESFET Level 1 Model:** This model includes parameters such as transconductance, threshold voltage, and channel length modulation to provide a more accurate representation of MESFET behavior.
- **MESFET Level 3 Model:** This model is more comprehensive and includes additional parameters to account for mobility variation, gate-source capacitance, and temperature effects.

MESFET models in SPICE are used to analyze MESFET behavior in various circuit applications, such as microwave and RF amplifiers, oscillators, mixers, and more. Depending on the accuracy required for your simulation and the complexity of the circuit, you can choose the appropriate MESFET model that suits your design needs. Keep in mind that while ideal MESFET models are simple to use, they may not capture all the nuances of real-world MESFET behavior.

2.2.1.7 MOSFET

In SPICE (Simulation Program with Integrated Circuit Emphasis), MOSFETs (Metal-Oxide-Semiconductor Field-Effect Transistors) are modeled to simulate the behavior of these semiconductor devices in electronic circuits. MOSFETs are a type of field-

effect transistor that uses the electric field generated by a gate terminal to control the flow of current between the source and drain terminals..

SPICE provides various MOSFET models to accurately represent their behavior under different operating conditions. MOSFET models in SPICE include idealized models as well as more advanced models that consider non-ideal characteristics, temperature effects, and other relevant parameters.

Here are some key MOSFET models in SPICE:

Ideal MOSFET Model

The ideal MOSFET model is a simplified representation of a MOSFET that captures its basic behavior. It assumes that the MOSFET operates in its linear (ohmic) region when biased and provides a constant current when on.

Non-Ideal MOSFET Models

More advanced MOSFET models in SPICE consider non-ideal characteristics such as channel length modulation, mobility variation, gate-source capacitance, temperature effects, and more.

Some of the advanced MOSFET models include:

- **MOSFET Level 1 Model:** This model includes parameters such as transconductance, threshold voltage, and channel length modulation to provide a more accurate representation of MOSFET behavior.
- **MOSFET Level 3 Model:** This model is more comprehensive and includes additional parameters to account for mobility variation, gate-source capacitance, and temperature effects.
- **BSIM (Berkeley Short-Channel IGFET Model):** This is a family of advanced MOSFET models developed at UC Berkeley. BSIM models provide even greater accuracy and include additional features for modeling submicron MOSFET behavior.

MOSFET models in SPICE are used to analyze MOSFET behavior in various circuit applications, such as amplifiers, switches, analog circuits, digital circuits, and more. Depending on the accuracy required for your simulation and the complexity of the circuit, you can choose the appropriate MOSFET model that suits your design needs. Keep in mind that while ideal MOSFET models are simple to use, they may not capture all the nuances of real-world MOSFET behavior.

2.2.1.8 Switches

In SPICE (Simulation Program with Integrated Circuit Emphasis), switches are components that can be used to represent the opening or closing of a connection in a circuit. Switches allow you to model changes in circuit topology over time, such as

toggling a component on or off, changing the path of current, or introducing and removing components based on certain conditions. SPICE provides various types of switches that you can use in your circuit simulations.

Here are the common types of switches in SPICE:

Voltage-Controlled Switch (S)

The voltage-controlled switch models a switch that opens or closes based on the voltage across its controlling nodes.

Syntax: S<name> <node1> <node2> <control_voltage_node1> <control_voltage_node2> <model>

Example: S1 N1 N2 Vctrl1 Vctrl2 SW1

Current-Controlled Switch (CS)

The current-controlled switch models a switch that opens or closes based on the current through its controlling nodes.

Syntax: CS<name> <node1> <node2> <control_current_node1> <control_current_node2> <model>

Example: CS1 N3 N4 Ictrl1 Ictrl2 SW2

Transmission Line Switch (SW)

The transmission line switch models a switch that can be used in transmission line simulations.

Syntax: SW<name> <node1> <node2> <model>

Example: SW1 N5 N6 TransmissionLineSwitch

Diode Switch (DS)

The diode switch models a diode-like switch that can turn on or off based on a controlling voltage.

Syntax: DS<name> <node_anode> <node_cathode> <control_voltage_node> <model>

Example: DS1 Anode Cathode Vctrl DiodeSwitchModel

Switches can be used to simulate various circuit scenarios, such as toggling components on and off, simulating the opening or closing of a circuit path, and modeling dynamic circuit behavior. The controlling nodes determine when the switch opens or closes based on specified conditions. It's important to provide an appropriate model for the switch to accurately represent its behavior.

When using switches, be sure to understand the behavior of the specific switch type and how it integrates with the rest of your circuit components.

2.2.2 Transmission Lines

n SPICE (Simulation Program with Integrated Circuit Emphasis), transmission lines are modeled to simulate the behavior of signal propagation along transmission lines in electronic circuits. Transmission lines are used to carry signals, such as high-frequency analog and digital signals, between different parts of a circuit or between different components. Modeling transmission lines in SPICE is important for analyzing signal integrity, impedance matching, and other high-frequency effects.

SPICE provides various transmission line models that take into account the characteristics of real transmission lines, such as impedance, delay, and reflection. The most common type of transmission line model in SPICE is the distributed model, which divides the transmission line into small segments and models the propagation of signals along these segments.

Here's an overview of how transmission lines are typically modeled in SPICE:

Distributed Transmission Line Model

Distributed models divide the transmission line into small segments and model the electrical behavior of each segment.

Each segment is represented by an inductor, a resistor, a capacitor, and a series resistor to account for resistance, capacitance, and inductance of the transmission line.

These models use the parameters of the transmission line, such as characteristic impedance, propagation delay, and length, to determine the behavior of the line.

Single-Conductor Line Model

For single-conductor lines (e.g., microstrip or stripline), models may include ground-plane effects and dielectric properties.

These models consider the geometry of the line, the dielectric material, and the surrounding environment.

Example: TL2 N1 N2 CONDUCTOR=1 DIELECTRIC=FR4 H=1.6 W=0.2

Transmission Line Components

SPICE provides specific components, such as the TLINE component, to model transmission lines.

These components allow you to specify parameters like characteristic impedance, propagation delay, and length.

S-Parameter Models

For more complex transmission lines, you can use S-parameter models to define the scattering parameters of the line.

S-parameters describe the relationship between incident and reflected signals at various frequencies.

Modeling transmission lines in SPICE helps to understand how signals behave in high-frequency circuits, consider impedance matching, and avoid signal integrity issues like reflections and signal distortion. The choice of model depends on the complexity of the transmission line and the level of accuracy required for your simulation. Keep in mind that accurate transmission line modeling becomes more important at higher frequencies and in circuits with long interconnects.

2.2.3 Voltage and Current Sources

In SPICE (Simulation Program with Integrated Circuit Emphasis), voltage sources and current sources are essential components used to represent external signals, power supplies, and stimulus in electronic circuit simulations. These sources provide the means to inject or measure electrical quantities into or from a circuit. They are crucial for analyzing the behavior and response of circuits under various conditions.

SPICE provides various types of voltage and current sources to model different scenarios.

Independent Sources

Linear Dependent Sources

Non-Linear Dependant Sources

2.2.3.1 Independent Sources

In SPICE (Simulation Program with Integrated Circuit Emphasis), independent sources are components that generate or inject signals into a circuit without being influenced by other circuit elements. Independent sources provide a means to simulate external stimulus, power supplies, and varying signals in electronic circuit simulations. These sources are essential for analyzing circuit behavior and response under different conditions.

SPICE offers various types of independent sources to model different signal types and behaviors.

DC Voltage/Current Source

A DC voltage source is an independent source that generates a constant DC voltage regardless of the current flowing through it. DC voltage sources are commonly used

to model power supplies, biasing voltages, and constant reference voltages in electronic circuit simulations.

Exponential Voltage/Current Source

An exponential voltage source (VEXP) and exponential current source (IEXP) are independent sources used to generate exponential voltage and current waveforms, respectively. These sources are useful for simulating circuits with exponential responses, such as charging or discharging behaviors in RC circuits.

The exponential waveform is determined by the time constant and initial value. The exponential waveform can represent charging or discharging behaviors of capacitors in RC circuits, for instance. The scaling factor allows you to adjust the magnitude of the exponential waveform.

These exponential sources are particularly useful for simulating transient behaviors in circuits where exponential changes occur, such as the charging or discharging of capacitors or inductor currents.

Piece-Wise Linear Voltage/Current Source

A Piece-Wise Linear (PWL) voltage source and a Piece-Wise Linear (PWL) current source are independent sources that generate voltage and current waveforms defined by a set of time and value pairs. These sources are useful for simulating signals with specific step changes or varying levels at different time points.

The PWL sources generate waveform segments defined by the specified time and value pairs. The waveform between two consecutive pairs is linearly interpolated.

Piece-Wise Linear sources are particularly useful for simulating signals with abrupt changes, step responses, or any other time-varying behavior that can be approximated using linear segments. These sources allow you to define complex voltage and current patterns that change over time, making them versatile tools for transient and dynamic circuit simulations.

Single Frequency FM Voltage/Current Source

You can simulate frequency modulation (FM) effects using sinusoidal voltage or current sources with varying frequencies over time. While SPICE does not directly offer a single-frequency FM source, you can achieve this effect by manipulating the frequency of a sinusoidal source using behavioral sources.

By manipulating the frequency over time, you simulate the effect of frequency modulation.

Keep in mind that while this approach allows you to simulate frequency modulation effects in SPICE, it might not be suitable for all scenarios. For more complex FM simulations or more accurate FM modulation, you might need specialized tools or software that support FM modulation directly.

Sinusoidal Voltage/Current Source

You can simulate sinusoidal voltage and current sources to generate sinusoidal AC signals in electronic circuit simulations. Sinusoidal sources are commonly used to represent AC signals, oscillators, and alternating currents in circuits.

The sinusoidal sources generate waveforms based on the magnitude and phase angle specified. The waveforms are sinusoidal AC signals with the specified peak amplitude and phase shift relative to time.

Sinusoidal sources are essential for simulating AC behavior in circuits, including oscillators, amplifiers, filters, and other AC-based components. When analyzing frequency response or transient AC behavior, these sources help you understand how circuits respond to sinusoidal signals at different frequencies and phases.

2.2.3.2 Linear Dependent Sources

Current-Controlled Current Sources

Current-controlled current sources (CCCS) are components that generate a current output based on the value of a controlling current in the circuit. CCCS are used to model circuits where the current flowing through one part of the circuit controls the current flowing through another part. These sources are particularly useful in amplifier design and feedback analysis.

CCCS are often used to model transistors and operational amplifier (op-amp) circuits, where a small input current controls a larger output current. They play a key role in analyzing amplifiers, feedback loops, and signal processing circuits. When working with CCCS, ensure that the controlling current and gain values are properly defined to accurately model the circuit's behavior.

Current-Controlled Voltage Sources

Current-controlled voltage sources (CCVS) are components that generate a voltage output based on the value of a controlling current in the circuit. CCVS are used to model circuits where the current flowing through one part of the circuit controls the voltage across another part. These sources are commonly used in amplifier designs, feedback analysis, and other applications.

CCVS are used to model various types of circuits, such as amplifiers, feedback networks, and control systems. They allow you to represent complex relationships between current and voltage in a circuit. When working with CCVS, ensure that the controlling current and gain values are properly defined to accurately model the circuit's behavior.

Voltage-Controlled Current Sources

Voltage-controlled current sources (VCCS) are components that generate a current output based on the value of a controlling voltage in the circuit. VCCS are used to model circuits where the voltage across one part of the circuit controls the current flowing through another part. These sources are commonly used in amplifier designs, feedback loops, and other applications.

VCCS are commonly used to model operational amplifier circuits, differential amplifiers, and other applications where the output current is controlled by a controlling voltage. When using VCCS, ensure that the controlling voltage and transconductance values are properly defined to accurately model the circuit's behavior.

Voltage-Controlled Voltage Sources

Voltage-controlled voltage sources (VCVS) are components that generate an output voltage based on the value of a controlling voltage in the circuit. VCVS are used to model circuits where the voltage across one part of the circuit controls the voltage at another part. These sources are commonly used in amplifier designs, feedback loops, and other applications.

VCVS are commonly used to model amplifiers, operational amplifier circuits, and other applications where the output voltage is controlled by a controlling voltage. When using VCVS, ensure that the controlling voltage and gain values are properly defined to accurately model the circuit's behavior

2.2.4 SubCircuits

Subcircuits (also known as subcircuits or subcircuits) are a powerful feature that allows you to create custom circuit components with their own internal network of devices. Subcircuits enable you to encapsulate complex circuits, reuse them in multiple designs, and organize your simulations more efficiently. They are particularly useful for modeling integrated circuits, complex devices, and custom components that are not available as built-in SPICE primitives.

Subcircuits allow you to create reusable building blocks for your circuits, improving simulation efficiency and organization. They are especially useful for complex devices like operational amplifiers, analog components, custom transistors, and more. Make sure to understand the syntax and conventions of your specific SPICE simulator for defining and using subcircuits effectively.

2.3 PCB Design

PCB design refers to the process of designing the layout and circuitry of a Printed Circuit Board (PCB), which is an electronic circuit board that is used to connect and control electronic components. PCB design involves creating a schematic diagram of the electrical connections between components, selecting appropriate components for the design, and laying out the physical placement of the components on the PCB.

The PCB design process involves a variety of software tools that help designers create and test the design before it is sent to be manufactured. This includes software for creating the schematic diagram, designing the PCB layout, and simulating the electrical behavior of the circuit. PCB design is an essential step in the development of electronic products, as the layout and functionality of the PCB can have a significant impact on the overall performance of the device.

Designing a PCB (Printed Circuit Board) involves several steps and requires a good understanding of electrical and electronic circuits. Here are the general steps to design a PCB:

Schematic Design

Create a schematic diagram of your circuit using a circuit design software. This will help you to identify the components, their connectivity, and the overall structure of the circuit.

PCB Layout

Once you have a schematic, you need to translate it into a physical layout of the PCB. You can use a PCB design software to place the components, draw the traces, and generate the Gerber files required for manufacturing.

Component Placement

Arrange the components on the board so that they fit and are easily accessible. Consider the size of the components, the heat dissipation, and any mechanical constraints. Place the components in a logical and organized manner to minimize the trace length.

Trace Routing

Connect the components using traces or wires on the board. You will need to route the traces in a way that minimizes interference, reduces noise, and ensures proper grounding. Use the design software's tools to route the traces, and make sure that you follow the design rules and constraints.

PCB Verification

Check the design for errors and verify that it meets the specifications of the circuit. Perform a design rule check (DRC) to ensure that the board adheres to the manufacturing rules and constraints. Verify that the connections are correct, and run simulations to check the performance of the circuit.

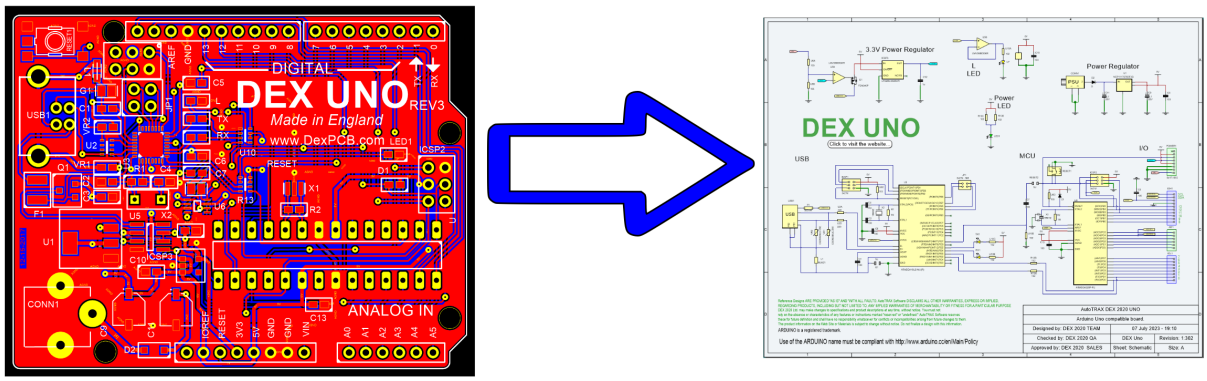
PCB Manufacturing

Once the design is complete and verified, it is ready for manufacturing. Generate the Gerber files required for manufacturing, and send them to a PCB manufacturer. Choose a manufacturer that meets your budget and quality requirements.

In summary, designing a PCB involves schematic design, PCB layout, component placement, trace routing, PCB verification, and PCB manufacturing. It can be a complex process, but with practice and patience, anyone can learn to design a PCB.

2.3.1 PCB Back Annotation

Back annotation in the context of printed circuit board (PCB) design is the process of transferring changes from the PCB layout back to the schematic.



PCB Back Annotation

When designing a PCB, you typically start by creating a schematic, which is a diagram that represents the electrical connections between different components. This schematic is then used to create a physical layout for the PCB.

Sometimes, changes are made during the PCB layout process. These might include changing component values, or reordering the components for space optimization, improved signal integrity, or other considerations. These changes can significantly affect the electrical properties and behavior of the circuit.

Back annotation is the process of taking these changes and updating the original schematic to reflect them. This is important for a few reasons:

Consistency

It's important to have a single, consistent representation of the design. If the schematic and layout don't match, it can cause confusion and mistakes.

Verification and Validation: Many of the analyses and checks that are done during the design process (like circuit simulation, design rule checks, etc.) are based on the schematic. If the schematic doesn't reflect the actual design, these analyses may not be accurate.

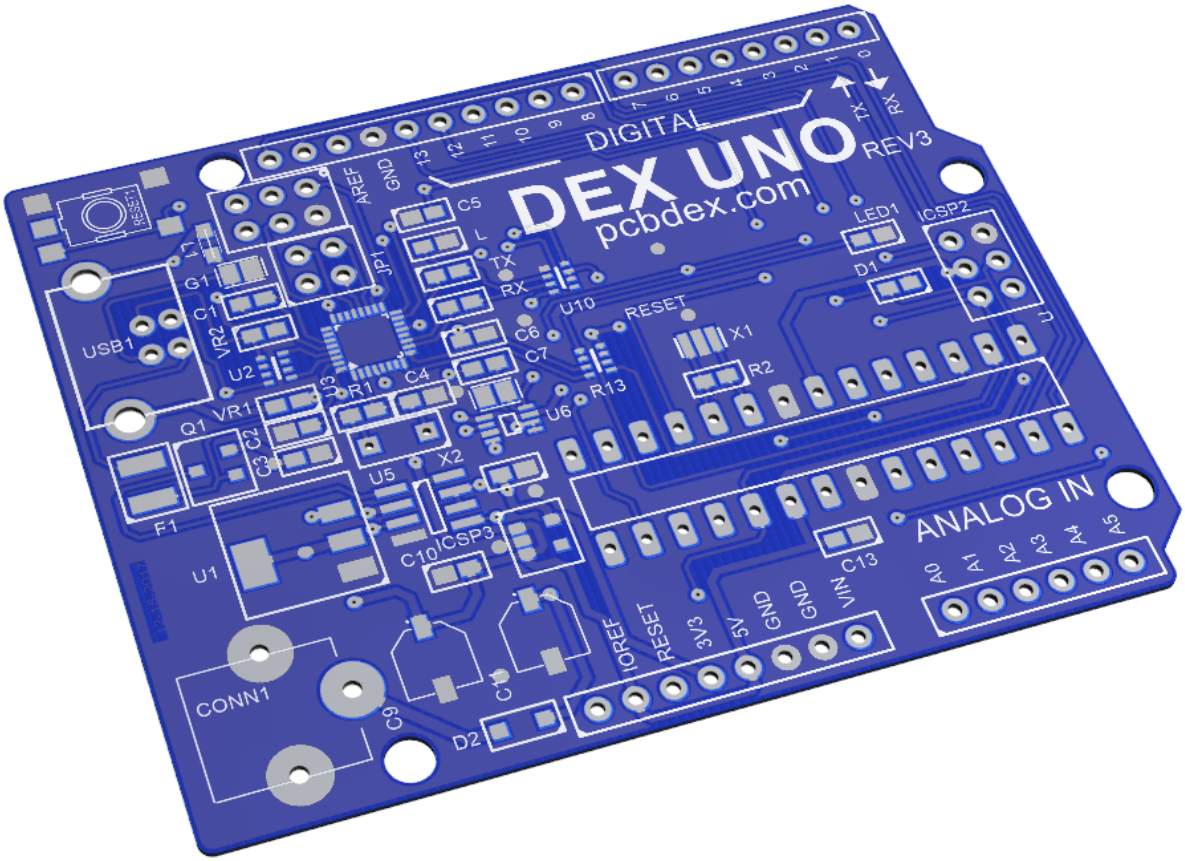
Documentation

The schematic is often used as a basis for creating other documentation, like the Bill of Materials (BOM), assembly drawings, and so on. If it's not up to date, these documents will be incorrect.

It's worth noting that the specifics of how back annotation is done can depend on the particular Electronic Design Automation (EDA) software you're using. In some cases, the software may be able to automatically apply changes from the layout to the schematic. In other cases, you may need to manually update the schematic based on a report of changes from the layout tool.

2.3.2 What is a PCB?

PCB stands for Printed Circuit Board. It is a board made of non-conductive material (usually fiberglass or plastic) that contains a pattern of conductive traces and components. PCBs are used to provide mechanical support and electrical connections for electronic components, and are found in a wide range of electronic devices, from simple toys and appliances to complex computers and communication systems.



Typical 2-Sided PCB

The conductive traces on a PCB are typically made of copper, and are arranged in a specific pattern that allows the electrical signals to flow between the components.

The components are mounted on the surface of the PCB using a variety of methods, including through-hole and surface mount technology (SMT).

PCBs can be single-sided or double-sided, depending on the complexity of the circuit. In addition, PCBs can have multiple layers of conductive traces and components, which are separated by insulating layers. These multilayer PCBs are used in more complex electronic devices, where space is limited and the circuitry is more intricate.

PCB design involves a range of factors, including selecting the appropriate components, arranging them on the board, routing the conductive traces, and ensuring that the board meets the necessary electrical and mechanical specifications. PCBs can be designed using specialized software tools, and are manufactured using a variety of processes, including etching, milling, and drilling.

A PCB (Printed Circuit Board) is a board made of insulating material such as fiberglass or plastic, with conductive pathways etched onto its surface. PCBs are used to provide a physical platform for electronic components to be mounted and connected in a circuit. The conductive pathways on the board are usually made of copper and are designed to carry electrical signals and power between the components.

PCBs are widely used in the manufacturing of electronic devices and equipment, including computers, smartphones, televisions, and medical equipment. They are often designed using specialized software and produced in large quantities using automated processes. PCBs can be single-layer or multi-layer, with the latter offering more complex and compact circuit designs.

Overall, PCBs are an essential component of modern electronics, enabling the efficient and reliable transfer of electrical signals between components.

2.3.3 What are PCBs Made Of

PCBs (Printed Circuit Boards) are typically made of a combination of different materials that serve various purposes in the construction of the board. The main materials used in the manufacturing of PCBs are as follows:

Substrate Material

FR-4 (Flame Retardant 4): FR-4 is the most common substrate material used in rigid PCBs. It is composed of woven fiberglass cloth impregnated with epoxy resin. FR-4 provides good electrical insulation, mechanical strength, and dimensional stability. It is also flame retardant, making it safe for many applications.

Polyimide: Polyimide is a flexible substrate material used in flexible PCBs. It offers excellent flexibility and high-temperature resistance, making it suitable for applications where the PCB needs to bend or flex.

PTFE (Polytetrafluoroethylene): PTFE, commonly known by the brand name Teflon, is used in high-frequency applications and RF circuits due to its low dielectric constant and loss tangent. It is also known for its high-temperature stability.

Copper Foil

Copper is the primary conductor material in PCBs. Copper foil is bonded to the substrate material, and the conductive traces and pads are etched from this copper layer during the manufacturing process.

Solder Mask

Solder mask is a protective layer applied to the PCB to insulate the conductive traces and prevent solder bridges during assembly. It is typically green, but other colors like blue, red, black, and white are also used.

Silkscreen

The silkscreen layer includes markings and labels on the PCB, such as component outlines, reference designators, part numbers, and logos. It is printed in white or other contrasting colors for visibility.

Surface Finish

Surface finish is a coating applied to the exposed copper traces and pads to prevent oxidation and ensure good solderability during assembly. Common surface finish types include HASL (Hot Air Solder Leveling), ENIG (Electroless Nickel Immersion Gold), and OSP (Organic Solderability Preservative).

Solder Paste

Solder paste is used in surface-mount technology (SMT) assembly. It is a mixture of solder particles and flux that allows components to be temporarily held in place before reflow soldering.

Adhesive

In multilayer PCBs, an adhesive material is used to bond individual layers together before they are laminated and pressed to form a single board.

Prepreg

Prepreg is a resin-impregnated fiberglass material used between copper layers in multilayer PCBs. It helps to insulate and bond the layers together during the lamination process.

These materials are carefully selected based on the specific requirements of the PCB, such as its intended application, complexity, mechanical strength, thermal characteristics, and cost considerations. The combination of these materials and their arrangement determines the overall performance and functionality of the printed circuit board.

2.3.4 What is FR-4

FR-4 is a widely used standard grade of flame-retardant glass-reinforced epoxy laminate material used as a substrate for printed circuit boards (PCBs). It is one of the most common and versatile materials used in the electronics industry for manufacturing rigid PCBs. The term "FR" stands for "Flame Retardant," and the "4" denotes the type or grade of the material.

FR-4 is a composite material that consists of a woven fiberglass cloth impregnated with epoxy resin. The fiberglass provides mechanical strength and rigidity to the PCB, while the epoxy resin acts as a binder that holds the fiberglass together and provides good electrical insulation properties.

Key characteristics and advantages of FR-4 material for PCBs include:

- **Flame Retardancy:** FR-4 is designed to be flame retardant, which means it has the ability to resist the spread of fire and self-extinguish. This property is crucial for ensuring the safety of electronic devices and preventing fire hazards.
- **Electrical Insulation:** FR-4 provides excellent electrical insulation properties, helping to prevent short circuits and ensuring proper signal integrity on the PCB.
- **Mechanical Strength:** The woven fiberglass cloth in FR-4 gives the material high mechanical strength and dimensional stability, making it suitable for supporting various electronic components and withstanding mechanical stress during handling and assembly.
- **Thermal Resistance:** FR-4 exhibits good thermal resistance, allowing it to withstand the heat generated during soldering processes and the normal operation of electronic devices.
- **Cost-Effectiveness:** FR-4 is a relatively affordable and readily available material, making it a cost-effective choice for a wide range of applications.
- **Compatibility with High-Frequency Signals:** FR-4 can be used in applications involving high-frequency signals, although its performance at very high frequencies might be limited compared to specialized materials like PTFE (Teflon).
- **Standardization:** FR-4 material is standardized, and its properties are well-documented, making it easier for PCB designers and manufacturers to work with and predict its behavior.

Due to its desirable properties, FR-4 is widely used in various electronic devices, from consumer electronics like smartphones and computers to industrial equipment and automotive systems. The specific grade of FR-4 may vary based on the application and performance requirements, but overall, it remains a popular and dependable choice for the majority of rigid PCBs in the electronics industry.

2.3.5 What is PCB Prepreg

PCB prepreg, short for "pre-impregnated," is an essential material used in the construction of multilayer printed circuit boards (PCBs). It plays a crucial role in bonding and insulating the different layers of the PCB together during the lamination process. Prepreg is typically a fiberglass cloth that has been pre-impregnated with a partially cured epoxy resin.

The prepreg material is used in conjunction with copper-clad laminates to build multilayer PCBs. Here's how it works:

- **Copper-Clad Laminates:** Copper-clad laminates are composed of layers of copper foil bonded to either side of a rigid core material, which is often FR-4 (flame-retardant glass-reinforced epoxy laminate). The copper layers serve as the conductive paths for the circuit, and the core material provides mechanical strength to the PCB.
- **Prepreg Layers:** Prepreg layers are placed between the copper-clad laminates in a multilayer PCB stackup. These prepreg layers are essentially the same fiberglass material used in FR-4, but instead of being fully impregnated with epoxy, they are only partially cured. The epoxy resin in the prepreg is partially cured to a semi-solid state, which allows it to flow and bond during the lamination process.
- **Lamination Process:** The PCB stackup, comprising alternating layers of copper-clad laminate and prepreg, is subjected to high temperature and pressure in a process called lamination. During lamination, the partially cured epoxy in the prepreg undergoes further curing, becoming fully solid and creating a strong bond between the copper layers and the core material.
- **Bonding and Insulation:** As the prepreg cures and solidifies, it acts as both a bonding agent and an insulator. It adheres the copper layers to the core material, creating a cohesive structure, and it also provides electrical insulation between adjacent copper layers, preventing short circuits.
- **Through-Hole Formation:** After lamination, the PCB is drilled to create holes known as vias. These vias allow electrical connections to be made between different layers in the multilayer PCB. The vias pass through the copper layers and prepreg, and they are typically plated with a conductive material to establish the electrical connections.

PCB prepreg ensures the integrity and mechanical stability of the multilayer PCB, enabling it to function as a single, cohesive unit. The number of prepreg layers and their thickness can be adjusted based on the specific design requirements of the multilayer PCB.

2.3.6 What are Copper-Clad Laminates

Copper-clad laminates are a type of composite material used in the construction of printed circuit boards (PCBs). They consist of layers of copper foil bonded to a substrate material, typically made of fiberglass impregnated with epoxy resin. The copper layers serve as the conductive pathways for the PCB, while the substrate material provides mechanical support and electrical insulation between copper traces on different layers of the PCB.

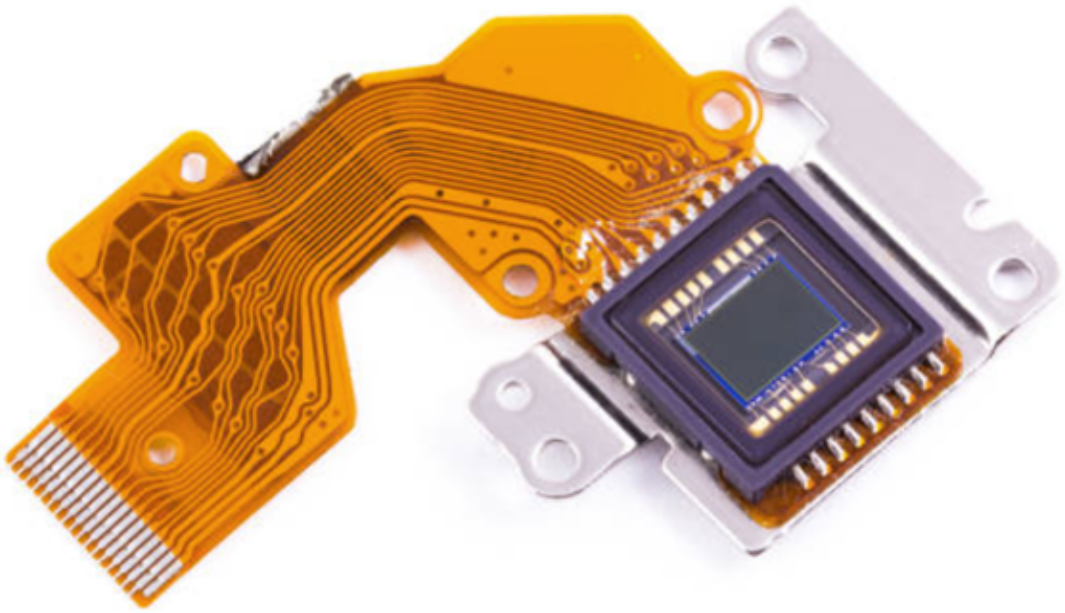
The construction of copper-clad laminates involves the following components:

- **Copper Foil:** Copper-clad laminates feature thin layers of copper foil on one or both sides. The copper foil acts as the conductive material for the PCB, providing the paths for electrical signals to flow between components.
- **Substrate Material:** The substrate material in copper-clad laminates is usually made of fiberglass cloth impregnated with epoxy resin. This material is commonly referred to as "FR-4" (Flame Retardant 4) and is widely used in the electronics industry due to its excellent mechanical properties, electrical insulation, and flame-retardant characteristics.
- The manufacturing process of copper-clad laminates involves the following steps:
- **Preparation:** The fiberglass cloth is treated and coated with a thin layer of epoxy resin to ensure proper adhesion with the copper foil.
- **Bonding:** Thin layers of copper foil are bonded to one or both sides of the prepared substrate material. The bonding process is achieved through heat and pressure, which creates a strong adhesion between the copper and the substrate.
- **Curing:** The copper-clad laminate is subjected to a curing process, during which the epoxy resin is fully solidified, forming a rigid and durable composite material.

Once the copper-clad laminate is manufactured, it serves as the foundational material for single-sided, double-sided, and multilayer PCBs. During the PCB fabrication process, conductive traces and pads are created on the copper surface through etching, leaving the desired circuit pattern. In multilayer PCBs, prepreg layers are used in conjunction with copper-clad laminates to bond and insulate the different layers together during the lamination process.

Copper-clad laminates are available in various thicknesses, copper weights (measured in ounces per square foot), and substrate material types, allowing PCB designers to choose the most suitable laminate for their specific application and performance requirements.

2.3.7 What are Flexible PCBs



A Flexible PCB

Flexible PCBs (Printed Circuit Boards) are a type of electronic interconnect technology that offers flexibility and versatility in comparison to traditional rigid PCBs. They are also known as flex circuits or flex boards. Flexible PCBs are designed to bend, twist, and conform to the shape of the device or system they are used in, making them ideal for applications where space and weight are critical or where the board needs to fit into unconventional shapes or contours.

The main features and benefits of flexible PCBs include:

- **Flexibility:** As the name suggests, flexible PCBs can bend, fold, and twist without losing functionality. This property makes them suitable for applications with limited space or irregular geometries.
- **Weight and Space Reduction:** The ability to create 3D shapes and reduce the number of connectors and cables often results in weight and space savings, which is crucial for portable devices and miniaturized electronics.
- **Improved Reliability:** Flexible PCBs have fewer solder joints and interconnects, leading to reduced points of failure and improved reliability.
- **Cost-Effectiveness:** In some cases, flexible PCBs can reduce the overall cost of the final product by eliminating the need for connectors and additional components.

- **Signal Integrity:** Flex circuits can offer good signal integrity and electrical performance, making them suitable for high-frequency applications.
- **Temperature Resistance:** Flexible PCB materials are typically designed to withstand a wide range of temperatures, making them suitable for environments with varying thermal conditions.
- **Durability:** The ability to withstand repeated bending and flexing makes flexible PCBs more durable in certain applications than traditional rigid PCBs.

Flexible PCBs can be manufactured using various materials, including polyimide, polyester, and PTFE (Teflon). The manufacturing process involves etching conductive traces on the flexible substrate, just like rigid PCBs, but with additional considerations for the material's flexibility.

Flexible PCBs find applications in various industries, including consumer electronics (smartphones, wearables), automotive, medical devices, aerospace, and military applications. As technology advances, the demand for flexible PCBs is likely to grow due to their advantages in addressing the evolving needs of electronic products and systems.

2.3.7.1 How are What are Flexible PCBs Made

Flexible PCBs are made through a specialized manufacturing process that involves building conductive traces and components on a flexible substrate. The process is similar to the manufacturing of rigid PCBs, but it includes additional steps to accommodate the flexibility of the final product. Here's a general overview of how flexible PCBs are made:

- **Design and Layout:** The first step is to design the flexible PCB using computer-aided design (CAD) software. The layout includes the placement of components, the routing of conductive traces, and the design of any required flexing areas.
- **Substrate Material Selection:** Flexible PCBs are typically made using flexible substrate materials like polyimide (e.g., Kapton) or polyester (e.g., Mylar). The choice of material depends on factors such as the intended application, required flexibility, and temperature resistance.
- **Substrate Preparation:** The chosen substrate material is cleaned and pre-treated to ensure good adhesion of the conductive traces and components.
- **Copper Cladding:** A thin layer of copper is added to one or both sides of the flexible substrate through a process called copper cladding. This copper layer will form the conductive paths on the PCB.
- **Etching:** A layer of photoresist is applied to the copper surface, and the PCB layout pattern is transferred onto it using a photomask. The areas not covered by the photomask are exposed to UV light, which hardens the photoresist. Afterward, the unexposed photoresist is removed, leaving the desired copper traces exposed.

- **Chemical Etching:** The exposed copper is etched away using an appropriate etchant, leaving behind the conductive traces following the PCB layout pattern.
- **Drilling:** If the flexible PCB requires through-hole components or vias, holes are drilled through the substrate at appropriate locations. These holes will later be plated to create electrical connections between different layers.
- **Component Placement:** Surface-mount and/or through-hole components are placed onto the flexible substrate at their designated locations. Automated pick-and-place machines are often used for precise component placement.
- **Soldering:** The components are soldered onto the conductive traces to create electrical connections. Depending on the design, soldering can be done using reflow soldering for surface-mount components or wave soldering for through-hole components.
- **Encapsulation and Protection:** To protect the flexible PCB and its components from environmental factors, such as moisture and physical damage, it can be encapsulated with a protective layer or conformal coating.
- **Testing:** The flexible PCB undergoes thorough testing to ensure that all connections are correctly established, and there are no manufacturing defects.
- **Cutting and Forming:** The flexible PCB is cut and formed to its final shape, conforming to the specific requirements of the application.

The manufacturing process for flexible PCBs requires precision and attention to detail to ensure that the resulting product meets the desired specifications and maintains its flexibility while functioning reliably. Depending on the complexity of the design and the required quantities, the manufacturing process can be done using manual, semi-automated, or fully automated methods.

2.3.8 What is PCB design software?

PCB design software is a type of computer software used by engineers and designers to create, test, and optimize printed circuit board (PCB) layouts. PCB design software typically includes a suite of tools for creating schematics, laying out components, routing traces, and verifying the design before it is sent to manufacturing.

There are many different PCB design software options available, ranging from simple, free tools to more advanced, feature-rich applications that can cost thousands of dollars.

PCB design software typically includes a range of features to help designers create efficient, optimized PCB layouts. These may include tools for component placement, trace routing, signal integrity analysis, and design rule checking. Some software packages may also include simulation and analysis tools to help designers verify the performance of their circuits before they are manufactured.

Overall, PCB design software is an essential tool for engineers and designers working on electronic circuits and systems. It allows for efficient, optimized PCB layouts that are reliable, easy to manufacture, and perform well under a range of conditions.

2.3.9 What is PCB Design?

The Basics of PCB Design

PCB design involves using computer-aided design (CAD) software to create a schematic diagram that accurately represents how all the components fit together on a circuit board. This includes placing components in their correct locations, connecting them to each other, and routing power and signal lines between them. Once this is done, the CAD file can be converted into Gerber files for production. Additionally, designers may also use simulation software to test their designs before they're sent off for manufacturing.

There are several factors that must be considered when designing a PCB, including component selection and placement, trace routing, copper area utilization, stackup design (the number of layers), via selection (the type used), drill size selection, impedance control requirements, etc. Additionally, there are several standards that must be followed when designing a PCB such as IPC-2221A/IPC-2222A for general requirements or IPC-7351B standard for component pad sizes. Following these standards ensures that your final product meets industry requirements and won't cause any issues down the line.

Designers must also consider any special requirements such as high speed signals or RF frequencies when designing a PCB; these require specific layout techniques such as controlled impedance traces in order to ensure optimal performance from your device. Failure to follow these guidelines could result in poor signal integrity which could cause your device to malfunction or even become damaged due to excessive heat build-up.

Conclusion: As you can see from this overview, there is a lot more involved in PCB design than meets the eye! It requires careful consideration of component placement, trace routing techniques, stackup configurations and much more in order to ensure that your device works properly once it's been manufactured and put into use by customers. With proper planning and forethought however, anyone with an interest in electronics can learn how to create successful designs with minimal risk of failure or delays during production! With that being said – happy designing!

2.3.10 How to Specify the Design of a PCB

To specify the design of a PCB, you need to provide clear and detailed information about the requirements, constraints, and specifications for the design. Here are some key aspects to consider when specifying a PCB design:

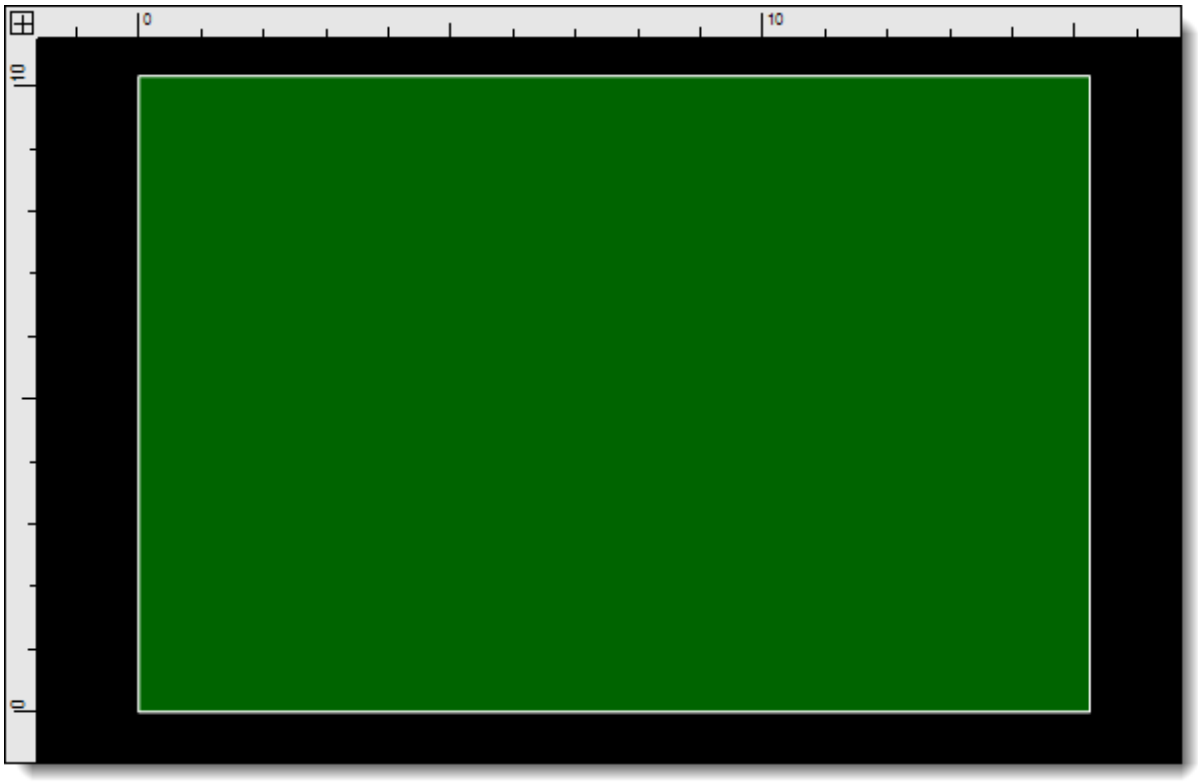
Purpose and Functionality

Clearly define the purpose and functionality of the PCB. Describe what the circuit is intended to do, its intended application, and any specific features or requirements it should fulfill.

Deciding on PCB Design and Functionality

Board Size and Shape

Specify the physical dimensions and shape of the PCB. Consider factors such as the available space in the enclosure or system where the PCB will be installed, as well as any size limitations or constraints.



PCB Board Size

Layer Configuration

Determine the number of layers required for the PCB. Consider the complexity of the circuit, the density of components, and the need for signal integrity. Specify the stack-up configuration, such as the placement of signal layers, ground planes, power planes, and any other specific requirements.

Layers 4 | 1 2 4 6 | Thickness 0.11 | Total 0.31

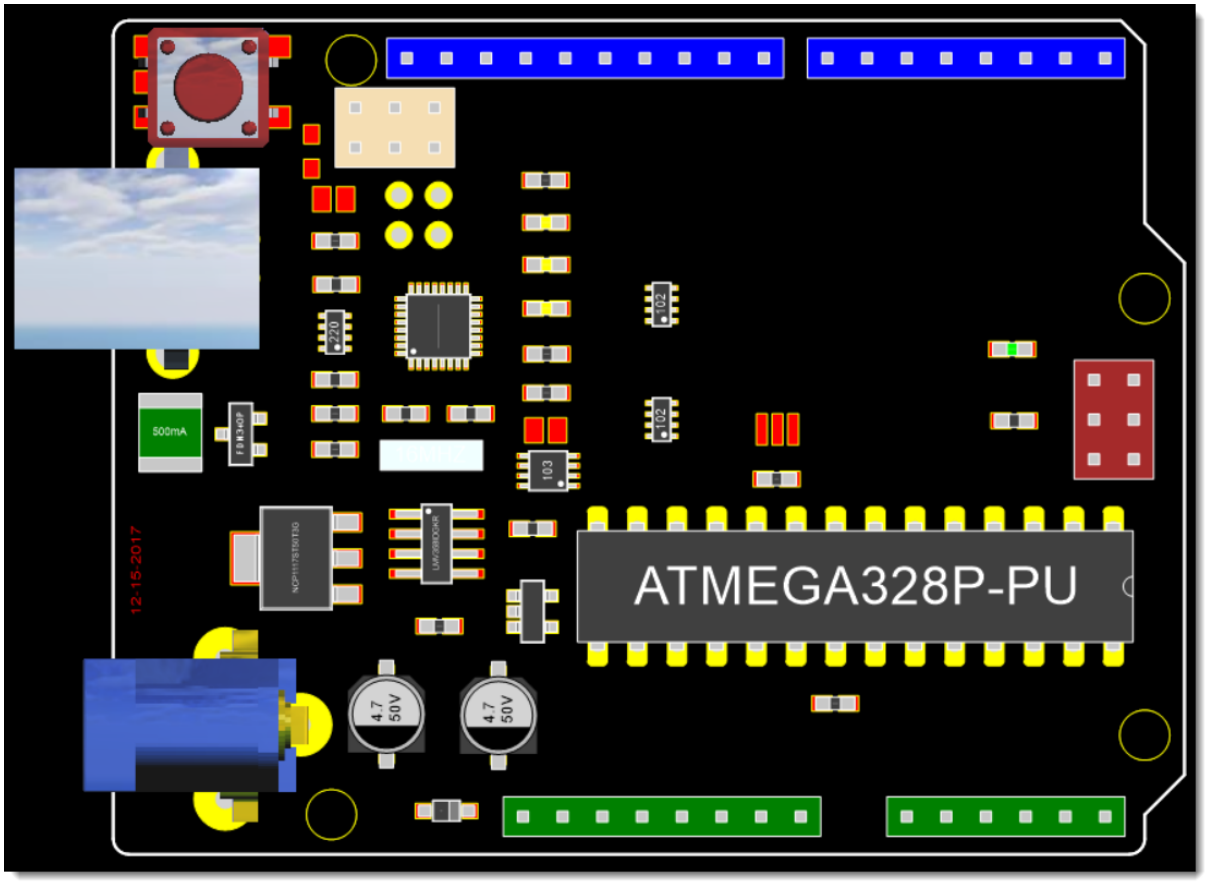
Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

PCB Layer Configuration

Component Placement

Specify any specific requirements for component placement. If there are critical components that require specific locations or orientations, mention them. Consider factors such as thermal considerations, signal routing, and accessibility for assembly and testing.



PCB Component Placement

Define the electrical specifications for the PCB. Specify the voltage and current requirements, impedance matching requirements, signal integrity considerations, and any specific power supply or signal conditioning requirements.

Design Constraints

Identify any design constraints or limitations. For example, specify any specific manufacturing constraints, such as minimum trace width and spacing, via size, and drill requirements. Consider any specific environmental or operational constraints that may impact the design.

Deciding on PCB Dersig Constraints

Connectivity and Interfaces

Specify the connectors, interfaces, or communication protocols required for the PCB. Identify any specific pin assignments, connector types, or signal levels that need to be incorporated into the design.

PCB Connectivity

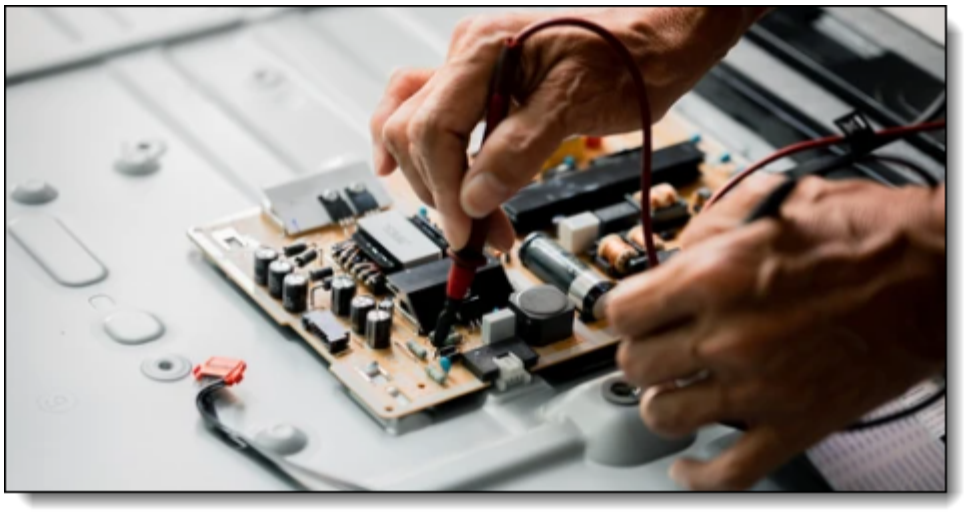
Manufacturing Requirements

Specify any specific manufacturing requirements, such as solder mask color, silkscreen legends, surface finish, and any special fabrication techniques or processes required.

PCB Manufacturing Requirements

Testing and Quality Assurance

Define any specific testing requirements or quality assurance procedures that need to be followed during the manufacturing and testing phases. Specify any testing points or test pads required for debugging or functional testing.



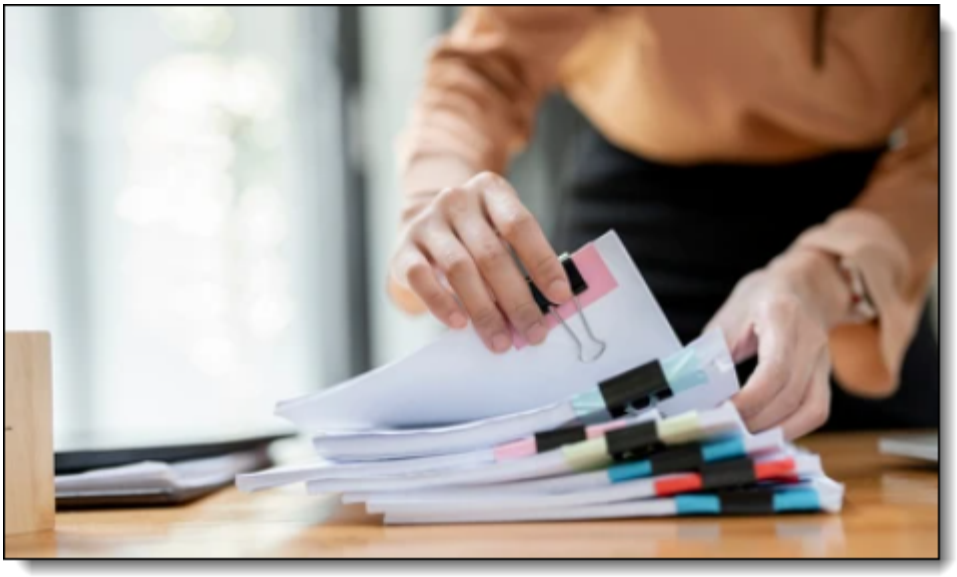
PCB Testing and Quality Assurance

Documentation

Clearly specify the level of documentation required for the PCB design. This may include schematic diagrams, Bill of Materials (BOM), assembly drawings, PCB layout files, and any other relevant documentation.

When specifying the design of a PCB, it's crucial to be as detailed and specific as possible. This helps ensure that the PCB designer or manufacturer can accurately understand and meet your requirements. If you are unsure about any specific

aspects of the design, consider consulting with an experienced PCB designer or engineer to assist you in the specification process.



PCB Documentation

2.3.11 PCB Cost and Design Complexity

Cost and design complexity are two critical factors that can greatly impact the process of developing a printed circuit board (PCB). Various elements of PCB design influence these factors. Here are some of the considerations related to cost and design complexity in PCB design:

Layer Count

More layers typically mean a higher cost due to the increased complexity in manufacturing. More layers might be necessary for complex designs with many components or for better electrical performance (like signal integrity or noise reduction).

Board Size and Shape

Larger boards are generally more expensive due to the increased material use. The shape of the PCB can also impact the cost. Simple rectangular or square boards are typically less expensive than irregular shapes, which can cause more waste during the manufacturing process.

Trace/Space Width

Smaller trace widths and spaces between traces often increase manufacturing costs due to the higher precision required. They can also increase the risk of manufacturing defects, which may require rework or result in a higher failure rate.

Materials

The type of substrate material used can greatly impact the cost. Standard FR-4 material is usually less expensive, but some designs may require special materials (like high-frequency materials, high-temperature materials, flexible materials, etc.), which can increase the cost.

Component Selection and Placement

The type, number, and placement of components can greatly impact both cost and complexity. Components with finer pitch or smaller size could be more difficult to assemble and may lead to a higher failure rate. Also, using surface mount components on both sides of the board may increase assembly costs.

Manufacturing Tolerances

Tighter tolerances (smaller annular rings, via sizes, alignment, etc.) can increase the cost due to the higher precision required in manufacturing.

Finishes

Different types of surface finishes can affect cost. For instance, ENIG (Electroless Nickel Immersion Gold) finish might be more expensive than HASL (Hot Air Solder Leveling).

Turnaround Time

The required speed of production can also affect the cost. Rush orders are typically more expensive than standard lead times.

Design for Manufacturability (DFM) and Design for Assembly (DFA)

A design that considers manufacturing and assembly processes from the start can significantly reduce cost and time. It helps to avoid design errors that can lead to production issues or require costly design revisions.

Volume

The number of units being produced can greatly influence the cost. Higher volumes typically lead to lower cost per unit due to economies of scale.

When designing a PCB, it's important to keep these factors in mind in order to balance cost, performance, reliability, and manufacturability. Working closely with your PCB manufacturer during the design phase can help you optimize your design to meet your specific needs and constraints.

2.3.12 PCB Design Rules

PCB (Printed Circuit Board) design rules are a set of guidelines that a designer must follow to correctly design a PCB that can be reliably manufactured and function as intended. These rules consider various factors related to the manufacturing process, electrical requirements, physical constraints, and more. Below are some key design rules for PCBs:

Minimum Traces and Spaces: These are the smallest width of the copper track and the smallest gap between two tracks that can be reliably manufactured. If the traces or spaces are too small, there may be issues during manufacturing, potentially leading to electrical shorts.

Drill Sizes

These define the size of the holes that can be drilled on the PCB. Minimum and maximum drill sizes are typically specified.

Annular Rings

This is the area of copper around a drilled hole on a PCB. There is typically a minimum annular ring size to ensure reliability.

Component Placement

Components should be placed in such a way to minimize the length of signal traces, and avoid interference between components.

Via Sizes

These are small holes drilled through a PCB that are plated with metal to create an electrical connection between different layers of the board.

Copper Thickness

This is the thickness of the copper layer on the PCB. Higher currents require thicker copper.

Clearance Rules

These are the distances that must be maintained between various elements on the board like traces, pads, vias, holes, etc.

Thermal Considerations

This includes designing the board to minimize heat buildup, which can be accomplished through the use of heat sinks, thermal vias, and other design strategies.

Impedance Control

For high-frequency signals, impedance matching is important to prevent signal reflection and loss.

Design for Manufacturability (DFM)

This involves designing the PCB in a way that makes it easy and cost-effective to manufacture.

These rules may vary based on the specific manufacturer's capabilities, the complexity of the board, and the intended application of the board. It is important for the designer to know these rules before starting the design process. Violating these rules can lead to a PCB that doesn't function as intended or that can't be reliably manufactured.

2.3.12.1 Minimum Traces and Spaces

"Minimum traces and spaces" is a key parameter in PCB (Printed Circuit Board) design. They are defined as the smallest width of the copper track (trace width) and the smallest gap (space) between two tracks that can be reliably manufactured and function correctly. These specifications are largely determined by the manufacturer's capabilities, and the specifics of the design.

Here are some considerations when working with minimum traces and spaces:

Manufacturer's Capabilities

Each PCB manufacturer will have specific limitations on their manufacturing process that dictate the minimum trace widths and spaces that they can reliably fabricate. For example, a common limit for many manufacturers is 6 mil (thousandths of an inch) for both minimum trace width and minimum spacing, though more advanced manufacturers may be able to go lower.

Current Carrying Capacity

The width of a trace determines its current-carrying capacity. Wider traces can carry more current without excessive heating. This needs to be considered, especially for power supply and high-current paths.

Impedance Control

For high-frequency or impedance-controlled designs, the trace width, together with the substrate height and dielectric, determines the characteristic impedance. This is important for signal integrity in high-speed designs.

Cost Considerations

Generally, designs with smaller traces and spaces are more expensive to manufacture because they require higher precision and have a greater likelihood of errors during manufacturing.

Reliability

Boards with smaller traces and spaces may also be less reliable, as they are more susceptible to damage and manufacturing defects can have a larger impact.

Crosstalk

Smaller spaces can increase the likelihood of crosstalk (unwanted transfer of signals between communication channels), which can impact signal integrity.

Before starting a design, it's important to understand the constraints of the manufacturing process, the requirements of the design, and the trade-offs between these factors. It's often best to use the largest traces and spaces that the design and manufacturing process will allow to improve reliability and reduce cost.

2.3.12.2 Annular Rings

The annular ring on a printed circuit board (PCB) is the area of copper pad that extends around the circumference of a drilled and finished (usually plated) hole. This ring is vital for providing a reliable connection between the hole (via or through-hole) and the conducting layers of the PCB.

Here are some key considerations and rules regarding annular ring design:

Size

The size of the annular ring is critical. If it's too small, there might be a poor connection or no connection at all. If it's too large, it may unnecessarily consume PCB real estate. The minimum annular ring size is generally defined by the manufacturer's capabilities and is often in the range of 1-6 mils (0.001-0.006 inches) for outer layers and 1-1.5 mils for inner layers.

Tolerance

Fabrication tolerances, including drilling accuracy and alignment, also play a role in defining the annular ring size. There can be a positional variance during drilling, so a larger annular ring allows for these potential misalignments.

Plating

Most vias are plated with copper to electrically connect different layers of the PCB. The plating process can reduce the size of the annular ring, so this must be taken into consideration when defining the initial size.

Mechanical Stress

Annular rings are more vulnerable to mechanical stress than other parts of the PCB. High levels of mechanical stress can cause the annular ring to fracture or separate from the PCB substrate, leading to circuit failure. Therefore, if your board is

expected to experience high levels of mechanical stress (like in a high vibration environment), you might want to use larger annular rings.

Thermal Stress

Thermal cycles can cause expansion and contraction of the PCB material and the copper annular ring. This mismatch can cause cracks in the annular ring or separation from the substrate, leading to circuit failure. If your design is intended for a high-temperature environment or will experience thermal cycles, you may need to use larger annular rings.

Cost and Design Complexity

As with other aspects of PCB design, larger annular rings increase reliability but can also increase cost and decrease the density of the design. So, a balance must be struck between reliability, cost, and design complexity.

In conclusion, the size of the annular ring in your PCB design will depend on a number of factors, including manufacturing capabilities and tolerances, the expected mechanical and thermal environment, and cost and design considerations. Always consult with your manufacturer to understand their specific requirements and capabilities.

2.3.12.3 Design Rules for Component Placement

When designing a PCB (Printed Circuit Board), several rules need to be considered during the component placement stage. These rules help to optimize the PCB layout for manufacturability, reliability, and performance. Here are a few key rules:

Decoupling Capacitors Placement

These should be placed as close as possible to the power pins of the respective ICs (Integrated Circuits) to minimize loop area and maximize their effectiveness.

Orientation

Components should be placed in the same orientation to prevent mistakes during the assembly process. This is particularly important when dealing with polarized components such as diodes, capacitors, and ICs.

Thermal Management

Power components that dissipate a lot of heat should be placed in such a way that they can be cooled efficiently. If a heat sink or forced air cooling is used, enough clearance must be provided.

Avoid Mechanical Stress

Avoid placing components near the board edges, holes, or cut-outs to reduce the risk of mechanical stress which could lead to failure. Also, consider the placement of heavy components - they may require additional board support to prevent damage.

Leave Enough Space

It's important to leave enough space between components to accommodate the size of the soldering tool and to allow for heat dissipation. Also, allow for enough space to route traces between components.

Analog and Digital Ground Separation: If a PCB contains both analog and digital components, keep them separated on the board and consider splitting the ground plane to avoid noise interference.

High-Frequency Components

For high-speed or high-frequency designs, consider the length and routing of the signal paths. Shorter paths are preferable, and the path of critical signals should be as direct as possible.

Design for Testability

Components should be placed in a way that test points are accessible for testing and debugging. This is especially important for automated testing setups.

Design for Manufacturing

Component placement should consider the capabilities and limitations of the manufacturer's assembly equipment. Some may have restrictions on component density, size, or orientation.

Remember that these are general rules, and the specific requirements for component placement can vary greatly depending on the complexity of the design, the application of the board, and other factors. Always consult with your PCB manufacturer to ensure your design can be produced reliably and cost-effectively.

2.3.12.4 PCB Clearance Rules

PCB clearance rules refer to the minimum spacing requirements between different elements on a PCB to ensure correct operation and avoid electrical problems such as short circuits, arcing, and electromagnetic interference. They are one of the critical aspects of PCB design.

These rules apply to different elements of the PCB:

Trace to Trace

The distance between adjacent traces is a crucial consideration to prevent short circuits and crosstalk. It will depend on factors such as the maximum operating voltage and the PCB's environment.

Trace to Pad

Traces must have a safe distance from pads not connected to them to prevent accidental bridging and short circuits during soldering.

Pad to Pad

Pads for different connections must be spaced apart appropriately to ensure that components can be soldered without causing shorts.

Via to Via, Trace, or Pad

Vias should be placed at a sufficient distance from each other and from traces and pads. The spacing helps avoid problems during the drilling process and helps prevent short circuits.

Component to Component

Components need to be placed at an adequate distance from each other to allow for thermal expansion, avoid mechanical interference, and prevent electrical interference.

Edge Clearance

Components, traces, and vias should be placed a safe distance from the edge of the PCB to avoid damage during fabrication and handling.

The specific clearance requirements can depend on several factors, including the maximum voltage that the PCB will handle, the type of signals (digital or analog), the frequency of operation, and safety regulations. The manufacturing process's limitations also play a role: different manufacturers may have different capabilities in terms of how closely they can reliably fabricate different elements.

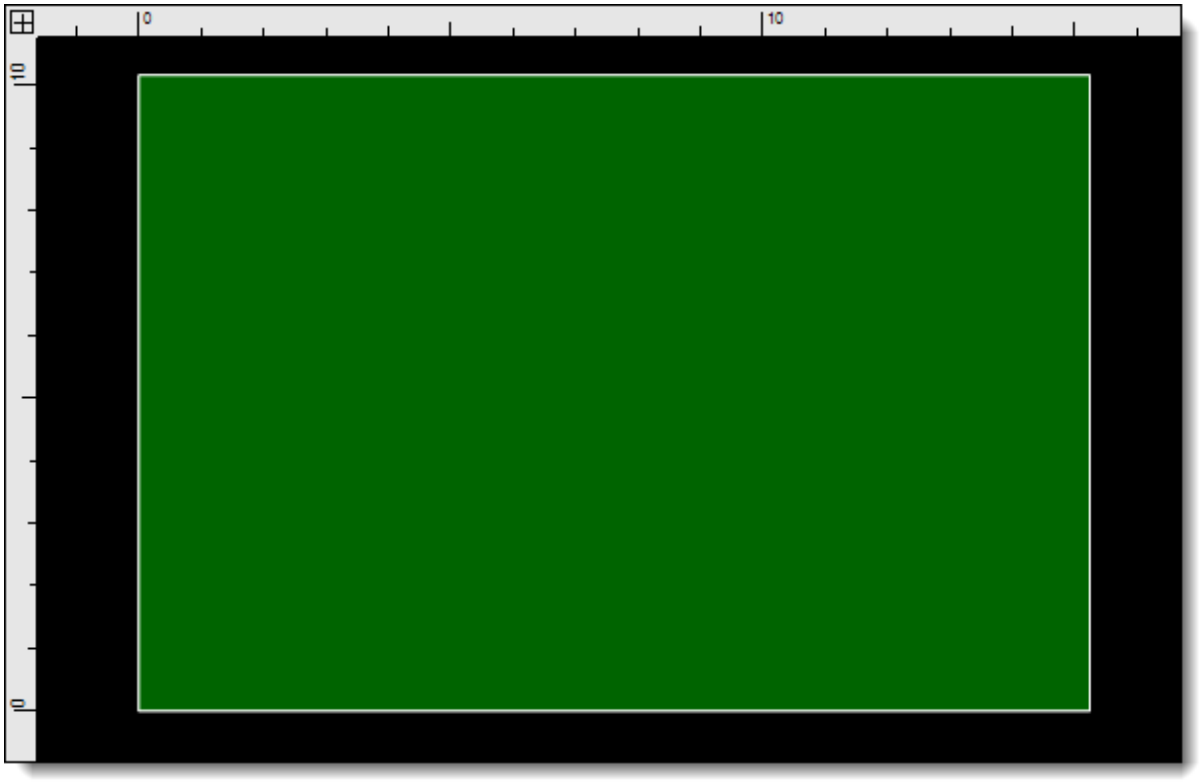
Always consult your PCB manufacturer's design rules and any relevant safety standards or regulations to ensure your PCB design meets all the necessary requirements. It's better to be safe and design with more significant clearances when possible. However, in compact designs, you might have to balance between minimum clearance requirements and the available space.

2.3.13 PCB board sizes and shapes

PCB board sizes and shapes can vary depending on the specific requirements and constraints of the application. Here are some common PCB board sizes and shapes:

Rectangular

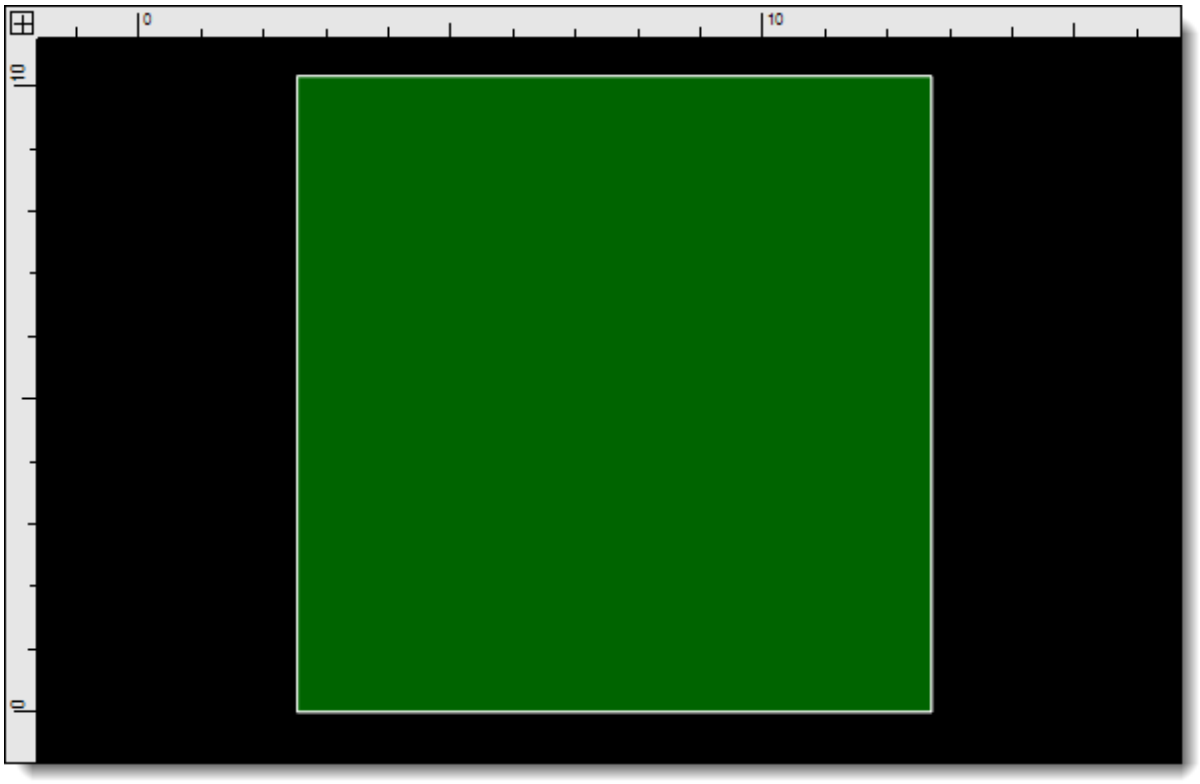
Rectangular PCBs are the most common shape. They can range from small sizes like 1 inch by 1 inch (25mm x 25mm) for compact circuits to larger sizes like 12 inches by 18 inches (305mm x 457mm) for more complex or expansive designs.



A Rectangular PCB

Square

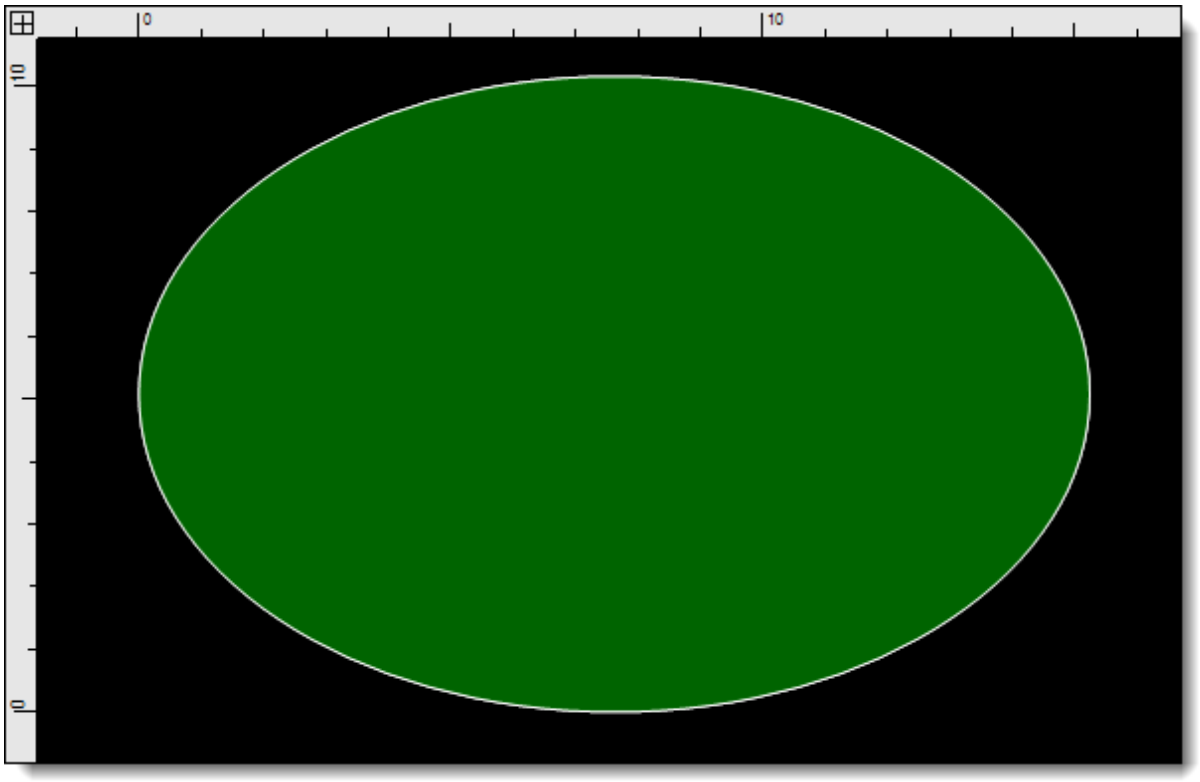
Square PCBs have equal side lengths and are often used when space is limited or for specific applications where a square shape is desirable. Square PCBs can range in size from small to large, depending on the requirements.



A Square PCB

Circular

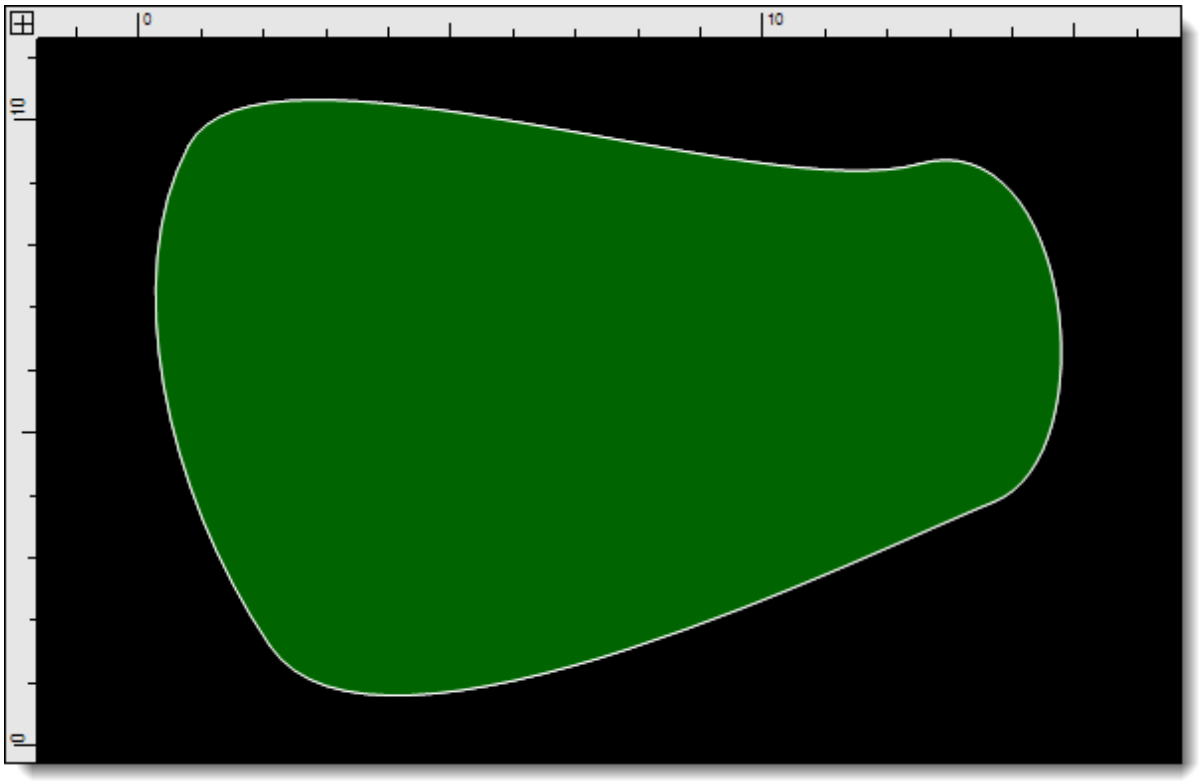
Circular PCBs have a round shape and are typically used for specialized applications or specific design requirements. They can vary in diameter, ranging from a few millimeters to several inches.



A Circular (Elliptical) PCB

Custom Shapes

PCBs can be designed in various custom shapes to fit specific enclosures or accommodate unique mechanical requirements. Custom shapes can include irregular polygons, curved edges, or cutouts for specific components or connectors.

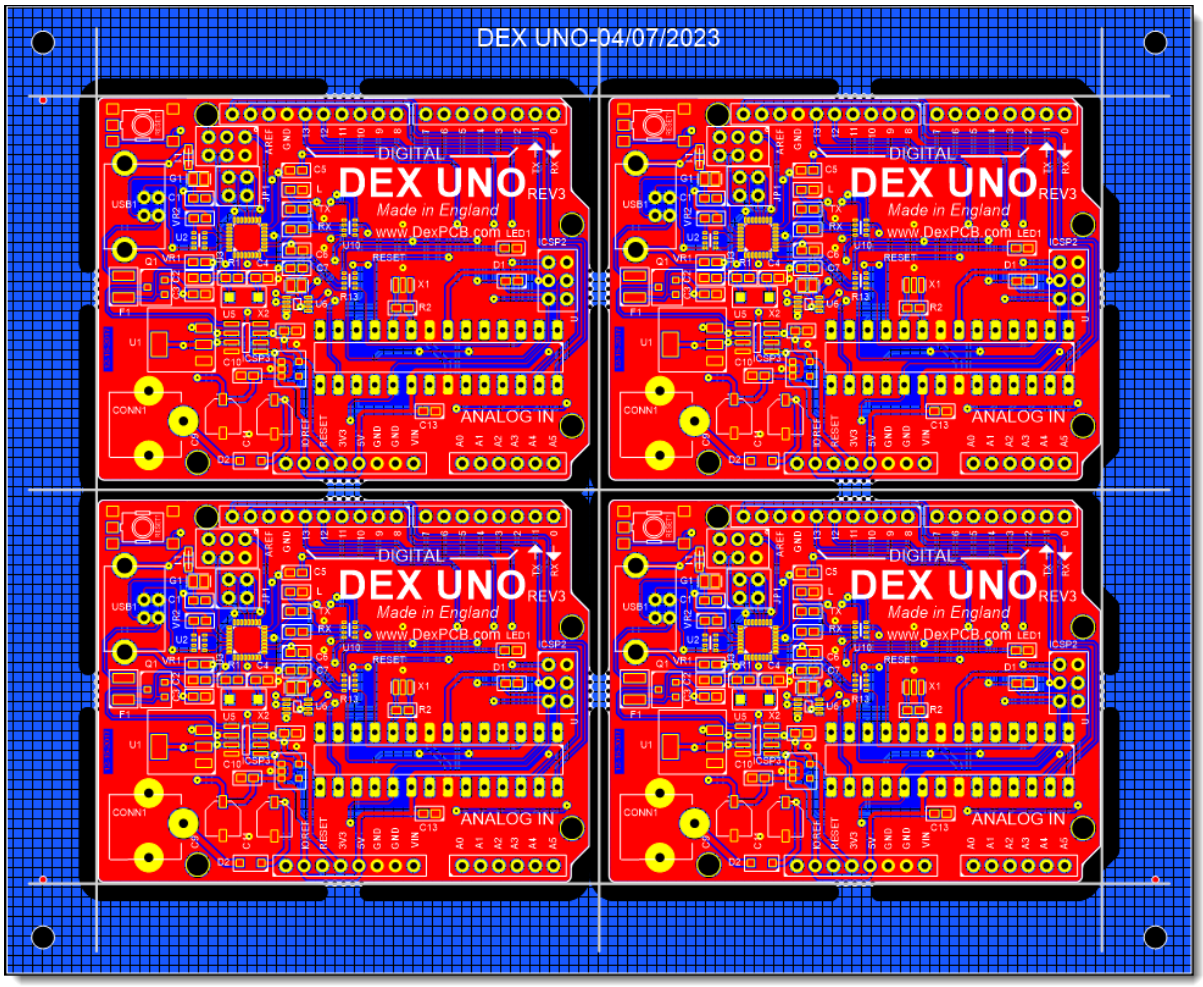


A Custom PCB Shapes

Panelized

In some cases, multiple smaller PCBs may be combined into a larger panel for efficient manufacturing and assembly. Panelization helps optimize the fabrication and assembly processes, especially for mass production.

It's important to note that the size and shape of the PCB should be determined based on the specific requirements of the circuit and the available space in the intended application or system. Factors such as the number and size of components, available enclosure or system space, and any mechanical or electrical constraints should be taken into account when deciding on the PCB board size and shape. Additionally, it's recommended to consult the manufacturer's capabilities and specifications to ensure that the desired size and shape can be manufactured effectively.



A 4x4 PCB Panel

2.3.14 Design of PCB Layers

Designing the layers of a printed circuit board (PCB) involves carefully planning the arrangement of conductive traces, components, and other elements to ensure proper functionality, signal integrity, and manufacturability. The number of layers in a PCB design depends on the complexity of the circuit and the specific requirements of the application. Here are the typical steps involved in designing PCB layers:

- **Schematic Design:** The first step is to create a schematic diagram of the circuit using schematic design software. This diagram represents the logical connections between components, and it serves as the blueprint for the PCB layout.
- **Component Placement:** Once the schematic is complete, components are placed on the PCB layout using PCB design software. During component placement, factors such as signal flow, thermal considerations, and mechanical constraints are taken into account.

- **Single-Layer PCBs:** For simple circuits with few components and connections, a single-layer PCB may suffice. In a single-layer PCB, all the traces are on one side of the board, and components are placed on the same side. However, single-layer PCBs are limited in complexity and are less common in modern electronic designs.
- **Double-Layer PCBs:** For more complex circuits, a double-layer PCB is often used. A double-layer PCB has conductive traces on both the top and bottom sides of the board, allowing for more compact layouts and increased routing flexibility. Vias (plated through-holes) are used to create connections between the top and bottom layers.
- **Multilayer PCBs:** As circuits become more complex or require better signal integrity, multilayer PCBs are used. A multilayer PCB consists of three or more layers of conductive traces separated by insulating layers (dielectric material). The number of layers can range from four to dozens, depending on the design requirements.
- **Routing and Signal Integrity:** During the PCB layout process, the designer must carefully route the traces to minimize signal interference, crosstalk, and impedance mismatches. High-speed signals, such as those in digital circuits or high-frequency applications, require special attention to maintain signal integrity.
- **Ground and Power Planes:** In multilayer PCBs, it is common to dedicate one or more layers to ground and power planes. These large copper areas act as low impedance paths for return currents and provide a stable voltage reference for components.
- **Signal Reference and Split Planes:** Signal reference planes are used to create regions with specific ground voltages for sensitive components or circuits. Split planes can be used to isolate different parts of the circuit to minimize interference.
- **Placement of Decoupling Capacitors**: Decoupling capacitors are strategically placed near ICs and other components to stabilize the power supply and reduce noise.
- **Silkscreen and Solder Mask:** The final step involves adding the silkscreen layer, which includes component labels, reference designators, and other markings for assembly and maintenance. The solder mask layer is applied to protect the copper traces from oxidation and to prevent solder bridges during assembly.

Throughout the design process, the PCB designer must consider factors such as signal integrity, thermal management, manufacturability, and cost to ensure the successful realization of the PCB design.

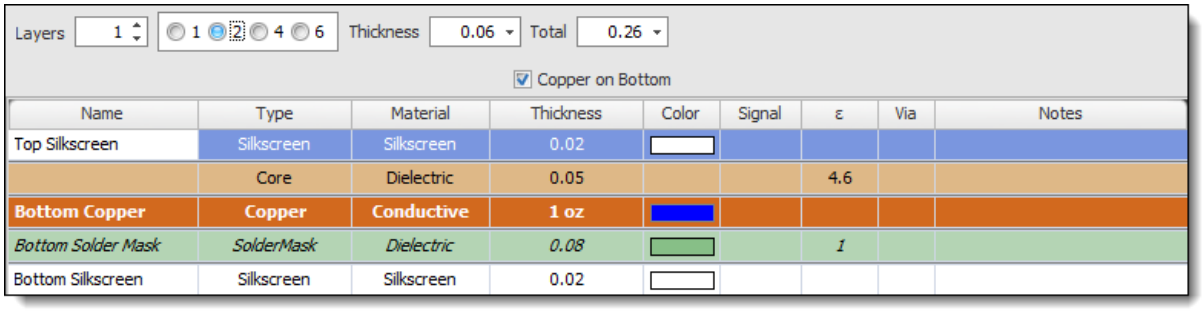
2.3.15 PCB Layer Configurations

PCB layer configuration refers to the arrangement and stack-up of the copper layers in a printed circuit board. The layer configuration is determined by the complexity of

the circuit, the density of components, and the specific design requirements. Here are some common PCB layer configurations:

Single-Layer PCB

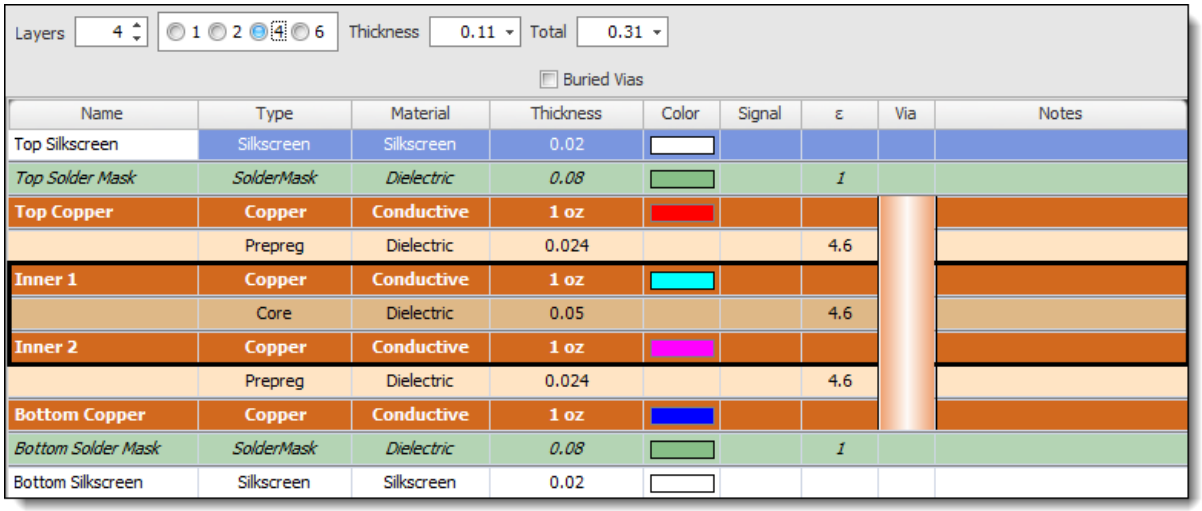
A single-layer PCB consists of only one layer of copper on one side of the substrate. This is the simplest and most cost-effective type of PCB, typically used for low-density and simple circuits.



A Single-Layer PCB Stack

Double-Layer PCB

A double-layer PCB has copper traces on both sides of the substrate. The two copper layers are connected through vias (plated through-holes). This configuration allows for more complex circuitry and increased component density compared to single-layer PCBs.



A Double-Layer PCB Stack

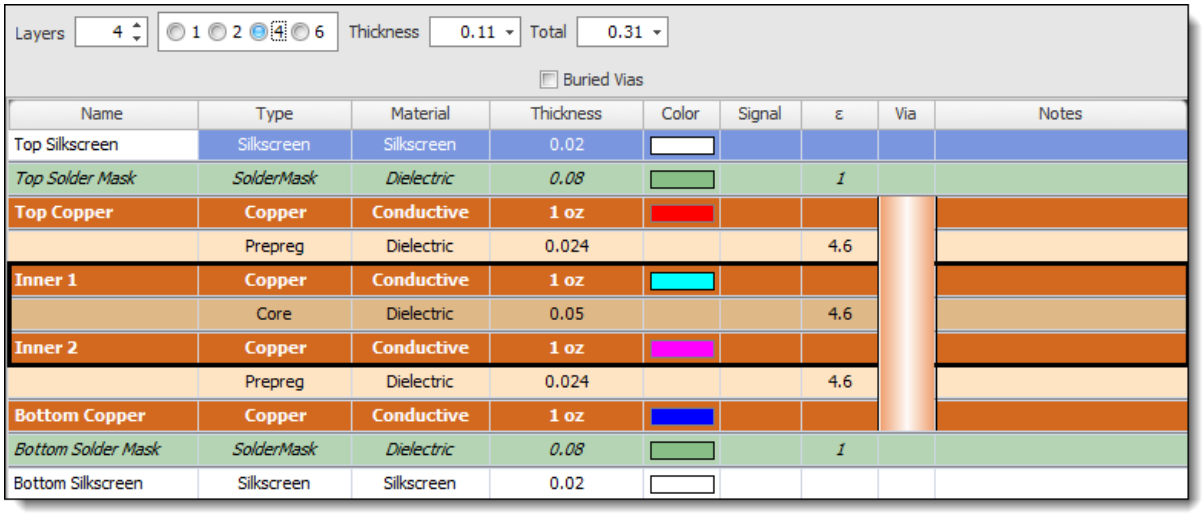
Multi-Layer PCB

Multi-layer PCBs have three or more copper layers separated by insulating material (usually a pre-preg). The number of layers can range from 4 to over 30, depending on the complexity of the circuit. The additional layers provide more routing space, better signal integrity, and improved power and ground plane distribution.

Common multi-layer configurations

Four-Layer PCB

A four-layer PCB typically consists of two internal signal layers and two external plane layers (usually power and ground planes). It is commonly used in medium-density designs.



Layers 4 | 1 2 4 6 | Thickness 0.11 | Total 0.31

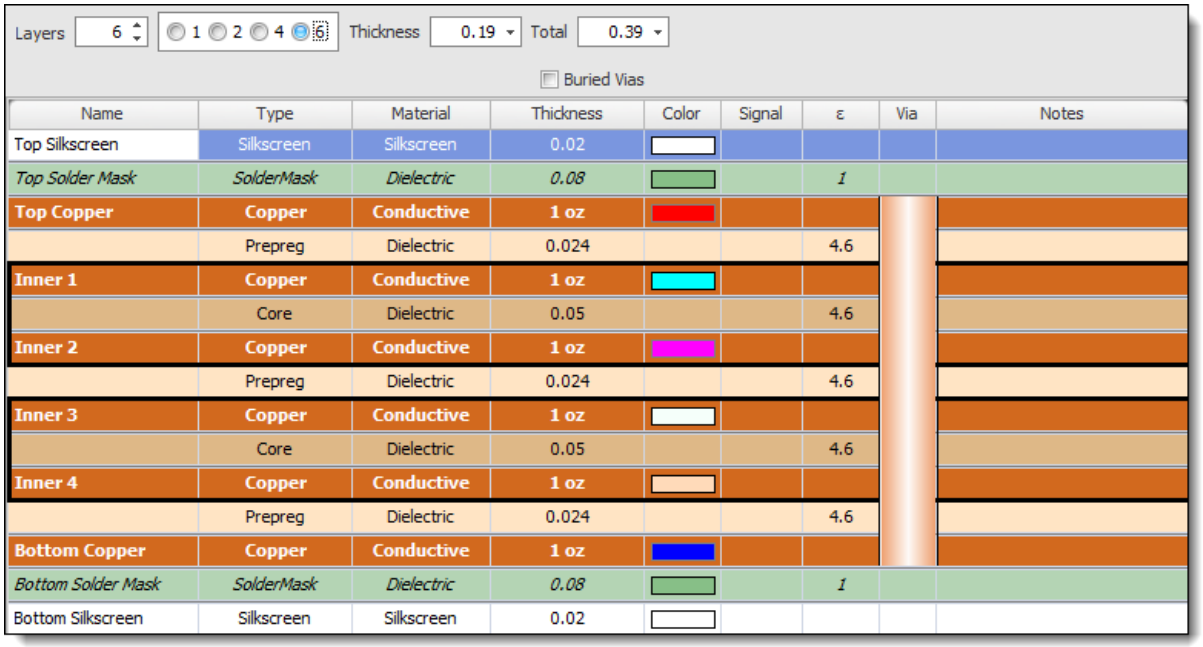
Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

A Four-Layer PCB Stack

Six-Layer PCB

A six-layer PCB adds two additional signal layers to the four-layer configuration. It offers more routing space and is suitable for higher-density designs.



Layers 6 | 1 2 4 6 | Thickness 0.19 | Total 0.39

Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 3	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 4	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

A Six-Layer PCB Stack

Eight-Layer or More

Layers 8 | 1 2 4 6 | Thickness 0.27 | Total 0.47

Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 3	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 4	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 5	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 6	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

An Eight-Layer PCB Stack

PCBs with eight or more layers provide even greater flexibility and space for routing, power distribution, and signal integrity optimization. These configurations are used for complex and high-density designs, such as advanced electronics or high-speed applications.

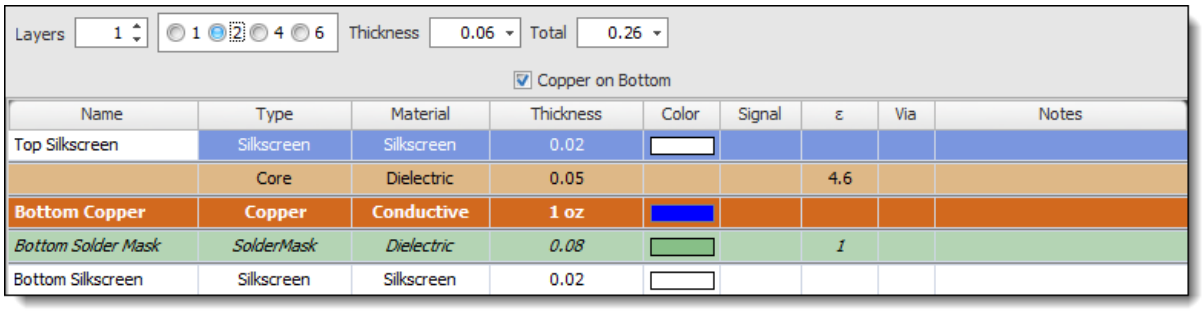
The choice of PCB layer configuration depends on the complexity of the circuit, the desired electrical performance, and the available space in the system or enclosure. It's important to consider factors such as signal integrity, power distribution, and thermal management when determining the appropriate layer configuration for your PCB design.

2.3.16 PCB Layer Stack

The PCB layer stack refers to the arrangement of the layers that make up a printed circuit board. A layer stack is determined by the number and types of layers required for the PCB design.

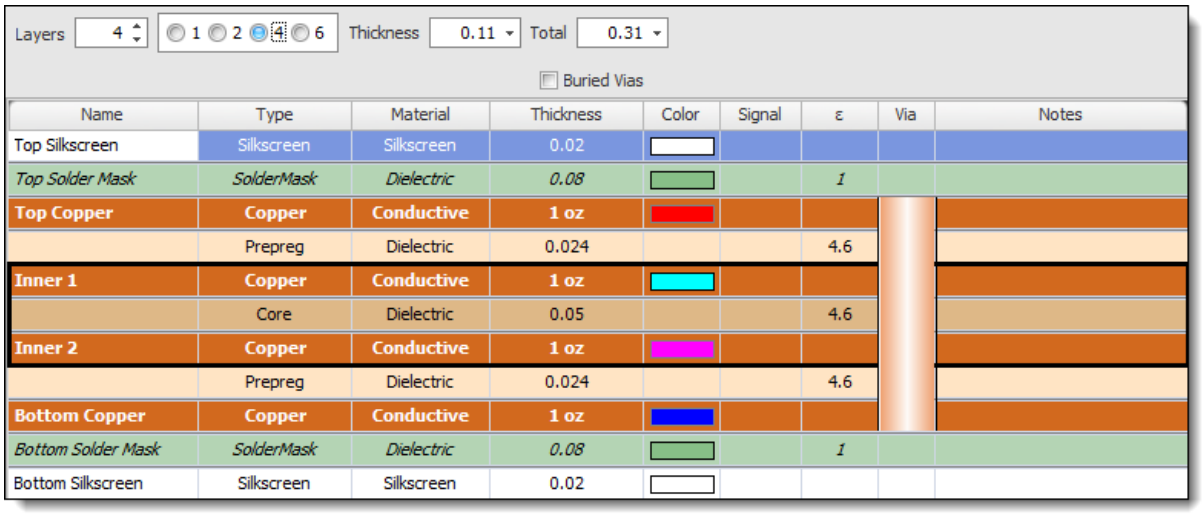
The layer stack is an essential consideration in PCB design as it affects the electrical performance, manufacturability, and overall functionality of the PCB. Here are some common layer stack configurations for PCBs:

Single-Layer PCB. This is the simplest type of PCB, with components mounted on the top and electrticsal tracks on thr bottom.



A Single-Layer PCB Stack

2-Layer PCB: This is the simplest type of PCB, with a top and bottom layer. 2-layer PCBs are typically used for simple designs with few components and short trace lengths.



A Double-Layer PCB Stack

4-Layer PCB: 4-layer PCBs have two signal layers and two ground or power layers. The power or ground planes provide a low impedance path for signals and reduce electromagnetic interference, resulting in improved signal integrity.

Layers 4 | 1 2 4 6 | Thickness 0.11 | Total 0.31

Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

A Four-Layer PCB Stack

6-Layer PCB: 6-layer PCBs have additional signal layers, providing more flexibility for routing traces and reducing the chances of signal interference.

Layers 6 | 1 2 4 6 | Thickness 0.19 | Total 0.39

Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 3	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 4	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

A Six-Layer PCB Stack

8-Layer PCB: 8-layer PCBs provide even more routing flexibility and can support more complex designs with high-density components and signal requirements.

Layers 8 | 1 2 4 6 | Thickness 0.27 | Total 0.47

Buried Vias

Name	Type	Material	Thickness	Color	Signal	ε	Via	Notes
Top Silkscreen	Silkscreen	Silkscreen	0.02					
Top Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Top Copper	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 1	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 2	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 3	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 4	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Inner 5	**Copper**	**Conductive**	**1 oz**					
	Core	Dielectric	0.05			4.6		
Inner 6	**Copper**	**Conductive**	**1 oz**					
	Prepreg	Dielectric	0.024			4.6		
Bottom Copper	**Copper**	**Conductive**	**1 oz**					
Bottom Solder Mask	*SolderMask*	*Dielectric*	*0.08*			*1*		
Bottom Silkscreen	Silkscreen	Silkscreen	0.02					

An Eight-Layer PCB Stack

When designing a layer stack, it is important to consider the following factors:

1. **Signal integrity:** The layer stack should be designed to minimize the potential for signal interference and ensure the desired signal quality.
2. **Power and Ground planes:** Adequate power and ground planes should be included in the layer stack to provide low impedance paths for signals, minimize noise, and improve the overall performance of the PCB.
3. **Manufacturing constraints:** The layer stack should be designed to comply with manufacturing constraints such as the minimum drill size, minimum trace width, and the ability to laminate the layers.
4. **Cost:** A layer stack with more layers is typically more expensive to manufacture. The design should be optimized to achieve the required electrical performance with the fewest layers possible to reduce the cost of manufacturing.

In summary, the PCB layer stack is a critical aspect of the PCB design process. It affects the electrical performance, manufacturability, and cost of the PCB. Careful consideration should be given to the layer stack design to achieve the desired electrical performance while ensuring manufacturability and keeping costs in check.

A Printed Circuit Layer Stack is a type of layered construction used to construct printed circuit boards (PCBs). The layers are typically made from copper foil, which is then etched and plated with other materials for protection. Typically, each layer

has its own purpose and may be used for power supply routing, signal traces, or mounting components. With this type of construction, it is possible to create very complex electrical systems with multiple layers of conductive material in a very small space.

The most common type of stack is the two-layer stackup. As the name suggests, it consists of two layers—the top-side layer and the bottom-side layer—which are separated by a dielectric material such as an epoxy resin. The top-side layer typically consists of a single sheet of copper foil which contains traces that are connected to components on the board. These traces can also be connected to ground plains or other external sources of electricity such as batteries. On the bottom-side layer, there may be additional traces that connect components to each other or to ground plains and external supplies.

In some cases, more than two layers may be used; however, these multi-layer PCBs tend to require more complex manufacturing processes and are not always cost effective for low volume production runs.

When designing and fabricating multiple-layer PCBs, several important considerations must be taken into account including trace spacing between layers, trace widths relative to power supply current requirements, isolation distances between traces and any nearby components or planes in order to avoid interference from stray currents and signals crossing over one another in adjacent layers. Additionally it’s important ensure adequate air gaps between the dielectric material used as insulation between all layers in order prevent short circuits due to capacitive coupling effects between those conductors.

For larger scale designs with multiple types of signals traveling across many different planes within different areas on a board - it’s sometimes necessary to use split planes which separate signals into their respective groups according to their frequency levels in order maintain good signal integrity while minimizing crosstalk issues between various components on the board. In addition - increased trace widths may need to utilized along with additional vias when using high speed signals so that they remain well within acceptable signal tolerances during operation over time without any distortion or loss in quality due transmission line effects arising from adjoining signal lines running parallel together with them on different layers within a stack up configuration.

Finally when making multiple copies of boards using this method - its crucial that each successive design adheres closely enough to the original layout so that assembly process can take place quickly enough without having to make too many changes from duplicate version run batch after batch - thus preserving overall cost effectiveness for large scale fabrication runs where accuracy is paramount for maintaining consistent product specifications throughout every production cycle for years into future if needed!

2.3.17 PCB Split Power Planes

A split power plane refers to a power plane that is divided into multiple sections or regions, each with a specific function. Split power planes are commonly used in PCB designs to isolate sensitive components, reduce noise, and improve the overall performance of the PCB.

Here are some advantages of using split power planes:

Noise reduction

Split power planes can reduce noise in the PCB by isolating sensitive components from noisy regions of the PCB. This can improve the signal integrity and reduce electromagnetic interference.

Thermal Management

Split power planes can improve the thermal management of the PCB by allowing for more efficient heat dissipation from components.

Flexibility

Split power planes provide greater flexibility in the placement of components and routing of traces. The designer can adjust the size and location of the split power planes to optimize the layout for the specific requirements of the PCB.

Protection against shorts: Split power planes can provide protection against shorts in the event of a failure or damage to the PCB.

When designing a PCB with split power planes, it is important to consider the following factors:

Size and Shape

The size and shape of the split power planes should be optimized to achieve the desired electrical performance while ensuring manufacturability and minimizing the cost.

Power Distribution

The power distribution network should be designed to ensure that each split power plane is properly connected to the power supply and other components.

Signal integrity

Care should be taken to avoid any signal interference caused by the split power planes, which can create noise and affect the signal quality.

Thermal Management

The placement and size of the split power planes should be optimized to achieve the desired thermal performance and prevent overheating of the components.

In summary, split power planes are a useful tool for PCB designers to reduce noise, improve thermal management, and protect against shorts. The designer should carefully consider the size and shape of the split power planes and ensure that they do not interfere with signal integrity or cause other issues during manufacturing.

2.3.18 Design for Testability

Design for Testability (DFT) is a methodology that involves designing a product in such a way that it can be easily and efficiently tested for faults or errors. In the context of PCB (Printed Circuit Board) design, this means planning and organizing the board layout, components, and circuitry in a way that makes it possible to effectively test the board for defects and functionality after it's manufactured.

Here are some key strategies involved in DFT:

Test Points

These are specific locations on the PCB that are accessible for probe testing. They should be strategically placed at crucial points in the circuit, such as the input and output of a critical component or subsystem. Adequate test points can make it easier to isolate and identify problems.

Testability Analysis

This is an evaluation of the design to determine how easily it can be tested. It may involve assessing the complexity of the design, the number and accessibility of test points, and the probability of detecting different types of faults.

Built-In Self-Test (BIST)

This involves designing hardware and software features into the board that allow it to conduct self-tests to check its own operation.

Boundary Scan

This is a method for testing interconnects (circuit board wiring and soldering) on digital boards that have boundary scan compatible chips. It involves shifting test data into dedicated test or boundary-scan registers and reading test responses out.

Fault Modeling and Simulation

This involves predicting the likely faults that could occur in the design and then simulating those faults to verify that they can be detected.

Design Partitioning

This involves dividing the design into smaller sections or blocks which can be individually tested. This can make it easier to isolate and diagnose faults.

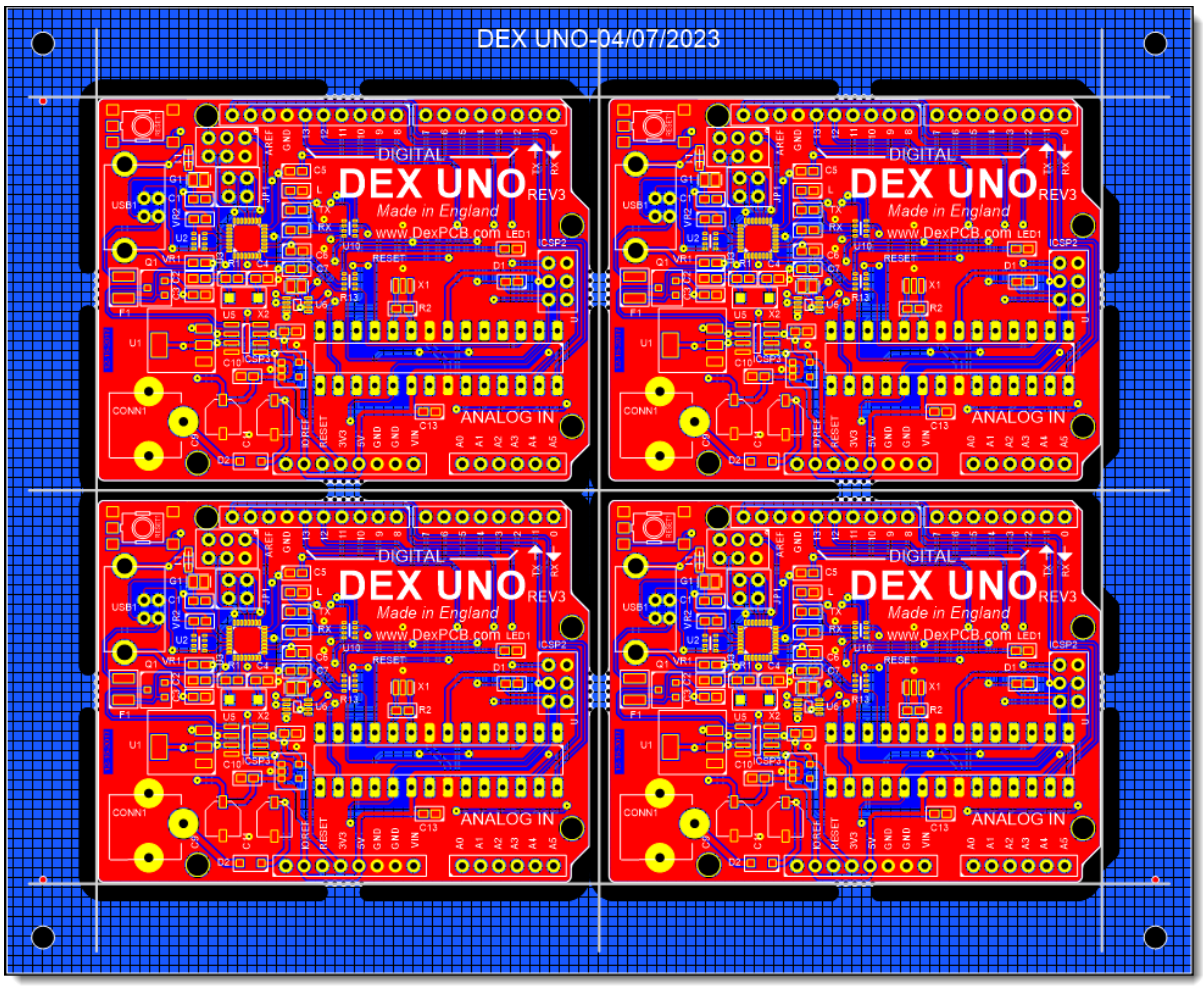
Consider the Manufacturing Process

By considering how the board will be manufactured, designers can ensure that tests will be able to detect potential defects related to the manufacturing process.

The main goal of DFT is to make it as easy as possible to test the PCB and accurately identify any defects. By implementing DFT strategies, designers can increase the likelihood that any faults are detected before the product reaches the consumer, thereby improving product quality and reliability.

2.3.19 What are Panelized PCBs

Panelized PCBs refer to the practice of combining multiple smaller PCBs into a larger panel for manufacturing and assembly purposes. Instead of fabricating and assembling individual PCBs, manufacturers often use panelization to increase efficiency and reduce costs. Here's an overview of panelized PCBs:



A 4x4 PCB Panel

Panelization Benefits

Panelizing PCBs offers several advantages:

- Efficient Manufacturing: Panelizing multiple PCBs on a single panel allows for efficient use of manufacturing equipment, such as automated assembly machines and stencil printers. This leads to faster production times and reduced costs.
- Improved Yield: Panelization helps minimize material waste and improves the overall yield during fabrication. It reduces the risk of damage to individual PCBs during handling and transport, as they are secured within the panel.
- Simplified Assembly: Panelized PCBs streamline the assembly process by allowing for simultaneous placement and soldering of multiple PCBs. This can save time and improve consistency in component placement.

Panelization Techniques

- There are different panelization techniques used in PCB manufacturing. Some common methods include:
- Array Panelization: In this method, multiple identical PCBs are arranged in a grid pattern within a larger panel. The individual PCBs are separated by routing tabs or breakaway tabs. Array panelization is commonly used for high-volume production.
- Depanelization: Once the PCBs are fabricated and assembled on the panel, they need to be separated or depaneled into individual units. Depanelization can be done using various methods, such as routing, scoring, or v-grooving. Special care must be taken during the depanelization process to avoid damaging the PCBs.

Design Considerations

When designing for panelization, it's important to consider the following:

- Panel Size: Determine the appropriate panel size based on the number of PCBs and the manufacturing capabilities of the PCB manufacturer. Ensure that the panel size fits within the manufacturing and assembly equipment used by the manufacturer.
- Panelization Method: Choose the panelization method based on the specific requirements of your project. Consider factors such as PCB size, quantity, ease of assembly, and depanelization techniques.
- Tooling Holes and Breakaway Tabs: Designate tooling holes on the panel for alignment and secure attachment during manufacturing and assembly processes. Include breakaway tabs or routing tabs to hold the individual PCBs in

place within the panel. These tabs should be designed to be easily removed during the depanelization process.

Panelization is commonly used in high-volume production or when multiple smaller PCBs are part of the same project. It offers efficiency gains, improved yield, and simplified assembly processes. Discuss panelization options and requirements with your PCB manufacturer to ensure proper implementation and successful production.

2.3.19.1 What are PCB V-cuts?

PCB V-cuts, also known as scoring or grooving, are cuts made in the surface of a printed circuit board (PCB) to facilitate the separation of the board into individual pieces. V-cuts are made by routing a V-shaped groove into the surface of the board along the line where the board is to be separated.

V-cuts are commonly used in the manufacturing of PCBs, where multiple boards are assembled on a larger panel and then separated into individual boards after assembly. The V-cuts allow the individual boards to be separated easily and cleanly, without damaging the components or traces on the board.

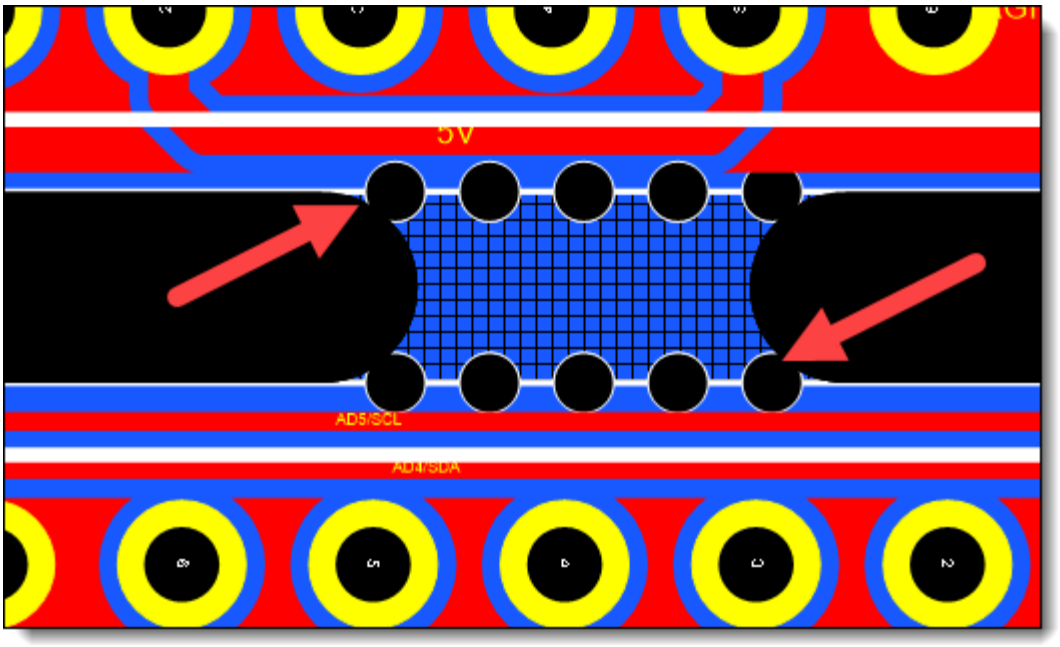
V-cuts can be made using a variety of methods, including mechanical milling or routing, laser cutting, or even manual cutting using a blade or saw. The depth and width of the V-cut are carefully controlled to ensure that the board can be cleanly separated without damaging the components or traces on the board.

In addition to facilitating board separation, V-cuts can also be used to create complex board shapes and to reduce the overall size of the board. By cutting away portions of the board in a V-shape, designers can create intricate shapes and patterns that would be difficult or impossible to achieve using other methods.

In summary, PCB V-cuts are cuts made in the surface of a printed circuit board to facilitate the separation of the board into individual pieces. V-cuts are commonly used in PCB manufacturing to separate boards from larger panels and to create complex board shapes.

2.3.19.2 What are PCB Mouse-bites?

"Mouse bites" in the context of Printed Circuit Boards (PCBs) refer to small, perforated holes that are used to hold multiple PCBs together during the manufacturing process. When PCBs are made, they are often produced in a panelized format for easier assembly. This means multiple PCBs are created as part of a larger panel that can be broken down into individual boards later.



Mouse-bites

When these panels are manufactured, it's often necessary to keep them connected in some way for easier handling and to ensure they all go through the same process at the same time. "Mouse bites" are one of the ways to do this. They're called that because they look like small nibbles a mouse might make.

After the manufacturing process is complete, these 'bites' allow the PCBs to be easily separated from each other by applying a small amount of force, while keeping the individual boards intact and undamaged. This breaking process is also referred to as "depaneling".

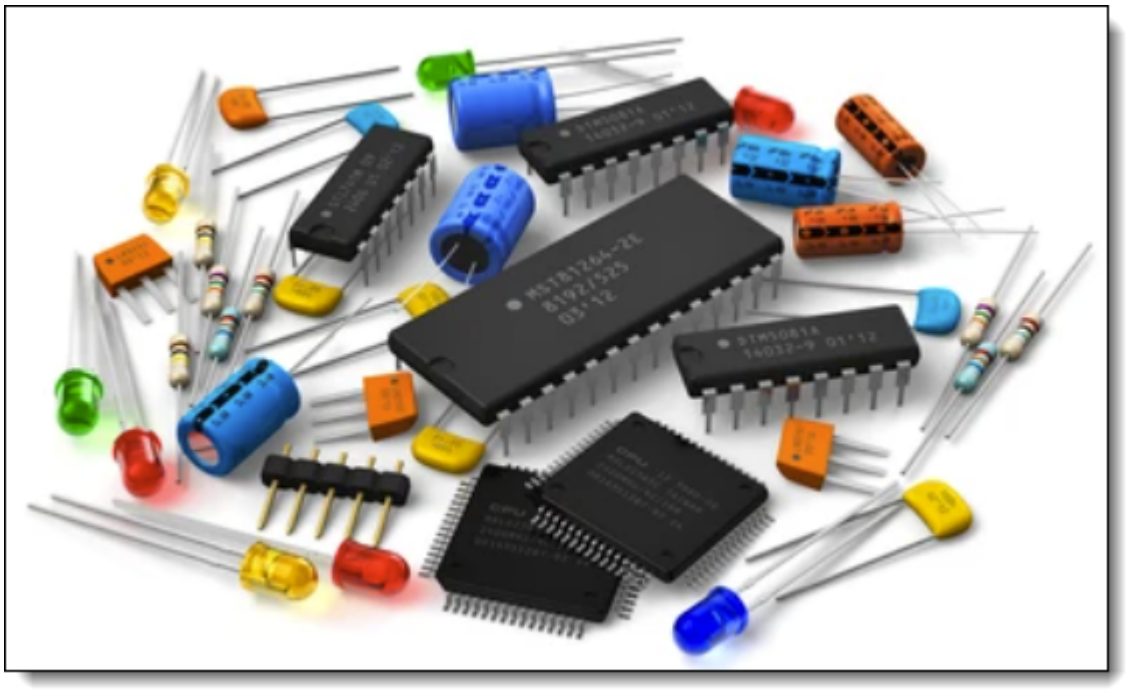
Just note that while mouse bites are a common method for de-paneling, there can be some concerns regarding the stress that breaking the panel can cause to the components on the board. Hence, other methods like V-groove or routing are also popular.

2.3.20 What is PCB Component Placement

PCB component placement refers to the process of arranging electronic components on a printed circuit board (PCB). It involves determining the optimal positions for each component to ensure proper functionality, efficient routing of traces, and ease of manufacturing and assembly. Here's an overview of PCB component placement:

Component Selection

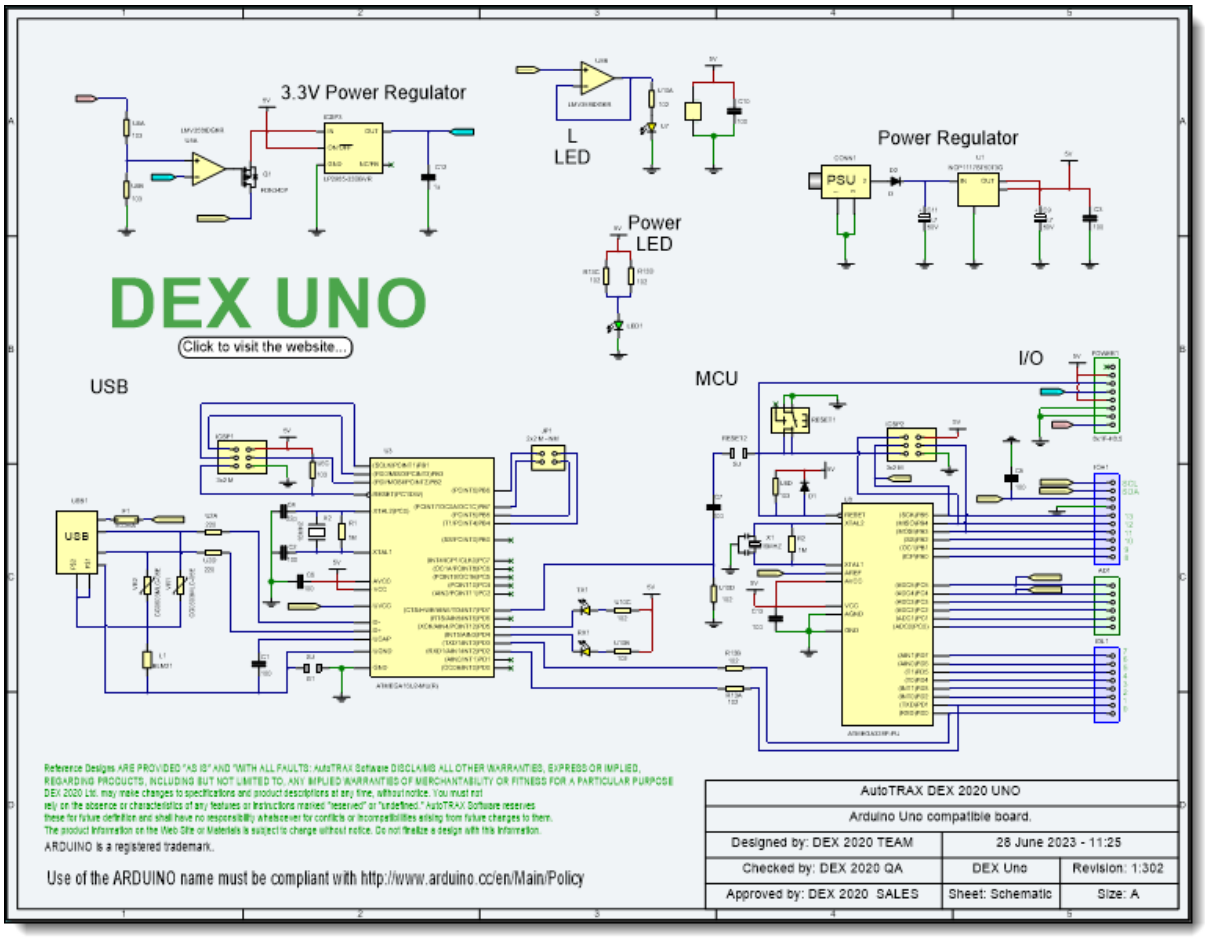
Before placement, you need to select the appropriate components for your circuit based on the design requirements. Consider factors such as component type, package size, electrical characteristics, and availability.



Component Selection

Schematic-Based Placement

Component placement is typically based on the schematic diagram of the circuit. The schematic provides the interconnections between components, allowing you to understand the relationships and dependencies between them.



Schematic-Based Placement

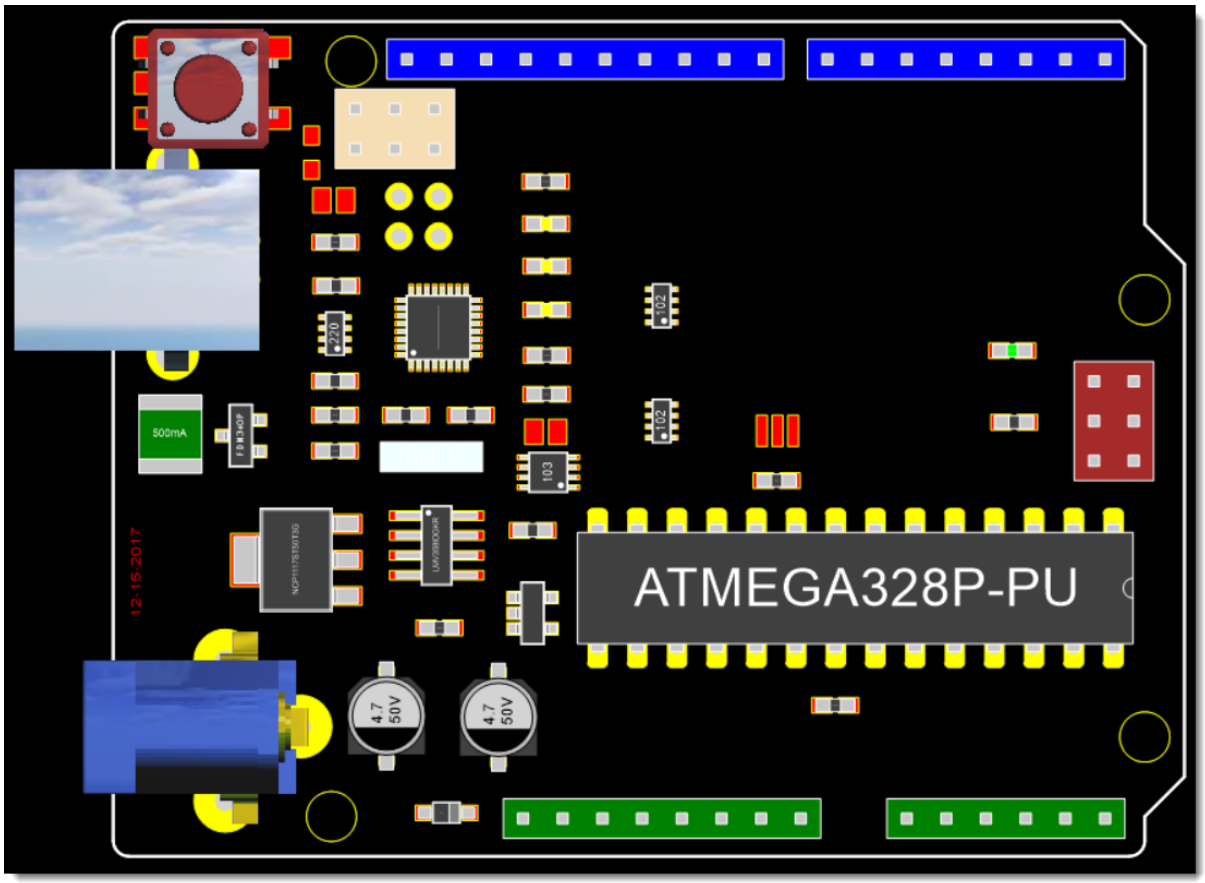
Component Positioning

Consider the following factors when positioning components on the PCB:

- **Functional Requirements:** Arrange the components to meet the functional requirements of the circuit. Group related components together and position them in a logical and intuitive manner.
- **Signal Integrity:** Place components to minimize signal degradation and optimize signal integrity. Consider factors such as trace length, impedance matching, and minimizing signal crosstalk or noise.
- **Thermal Considerations:** Consider the thermal requirements of the components. Place heat-generating components, such as power devices or ICs, in locations that allow for efficient heat dissipation. Avoid placing heat-sensitive components near heat sources.
- **Mechanical Constraints:** Account for any mechanical constraints, such as the size and shape of the enclosure or system where the PCB will be installed.

Ensure that the component placement fits within the available space and allows for proper assembly and maintenance.

- **Assembly and Manufacturing:** Position components in a way that facilitates efficient assembly and manufacturing processes. Consider factors such as access for soldering, test point placement, and avoidance of overcrowding that may impede assembly or inspection.



PCB Component Positioning

Component Orientation

Determine the appropriate orientation of components. Consider factors such as pin configurations, polarized components (e.g., diodes or capacitors), and alignment with reference designators and silkscreen markings.

Design Iteration

PCB component placement often involves an iterative process. You may need to revise and optimize the placement based on factors such as signal routing constraints, manufacturability issues, or design rule violations.

PCB Design Iterations

PCB Design Software

Use PCB design software tools that offer features to assist with component placement. These tools often provide automated or manual placement capabilities, design rule checks, and visualization tools to aid in component positioning and optimization.

It's important to strike a balance between functional requirements, signal integrity, thermal considerations, mechanical constraints, and manufacturing efficiency when placing components on a PCB. Consider consulting with experienced PCB designers or engineers and utilizing appropriate software tools to ensure an effective and optimized component placement.

Find out more...

2.3.21 How to avoid Avoid Mechanical Stress in PCBs

Mechanical stress can lead to various issues in PCBs, including fractured traces, delamination, cracked solder joints, and damaged components. Thus, it is crucial to design and handle PCBs in a way that minimizes mechanical stress. Here are several ways to avoid mechanical stress in PCBs:

Component Placement

Avoid placing components near the edges of the board, as these areas are more likely to be exposed to mechanical stress. Similarly, components should not be placed near holes or slots on the board.

Support Heavy Components

Heavy components, like transformers or large capacitors, can cause the board to bend or warp under their weight. If possible, these components should be supported with additional hardware to distribute their weight more evenly.

Avoid Sharp Corners

Sharp corners in the board layout are more likely to crack under stress. Using rounded corners can help distribute stress more evenly.

Consider Thermal Expansion

Different materials on a PCB expand at different rates when heated. This differential thermal expansion can lead to mechanical stress. Choose materials with similar thermal expansion coefficients and design your board with thermal management in mind.

Proper Handling and Assembly

Mechanical stress can also be introduced during the assembly and handling of the PCB. Be sure to handle boards carefully, and consider using fixtures during assembly to prevent bending or warping of the board.

Vibration and Shock

Consideration should be given to the environment in which the PCB will operate. If the board is likely to be exposed to vibration or shock, use mounting hardware that can absorb some of these forces, and consider using components that are rated for such environments.

Proper De-panelization Methods

PCBs are typically manufactured in panels that contain multiple boards. The process of separating, or de-panelizing, these boards can introduce mechanical stress. Use methods that minimize this, such as routing, laser cutting, or V-scoring.

Designing for Uniform Copper Distribution

Uneven copper distribution can lead to board warping during the heat cycles of manufacturing or during operation. Try to keep copper distribution as uniform as possible.

In conclusion, avoiding mechanical stress in PCBs requires thoughtful design and careful handling. By considering potential sources of stress in the design stage and addressing them proactively, you can significantly increase the reliability and lifespan of your PCBs.

2.3.22 PCB Layout

PCB layout refers to the process of arranging the components, traces, and other elements on a printed circuit board (PCB) in a way that optimizes the performance, manufacturability, and reliability of the PCB. Here are some key considerations when designing a PCB layout:

Component placement

The placement of components on the PCB is critical for achieving the desired electrical performance and signal integrity. Components should be placed strategically to minimize the length of the traces and the number of vias needed to connect them. Components should also be placed in a way that allows for efficient heat dissipation and does not interfere with other components.

Trace routing

The routing of traces on the PCB should be optimized to minimize the length of the traces and reduce the number of vias required. Traces should be routed in a way that minimizes crosstalk and interference, and they should be designed to handle the expected currents and voltages without creating excessive heat.

Ground and power planes

Ground and power planes should be designed to provide a low impedance path for signals and to reduce electromagnetic interference. The size and placement of the planes should be optimized to achieve the desired electrical performance and thermal management.

Design rules

Design rules should be established for the PCB layout to ensure manufacturability and reliability. These rules should include minimum trace widths and clearances, minimum hole sizes, and other specifications that are necessary for the fabrication and assembly of the PCB.

Design for testability

The PCB layout should be designed to facilitate testing and troubleshooting of the PCB. This may include the placement of test points and the use of special features, such as JTAG or boundary scan, to aid in testing.

When designing a PCB layout, it is important to consider all of these factors and to optimize the layout for the specific requirements of the application. PCB layout software can be used to facilitate the design process and to ensure that the layout meets the necessary specifications and design rules.

Printed circuit board (PCB) layout is a critical component of successful electronics design. It is the process of arranging and connecting electronic components to create a functional printed circuit board. It involves designing for component

placement, routing traces between components, creating ground planes, mounting holes, and vias, as well as other features required for a complete functioning device.

The most common way to achieve a high-quality PCB layout is to follow a set of guidelines known as "Design Rules." These rules serve as the foundation on which all successful PCB designs are built. They ensure that all aspects of the layout - component placement, trace routing, ground planes, vias, etc. - are within acceptable tolerances and specifications. Design rules also enable engineers to identify potential problems with their design before manufacturing begins.

When it comes to PCB layout specifically, there are several key steps that must be taken in order to produce an efficient and effective design. The first step is to determine the overall size and shape of the board based on the components being used and the intended purpose of the device being designed. This will help establish the boundaries for component placement and routing later on in the process.

Next, it's important to select an appropriate trace width based on the current requirements of each signal line. In general, thicker traces are better suited for higher current applications while thinner traces may be suitable for lower current lines or signals with less noise sensitivity requirements. It's important that this step be done carefully in order to ensure that no signal integrity issues arise during manufacturing or operation later on down the line.

It's then time to begin actually laying out components onto the board itself while following any applicable design rules such as those outlined in IPC-A-600 or other industry standards documents like JEDEC’s JESD22-B111 series . Component placement should be done using proper spacing so that traces can be routed efficiently between them without running into any interference issues along the way. Additionally, it's important that considerations such as thermal effects from heat generating components are taken into account when placing them onto a board since these could impact device reliability if not addressed properly .

The last step in performing an effective PCB layout is trace routing which involves connecting each component together using conductive pathways known as traces . Here it's important that all necessary connections have been made as specified by either schematics or circuit diagrams provided by engineering teams or customers prior to beginning this phase of work . Furthermore , track widths should also match up with those chosen during Step 2 when selecting trace sizes earlier on in order not to introduce any signal integrity issues here either . Once completed , it’s time at long last check one final time against all applicable industry standards documents - especially Design Rules - before sending off your design files for fabrication !

2.3.23 What are PCB Electrical Requirements

PCB electrical requirements refer to the specifications and considerations related to the electrical characteristics and performance of a printed circuit board. These requirements ensure that the PCB functions properly, meets the design goals, and maintains signal integrity. Here are some common PCB electrical requirements:

Voltage and Current Ratings: Specify the voltage and current ratings for the PCB. This includes determining the maximum and minimum voltage levels that the PCB will handle, as well as the current-carrying capacity of the traces and power planes. Ensure that the PCB materials and traces are appropriately sized to handle the expected voltage and current levels.

Signal Integrity: Signal integrity refers to maintaining the quality and reliability of signals as they travel through the PCB. Consider the following factors to ensure good signal integrity:

Impedance Control: For high-speed digital signals or analog circuits, it is important to control the impedance of transmission lines to prevent signal reflections and maintain signal integrity. Use appropriate trace widths, spacing, and PCB materials to achieve the desired impedance values.

Crosstalk Minimization: Minimize the coupling of signals between adjacent traces to avoid crosstalk. Ensure sufficient spacing between high-speed signal traces, use ground or power planes as shielding, and employ appropriate termination techniques.

Signal Integrity Analysis: Use simulation and analysis tools to evaluate signal integrity, checking for issues such as signal degradation, reflections, and excessive noise. Analyze the PCB layout for timing and impedance discontinuities.

Power Delivery: Ensure proper power delivery to all components on the PCB. Consider the power requirements of each component and design an appropriate power distribution network (PDN). This includes designing power planes, optimizing the placement of decoupling capacitors, and minimizing voltage drops.

EMI/EMC Considerations: PCBs should meet electromagnetic interference (EMI) and electromagnetic compatibility (EMC) requirements. Implement appropriate measures to minimize radiated and conducted emissions and ensure compatibility with other components or systems. Consider grounding techniques, signal shielding, and proper filtering to mitigate EMI/EMC issues.

Thermal Considerations: Account for thermal requirements to prevent overheating of components and ensure reliable operation. Adequate thermal management techniques, such as proper component placement, thermal vias, heat sinks, and thermal pads, should be implemented.

Environmental Considerations: Identify any specific environmental conditions that may impact the PCB's electrical performance. This includes factors like temperature, humidity, vibration, and exposure to chemicals. Design the PCB to withstand these conditions and ensure its long-term reliability.

Design for Testability (DFT): Incorporate features and test points in the PCB design to facilitate testing and troubleshooting during manufacturing and maintenance. Include appropriate test pads, access points, and boundary scan capabilities as needed.

When specifying the electrical requirements for a PCB, it's crucial to consider the specific characteristics and constraints of the circuit and its intended application. Consulting with experienced PCB designers or engineers and utilizing simulation tools can help ensure that the PCB design meets the desired electrical requirements.

2.3.24 PCB Design Guidelines

Printed Circuit Board (PCB) design guidelines are a set of rules and recommendations that help ensure the functionality, manufacturability, and reliability of a PCB. These guidelines are used by PCB designers to create circuits that meet the requirements of the end-product, while also considering the capabilities of the manufacturing process.

Some common PCB design guidelines include:

Component Placement

Components should be placed strategically on the PCB to reduce noise, minimize signal distortion, and ensure thermal management.

Trace Routing

PCB traces should be routed in a way that reduces crosstalk, minimizes signal reflections, and ensures signal integrity. Trace widths and spacing should be designed to meet the required current carrying capacity and impedance.

Grounding

Ground planes should be used to provide a low impedance path for return currents, and to reduce electromagnetic interference (EMI). The ground plane should be connected to the chassis or system ground at a single point.

Power Distribution

Power planes should be used to provide low impedance paths for power distribution, and to reduce noise. High current traces should be designed with sufficient width and thickness to handle the required current.

Thermal Management

The PCB layout should consider thermal management requirements to ensure that heat is dissipated properly. This may involve the use of thermal vias, heat sinks, or other cooling mechanisms.

Design for Manufacturability

PCB designers should consider the capabilities of the manufacturing process and design for ease of assembly, testability, and cost-effective production.

By following these guidelines, PCB designers can create circuits that are functional, reliable, and easy to manufacture.

2.3.25 PCB Signal Integrity

Signal integrity refers to the ability of a PCB to transmit signals without loss or distortion. Poor signal integrity can result in a range of problems, including noise, crosstalk, signal reflections, and timing errors. Here are some factors that can affect the signal integrity of a PCB:

1. Trace length and routing: The length and routing of traces on the PCB can have a significant impact on signal integrity. Traces should be routed in a way that minimizes their length and reduces the number of vias required. Traces should also be routed away from noise sources and other sources of interference.
2. Impedance matching: Impedance matching is important for ensuring that the signal is transmitted without reflections or distortion. Impedance matching can be achieved through the use of transmission lines, impedance-controlled traces, and termination resistors.
3. Ground and power planes: Ground and power planes should be designed to provide a low impedance path for signals and to reduce electromagnetic interference. The size and placement of the planes should be optimized to achieve the desired electrical performance and thermal management.
4. Crosstalk: Crosstalk occurs when signals on adjacent traces interfere with each other. Crosstalk can be reduced through the use of spacing between traces, shielding, and ground planes.
5. EMI/EMC: Electromagnetic interference (EMI) and electromagnetic compatibility (EMC) can affect the signal integrity of a PCB. EMI can be reduced through the use of shielding, ferrite beads, and other techniques, while EMC can be achieved through compliance with regulatory requirements and design standards.

To ensure good signal integrity, it is important to consider all of these factors when designing a PCB. PCB design software can be used to simulate the electrical performance of the PCB and to identify potential signal integrity issues before the PCB is fabricated. It is also important to test the PCB for signal integrity after it is fabricated and assembled to ensure that it meets the necessary specifications and requirements.

2.3.25.1 What is PCB Signal Integrity

PCB signal integrity refers to the ability of a printed circuit board (PCB) to preserve the quality and reliability of electrical signals as they propagate through the PCB traces, vias, and components. Maintaining good signal integrity is crucial for

ensuring the proper functioning of high-speed digital signals and analog circuits. Here are some key aspects of PCB signal integrity:

- **Signal Reflections:** When a signal encounters an impedance mismatch or a sudden change in impedance along its transmission path, it can reflect back, causing signal degradation and potential errors. Proper termination techniques, controlled impedance traces, and careful PCB layout can minimize signal reflections.
- **Signal Delay and Skew:** High-speed signals traveling through PCB traces can experience delays and skew, leading to timing errors. Equal-length signal traces, careful routing, and matched trace lengths help reduce signal delay and skew, maintaining timing integrity.
- Crosstalk: Crosstalk occurs when signals on adjacent traces interfere with each other due to capacitive or inductive coupling. It can lead to signal distortion and noise, affecting signal quality. Adequate spacing between high-speed signal traces, ground or power plane shielding, and proper isolation techniques help minimize crosstalk.
- **Signal Integrity Analysis:** Simulation and analysis tools are used to evaluate and verify the signal integrity of a PCB design. These tools analyze factors like signal propagation, timing, impedance, and noise. They help identify potential signal integrity issues early in the design stage, allowing for adjustments and optimizations.
- **Power Integrity:** Power integrity is closely related to signal integrity since power delivery plays a crucial role in maintaining stable and noise-free signals. Proper decoupling capacitors, power planes, and low-impedance power distribution networks (PDNs) are essential to ensure reliable power delivery and minimize power-related noise and voltage drops.
- **EMI/EMC Considerations:** PCB signal integrity is also connected to electromagnetic interference (EMI) and electromagnetic compatibility (EMC) issues. Proper grounding techniques, signal shielding, and filtering mechanisms help reduce radiated and conducted emissions, ensuring that signals are not compromised by external interference.
- **High-Speed Design Considerations:** As signal frequencies increase, additional signal integrity considerations arise. These include controlled impedance routing, differential signaling for noise immunity, controlled skew for high-speed parallel buses, and careful selection of PCB materials with appropriate dielectric properties.

Maintaining good signal integrity in PCB design requires careful consideration of routing techniques, impedance control, grounding, decoupling, and thorough analysis. Simulation tools, design guidelines, and consultation with experienced PCB designers or engineers can assist in achieving reliable signal integrity and optimal PCB performance.

2.3.25.2 How to ensure PCB Signal Integrity

Ensuring PCB signal integrity requires careful consideration and implementation of various design techniques and best practices. Here are some steps you can take to enhance signal integrity in your PCB design:

- **Controlled Impedance Design:** Determine the appropriate trace widths, spacing, and layer stack-up to achieve controlled impedance matching for high-speed signal traces. Use PCB design software or online calculators to calculate the required trace dimensions based on the desired impedance.
- **Signal Routing:** Pay attention to the routing of high-speed signal traces. Ensure that they are kept as short as possible and follow a direct and straight path to minimize signal delay and skew. Avoid sharp bends, vias, and stubs that can cause signal reflections and degradation. Maintain consistent trace widths and spacing to avoid impedance discontinuities.
- **Differential Signaling:** Use differential signaling for high-speed data buses to improve noise immunity and common-mode rejection. Ensure that the differential pairs are properly matched in length and impedance, and maintain consistent spacing and routing guidelines.
- **Grounding and Power Distribution:** Implement a solid ground plane or planes to provide a low-impedance reference for signal return paths. Separate analog and digital ground planes if necessary and properly connect them at a single point. Place decoupling capacitors strategically near power pins to provide localized power stability and minimize noise.
- **EMI/EMC Considerations:** Employ proper grounding techniques, shielding, and filtering to minimize electromagnetic interference (EMI) and ensure electromagnetic compatibility (EMC). Follow EMI/EMC guidelines and regulations specific to your application or industry.
- **Signal Termination:** Apply appropriate termination techniques to prevent signal reflections and ringing. For example, use series terminations (e.g., series resistors) or parallel terminations (e.g., parallel termination resistors) at the ends of transmission lines to match the characteristic impedance and minimize signal reflections.
- **Simulation and Analysis:** Utilize PCB design software tools that offer simulation and analysis capabilities to evaluate signal integrity. Perform signal integrity analysis to identify potential issues such as signal reflections, crosstalk, and timing violations. Use tools like eye diagrams, signal integrity plots, and power integrity analysis to validate and optimize your design.
- **Component Placement:** Carefully consider component placement to minimize signal path lengths, reduce trace crossings, and ensure proper signal routing. Group related components together and follow guidelines for placement to optimize signal integrity.

- **Design for Manufacturing (DFM) Considerations:** Collaborate with your PCB manufacturer to ensure that the PCB design is manufacturable with high signal integrity. Follow their DFM guidelines and consult with them regarding material selection, stack-up design, and other factors that impact signal integrity.
- **Prototyping and Testing:** Build prototypes of your PCB design and perform thorough testing to validate signal integrity. Use oscilloscopes, signal generators, and other test equipment to measure and analyze signal characteristics, timing, and noise levels.

By following these guidelines and utilizing appropriate design tools and techniques, you can enhance PCB signal integrity and improve the performance and reliability of your electronic circuits. It is also recommended to work closely with experienced PCB designers and engineers to ensure the best possible signal integrity for your specific application.

2.3.25.3 PCB Differential Pair Routing

Differential pairs are an important technique in PCB design for transmitting high-speed digital and analog signals with a high degree of immunity to noise and interference. They consist of two conductive traces (signal and its inverse), designed to carry equal and opposite signals. The receiving circuit detects the difference between the two signals, so common noise induced on both lines can be rejected.

When routing differential pairs on a PCB, it's crucial to adhere to certain guidelines to ensure signal integrity. Here are some key points to consider:

Trace Length Matching

The two traces in a differential pair should be the same length to prevent timing skew between the signals. Timing skew can degrade the performance of the differential pair by causing the receiver to incorrectly interpret the difference between the signals.

Spacing Between Traces (Coupling)

The two traces should maintain a constant separation along their length to maintain consistent impedance. This is often called "edge-coupled" or "over/under-coupled" in a multilayer board. The spacing depends on the differential impedance required by the design, typically between 80 and 120 ohms for most differential pair applications.

Routing on the Same Layer

The two traces of a differential pair should be routed on the same layer of the PCB. This is to ensure that they are exposed to the same electrical environment, thereby ensuring that any noise is induced equally in both traces.

Avoiding Right Angles

As with any high-speed signal, it's best to avoid routing differential pairs with 90-degree angles, as they can cause impedance changes and signal reflections. Use 45-degree or curved bends instead.

Minimize Vias and Other Discontinuities

Each via or other discontinuity (like a change in trace width) can cause a change in impedance, potentially leading to signal reflection. It's best to minimize these where possible.

Guard Traces

To protect differential pairs from crosstalk, it's often a good idea to route them with "guard traces" - ground traces on either side of the pair. The ground traces act as a barrier to prevent signals in nearby traces from interfering with the differential pair.

Differential Termination

It’s important to correctly terminate the differential pair to prevent signal reflections. The termination resistance should match the differential impedance of the pair.

Remember, these are guidelines and not strict rules. Each design might need specific tweaks depending on various factors. Also, make use of the Differential Pair Routing feature available in most advanced PCB design software, which helps to maintain the required spacing and length equality automatically.

2.3.26 PCB Design Guidelines for EMI and EMC

Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) are important considerations when designing PCBs (Printed Circuit Boards). EMI is unwanted interference caused by electromagnetic radiation emitted from electronic devices, while EMC is a device's ability to function properly within its electromagnetic environment without introducing intolerable EMI.

Here are some key PCB design guidelines to reduce EMI and ensure EMC.

Proper Grounding

A good ground layout can greatly reduce EMI. Ground planes should be wide and continuous, and a multi-layer PCB design often includes a dedicated layer for ground. Ground loops should be avoided, as they can act as antennas and radiate EMI. Star grounding or ground grid techniques can be employed.

Trace Routing

High-frequency signals should be routed in shortest paths and avoided from board edges. Keep analog, digital, and power traces separate from each other and don't route high-speed signals across gaps in the ground plane.

Control of Impedance

Controlled impedance lines should be used for high-frequency signals. This can be achieved by keeping the trace width, thickness and distance to ground plane constant along the signal path.

Decoupling Capacitors

These are used to filter out high-frequency noise in power supply lines. They should be placed as close as possible to the power pins of the device they're intended to decouple.

Proper Power Distribution

Similar to grounding, the power distribution network should be designed to ensure all ICs receive clean, stable power. This usually involves careful planning and placement of power planes or power rails.

Component Placement

Place critical, high-speed, or sensitive components first and arrange them in a way that minimizes high-speed signal path lengths. Keep digital and analog components separate. Clock generators or other high-frequency components should be located centrally on the PCB to avoid radiation from the board edges.

Shielding and Filtering

For very sensitive circuits or in high noise environments, shielding can be used to prevent the ingress or egress of EMI. Filtering on I/O lines can prevent EMI from entering or leaving the board through cables.

Layer Stacking

In a multi-layer PCB, the arrangement of signal, ground, and power layers can have a significant impact on EMI. A common practice is to alternate power/ground and signal layers.

Use of Via: Vias inherently cause impedance discontinuities, which can lead to signal reflections and increased EMI. Use them sparingly on high-speed signal paths, and use ground vias to connect different ground planes.

Termination Techniques: Terminate high-speed signal lines properly to prevent signal reflections which can cause EMI. Techniques can include series termination, parallel termination, Thevenin termination, AC termination, etc.

These guidelines are a starting point, but EMI/EMC design is a complex field. For best results, consider using simulation software to predict EMI issues, and always perform EMI/EMC testing on prototypes to confirm that your design meets the required standards.

2.3.26.1 PCB EMC

PCB EMC stands for Printed Circuit Board Electromagnetic Compatibility. It refers to the ability of a printed circuit board to function correctly in an electromagnetic environment without causing or suffering from electromagnetic interference (EMI).

EMI can cause unwanted noise and distortion in electronic circuits, leading to system malfunction or failure. PCB designers must consider EMC during the design process to ensure that the circuit operates correctly and does not cause interference with other components or systems.

Here are some tips for PCB EMC design:

1. *Grounding:* Proper grounding is essential to reduce electromagnetic interference. A ground plane or multiple ground planes can be used to provide a low impedance return path for signals and reduce the electromagnetic field around the PCB.
2. **Signal Integrity:** Signal integrity is important in PCB design to ensure that signals are not degraded due to electromagnetic interference. The trace routing should be optimized to minimize signal crosstalk and signal reflections, which can cause noise in the circuit.
3. **Shielding:** The use of shielding can significantly reduce electromagnetic interference. The shield should be grounded to the PCB ground plane and should be placed around the sensitive components or areas of the PCB.
4. **EMI Filters:** EMI filters can be used to suppress electromagnetic interference. They can be placed on the input or output of the circuit to reduce EMI noise from entering or leaving the circuit.
5. **Component Selection:** The selection of components can also affect the PCB's EMC performance. Components with good EMC characteristics should be selected, and the use of components with built-in filtering or shielding can also help reduce EMI.

In summary, PCB EMC is an important consideration for PCB designers. Proper grounding, signal integrity, shielding, the use of EMI filters, and careful component selection can all help to minimize electromagnetic interference and ensure the circuit operates correctly in the intended electromagnetic environment.

Printed circuit boards (PCBs) are essential components of many electronic systems, and they must demonstrate electromagnetic compatibility (EMC) in order to function correctly. PCBs must have the ability to resist interference from external electromagnetic signals, as well as to not interfere with other systems or electronics. EMC plays a crucial role in ensuring that all connected devices can communicate and exchange data without any hindrance due to the presence of electromagnetic fields.

When it comes to printed circuit boards, there are several factors that can affect their electromagnetic compatibility. The most important factor is the layout of the board

itself; for instance, having sufficient gaps between tracks and components can minimize the interaction between different elements on the board and reduce any chance of interference. Additionally, proper shielding should be implemented when designing PCBs, as this will ensure that any unwanted emissions are suppressed and kept away from other devices within a system. Furthermore, using materials that possess low dielectric constants helps reduce the propagation of electrical signals across a PCB and minimizes the potential for cross-talk between different parts of a board. Additionally, using high quality components on a board also contributes towards better EMC performance; higher-grade components tend to have tighter tolerances which results in better thermal efficiency and improved signal integrity.

In summary, implementing good EMC practices into printed circuit board design is essential for creating reliable electronic systems. It is important for designers to pay close attention to details such as component placement, shielding solutions, dielectric constants and component quality in order to ensure sufficient electromagnetic compatibility across their circuits. Taking these steps will help guarantee that PCBs provide strong interference immunity while still allowing smooth communication between different parts within an electronic system.

2.3.26.2 PCB routing guidelines to reduce EMU and EMC

Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) are critical considerations for Printed Circuit Board (PCB) design. EMI is the interference caused by electromagnetic radiation from electronic devices, and EMC is the ability of a device to function correctly within its electromagnetic environment without introducing intolerable EMI.

Proper routing of PCB traces plays a significant role in controlling EMI and ensuring EMC. Here are some routing guidelines:

Keep Paths Short

High-speed signal paths should be kept as short and direct as possible. The longer the trace, the higher the chance of it acting as an antenna and radiating or picking up interference.

Avoid 90-degree Corners

Instead of 90-degree corners, use 45-degree corners or curves. 90-degree corners have higher inductance and can cause reflections and signal integrity problems.

Maintain Trace Impedance

For high-speed signals, controlled impedance lines should be used. This requires maintaining the trace width, thickness, and distance to the ground plane consistently along the signal path.

Separation of Traces

Keep digital, power, and analog traces separate. Avoid running sensitive signal traces parallel to high-speed or noisy traces.

Avoid Board Edges

High-speed traces should be kept away from the edges of the board, where they are more likely to radiate EMI.

Respect Ground Planes

Never route high-speed signals across gaps or splits in the ground plane. This can cause the return current path to be long and circuitous, increasing inductance, radiation, and susceptibility to interference.

Reduce Use of Vias

Vias cause impedance discontinuities and should be minimized in high-speed signal paths. If vias are necessary, make sure to have nearby ground vias to provide a path for the return current.

Differential Pair Routing

When routing differential pairs, maintain the same length and parallelism for both traces to ensure signal integrity.

Terminate High-Speed Traces Properly

Terminations are necessary to prevent signal reflections, which can degrade signal integrity and cause EMI. Methods can include series termination, parallel termination, Thevenin termination, etc.

Follow Clock Signal Routing Guidelines

Clock signals are often sources of EMI due to their high switching frequencies. They should be routed directly to their destinations (no daisy-chaining), and traces should be shielded by ground planes or traces.

Remember, these are general guidelines. For specific applications, other factors may come into play, and more detailed analysis may be needed. Using EMC simulation software can help predict potential issues, but physical testing is always recommended to verify EMC compliance.

2.3.27 PCB Thermal Management

Thermal management is a critical aspect of PCB design as electronic components generate heat during operation, and excessive heat can degrade the performance of the PCB and even cause failures. Therefore, it is important to implement effective thermal management techniques to ensure that the temperature of the components remains within safe limits. Here are some common techniques used for PCB thermal management:

1. **Heat sinks:** Heat sinks are passive thermal management devices that are attached to components to dissipate heat by increasing the surface area exposed to air or liquid. Heat sinks are commonly used for high-power components, such as power amplifiers and voltage regulators.
2. **Thermal vias:** Thermal vias are used to provide a low thermal resistance path for heat to transfer from the component to the internal or external layers of the PCB. Thermal vias can be used to connect a component to a copper plane, which acts as a heat sink, and they can also be used to connect multiple copper planes to improve heat dissipation.
3. **Copper pours:** Copper pours are areas of copper that are not used for routing traces but instead are filled with copper to act as a heat sink. Copper pours can be placed near components that generate heat or on the inner layers of the PCB to distribute heat more evenly.
4. **Fan or liquid cooling:** Fan or liquid cooling systems can be used to dissipate heat from the PCB by circulating air or liquid over the components. Fan cooling is a passive method that is commonly used for low to medium-power applications, while liquid cooling is an active method that is used for high-power applications.
5. **Component placement:** The placement of components on the PCB can also affect the thermal performance. Components that generate more heat should be placed away from other components and near the edge of the PCB to facilitate heat dissipation.

When designing a PCB, it is important to consider the thermal performance of the PCB and implement appropriate thermal management techniques to ensure the reliable operation of the components. The designer should consider the power dissipation of each component, the ambient temperature, the thermal conductivity of the materials used, and the available space for implementing thermal management techniques.

Printed circuit boards (PCBs) are a key part of any electronic device, and their thermal design is critical to the successful operation of any system. In order to ensure that the PCBs in an electronic system are operating within their acceptable temperature range, careful consideration must be given to their thermal design.

The most important factor in thermal design of a PCB is providing adequate airflow around the components on it. This means ensuring that there is sufficient space for air to circulate freely around all components, as well as between layers of the board itself. It also means maximizing the surface area of each component so as to increase its ability to dissipate heat effectively. Additionally, air-flow can be optimized by including well placed vents in both the board and enclosures used with it, which will provide channels for air to move around more easily and help keep temperatures under control.

The materials used for a PCB also play an important role in its thermal performance. Boa rds made from materials with high thermal conductivity ratings are best suited for applications where heat needs to be quickly dissipated away from components

on the board. On the other hand, boards made from materials with low thermal conductivity ratings may be better suited for applications where slow heat dissipation is desired or when lower temperatures are desired or needed over longer periods of time.

In addition to optimizing airflow and material selection, there are several hardware techniques that can be employed during PCB thermal design such as adding heatsinks or active cooling systems like fans or liquid cooling systems that draw warm air away from components on the board, keeping them at lower temperatures over longer periods of time and preventing them from overheating and failing prematurely.

Finally, software techniques such as dynamic power management can also have a significant effect on how much heat is generated by different components on a PCB at different times throughout its life cycle. By predicting peak loads before they occur based on user input and appropriately adjusting active power consumption accordingly, power-hungry microprocessors can draw less electricity when running at full capacity but still benefit from improved system performance due to shorter processing times. Ultimately this results in less heat being generated overall which helps keep temperatures down on boards with limited airflow options available due to space constraints or other design considerations.

2.3.28 PCB Copper Pour

PCB copper pour refers to the process of filling the unused areas of a printed circuit board with copper, creating a solid copper plane. Copper pour can improve the electrical performance of a PCB by providing a low impedance path for signals and reducing electromagnetic interference.

Here are some advantages of PCB copper pour:

- **Ground plane:** Copper pour can be used to create a solid ground plane on the PCB, providing a low impedance return path for signals and reducing electromagnetic interference.
- **Thermal management:** Copper has excellent thermal conductivity, and copper pour can be used to improve the thermal management of the PCB. The copper pour can be connected to a heatsink or thermal vias to dissipate heat from the PCB.
- **Shielding:** Copper pour can be used to create a shield around sensitive components, providing additional protection against electromagnetic interference.
- **Power distribution:** Copper pour can be used to create a power distribution network on the PCB, reducing voltage drop and improving the power delivery to the components.

When designing a PCB with copper pour, it is essential to follow some guidelines to ensure that the copper pour does not cause any issues. Here are some tips for PCB copper pour design:

- **Keep it simple:** Complex copper pour designs can be challenging to manufacture and may increase the cost of the PCB. Designers should strive to keep the copper pour as simple as possible, minimizing the number of corners and reducing the number of isolated areas.
- **Avoid signal interference:** Copper pour can create a conductive path between signals, leading to signal interference. Care should be taken to avoid copper pour areas that are adjacent to sensitive signals.
- **Thermal relief:** Thermal relief connections should be used to connect the copper pour to the components to avoid soldering issues.
- **Minimize the use of vias:** Vias in the copper pour can create a discontinuity in the solid copper plane, leading to increased impedance. The use of vias should be minimized, and when necessary, they should be placed strategically to avoid signal interference.

In summary, PCB copper pour can improve the electrical performance of a PCB by providing a low impedance path for signals, reducing electromagnetic interference, improving thermal management, and providing power distribution. However, careful consideration should be given to the copper pour design to avoid signal interference and ensure manufacturability.

PCB Copper Pour and Warped PCBs

PCB copper pour can sometimes lead to the issue of warped or distorted PCBs. Warping occurs when the PCB experiences a change in shape, resulting in a curvature or bowing of the board. There are several factors related to copper pour that can contribute to PCB warping:

- **Uneven Copper Distribution:** When copper pour is applied to a large area of the PCB, it can create an uneven distribution of copper across the board. The variation in copper density can cause differential expansion and contraction during temperature changes, leading to warping.
- **Thermal Imbalance:** Copper has a higher coefficient of thermal expansion (CTE) than the PCB substrate material (typically FR-4). During soldering or reflow processes, the copper pour absorbs and dissipates heat differently from the surrounding substrate, causing temperature imbalances that can contribute to warping.
- **Uneven Heating during Soldering:** During the assembly process, such as wave soldering or reflow soldering, the uneven distribution of copper in the copper pour can lead to non-uniform heating. This non-uniform heating can cause localized expansion and contribute to warping.
- **Improper Copper Pour Design:** Poorly designed copper pours that have inadequate clearance from nearby components or traces can result in uneven stress distribution during temperature changes, leading to warping.

To minimize the risk of PCB warping due to copper pour, consider the following practices:

- **Use Copper Pour Sparingly:** Avoid excessively large copper pour areas that cover a significant portion of the PCB surface. Instead, use smaller, localized copper pours that serve specific functions (e.g., ground planes around sensitive components).
- **Maintain Copper Balance:** Ensure that the copper distribution across the PCB is balanced and symmetrical. This can help reduce thermal stresses during heating and cooling cycles.
- **Appropriate Thermal Relief:** Implement thermal relief connections for vias and pads connected to the copper pour. This allows better heat dissipation during soldering, reducing thermal stress.
- **Clearance and Spacing:** Provide sufficient clearance and spacing between copper pour regions, traces, and components to reduce the risk of uneven stress distribution.
- **Controlled Impedance Design:** In high-frequency applications, consider controlled impedance design principles to ensure signal integrity while minimizing the effects of thermal expansion.
- **Consider Material Selection:** Some advanced PCB materials with lower CTE values can help mitigate warping issues. However, they may be more expensive than standard FR-4 materials.
- By carefully designing and implementing the copper pour and considering these factors, PCB warping can be minimized, resulting in a more reliable and mechanically stable printed circuit board.

2.3.29 PCB Test Points

Test points on a printed circuit board (PCB) are designated locations that are used for testing the functionality and diagnosing issues of the PCB during manufacturing and troubleshooting. They provide a way for test probes to make a reliable contact to the circuitry on the PCB.

Test points can come in several forms, but the most common types are:

Surface Test Points

These are typically exposed copper pads placed on the surface of the PCB. They can be probed with a spring-loaded test probe, known as a "pogo pin", in a test fixture.

Via Test Points: These are vias that have been designated as test points. Like surface test points, they can be probed with a pogo pin. Vias can be a convenient

way to add a test point without using additional board space, but they may not provide as reliable a contact as a dedicated surface test point.

Edge Connectors or Gold Fingers

These are contacts placed along the edge of the PCB. They're typically used for functional testing, where the PCB is plugged into a test fixture.

Test points are typically placed at strategic locations throughout the PCB to give access to critical signals, power rails, ground, digital buses (like I2C or SPI), microcontroller pins, and other important nodes. The selection and placement of test points is an important part of the PCB design process. It needs to balance the need for test coverage with considerations like board space, cost, and potential interference with the operation of the circuit.

Once a PCB design is complete and manufactured, automated test systems (like a "Bed of Nails" tester or a Flying Probe tester) can use these test points to quickly and reliably test each board for manufacturing defects or design issues.

Bed of Nails

Bed of nails testing, also known as in-circuit testing (ICT), is a method of testing printed circuit boards (PCBs) to ensure that they were manufactured correctly.

The term "bed of nails" comes from the test fixture used, which consists of a series of spring-loaded pogo pins arranged in a pattern that matches the location of the test points on the PCB. When the PCB is pressed onto this bed of nails, each pogo pin makes contact with its corresponding test point, allowing the testing equipment to measure the electrical characteristics of the circuit.

The bed of nails tester can check for a variety of potential issues, including:

Short Circuits

The tester applies a small amount of current to each circuit and measures the voltage drop to check for short circuits.

Open Circuits

The tester checks for continuity between each pair of points that should be connected.

Component Values

The tester can measure the values of certain components like resistors and capacitors to make sure they are within their specified tolerances.

Digital Logic

For digital circuits, the tester can apply different logic levels (high or low) and check the response of the circuit to make sure it is functioning correctly.

The advantage of bed of nails testing is that it provides a quick and reliable way to test a PCB for manufacturing defects. However, creating the bed of nails fixture can be time-consuming and costly, especially for complex boards. This method is typically used for high volume production where the initial cost of the fixture can be spread over many units. It's also more effective for through-hole and larger surface mount components, as modern circuits with tiny, high-density components can be difficult to test using this method.

2.3.30 What is it PCB net?

In electronics, a PCB net refers to a collection of interconnected electronic components or traces on a printed circuit board (PCB) that are electrically connected together to form a functional circuit.

When designing a PCB, the designer creates a netlist, which is a list of all the connections that need to be made on the board. This includes information about which components need to be connected together and how they should be connected. The netlist is used by the PCB design software to create the physical layout of the board, which includes routing the traces to connect the components according to the netlist.

Each net on the PCB has a unique identifier, which is used to associate the physical traces on the board with the corresponding electrical connections in the netlist. The net identifier is usually a name or number that is assigned to the net by the designer.

PCB nets are important in electronics design because they ensure that the electrical connections between components are correctly established and that the circuit functions as intended. PCB design software includes tools for verifying the connectivity of the nets and checking for errors or inconsistencies in the design, helping to ensure the quality and reliability of the finished PCB.

2.3.31 What are PCB vias?

PCB vias are small holes in a printed circuit board (PCB) that are used to connect different layers of copper within the board. Vias can be thought of as the electrical equivalent of a tunnel, allowing signals to pass between different layers of the board.

Vias can be created in a number of different ways, including drilling, laser drilling, or using a specialized type of conductive ink. Vias can be placed anywhere on the board, and can be used to connect different layers of copper, or to connect components on different layers.

There are two main types of vias: through-hole vias and blind vias. Through-hole vias are drilled all the way through the board, connecting all layers of copper. Blind vias are only drilled partway into the board, connecting the top and bottom layers of copper to one or more inner layers, but not all of them.

Vias can also be stacked on top of each other to create what is known as a "via-in-pad" design. This allows components to be mounted directly on top of the via, saving space and allowing for more compact designs.

Overall, vias are an important feature of modern PCBs, allowing for complex designs with multiple layers of copper and components. They are critical to the proper functioning of many electronic devices and are used in a wide range of applications, from consumer electronics to aerospace and defense systems.

Printed circuit board (PCB) vias are essential elements of a printed circuit board. They are used to create electrical pathways that facilitate the transmission of signals between different layers of the board. These vias come in two main types: through-hole and surface mount.

Through-hole vias are commonly found in traditional PCBs, and they involve drilling holes in the circuit board before soldering components into them. The holes must be placed at specific locations so that an electrical connection is established between components on different layers of the board. This type of via is usually made out of copper plating or carbon-filled epoxy.

Surface mount vias, on the other hand, involve soldering components directly onto the printed circuit board’s surface. This type of via is much more common in modern high-density PCB designs due to its small size and ability to provide a strong mechanical connection between components. Surface mount vias typically consist of copper foil or silver ink for better electrical conductivity and reliability.

In addition to providing electrical pathways between different layers on a PCB, vias can also serve as heat sinks for dissipating heat from power transistors or integrated circuits during operation. By using multiple vias, manufacturers can create thermal management systems that ensure reliable operation even under extreme conditions. Furthermore, strategically placed vias can help reduce electromagnetic interference caused by noise generated by high-speed digital signals on the printed circuit board’s signal traces and components.

Overall, printed circuit board vias play an important role in helping engineers design reliable electronic products with optimal performance characteristics while keeping their costs low through efficient manufacturing processes. When designing a PCB, it is important to consider how best to use these versatile features, as they can affect both your product's cost and performance significantly over its lifetime.

2.3.31.1 What are PCB buried vias?

PCB buried vias are a type of via used in printed circuit boards (PCBs) that are hidden or "buried" within the layers of the board. Unlike through-hole or blind vias, which are visible on the surface of the board, buried vias are located entirely within the internal layers of the board.

Buried vias are typically used in multilayer PCBs, where there are multiple layers of copper separated by layers of insulating material. By using buried vias, designers

can create complex circuits that require multiple layers of copper without sacrificing board space or increasing the board's thickness.

The process of creating a buried via involves drilling a small hole into the board, typically using a laser or other specialized drilling tool, and then depositing a conductive material, such as copper, into the hole. This creates a connection between the different layers of copper within the board, allowing signals to pass between them.

One of the advantages of buried vias is that they can help to reduce signal interference and noise in a circuit. By keeping the via hidden within the layers of the board, there is less chance of interference from other components or external sources.

Overall, buried vias are an important feature of modern PCB design, allowing for complex, high-density circuits that are both reliable and efficient. They are used in a wide range of electronic applications, from consumer electronics to industrial control systems and medical devices.

2.3.31.2 What are PCB blind vias?

PCB blind vias are a type of via used in printed circuit boards (PCBs) that connect the outer layer of the board to one or more inner layers, but do not go all the way through the board. Blind vias are drilled from one side of the board only, and are typically used to connect surface-mount components on the top or bottom layer of the board to inner layers of the board.

Blind vias are useful in multilayer PCBs where there are multiple layers of copper separated by layers of insulating material. By using blind vias, designers can create complex circuits that require multiple layers of copper without sacrificing board space or increasing the board's thickness.

The process of creating a blind via involves drilling a small hole into the board from one side only, typically using a laser or other specialized drilling tool. The hole is then coated with a conductive material, such as copper, to create a connection between the different layers of copper within the board.

One of the advantages of blind vias is that they can help to reduce signal interference and noise in a circuit. By connecting only the necessary layers of copper within the board, there is less chance of interference from other components or external sources.

Overall, blind vias are an important feature of modern PCB design, allowing for complex, high-density circuits that are both reliable and efficient. They are used in a wide range of electronic applications, from consumer electronics to aerospace and defense systems.

2.3.32 What are PCB plated through holes?

Plated through holes (PTHs) are a type of hole used in printed circuit boards (PCBs) that are plated with metal to create a connection between different layers of the board. PTHs are used to connect the top and bottom copper layers of the board to the inner layers, allowing components on different layers to be electrically connected.

The process of creating a plated through hole involves drilling a hole through the entire thickness of the PCB, and then plating the inside of the hole with a conductive material such as copper. This creates a metal lining that connects the different layers of the board, and allows electrical signals to pass through the hole.

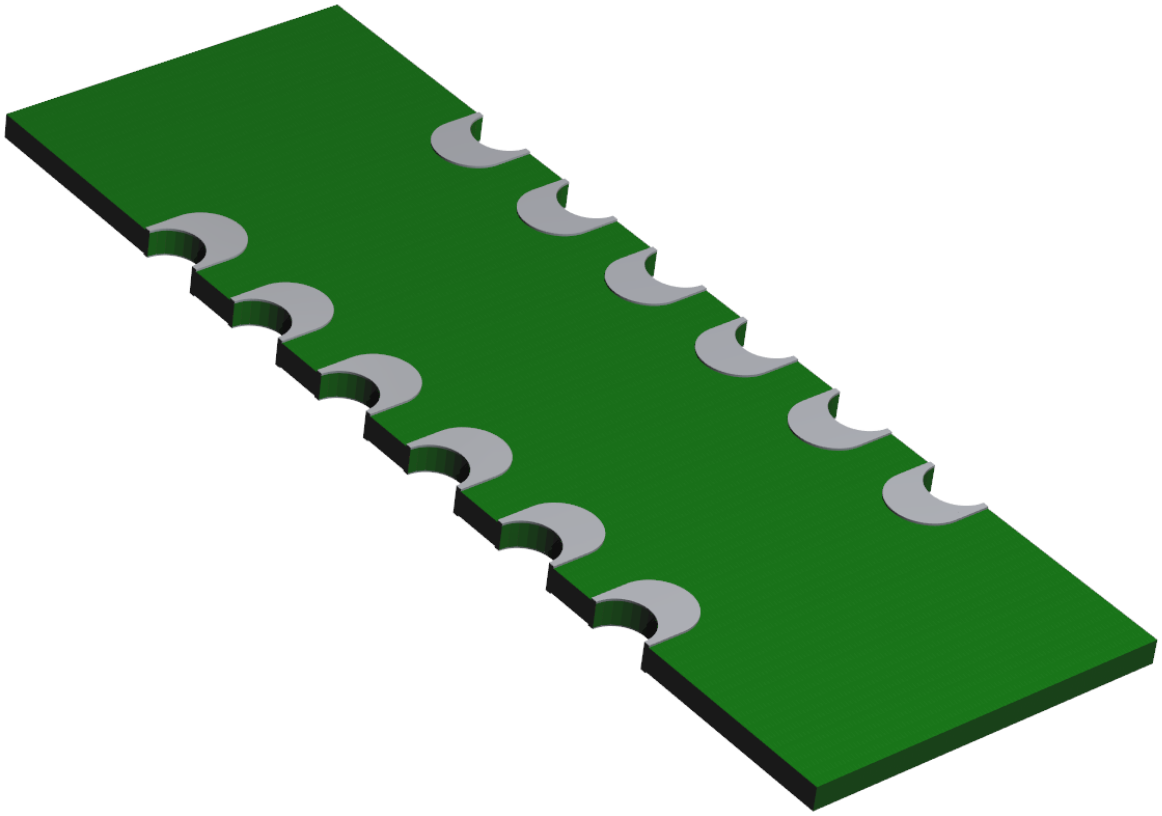
PTHs are commonly used in multilayer PCBs, where there are multiple layers of copper separated by layers of insulating material. By connecting the different layers of copper using PTHs, components on different layers can be connected to each other, and the overall circuit can be more compact and efficient.

In addition to providing electrical connectivity, PTHs can also be used for mechanical support and to aid in the assembly process. For example, they can be used to mount components or to connect the PCB to a larger system.

Overall, plated through holes are an important feature of modern PCBs, allowing for complex and compact circuit designs that can be used in a wide range of electronic applications.

2.3.33 What are Castellated Holes on a PCB?

Castellated holes, or castellations, are a special form of semi-plated indentations made along the edges of Printed Circuit Board (PCB) modules. They serve as crucial links to mount one PCB on top of another during the assembly process. These unique holes can take the form of half holes or resemble varying portions of a fragmented circle, depending on the application. They ensure accurate alignment between boards during the soldering process, a procedure known as board-to-board soldering. These features are often found on PCB modules such as Bluetooth or Wi-Fi modules, enabling their use as separate components during the assembly process.



A Castellated PCB

Common methods used in PCB assembly include Through Hole Technology (THT) and Surface Mount Technology. However, when it becomes necessary to stack one PCB over another, board-to-board soldering comes into play. Here, castellations act as connectors between the module and the board it's to be soldered onto.

Why use Castellated holes?

These edge holes are highly beneficial for replicating specific sections of a PCB circuit. For instance, if a circuit includes an inverter, filter, or feedback loops, these sub-circuits can be mass-produced, tested, and when required, soldered onto the main PCB containing other circuit parts.

Applications of Castellated Holes

They serve as breakout boards for certain sections of a larger PCB.

They offer flexibility in changing the Pin layout as per user requirements.

Integrated modules can be mass-produced on a single PCB using castellation, for use in another assembly during production.

PCBs with castellated holes are easily mounted onto another PCB during final production.

They assist in combining two boards to assess the quality of solder joints.

They are ideal for small modules or breakout boards, such as Wi-Fi modules.

They facilitate the creation of wireless PCB to PCB connections.

Designing Castellated Holes

The process of creating castellated holes or vias involves normal via procedures like drilling and copper plating. These holes appear as semi-circles on the board edges because they are cut to form a half hole or partial hole, creating an edge opening. Castellated holes can be formed in several ways.

Recommended Specifications for Castellated Holes

Certain design guidelines should be followed for castellated holes

- **Size.** Select the largest size possible.
- **Surface Finish.** This depends on the board's intended application, but electroless nickel immersion gold finish is typically recommended.
- **Pad Design.** Using the largest possible pad on both top and bottom is recommended.
- **Number of Holes.** The number of holes depends on the design. It's crucial to find a balance; too many can complicate alignment and assembly.

Use of Castellated holes

Castellated holes on PCBs are widely used across various industries, including telecommunication, computer applications, industrial control, power, automotive, high-end consumer electronics, and more.

2.3.34 What are PCB jumpers?

PCB jumpers, also known as solder jumpers, are small pieces of wire or conductive material that are used to create a temporary or permanent connection between two points on a printed circuit board (PCB). Jumpers are typically used when a circuit requires a connection that cannot be made using a trace on the PCB, or when a connection needs to be made after the PCB has been assembled.

Jumpers can be created using a variety of materials, including thin wire, conductive ink, or small pieces of metal or foil. They are typically placed on the surface of the board, and are often soldered in place to create a permanent connection.

There are two main types of jumpers: shunt jumpers and cut jumpers. Shunt jumpers are typically used to create a temporary connection between two points on the board. They consist of a small piece of metal or foil that is inserted into a pair of jumper pins or pads on the board. When the jumper is in place, it creates a connection between the two points.

Cut jumpers, on the other hand, are used to create a permanent connection between two points on the board. They are typically created by cutting a trace on the board, and then soldering a small piece of wire or conductive material across the cut.

Overall, jumpers are an important tool for PCB designers and assemblers, allowing for flexibility in circuit design and assembly. They are used in a wide range of electronic applications, from consumer electronics to industrial control systems and medical devices.

2.3.35 What are PCB Mounting Holes?

PCB mounting holes are an important part of PCB design, as they allow the board to be securely mounted to a larger system or enclosure. The location and size of the mounting holes are typically specified in the PCB design, and can vary depending on the specific application and mounting requirements.

Mounting holes can be created using a variety of methods, including drilling, routing, or punching. The holes are typically plated to ensure good electrical conductivity and to prevent corrosion. The holes may also be surrounded by a copper pad or annular ring to provide additional support and stability.

In summary, PCB mounting holes are holes that are drilled or routed into a printed circuit board to allow the board to be mounted to a chassis or other object. They are an important part of PCB design and are specified in the design to ensure proper mounting and support of the board.

2.3.36 PCB Bow and Twist

Bow and twist are terms used to describe the amount of deformation or warpage in a printed circuit board (PCB) that can occur during the manufacturing process.

Bow refers to the deformation of a board along its horizontal plane, meaning it is curved along either its x or y axis, resembling the shape of an archer's bow.

Twist refers to the deformation of a board along its vertical plane, meaning the board has a twist from one end to the other, similar to the way you'd twist a wet cloth to wring out water.

The bow and twist in a PCB can be caused by several factors, including:

Uneven distribution of copper across the layers of the PCB

Uneven distribution of heat during the manufacturing process

Differences in the coefficients of thermal expansion of the materials used in the PCB

Physical stress during the manufacturing process, such as during depaneling

Excessive bow or twist can cause issues with the assembly and operation of the PCB. For instance, it can lead to problems with component placement and soldering, particularly for surface mount devices. In severe cases, it can even lead to mechanical or electrical failures.

To minimize these issues, industry standards such as the IPC-600 (Acceptability of Printed Boards) and IPC-6010 series (Performance and Quality Specifications for Printed Boards) define acceptable limits for bow and twist.

The IPC standard generally allows a maximum of 0.75% bow and twist for rigid boards, but the value can vary depending on the specific application and class of the PCB. For example, boards used in high precision or high reliability applications may have stricter requirements.

If you're seeing issues with bow and twist in your PCBs, it might be worth discussing with your manufacturer. They might be able to adjust their processes or materials to help minimize these issues.

2.3.37 PCB Design for Manufacturability

Design for Manufacturability (DFM) is a critical aspect of PCB design that focuses on creating a design that is optimized for efficient and cost-effective manufacturing. PCB designers need to consider the capabilities of the manufacturing process while designing the board to ensure that it can be produced efficiently and with high quality.

Here are some tips for PCB Design for Manufacturability:

Design rules

PCB designers need to follow specific design rules to ensure that the PCB can be manufactured without any issues. The design rules should be consistent with the capabilities of the manufacturing process and should be documented in a design rules file.

Component placement

Proper placement of components can significantly impact the PCB's manufacturability. Components should be placed in a way that allows easy access for assembly, inspection, and test.

Trace routing

Trace routing is another crucial aspect of PCB design. The designer needs to ensure that the routing is optimized for efficient assembly and that the traces are wide enough to handle the required current.

Keep it simple

Complex PCB designs can be challenging to manufacture, and they may lead to higher costs and longer lead times. Designers should strive to simplify the design wherever possible, minimizing the number of layers, vias, and complex features.

Panelization

Panelization is the process of placing multiple PCBs on a single panel to improve manufacturing efficiency. The designer needs to consider the panel size, the number of PCBs per panel, and the location of the fiducial marks.

Test points

Test points are essential for PCB testing and debugging. The designer should include adequate test points to allow for efficient testing and debugging during the manufacturing process.

In summary, designing a PCB for manufacturability requires careful consideration of the manufacturing process's capabilities and limitations. By following the tips outlined above, PCB designers can create a design that is optimized for efficient and cost-effective manufacturing.

2.3.38 Understanding Manufacturing Tolerances on a PCB

When fabricating a printed circuit board (PCB), a manufacturer must follow certain tolerances to ensure that the final product works as intended. Tolerances are the allowable amount of variation in a physical dimension, which is inherent in all manufacturing processes. In PCB manufacturing, these tolerances are defined by a range of acceptable measurements rather than an exact number.

Here are some of the key manufacturing tolerances in PCB production.

Drilling tolerances

Drilling tolerances are crucial in PCB manufacturing because they can affect the fit of the components and the alignment of the layers. There are two types of drilling processes in PCB manufacturing: mechanical drilling and laser drilling, each having its own tolerances.

Mechanical Drilling Tolerances

This is the conventional method of drilling holes in PCBs, and it's commonly used for through-holes and larger vias. For standard PCB manufacturing processes, the typical drilling tolerances for mechanically drilled holes might be as follows:

Hole size: ±0.003" to ±0.006" (±0.075mm to ±0.15mm)

Hole location: ±0.003" to ±0.005" (±0.075mm to ±0.125mm)

Laser Drilling Tolerances

Laser drilling is used for smaller holes, often in high-density interconnect (HDI) PCBs. The tolerances for laser-drilled vias (often called microvias) are usually much tighter. Typical tolerances might be:

Hole size: ±0.0005" (±0.013mm)

Hole location: ±0.001" (±0.025mm)

It's important to note that these values are approximate, and the specific tolerances can vary based on the manufacturer and the specific processes they use. As a PCB designer, you should always check with your manufacturer for the exact specifications they can meet.

In addition, keep in mind that tighter tolerances often mean higher manufacturing costs, so it's a good idea to design with the maximum tolerances that your design can afford. Lastly, make sure that your design software is set to use the same units (imperial or metric) that your manufacturer uses to avoid any conversion errors.

Track width/spacing tolerances

The track (or trace) width and spacing tolerances refer to the variation allowed in the width of the conductive tracks and the space between them. Depending on the manufacturer and the specific production methods, typical tolerances can be around ±10-20%.

The track width and spacing tolerances on a PCB are essential for ensuring that the board performs as intended. They directly impact the board's electrical performance and reliability.

Track width is the width of the conductive paths on the PCB, and it's crucial for determining the maximum current that the track can carry. On the other hand, track spacing is the distance between two adjacent tracks, and it must be maintained to avoid electrical shorts and meet the requirements for electrical clearance.

The tolerances on track width and spacing can vary depending on several factors, including the manufacturing capabilities, the PCB material, the copper thickness, and the complexity of the design.

Here are some general values for track width/spacing tolerances:

Track Width Tolerance

For standard PCB manufacturing processes, a typical tolerance on track width could be around ±10% to ±20%. For example, if a track width is specified to be 0.010" (10 mils), a tolerance of ±20% would mean the actual track width could vary between 0.008" and 0.012".

Track Spacing Tolerance

This also might be in the range of ±10% to ±20% depending on the manufacturing capabilities.

It's important to keep in mind that these are just general values, and actual tolerances can vary from manufacturer to manufacturer. Manufacturers with more advanced capabilities may be able to achieve tighter tolerances. Moreover, tighter tolerances can often lead to higher manufacturing costs. Therefore, when designing your PCB, it's essential to understand the tolerances that your chosen manufacturer can achieve and design your board accordingly.

Layer-to-layer registration tolerances

PCB layer-to-layer registration refers to the alignment of the different layers within a multilayer PCB. In other words, it ensures that each layer of the board lines up correctly with all other layers. This is important for the proper functioning of the PCB, as any misalignment could lead to shorts, opens, or other failures.

Tolerances for layer-to-layer registration can vary significantly depending on the specific manufacturer, design complexity, the board's size, the number of layers, and the technology used in the fabrication process. However, to give you a general idea, here are some typical values:

For standard multilayer PCBs (4-16 layers): around ±4 to ±6 mils (0.1 to 0.15 mm)

For high-density interconnect (HDI) PCBs: as tight as ±2 to ±3 mils (0.05 to 0.075 mm)

Remember that these are general values and the specific tolerances can vary. In any case, if you are designing a PCB, it is recommended to check with your chosen PCB manufacturer to get the exact specifications.

Finally, it's important to note that more precise registration (lower tolerances) often implies higher manufacturing costs. Therefore, when designing, it's recommended to allow the maximum possible tolerance that your design can afford to keep the production costs reasonable.

Impedance tolerances

Impedance control is essential for high-speed digital, RF, and microwave circuits. The impedance of the traces must be controlled to ensure proper signal integrity. Typical impedance tolerances range from ±10% to ±15%.

Impedance control in PCB manufacturing refers to the process of controlling the impedance of signal paths to ensure signal integrity, particularly in high-speed digital, RF, and microwave circuits. If the impedance isn't matched properly between the source, transmission line (the trace on the PCB), and the load, it can result in signal reflections, leading to data loss and degraded performance.

Impedance control is typically important for traces that are longer than 1/6 to 1/10 of the signal's wavelength. For digital signals, it's generally necessary when the signal's rise time is less than half the time it takes for the signal to propagate down the length of the trace.

PCB impedance is affected by several factors, including the width and thickness of the traces, the thickness and dielectric constant (Er) of the substrate material, and the distance to the reference plane (usually a ground or power plane).

When it comes to tolerances, typically manufacturers may offer:

Impedance tolerance

Generally, a standard tolerance range for impedance control in PCB manufacturing is ±10%. However, more advanced manufacturers may offer tighter tolerances, such as ±7.5% or even ±5%.

Keep in mind, achieving controlled impedance with tight tolerances often requires more advanced manufacturing techniques and therefore can increase the cost of the PCB.

The specific values depend heavily on the manufacturer's capabilities and the specific technology they use in their manufacturing process, so it's always a good idea to check with your manufacturer for their specific capabilities.

If you are designing a PCB that requires impedance control, be sure to specify this in your design files and communicate this requirement clearly to your manufacturer. You should also provide them with the target impedance value(s) and any specific tolerances required.

PCB thickness tolerances

The overall thickness of a PCB, as well as the thickness of individual layers, can also have specified tolerances. A typical tolerance on overall PCB thickness might be around ±10%.

PCB thickness tolerances refer to the allowable variation in the thickness of the PCB. This is important as the PCB thickness can influence a variety of factors, including the rigidity of the board, the size of the vias that can be used, and the space available for traces.

The overall thickness of a PCB, as well as the thickness of individual layers, can have specified tolerances. The actual values of these tolerances can vary depending on the manufacturer and the specific manufacturing processes used, but some general values might include:

Overall PCB Thickness Tolerance

For standard PCBs, the overall thickness tolerance is typically around ±10%. This means that if your specified PCB thickness is 1.6mm, the actual thickness could vary between 1.44mm and 1.76mm. For more precise applications, some manufacturers might offer tighter tolerances, such as ±5%.

Copper Layer Thickness Tolerance

The copper layer thickness, often specified in ounces (oz), can also have tolerances. For example, a typical 1 oz copper layer should have a thickness of 1.4 mils (or about 0.035mm), but the actual thickness might have a tolerance of ±10%.

Dielectric Layer Thickness Tolerance

The thickness of the dielectric layer (the non-conductive layer that separates the copper layers) can also have specified tolerances. This tolerance can impact the impedance of the board and is typically around ±10%.

Again, these values are just typical examples, and the specific tolerances can vary based on the manufacturer and the specific processes they use. As with all aspects of PCB design, it's important to check with your manufacturer for their specific capabilities and to make sure that your design can accommodate these tolerances.

Solder mask tolerances

This refers to the alignment and dimensions of the solder mask, which covers and protects most of the circuitry. Tolerances in the solder mask can affect component assembly and soldering processes.

Solder mask is an insulating protective layer applied to the copper traces of a printed circuit board (PCB) to prevent solder bridges from forming between closely spaced solder pads. It helps to prevent corrosion and electrical shorts, and also provides a professional appearance to the PCB.

Tolerances on solder mask application can have an impact on how accurately the solder mask aligns with the pads and how precisely it covers the traces and spaces. Misalignment or inconsistency can cause issues during the soldering process, leading to poor connection or potential electrical shorts.

Here are some typical tolerances related to solder mask:

Solder Mask Alignment Tolerance

This refers to how accurately the solder mask aligns with the copper features on the board. A typical tolerance might be around ±2 to ±3 mils (0.05 to 0.075 mm), but this can vary depending on the manufacturer and the specific processes they use.

Solder Mask Clearance Tolerance

This refers to the gap or clearance between the solder mask and the solder pad. This is typically specified in the design with a solder mask expansion or solder mask swell. The tolerance on this can be around ±1 to ±2 mils (0.025 to 0.05 mm).

Solder Mask Thickness Tolerance

The thickness of the applied solder mask layer can also have tolerances. The typical range might be around 0.8 to 1.2 mils (0.02 to 0.03 mm), but this can vary depending on the specific solder mask material and the application process.

Again, these are just general values, and the specific tolerances can vary based on the manufacturer and the specific processes they use. It's important to check with your manufacturer for their specific capabilities and to design your board with these tolerances in mind.

Copper weight/foil thickness tolerances

This refers to the thickness of the copper layers in the PCB. Higher copper weights allow for larger current carrying capacity but can be more challenging to manufacture with tight tolerances.

It's important to consult with your PCB manufacturer to understand the specific tolerances they can achieve, as this can significantly affect the performance, reliability, and cost of the final PCB. Different manufacturers may also have different capabilities when it comes to achieving tight tolerances, so it's worth shopping around and finding a manufacturer that can meet the specific needs of your design

The copper weight or foil thickness on a PCB refers to the thickness of the copper layer used on the board. This is important because it can affect the electrical performance of the board, including the current-carrying capacity and impedance.

Copper weight is typically specified in ounces (oz), representing the weight of copper spread evenly over an area of one square foot. Common copper weights used in PCBs include 0.5 oz, 1 oz, 2 oz, and even up to 4 oz for high-power applications.

Each of these weights corresponds to a specific thickness:

0.5 oz ≈ 0.0007 inches ≈ 18 µm

1 oz ≈ 0.0014 inches ≈ 35 µm

2 oz ≈ 0.0028 inches ≈ 70 µm

4 oz ≈ 0.0056 inches ≈ 140 µm

The tolerance on the copper weight or foil thickness refers to the allowable variation in this thickness. This can vary depending on the manufacturer and the specific manufacturing processes used. However, some general values might include:

Copper Weight/Foil Thickness Tolerance

The copper foil thickness tolerance can typically be around ±10% to ±20%. However, with advanced manufacturing processes, some manufacturers may be able to achieve tighter tolerances.

So, if you specified a 1 oz copper weight, the actual thickness could vary between 0.00126 inches and 0.00154 inches (or between 32 µm and 38 µm) at a ±10% tolerance.

These are just general values and actual tolerances can vary. It's important to check with your specific PCB manufacturer for their capabilities, as achieving precise

copper weights may involve additional costs. As with all aspects of PCB design, it's important to ensure that your design can accommodate these tolerances.

2.3.39 PCB Finished Hole Size Tolerances

Finished hole size tolerances refer to the acceptable variation in the final dimensions of drilled holes in a printed circuit board (PCB) after all manufacturing processes are complete. This includes not just the drilling of the hole, but also any plating or coating processes that might affect the hole's final dimensions.

Hole size is an essential parameter in PCB design because it directly impacts the fit of through-hole components and the alignment of vias connecting different layers of the board.

The specific values of the finished hole size tolerances can vary depending on the manufacturer and the specific manufacturing processes used. However, for standard PCB manufacturing processes, typical tolerances might look something like this:

Finished Hole Size Tolerance (PTH, Plated Through Hole)

This is typically around ±0.003" to ±0.005" (±0.075mm to ±0.125mm).

Finished Hole Size Tolerance (NPTH, Non-Plated Through Hole)

This is often a bit tighter, around ±0.002" to ±0.003" (±0.05mm to ±0.075mm) as there is no plating process involved that could change the hole size.

For smaller holes, such as those used for microvias in high-density interconnect (HDI) boards, the tolerances can be much tighter. For these, laser drilling is often used and the typical tolerance might be around ±0.0005" (±0.013mm).

As always, it's important to check with your specific PCB manufacturer for their capabilities, as achieving precise hole sizes may involve additional costs. It's also crucial to ensure that your PCB design can accommodate these tolerances to ensure proper fit and function of components and connections.

2.3.40 Nominal Hole Size versus available Drill Bit Sizes

When designing a printed circuit board (PCB), the engineer or designer specifies a nominal hole size for each hole that needs to be drilled. This nominal hole size is the desired finished size of the hole after all manufacturing processes are complete.

However, the actual sizes of drill bits available from a PCB manufacturer may not correspond exactly to the nominal hole sizes specified in the design. Drill bit sizes are usually standardized, with a fixed set of sizes available. The selection of

available drill bit sizes can vary depending on the manufacturer and the specific manufacturing process used.

If the desired hole size is not available, the manufacturer will typically use the next larger drill bit size. This is because subsequent processes, such as plating, will decrease the hole size slightly. For example, if a design calls for a finished hole size of 0.020" and the manufacturer has drill bit sizes of 0.019" and 0.021", the manufacturer would likely use the 0.021" drill bit. After plating, the final hole size would be closer to the desired 0.020".

However, this might vary based on the manufacturer's process capabilities and the specific tolerances required in the design. In some cases, it might be necessary to have a custom drill bit made, or to adjust the design to use a standard drill bit size.

As always, it's advisable to work closely with your PCB manufacturer to understand their capabilities and processes, and to ensure that your design can be accurately and reliably produced.

2.3.41 PCB Drill Bit Size Tolerance

Drill bit size tolerance refers to the allowable variation in the diameter of the drill bit used in drilling holes in a printed circuit board (PCB). The tolerance is important because it affects the accuracy of the drilled hole size, which in turn can affect the fit of components and the electrical connectivity between different layers of the PCB.

In general, drill bit sizes for PCB manufacturing are fairly precise, with typical tolerances ranging from ±0.0005" to ±0.001" (±0.013mm to ±0.025mm). However, the exact tolerance can vary depending on the specific drill bit, the manufacturer, and the drilling process used.

It's important to note that the drill bit size tolerance is just one factor that can affect the final hole size in a PCB. Other factors include drill bit wear, drilling technique, and subsequent manufacturing processes such as plating.

To ensure accurate hole sizes, PCB manufacturers will typically use quality control measures such as regular drill bit inspection and replacement, as well as post-drilling inspection of hole sizes. Additionally, PCB designers can account for these tolerances in their designs to ensure that the final product meets their specifications.

As always, it's advisable to communicate with your PCB manufacturer about their specific capabilities and tolerances, to ensure that your design can be accurately and reliably produced.

2.3.42 PCB Drill Bit Wear

Drill bits used in PCB manufacturing can wear out over time, which can affect the quality of the holes they produce. There are several factors that contribute to drill bit wear in PCB manufacturing:

Material Hardness

The harder the material being drilled, the faster the drill bit will wear out. For instance, some high-performance PCB materials are much harder than standard FR-4, leading to increased drill bit wear. Also, the copper layer itself is harder than the substrate material, which can contribute to uneven wear on the bit.

Drill Speed and Feed Rate

Faster drill speeds and higher feed rates can increase drill bit wear. While faster drilling can increase production speed, it also generates more heat and can lead to more rapid bit wear.

Hole Size

Smaller holes require smaller drill bits, which can wear out more quickly than larger bits.

Hole Quantity

The more holes a drill bit has to make, the more wear it will experience.

Type of Drill Bit

The material of the drill bit also affects its wear. Common materials for PCB drill bits include solid carbide and carbide-tipped bits, which are harder and last longer than high-speed steel bits.

Drill bit wear can affect the quality of the holes in several ways. As a drill bit wears, it can produce holes that are larger, more irregular, or off-center. It can also produce rougher hole walls, which can impact the quality of the hole plating.

To maintain quality, PCB manufacturers will typically monitor drill bit wear and replace bits as needed. This is typically done by keeping track of the number of hits (holes drilled) and/or by visually inspecting the bits for signs of wear.

If you're experiencing issues with hole quality in your PCBs, it may be worth discussing with your manufacturer to see if drill bit wear could be a contributing fact

2.3.43 PCB Hole Cleaning (Desmear)

The term "desmear" refers to a stage in the PCB manufacturing process that's used to clean and prepare the holes drilled into the board for further processing. This step is especially important when preparing the holes for the plating process.

During the drilling process, heat is generated that can cause residues from the base material (commonly epoxy and glass for FR4 boards) to be deposited on the hole walls. These residues, often referred to as "smear", can be a barrier to the adhesion of the plating in the hole. The desmear process is designed to remove this smear and provide a clean, rough surface for good plating adhesion.

There are a few different methods commonly used for desmear:

Chemical Desmear

In this process, the PCB is immersed in a chemical solution (often a mixture of sulfuric acid and potassium permanganate) that dissolves the smear. The board is then rinsed thoroughly to remove any remaining chemicals.

Plasma Desmear

This process uses a plasma (ionized gas) to remove the smear. This is often used for high-density interconnect (HDI) boards and other applications with small or blind vias, where a chemical desmear process might not be as effective.

Mechanical Abrasion (Scrubbing)

This process uses a brush or other abrasive to physically remove the smear. This is often used in combination with a chemical or plasma desmear process for added effectiveness.

After the desmear process, the holes typically go through an etching process to further roughen the hole walls and improve adhesion for the plating process. The board then goes through a series of chemical baths to deposit a thin layer of copper onto the hole walls, which seves as the base for the electroplating process.

It's important to note that the desmear process is a critical stage in PCB manufacturing, as inadequate smear removal or hole preparation can lead to weak plating adhesion and potential reliability issues in the finished board.

2.3.44 Plating of PCB Holes and the Copper Balance

The process of plating holes in a printed circuit board (PCB) is crucial for creating plated-through holes (PTH), which are used to electrically and mechanically connect different layers of the PCB. Plating adds a layer of copper to the walls of the drilled holes, allowing electrical current to pass through.

The plating process typically involves several steps:

Desmear and Etch

As discussed earlier, the drilled holes are first cleaned (desmeared) and etched to provide a clean, rough surface for good plating adhesion.

Copper Seed Layer Deposition

A thin layer of copper is chemically deposited onto the hole walls. This serves as a "seed" layer for the subsequent electroplating process.

Electroplating

The PCB is immersed in an electroplating bath, which contains a solution of copper ions. An electric current is passed through the bath, causing the copper ions to deposit onto the seed layer and form a thicker layer of copper.

Electroless Nickel/Immersion Gold (ENIG) or Other Surface Finish

To protect the copper from oxidation and improve solderability, a surface finish such as ENIG, HASL (Hot Air Solder Leveling), or OSP (Organic Solderability Preservatives) might be applied.

It's crucial to maintain a balance of copper across the PCB during the plating process. Copper imbalance can lead to issues like warping or bowing of the PCB due to different rates of thermal expansion and contraction. Copper imbalance can also lead to uneven plating, where some areas of the PCB (including some holes) might have more or less plating than others.

To achieve copper balance, PCB designers aim to evenly distribute copper across the PCB's surface and throughout its layers. This may involve adding copper pours or thieving patterns in areas without active circuitry. It's also important for the PCB manufacturer to carefully control their plating process to ensure even and consistent plating thickness.

Failure to maintain copper balance can lead to issues with the PCB's mechanical stability and electrical performance, so it's a crucial consideration in PCB design and manufacturing.

2.3.45 The Final Surface Finish of the PCB

The final surface finish of a printed circuit board (PCB) plays a crucial role in its performance. The surface finish is applied to the exposed copper of the PCB to protect it from oxidation, enhance its solderability, and provide a suitable surface for component assembly. There are several different types of surface finishes that can be used, each with its own benefits and drawbacks. Here are a few of the most common:

Hot Air Solder Leveling (HASL)

This is one of the most commonly used surface finishes. In this process, the PCB is dipped into molten solder and then leveled off with hot air knives. HASL provides a reliable, solderable surface, but the high temperatures involved can potentially lead to thermal shock. It's also not ideal for fine-pitch components due to surface planarity issues. There's a variant known as Lead-Free HASL (LF-HASL), which uses a lead-free solder alloy.

Electroless Nickel Immersion Gold (ENIG)

ENIG provides a flat surface, making it suitable for fine-pitch and BGA components. It's a two-layer coating with a nickel barrier layer beneath a thin layer of immersion

gold. ENIG has excellent shelf life and provides good oxidation resistance. However, it's a more expensive process than HASL.

Immersion Silver

Immersion silver provides a flat surface suitable for fine-pitch components. It's cheaper than ENIG but requires careful handling and storage conditions to prevent tarnishing.

Organic Solderability Preservative (OSP)

OSP is a water-based, organic surface finish that is selectively applied to the copper areas to be soldered. It's a cost-effective option for surface finish, providing a flat surface. However, it has a limited shelf life and is not as robust as other finishes against multiple reflow cycles.

Electrolytic Nickel/Gold (Hard Gold)

This finish is typically used for edge connectors due to its durability and wear resistance.

Immersion Tin

This finish is flat and suitable for fine-pitch components, but it is susceptible to tin whiskers and is not recommended for fine-pitch components.

Choosing the right surface finish depends on the specific requirements of the PCB, including the type of components used, the intended application, the operating environment, and cost considerations. It's also important to work closely with the PCB manufacturer to understand the capabilities and limitations of each option.

2.3.46 What are Gerber files?

Gerber files are a set of industry-standard file formats used in the manufacturing of printed circuit boards (PCBs). They are used to communicate the design specifications for a PCB to a PCB manufacturer or fabricator.

Gerber files typically include several different files, each of which contains different information about the PCB design. These files include:

- Top layer (or top copper) file: This file contains the pattern for the top layer of copper on the PCB.
- Bottom layer (or bottom copper) file: This file contains the pattern for the bottom layer of copper on the PCB.
- Silkscreen top file: This file contains the markings and labels for the top layer of the PCB.
- Silkscreen bottom file: This file contains the markings and labels for the bottom layer of the PCB.

- Solder mask top file: This file contains the pattern for the solder mask layer on the top side of the PCB.
- Solder mask bottom file: This file contains the pattern for the solder mask layer on the bottom side of the PCB.
- Drill file: This file contains the location and size information for all of the holes drilled into the PCB.

Gerber files are typically created using specialized PCB design software, and are then used by the PCB manufacturer to produce the physical PCB. The manufacturer uses the Gerber files to create the copper layers, silkscreen markings, and solder mask layers on the PCB, as well as to drill the necessary holes.

In summary, Gerber files are a set of industry-standard file formats used to communicate the design specifications for a printed circuit board to a manufacturer. They typically include files for the copper layers, silkscreen markings, solder mask layers, and drill holes.

The Gerber File Format

The Gerber file format is a standard file format used by the printed circuit board (PCB) industry for designing, manufacturing, and testing printed circuit boards. It's a 2D binary vector format that represents the copper layers, solder mask, legend, and drill and route data of a PCB design.

The Gerber file format originated from Gerber Systems Corp., hence the name. Over the years, it has seen a few revisions. The current standard as of my knowledge cutoff in September 2021 is Gerber X2, which is an extension of the previous standard Extended Gerber, or RS-274X.

Here's a little more detail on what each file in a typical Gerber set represents:

- **Copper Layers:** These files define the conductive areas or traces of the PCB.
- **Solder Mask:** These files identify where the solder resist will be applied. This coating is typically green and prevents solder bridges from forming between closely spaced solder pads.
- **Silkscreen:** These files (also known as the Legend), contain the human-readable characters and symbols (like part names and pin numbers) on the PCB.
- **Drill Files:** These files specify the locations and sizes of the holes to be drilled on the PCB.
- **Board Outline:** This file defines the physical size and shape of the PCB.

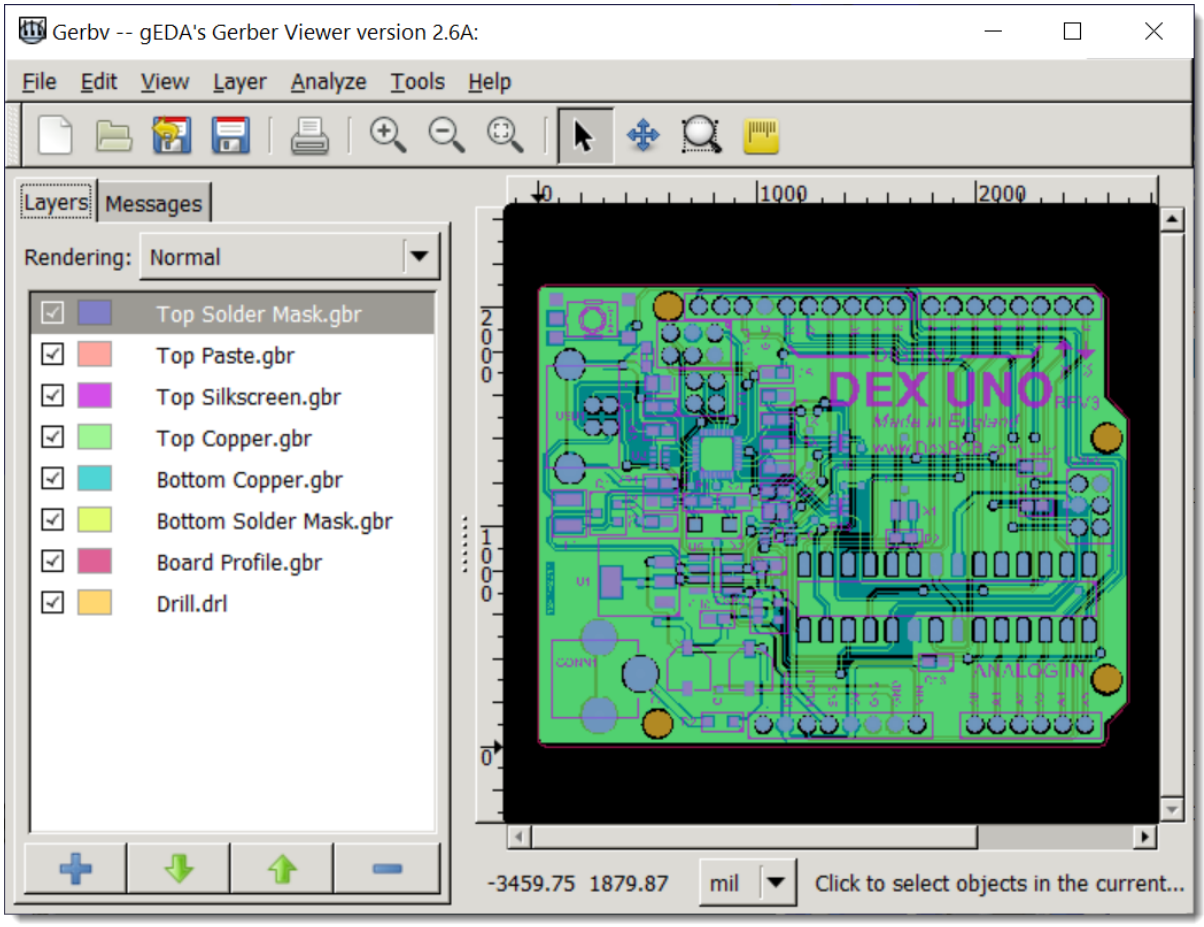
The Gerber file format is widely supported by PCB fabrication houses and is the de facto standard for PCB data exchange. However, as a simple image representation of the PCB data, it doesn't carry any high-level information about the components or the circuit, which is often needed for more advanced manufacturing and assembly

processes. This is one of the reasons for the emergence of more advanced file formats such as IPC-2581 and ODB++ that try to address this limitation.

Download Specifications...

Find out more...

A Free/Open Source Gerber Viewer- gerbv



A Free/Open Source Gerber Viewer- gerbv

Gerbv is an open source Gerber file viewer. Gerber files are standard files used in the electronic industry to communicate the details of printed circuit board (PCB) designs to manufacturing facilities.

Gerbv provides the ability to view, print and do some basic manipulations with these files. It is part of the gEDA project, which is a full GPL'd suite of Electronic Design Automation tools.

As of my knowledge cutoff in September 2021, the tool could handle the Gerber RS-274X format, Excellon drill files, pick and place files, and a few other types.

It's a valuable tool for electronic engineers and anyone involved in the process of PCB manufacturing because it allows for a detailed examination of PCB designs

before they are manufactured, potentially catching and correcting errors early in the process.

Download gerbv...

Ucamco

Ucamco, a company based in Ghent, Belgium, is a global leader in photoplotting and direct imaging systems for PCB (Printed Circuit Board) fabrication. They also develop software solutions for the PCB industry, including communication tools and quality control equipment.

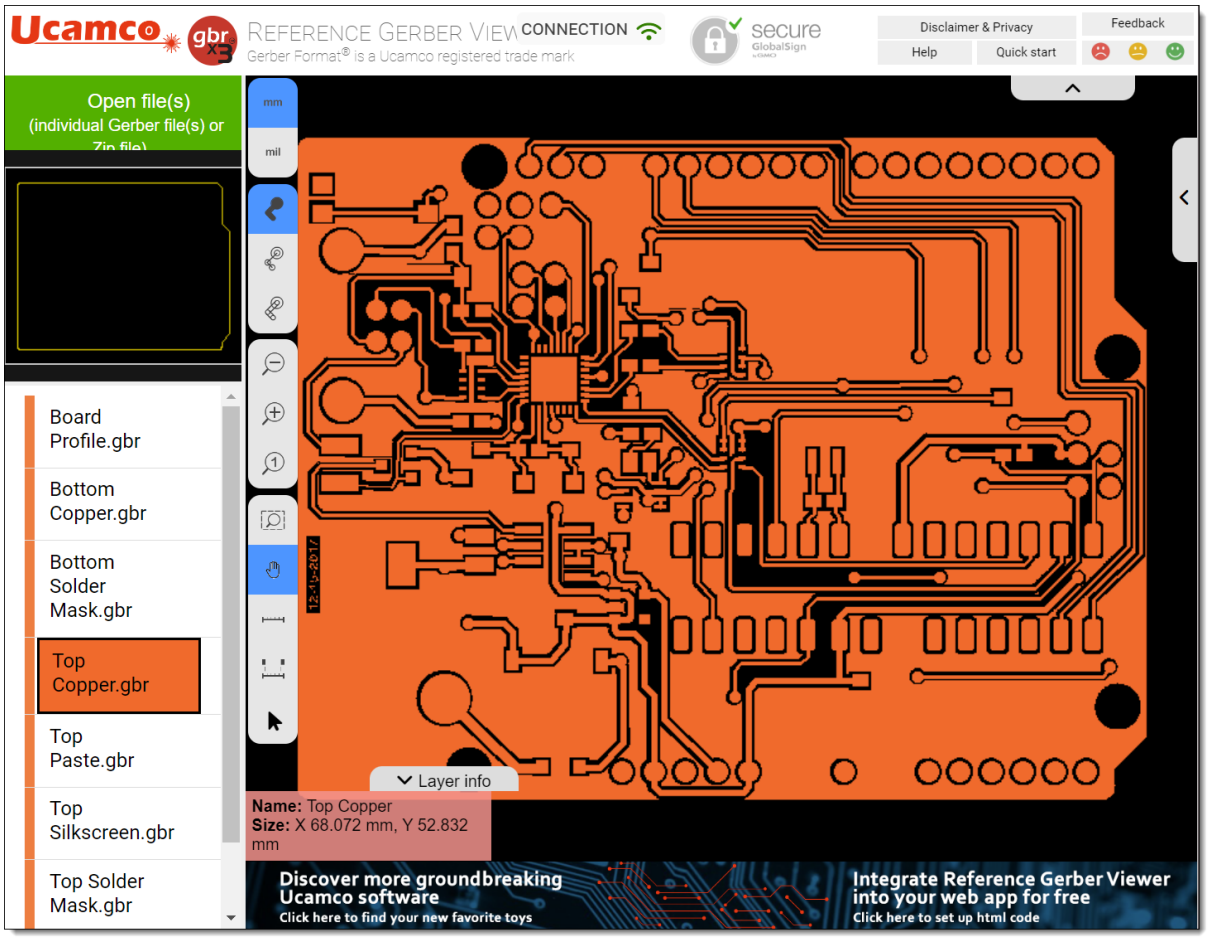
One of their significant contributions to the PCB industry is the maintenance and development of the Gerber file format. The Gerber file format is a widely used, standard file format for communicating PCB design information to manufacturing. It is used to represent the copper layers, solder mask, legend, and drill and rout data in 2D images.

Originally, the Gerber format was a proprietary format developed by Gerber Systems Corp., but it has since been developed into an open ASCII vector format by Ucamco, which took over maintenance of the Gerber format. The latest version as of my last update in 2021 is the Extended Gerber, or RS-274X, which includes embedded aperture information, making it a more complete and less error-prone file format for PCB production.

Ucamco also defined the Gerber Job file format, which is a JSON-based file format for exchanging PCB job parameters. This complements the Gerber image format by providing a way to package all the parameters for a PCB job into one file.

For more recent information, it would be best to visit https://www.ucamco.com/ or reach out to them directly.

The Free Ucamco Gerber Reference Viewer



The Free Ucamco Gerber Reference Viewer

[Open Free Ucamco Gerber Reference Viewer..](#).

The Gerber RS-274X File Format

RS-274X, also known as Extended Gerber or X-Gerber, is the most widely used standard for data exchange in PCB manufacturing. It's an extension of the original Gerber format, RS-274-D, also known as Standard Gerber or just Gerber.

RS-274X introduced a number of key improvements over the original format. While Standard Gerber required a separate aperture file to interpret the design information, RS-274X embeds the aperture information into the file itself, making the format self-contained and less prone to errors and confusion. This was a huge benefit and made the format more reliable and easier to use.

In an Extended Gerber file, each line of code corresponds to a command for a photoplotter, a machine that uses light to draw an image onto a photosensitive material. The commands include things like:

- "Move to coordinates X,Y"
- "Draw a line to coordinates X,Y"

- "Flash (draw) an aperture (shape)"
- "Interpolate a circle or arc with a specific radius to coordinates X,Y"

These commands collectively describe all of the shapes and patterns on the individual layers of a PCB.

RS-274X also allows for the definition of custom apertures, providing designers with much more flexibility in their designs. It also introduces the ability to specify polarity (dark or clear) for each shape, allowing for the creation of negative images.

Despite being an older format, Extended Gerber is still widely used and is accepted by virtually all PCB manufacturers. However, as mentioned before, more advanced formats like Gerber X2, IPC-2581, and ODB++ are increasingly used due to their ability to convey more complex design information.

The Gerber X3 File Format

Introduced in 2020, Gerber X3 stands as a leading data file format for the transfer of printed circuit board (PCB) design data. It has gained acceptance as a default standard due to its inclusion of both PCB layout data along with the Bill of Materials (BOM) and Component Placement List (CPL) data in one comprehensive and easily accessible file format.

The Gerber X3 standard expands its capabilities by incorporating component data within the Gerber data. This includes geometric data such as the centroid, outline, fiducial locations, and footprints, all of which can naturally fit into a Gerber image file.

The major benefit of combining raw board data and component data in Gerber files is the ability to holistically review the final board. This results in a streamlined workflow, facilitating the transfer of component information from the Computer-Aided Design (CAD) system to the Assembly Computer-Aided Manufacturing (CAM) system.

The key advantage here is the unification of data into a single format that can be easily imported, analyzed, and visualized by the CAM system. This allows for the automatic generation of outputs for various assembly equipment, leading to significant time and cost savings.

Maintaining the tradition of the original Gerber format, X3 is a straightforward, human-readable ASCII format. This enables engineers to quickly identify and rectify any anomalies. Additionally, Gerber X3 ensures backward compatibility, allowing users of older systems to read and use its data, albeit limited to features supported by their current system.

Gerber X3 files incorporate a new data set known as Component Layers (Top & Bottom). Within this set, there are dedicated attributes for components where all necessary information about each component on every layer can be defined. This further enhances the format's capability and utility in PCB design.

2.3.47 Gerber File History and Future

History

Gerber files play an integral role in transferring PCB designs from designers to fabricators globally, accounting for over 95% of all transfers. These files are generated automatically by most CAD systems, often without requiring designers to understand their intricate workings. This testament to the power and prevalence of the Gerber file format does not negate occasional complications, necessitating some understanding of the format's structure. Furthermore, there are future plans for enhancing the format's utility.

The Gerber file format derives its name from Joe Gerber (1924-1996), a US inventor who escaped from Austria in 1940. During his student years, he was deeply interested in accurate data plotting. In the 1950s, he created the digital XY co-ordinate table, which eventually formed the foundation for his future business, Gerber Scientific. The company's debut product was a digital drafting machine, one of the first of its kind globally. Subsequent products included automatic cloth cutting machinery and computerized spectacle lens machining equipment, both of which are still widely used today.

In the 1960s, Gerber devised a novel application for his XY table - the world's first NC photoplotter. The device was designed to create phototools for producing PCBs, using a moving optical head that exposed pads and drew tracks on the film. This system is why we still refer to "aperture tables," "flashes," and "draws" today. The photoplotters, known as vector plotters, followed the PCB pattern. The system was based on the RS-274-D format, developed by the US Electronic Industries Association (EIA), and initially operated using punched cards.

The advent of PCB Computer-Aided Design (CAD) systems in the early 1980s began to phase out hand-taped 2:1 artworks. These CAD systems could output data directly to a photoplotter, creating phototools. At the time, most photoplotters were Gerber devices. Gerber had published a complete specification of their format in 1980, so Gerber RS-274-D became the universal standard.

The format, however, had one major flaw: the size, shape, and number of apertures were restricted by the physical aperture wheel. This limitation made handling new surface-mount components with varying rectangular pad sizes challenging. The workaround involved "painting" pads with tiny draws. While this strategy worked for simple plane layers, it was ineffective for mixed plane layers or signal layers. These layers required filling with draws, resulting in large images that took hours to plot.

A new format and photoplotter were conceived as a solution. The raster photoplotter employed a light source, typically a laser, to raster-scan the film continuously. With this system, any shape could be plotted using raster pixels. This method is now the

industry standard for photo-imaging PCBs, with laser photoplotters capable of resolutions down to 50,000 dots per inch or more.

The advent of this technology allowed for the creation of a more flexible and designer-friendly Gerber format. The Extended Gerber or RS-274X, launched in 1991, allowed users to define and image any shape - a pad, track, or polygon (plane). The aperture definitions were no longer constrained by a physical wheel and could be generated automatically from the CAD job.

RS-274X is the prevailing PCB layer image data transfer format used today. It is transparent, unequivocal, and user-readable. Each file is self-contained and enables the drawing of any desired pad shape or copper area.

However, the old Standard Gerber RS-274-D persists despite its limitations, such as needing a separate aperture table, generating cumbersome files, and potential requirement for the merging of positive and negative images. Most PCB fabricators still accepts this older format for previous jobs but prefers Extended Gerber, RS-274X, due to its lack of these limitations. RS-274X, complete with its embedded aperture definition, is compatible with PCB Visualizer, facilitating all advantages of our advanced data checking technology. Most current and older CAD systems generate RS-274X output. If your CAD system is still producing the old-style Gerber RS-274-D, it may be possible to switch to RS-274X in the output settings, although different systems may use different terminology.

Today

Today, RS-274X serves as the standard data transfer format for PCB layer images. It stands out for its clarity, lack of ambiguity, and readability. Each RS-274X file is self-contained, enabling the depiction of any desired pad shape or copper area.

Despite its limitations, the older Standard Gerber RS-274-D persists. These limitations include a reliance on a separate aperture table that frequently goes missing, creating large and unwieldy files, and potentially necessitating the merging of positive and negative images. The latter can lead to extensive clean-up operations or, even worse, easily overlooked errors.

While Eurocircuits can still process the older RS-274-D format when necessary, for instance, for prior projects, it is incompatible with our PCB Visualizer. As such, we prefer the Extended Gerber, RS-274X. It is free from the constraints of RS-274-D and integrates the aperture definition within the file, making it compatible with the PCB Visualizer. This compatibility unlocks all advantages of our advanced data checking technology. Most present-generation and several older CAD systems generate RS-274X output. If your CAD system still outputs the outdated Gerber RS-274-D, consider reviewing your output settings. It may be feasible to switch from RS-274-D to RS-274X. However, different systems may use different terminologies. If you have any uncertainties, don't hesitate to contact us.

The Future

Extended Gerber, or RS-274X, offers a precise and clear depiction of a PCB's layers. However, it doesn't incorporate all the information necessary for fabrication, particularly for automated data preparation.

Examples of missing information could be:

The function of the layer in question, such as top copper, top solder mask, etc.

Whether the image represents a single PCB or a delivery panel.

The purpose of a particular object, such as whether it's an SMD pad, via pad, fiducial, etc.

The board's profile. Automated recognition software like PCB Visualizer can identify rectangular profiles but struggles with complex shapes.

Drill tolerances for a specific hole. For instance, it might be a press-fit hole.

The tracks requiring impedance control.

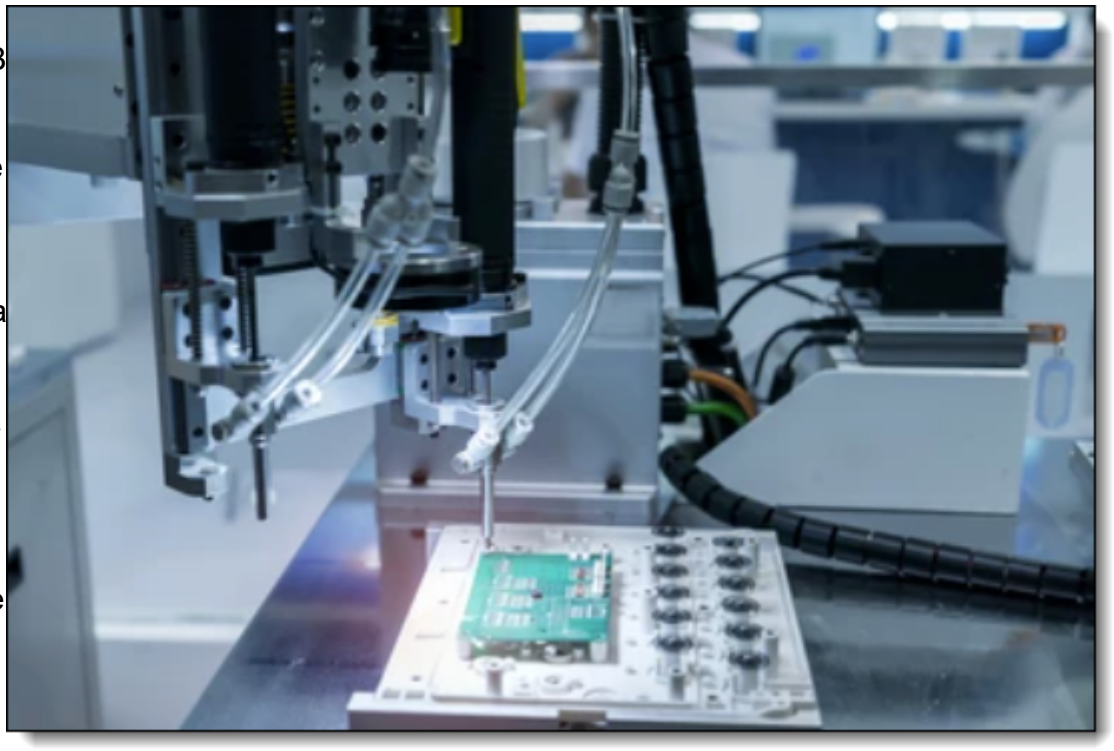
The vias that need filling.

The next progression is to integrate such information into the data transfer format. Any extension of the format must remain compatible with the existing format and current CAD systems. While alternative formats incorporating non-image information have been suggested, the Gerber format's widespread use and effectiveness mean it isn't easily replaceable, much like QWERTY, QWERTZ, and AZERTY keyboards.

Today, the Gerber format is managed and evolved by Ucamco, a Belgian company that acquired Gerber Scientific's PCB Division in 1997. Ucamco has recently unveiled the blueprint for the next version of RS-274X, known as Gerber RS-274X2. This iteration introduces attributes into the format that encapsulate the information mentioned above.

2.3.48 What is a printed circuit board drill file?

A printed circuit board (PCB) drill file is a type of file used in the manufacturing of PCBs. It contains information about the location and size of all the holes that need to be drilled into the board during the manufacturing process.



PCB Drill Robot

When a PCB design is created, the design software generates a drill file that contains the coordinates and sizes of all the holes required for the PCB. This includes holes for through-hole components, mounting holes, and vias that connect different layers of the board.

The drill file is used by the PCB manufacturer to guide the drilling process. The manufacturer uses a computer-controlled drilling machine to drill the holes into the board according to the specifications in the drill file.

The drill file typically includes information about the size of each hole, the position of the hole on the board, and the layer on which the hole is located. The file may also include information about the type of drill bit to be used for each hole, the drilling speed, and other parameters.

Overall, the drill file is an essential part of the PCB manufacturing process. It ensures that the holes are drilled accurately and precisely, and that the finished PCB meets the specifications of the original design.

Drill Formats

NC format is a set of instructions used by a Numerical Control machine in the manufacturing process. This is typically G-code or a similar language, which defines things like coordinates, tool paths, drilling operations, milling operations, etc.

For PCB manufacturing, different machine manufacturers may use slightly different NC formats, but they all tend to be variants of the G-code language. Common NC files for PCB production include drill files and milling files. These tell the machine where to drill holes, mill slots, etc.

Drill files (Excellon format)

The most common format for PCB drilling data is the Excellon format. It's an industry standard for delivering drilling (and routing) data on PCB fabrication data. Excellon files control CNC drilling machines and are ASCII text based.

Gerber Files

While not strictly an NC format, Gerber is the de-facto standard for PCB industry to define the copper layers, solder mask, legend, and drill and route data. Gerber format files are used to provide information to the machines creating the PCB about where copper needs to be placed or removed.

Milling Files

These are often G-code files that define the paths for milling operations, if those are required.

Remember that the specific formats used can depend on the PCB manufacturer's specific machinery and software. Always consult with your manufacturer to make sure your design files are in a format they can use.

2.3.49 What is ODB++

ODB++ is a proprietary, hierarchical data model and file format used in the printed circuit board (PCB) manufacturing industry. The acronym stands for Open DataBase and the "++" signifies that it's an advancement from the original ODB format.

The ODB++ format was created by Valor Computerized Systems, Ltd. as a comprehensive, unified data exchange format to replace older formats like Gerber, Excellon, and GENCAD. The goal was to convey more complex design information in a structured way to help eliminate errors and ambiguities during the manufacturing process.

Unlike more basic formats such as Gerber, which only contain graphical representations of the PCB layers, ODB++ captures a wide variety of data including:

Stackup

Details of the layering in the PCB, including copper, dielectric, and other types of layers.

Components: Information about all components in the design, including their values, part numbers, and package types.

Netlist

The list of electrical connections between components.

Drills and Routings

Information about all the holes to be drilled and paths to be routed on the PCB.

Placement

Information about where components should be placed, which can be used by pick-and-place machines.

Bill of Materials (BOM)

List of all the parts and materials required to manufacture the PCB.

Because it contains such a wide variety of information, ODB++ can be used to drive a wide range of manufacturing processes, including fabrication, assembly, and test. It's supported by a wide range of Electronic Design Automation (EDA) tools, as well as by many PCB manufacturers. The latest version of the standard is ODB++X, which is a more advanced version based on the XML format.

2.3.50 What is IPC-2581

IPC-2581 is a generic standard for printed circuit board assembly products' manufacturing description data and transfer methodology. It was developed by the IPC association, which is an international trade association for the electronics industry.

This standard was created to address the modern complexities of PCB design and manufacturing processes, which were not sufficiently addressed by older formats like Gerber and Excellon. It's a single, comprehensive file format that encapsulates all necessary data to manufacture and inspect a PCB assembly.

The IPC-2581 standard aims to improve efficiency, reduce data re-entry and transcription errors, and enhance traceability. It's a robust format that includes detailed information about:

PCB Stackup and Material Details

Information about the layers in the PCB, including copper, dielectric, and other types of layers.

Netlist

A list of the electrical connections between the different components on the PCB.

Components Information

Data about all the components to be assembled onto the PCB, including values, part numbers, package types, and positions.

Drill and Route Information

Data about all the holes to be drilled and the paths to be routed on the PCB.

Test Points and Probe Information

Details about the test points used for testing the PCB after assembly.

Manufacturing Notes and Assembly Instructions

Any additional notes or instructions required for manufacturing or assembling the PCB.

IPC-2581 files are XML-based, making them easy to generate and read with a variety of software tools. The IPC-2581 standard is gaining increasing acceptance in the industry, and more and more EDA tools and PCB manufacturers are starting to support it.

2.3.51 Automated Pick and Place

Automated Pick and Place is a critical process in the assembly of printed circuit boards (PCBs). The Pick and Place machine is designed to accurately place electronic components on the PCB, quickly and precisely, enabling high-speed assembly of complex electronics.

A Pick and Place Machine

Part Placement Using Pick and Place

Pick and Place CSV Files

How do Pick and Place Machines Work

How does Pick and Place Machine Vision Work

Pick and Place Machine File Format Standards

2.3.51.1 Part Placement Using Pick and Place

A Pick and Place machine (PnP machine) is a robotic device used primarily in the manufacturing of printed circuit boards (PCBs) and other electronic devices. The main function of this machine is to place surface-mount components (SMCs) onto a PCB in a rapid, precise, and automated manner.

Here is how it generally works:

Setup

First, reels of components are loaded into the machine. Each reel contains a specific component that will be placed onto the PCB. The machine also needs to be programmed with the details of the PCB design, including where each component needs to be placed, and in what orientation.

The machine operates by picking up components from feeders or trays using a vacuum or mechanical gripper and then accurately placing them on the specified locations on the PCB.

Here's a more detailed breakdown of how a Pick and Place machine operates:

Component Loading

The machine is loaded with reels or trays of components that will be placed onto the PCB. Each reel is filled with one type of component.

PCB Loading

The bare PCB, often coated with a layer of solder paste, is loaded into the machine.

Component Picking

The machine uses a robotic arm with a vacuum or mechanical nozzle to pick up an individual component from the reels.

Component Inspection

Many PnP machines have optical inspection capabilities, using cameras to ensure the correct component has been picked and is properly oriented. If a component is incorrect or misaligned, the machine can adjust or replace it as necessary.

Component Placement

The machine then moves the component to the correct location on the PCB and places it onto the solder paste on the board.

Soldering

After all components have been placed, the PCB is usually sent through a reflow oven, where the solder paste melts and forms a mechanical and electrical connection between the components and the board.

Pick and Place machines can accurately place hundreds or even thousands of components per hour, making them a key part of any high-volume electronics production line. This process, which would be time-consuming, tedious, and error-prone if done by hand, can be done quickly and accurately using a Pick and Place machine.

2.3.51.2 Pick and Place CSV Files

Pick and place machines for PCB (Printed Circuit Board) assembly use pick and place files to know where each component goes on the board. These files are often in CSV (Comma Separated Values) format, which is a widely accepted format that can be read by many machines and software.

The information in a typical pick and place CSV file includes:

Designator

This column contains the reference designator for each part, such as "R1", "C1", etc. This tells the machine what the name of the part is on the board.

Mid X and Mid Y

These are the coordinates that indicate where on the board the part should go. The position can be in units of millimeters or inches depending on the machine used.

Layer

This indicates whether the component goes on the top or bottom side of the board.

Rotation

This specifies how much the part should be rotated before it is placed on the board.

Value

This field is typically the value or name of the component, such as "10K resistor" or "0.1uF capacitor".

Footprint

This field might contain information about the component's physical size or the type of footprint it has.

Comment

This is typically an optional field for additional information that might be helpful to the machine operator.

The exact contents of a pick and place file can vary depending on the machine and the software used to generate the file. For example, some machines may require additional fields such as feeder number or package type.

A sample CSV pick and place file might look something like this:

Designator,Mid X,Mid Y,Layer,Rotation,Value,Footprint,Comment

R1,10,20,top,90,10K,0805,

C1,30,40,top,0,0.1uF,0603,

U1,50,60,top,270,OPA134,SOIC-8,

In this file, the resistor R1 is located at X=10, Y=20, on the top side of the board, and should be rotated 90 degrees before placement. The value of the resistor is 10K and it has an 0805 footprint. The other lines follow a similar format for the other components.

2.3.51.3 How do Pick and Place Machines Work

Pick and Place machines are used in the assembly of printed circuit boards (PCBs) and are designed to rapidly place components on a PCB with high precision. The overall operation can be broken down into several steps:

Programming

Before starting the operation, the machine needs to be programmed with the data from a pick and place file. This file provides information on component locations, orientations, and types. The file format can be CSV (Comma Separated Values), TXT (Text), or other formats, depending on the machine manufacturer. The file often includes the X-Y coordinates of each part, its identifier, rotation, and whether it's placed on the top or bottom side of the board.

Feeding

The electronic components to be placed on the PCB are loaded into the machine using a variety of feeders. Common types of feeders include tape feeders, stick or tube feeders, and tray feeders, each designed to handle components packaged in different ways. The machine is able to index these feeders to retrieve components when needed.

Picking

The machine uses a motorized head equipped with a vacuum nozzle to pick up individual components from the feeders. The suction allows the machine to pick up the component securely, even if it's very small. High-end machines may have multiple heads that can pick and place multiple components at once.

Vision System

After a component is picked, the machine uses a camera and vision system to check the component. The vision system can correct for any variations in the component's position on the nozzle, confirm the component is the correct type, and inspect the component for any visible defects. It can also make sure the rotation of the component is correct.

Placing

The machine moves the component to the correct position over the PCB, using the data from the pick and place file. The component is then placed onto the board, usually onto a layer of solder paste. The precision of this operation is very high, allowing components to be placed accurately even on high-density boards.

Reflow Soldering

After all the components have been placed, the PCB goes through a reflow oven, which melts the solder paste, creating a secure electrical and mechanical connection between the components and the board.

Pick and place machines play a critical role in modern electronics manufacturing, allowing for high-speed, high-accuracy assembly of PCBs. Without them, the assembly of modern electronics would be much slower and less reliable.

2.3.51.4 How does Pick and Place Machine Vision Work

The machine vision system of a pick and place machine plays a crucial role in ensuring accurate placement of components onto a PCB. Here's an overview of how it works:

Image Capture

The vision system starts with an image capture device, usually a high-resolution camera. The camera captures images of the component after it's been picked up by the machine's head. In some sophisticated systems, there might be multiple cameras to capture different angles or both sides of the component and the PCB.

Lighting

Adequate lighting is essential for capturing a good image. Different types and orientations of lighting can help to highlight certain features or contrasts in the component. The lighting system can include direct lighting, back lighting, or diffused lighting, depending on the specific requirements.

Image Processing

Once the image is captured, it's processed using a range of techniques. The goal is to extract key information from the image, such as the component's position, orientation, and condition. Digital image processing techniques such as edge

detection, blob analysis, and pattern matching are often used. The system can also compare the captured image with a stored image to verify that the correct component has been picked.

Fiducial Recognition

Fiducials are markers that are added to the PCB and used as reference points. By identifying these fiducials, the vision system can correct for any slight misalignment of the PCB and ensure that the components are placed in the correct position. The fiducials are usually small solid circles, crosses or squares that can be easily identified by the vision system.

Feedback

The results of the image processing can then be used to adjust the placement of the component. For example, if the vision system determines that the component is slightly rotated, it can instruct the machine to adjust the rotation before placing the component. The feedback loop is fast enough to correct any errors in real-time.

Quality Control

In addition to assisting with component placement, the vision system can also be used for quality control. For example, it can check that the components have been placed correctly and are properly aligned. Any discrepancies can be logged for further inspection.

Overall, the machine vision system is an integral part of modern pick and place machines, providing accuracy, speed, and quality control. It allows the machine to quickly adapt to the different types of components and PCB designs.

2.3.51.5 Pick and Place Machine File Format Standards

When it comes to the file formats used by pick and place machines, there isn't a universal standard across all manufacturers. The file format used often depends on the specific machine or software used to generate the file. However, there are some widely used formats that are generally accepted across many machines.

CSV (Comma Separated Values)

This is a simple, widely used format that can be read by many machines and software. The file typically includes data such as the component designator, X and Y coordinates, rotation, layer (top or bottom), and other details. The exact format can vary, so it may need to be customized to match the requirements of the specific machine.

TXT (Text)

Similar to CSV, but the data fields are usually separated by spaces or tabs instead of commas. Again, the format can vary depending on the machine.

Excel (XLS or XLSX)

Some machines can read data directly from Excel files. However, this is less common due to the complexities of the Excel file format.

EIA-481-D (Electronic Industries Alliance Standard)

This standard defines a format for component feeder data, which can be used by pick and place machines to set up the component feeders. It doesn't contain the placement data, but is often used in conjunction with another file that does.

EIA-481-D is a standard published by the Electronic Industries Alliance (EIA), which is now part of the Electronic Components Industry Association (ECIA). The "D" in the name represents the latest revision of the standard.

EIA-481-D specifies the standard for the embossed carrier taping of surface-mount components for automatic handling. This includes the tape that holds the components, the reel that the tape is wound onto, and the leader and trailer tape sections. The standard defines the physical properties of these tapes, including dimensions, materials, and mechanical characteristics.

Surface mount components are typically supplied on these tapes and are fed into automated pick-and-place machines for assembly onto printed circuit boards (PCBs). By adhering to this standard, component manufacturers ensure their products can be easily used by a variety of automated machinery in electronics manufacturing processes around the world.

Please note, however, that EIA-481-D doesn't directly relate to the data files used by pick and place machines to place components on a PCB. Those files, which may be in CSV, TXT, or other formats, contain information about the position, orientation, and type of each component on the board.

Gerber (RS-274X)

While not strictly a pick and place file format, Gerber files are often used in conjunction with pick and place data. The Gerber files provide a graphical representation of the PCB layout, including the copper layers, solder mask, silkscreen, and drill holes. This data can be useful for setting up the machine and verifying the placement data.

CAD Files (e.g., ODB++, IPC-2581)

Some machines can work directly with data from CAD (Computer Aided Design) systems. These file formats include much more information than a simple pick and place file, including the complete PCB layout and sometimes even the 3D models of the components. However, working with these formats usually requires more complex software and hardware.

In practice, you would typically generate the pick and place file from the same CAD software that you used to design the PCB. This ensures that the data matches

exactly with your design. You may need to customize the output format to match the requirements of your specific pick and place machine.

2.3.52 How are printed circuit boards made?

Printed circuit boards (PCBs) are made through a process that involves several steps, including design, fabrication, and assembly. Here is a general overview of the steps involved in making a PCB:

Design

The first step in making a PCB is to create a design using PCB design software. The software allows the user to create a schematic diagram of the circuit, and then convert it into a PCB layout.

Printing

Once the PCB layout is complete, it is printed onto a special film using a printer that produces a high-resolution image. The image is then transferred onto the surface of a copper-clad board, which is typically made of fiberglass or a similar material.

Etching

The next step is to remove the excess copper from the board, leaving only the copper traces that make up the circuit. This is done using a chemical etching process, which involves applying a special etchant solution to the board. The etchant dissolves the exposed copper, leaving behind only the desired copper traces.

Drilling

After the copper traces have been etched, the next step is to drill holes in the board where components will be mounted. These holes are typically drilled using a computer-controlled drill press.

Plating

Once the holes have been drilled, the board is plated with a thin layer of metal, usually copper, to create a surface for the components to be soldered to. This is typically done using an electroplating process.

Soldering

The final step is to mount and solder the components onto the board. This is typically done using a combination of automated and manual processes, depending on the complexity of the board and the number of components.

Overall, the process of making a printed circuit board is a complex and precise one that involves several steps, each of which is critical to the final outcome. The end

result is a custom-made PCB that can be used in a wide range of electronic devices and applications.

2.3.52.1 Solder

PCB solder is a type of fusible alloy used to create a strong, conductive bond between component leads and the traces on a printed circuit board (PCB). The process of using the solder to connect these elements is known as soldering.

The composition of solder varies, but it typically includes elements such as tin, lead, silver, and copper. The most common type used in electronics was traditionally a 60/40 (tin/lead) blend, but due to health and environmental concerns related to lead, lead-free solders (usually containing tin, silver, and copper) have become more prevalent.

Solder comes in several forms for different applications:

Wire Solder

This is the most common form, used in hand-soldering applications. It often has a core of flux, which helps clean the components and improve the flow of the solder when heated.

Real of Wire Solder

Solder Paste

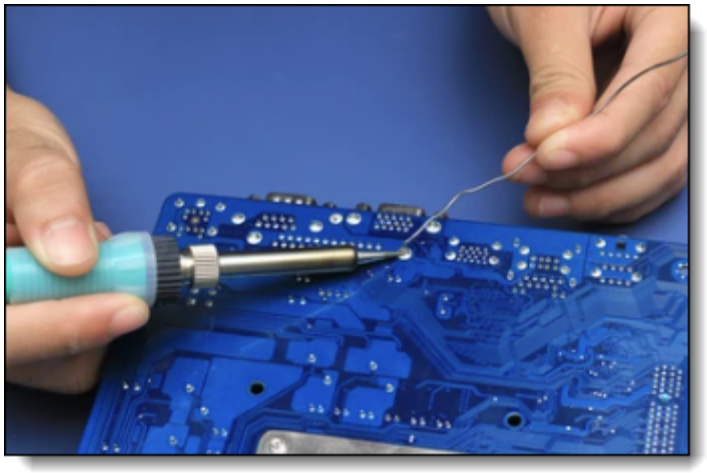
This is a mixture of small solder particles and flux used in reflow soldering, a common technique in surface-mount technology (SMT). The paste is applied to the PCB, the components are placed on top, and the whole assembly is heated in a reflow oven to melt the solder and create the electrical connections.

Solder Bar

Used in wave soldering machines, which are typically used for mass production of PCBs. The bar is melted and a "wave" of solder is used to solder all components to the PCB simultaneously.

It's important to note that solder doesn't just glue the components onto the board—it creates an electrical connection between the component and the circuit traces on the PCB. This is essential for the functionality of any electronic device.

2.3.52.2 Hand Soldering



Hand Soldering

Soldering a Printed Circuit Board (PCB) by hand is a valuable skill for electronics hobbyists, and while it's not overly complicated, it does require some care and attention to detail to do it correctly. Here is a basic step-by-step guide:

Materials:

Soldering Iron - A tool with a heated metal tip used to melt solder.

Solder - A fusible metal alloy used to create a permanent bond between metal workpieces.

Soldering Stand - To hold the soldering iron when not in use.

Soldering Sponge - To clean the soldering iron tip.

PCB - The board onto which components will be soldered.

Electronic Components - The parts to be soldered onto the PCB.

Desoldering Pump or Wick - For correcting mistakes.

Safety Glasses - Protects your eyes from any splashes of solder or cut component leads.

Tweezers - To hold tiny components.

Wire Cutters - To trim the leads of components after soldering.

Steps:

Prepare Your Workspace: Make sure your work area is well-ventilated and free of flammable materials. Place your soldering iron in its stand and plug it in. Ensure the tip is clean and tinned (coated with a thin layer of solder).

Insert the Components: Push the leads of your components through the appropriate holes in the PCB. Resistors, diodes, and some capacitors are not polarized, meaning they can be inserted either way. Other components, like electrolytic capacitors and ICs, are polarized and must be inserted with the correct orientation.

Secure the Components: Bend the leads on the other side of the PCB to keep the components in place. Make sure the components sit flat against the PCB.

Apply Heat: Touch the soldering iron tip to the component lead and the copper pad at the same time. Wait a second or two for them to heat up.

Apply Solder: Touch the end of your solder to the joint (but not directly to the iron tip). The solder should melt and flow freely around the lead and onto the pad. Remove the solder, then remove the iron. Allow the joint to cool naturally, without blowing on it.

Inspect the Joint: A good solder joint will look shiny and have a concave shape. A bad joint could look like a ball of solder, be dull or grainy, or not fully cover the pad.

Trim the Leads: Once the joint is cool and you've inspected it, use your wire cutters to trim off the excess lead close to the solder joint.

Repeat: Repeat these steps for all of your components.

Clean the PCB: After you've soldered all your components and the board is cool, you can clean off any flux residue with isopropyl alcohol and a brush.

Check Your Work: Use a multimeter to check for shorts and correct component values.

Safety Tips:

Always wear safety glasses when soldering.

Never touch the tip of the soldering iron.

Wash your hands after soldering to remove any lead residue.

Don't inhale the smoke that's produced when soldering. Use a fume extractor if possible.

Turn off and unplug the soldering iron when you're finished with it.

Remember that practice makes perfect. Soldering can be tricky to get right the first time, but with patience and experience, you'll get better at it.

2.3.52.3 Reflow Soldering

"Reflow soldering" is a commonly used method for attaching surface mount components to PCBs and is preferred for manufacturing high volume products or products with many small components.

PCB at the output of air convection reflow oven

The term "reflow" comes from the process where the solder is first pasted onto the board in a paste form and then later "reflows" into liquid form during the heating process to make the connection.

Reflow soldering is a process used in the manufacturing of printed circuit boards (PCBs). During this process, solder paste (a mix of solder particles and flux) is applied to the places where components will be mounted on the PCB. The components are then placed on the board, and the entire assembly is heated in a reflow oven.

The goal of reflow soldering is to melt the solder paste, allowing it to flow and create electrical and mechanical connections between the components and the board. The process usually involves several stages or zones:

Preheat Zone

The board and components are gradually heated to a specific temperature range, typically between 150°C to 200°C. This step is done gradually to prevent thermal shock to the components and the board.

Preheating is the initial stage in the reflow soldering process. During this phase, the temperature of the entire board assembly gradually increases towards a designated soak or pre-reflow temperature. The primary objective of the preheating phase is to safely and uniformly elevate the temperature of the assembly to the pre-reflow stage. Additionally, preheating allows the volatile solvents present in the solder paste to be released or "outgas."

To effectively allow solvent expulsion and safely reach pre-reflow temperatures, the PCB should be heated consistently at a linear rate. This rate is often referred to as the temperature slope rate and is commonly measured in degrees Celsius per second (°C/s). The ideal slope rate depends on several variables, including the target processing time, the volatility of the solder paste, and component considerations.

Component considerations often take precedence because many components could crack if the temperature changes too abruptly. The maximum rate of temperature change that the most temperature-sensitive components can tolerate becomes the slope rate's upper limit. However, if the assembly doesn't include thermally sensitive components and the main goal is to maximize throughput, more aggressive slope rates can be applied to expedite processing time. As such, many manufacturers increase the slope rate up to the commonly acceptable maximum of 3.0 °C/s.

On the other hand, if a solder paste with highly volatile solvents is used, excessively fast heating could destabilize the process. When volatile solvents outgas too rapidly, they might cause solder to splatter off the pads and onto the board, leading to solder-balling. This is a primary concern during the preheat phase.

Once the board has been uniformly heated to the target temperature during preheating, the process transitions to the soak or pre-reflow phase.

Thermal Soak Zone

The assembly is kept at a stable temperature to activate the flux in the solder paste, which cleans the component leads and PCB pads to ensure good soldering. This phase also helps to minimize temperature differences across the board, avoiding potential issues related to thermal expansion.

The thermal soak phase is the second stage in the reflow soldering process. This phase typically involves exposing the assembly to heat for 60 to 120 seconds. The heat serves two primary purposes: to dissipate volatile components in the solder paste and to activate the fluxes. Once activated, the fluxes initiate the reduction of oxidation on the leads of the components and the pads.

Maintaining an optimal temperature during this phase is crucial. If the temperature is excessively high, it can cause solder spattering or balling and induce oxidation in the

paste, the component terminations, and the attachment pads. Conversely, if the temperature is too low, the fluxes may not fully activate.

The objective at the end of the thermal soak zone is to achieve thermal equilibrium across the entire assembly, preparing it for the subsequent reflow zone. Utilizing a soak profile is advised to reduce the temperature difference, or delta T, between components of varying sizes or in cases where the PCB assembly is particularly large. It is also recommended to use a soak profile to minimize voiding in area array type packages.

Reflow Zone

The temperature is increased to the point where the solder paste melts or "reflows." This is typically in the range of 220°C to 245°C, depending on the type of solder paste used. The solder forms a liquidus bond between the component leads and the PCB pads.

The third segment, known as the reflow zone, is also called the "time above reflow" or "temperature above liquidus" (TAL). This is where the process reaches its highest temperature. A key factor to bear in mind is the peak temperature, which is the absolute highest temperature permissible in the whole process. A common peak temperature is 20–40 °C above the liquidus. This maximum limit is set by the component in the assembly with the least resistance to high temperatures, that is, the component most prone to thermal damage. A common practice is to reduce 5 °C from the highest temperature that the most vulnerable component can endure, establishing the process's maximum temperature. Constantly monitoring the process temperature to prevent exceeding this limit is crucial. Moreover, exceedingly high temperatures (beyond 260 °C) can cause damage to the internal dies of SMT components and promote intermetallic growth. Conversely, a temperature that's not sufficiently high may hinder the adequate reflow of the paste.

The measure of how long the solder remains a liquid is the Time above Liquidus (TAL), or time above reflow. The flux reduces surface tension at the juncture of metals, enabling metallurgical bonding and allowing the individual solder powder spheres to combine. If the profile time surpasses the manufacturer's specification, it may lead to premature flux activation or depletion, which can essentially "dry out" the paste before the solder joint is formed. A deficient time/temperature relationship can reduce the flux's cleaning action, resulting in poor wetting, insufficient solvent and flux removal, and potentially defective solder joints.

Experts generally advise the shortest TAL possible, but most pastes suggest a minimum TAL of 30 seconds, though the rationale behind this specific time isn't apparent. One theory is that there might be areas on the PCB that aren't measured during profiling, and hence, setting the minimum allowable time to 30 seconds lowers the risk of these unmeasured areas not reflowing. A high minimum reflow time also offers a safety margin against changes in oven temperature. Ideally, the wetting time should stay under 60 seconds above liquidus. Excessive time above liquidus may lead to too much intermetallic growth, resulting in joint brittleness. The board and components might also get damaged with prolonged temperature over

liquidus, and most components have a well-defined time limit for exposure to temperatures over a certain maximum. Insufficient time above liquidus may trap solvents and flux, creating the potential for cold or dull joints and solder voids.

Cooling Zone

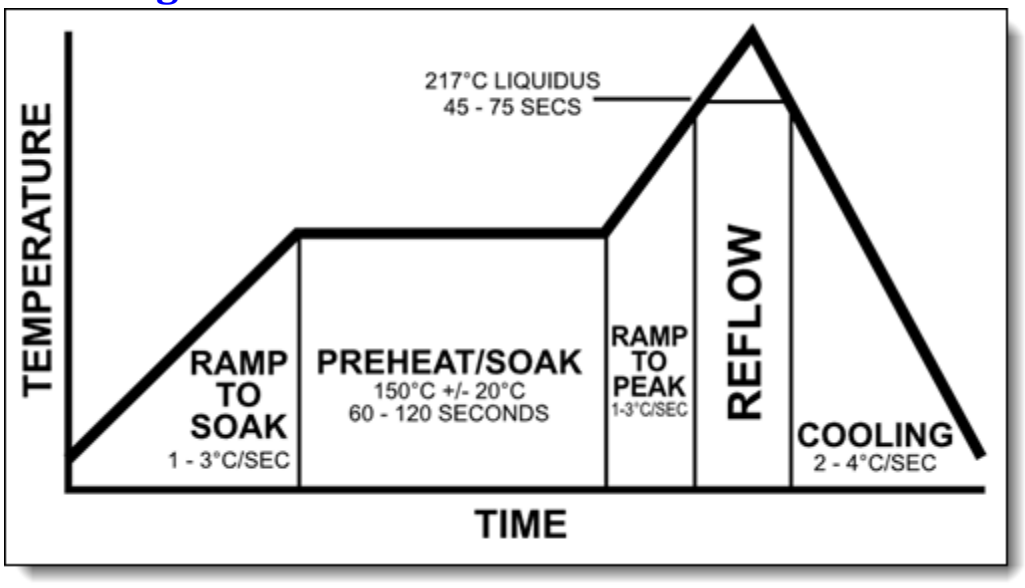
The assembly is cooled down to solidify the solder joints. The cooling should be controlled to avoid thermal shock and to create a fine grain structure in the solder for the best mechanical strength.

The final zone in the process is a cooling zone, which gradually cools down the treated board to solidify the solder joints. Ensuring proper cooling is essential to prevent excessive intermetallic creation or thermal shock to the components. The typical temperatures within the cooling zone usually fluctuate between 30–110 °C (86–212 °F). Selecting a quick cooling rate is ideal as it leads to the formation of a fine grain structure, which provides the best mechanical integrity.

While often overlooked, the ramp-down rate is just as important as the maximum ramp-up rate. Above certain temperatures, the ramp rate may not seem crucial, but the maximum permissible slope for any component should be adhered to regardless of whether the component is heating or cooling. A frequently recommended cooling rate is 4 °C/s. This factor should be taken into consideration when evaluating the outcomes of the process.

Thermal Profiling

This is the method of precisely charting or controlling the temperature of the PCB throughout the reflow soldering process. It is critical to ensure that all solder joints on the board reach the appropriate temperatures without overheating any components.



Example of reflow soldering thermal profile

Thermal profiling involves recording temperature readings at various locations on a circuit board to track its thermal journey during the soldering process. In the realm of electronics manufacturing, Statistical Process Control (SPC) is used to assess if the

process is being managed effectively, by comparing it against the reflow parameters set by soldering technologies and component specifications.

With advancements in technology, contemporary software tools have made it possible to capture a thermal profile and then automatically optimize it using mathematical simulations. This significantly decreases the time required to define the most effective settings for the process.

Reflow Ovens

A reflow oven is a device primarily utilized for reflow soldering, which involves attaching surface mount electronic components to printed circuit boards (PCBs).

In large-scale commercial applications, reflow ovens are typically long tunnel structures with a conveyor belt that transports PCBs. For prototype development or hobbyist use, PCBs can be inserted into a compact oven with a door.

These commercial conveyor-based reflow ovens have several independently heated zones, each with its own temperature control. As PCBs travel through the oven and its various zones, the speed of the conveyor and temperature of the zones are adjusted to achieve a specific time and temperature profile. The profile used can vary based on the requirements of the PCBs being processed.

Types of Reflow Ovens

Infrared and Convection Ovens

In infrared reflow ovens, ceramic infrared heaters located above and below the conveyor serve as the heat source, radiating heat to the PCBs.

Convection ovens, on the other hand, heat air within chambers and then use this air to transfer heat to the PCBs via convection and conduction. They may have fans to manage the airflow inside the oven. This indirect heating method provides more precise temperature control than direct infrared radiation, as the absorption rate of infrared radiation varies among PCBs and components.

A convection industrial reflow oven

There are ovens that use a combination of both infrared radiative heating and convection heating, and these are referred to as 'infrared convection' ovens.

Some ovens are designed to reflow PCBs in an oxygen-free environment, often using Nitrogen (N2) to achieve this. This approach minimizes the oxidation of solder surfaces. Nitrogen reflow ovens take a few minutes to decrease oxygen concentration within the chamber to acceptable levels, thereby reducing defect rates.

Vapour Phase Oven

In vapour phase ovens, PCBs are heated via thermal energy released during the phase transition (e.g., condensation) of a heat transfer liquid on the PCBs. The liquid selected has a predetermined boiling point to suit the solder alloy being reflowed.

Vapour phase soldering comes with several benefits:

It offers high energy efficiency due to the high heat transfer coefficient of vapour phase media.

The soldering process is oxygen-free, eliminating the need for protective gases like nitrogen.

Overheating of assemblies is avoided. The maximum temperature that assemblies can attain is capped by the boiling point of the medium.

This method is also known as condensation soldering.

2.3.52.4 Wave Soldering

Wave soldering is a popular method for mass soldering in the manufacturing process of printed circuit boards (PCBs). The PCB is moved over a tank of molten solder, and a pump in the tank produces an upward surge of solder resembling a standing wave. When the PCB makes contact with the solder wave, the components get soldered onto the board. This method is used for both through-hole and surface mount assemblies. In the latter case, the components are initially glued onto the PCB's surface by placement machinery, before being exposed to the solder wave.

Though wave soldering has largely been replaced by reflow soldering techniques with the rise of surface mount components, it is still heavily used where surface-mount technology (SMT) isn't suitable, such as for large power devices and high pin count connectors, or in scenarios where simple through-hole technology is predominant, like in some major appliances.

The wave soldering process involves a variety of equipment, including a conveyor to move the PCB through different zones, a solder pan, a pump to create the solder wave, a flux sprayer, and a preheating pad. The solder is typically a mix of metals, with a common leaded solder being composed of 50% tin, 49.5% lead, and 0.5% antimony. However, regulations like the Restriction of Hazardous Substances Directive (RoHS) have spurred a shift towards lead-free alternatives, such as tin-silver-copper and tin-copper-nickel alloys.

The process also includes fluxing, where the goal is to clean the components to be soldered, particularly any oxide layers that may have formed. Flux can be either corrosive or noncorrosive, each with its own benefits and drawbacks. Preheating is essential to speed up the soldering process and to avoid thermal shock to the components. The quality of the soldering is dependent on the correct heating temperatures and properly treated surfaces.

Flux residues post-soldering might need to be cleaned off using solvents or deionized water. Some "no-clean" fluxes leave behind harmless residues, eliminating the need for the cleaning stage.

Different solder alloys, including combinations of tin, lead, and other metals, are used based on the required properties. A popular choice is a eutectic alloy of 63% tin and 37% lead, known for its strength, low melting range, and quick setting. Other compositions are selected for their low melting points and are helpful for soldering heat-sensitive components.

The cooling rate of the PCBs post-soldering is critical. Rapid cooling can lead to warping of the PCB and compromised solder joints, while slow cooling can make the PCB brittle and risk heat damage to some components. Hence, appropriate cooling methods, like a fine water spray or air cooling, are used.

Thermal profiling is an important step, where multiple points on a PCB are measured to determine its thermal journey through the soldering process. Tools like the Solderstar WaveShuttle and Optiminer can help production engineers to establish and control the wave solder process.

Lastly, the height of the solder wave is a crucial parameter. It is usually controlled by adjusting the pump speed on the machine. The contact time between the solder wave and the assembly being soldered is typically set between 2 and 4 seconds, and this is regulated by the conveyor speed and wave height. To get more detailed records, there are fixtures available which digitally record contact times, height, and speed. Furthermore, certain wave solder machines provide operators with an option between a smooth laminar wave or a slightly higher-pressure "dancer" wave.

2.3.52.5 Electroless Nickel Immersion Gold

Electroless Nickel Immersion Gold (**ENIG**) is a type of surface finish used on printed circuit boards (PCBs). It's a two-layer metallic coating where the first layer is electroless nickel and the second is immersion gold.

The purpose of this dual layer is to protect the copper surface of the PCB, while providing a solderable surface for component assembly. Here's a more detailed breakdown of each layer:

Electroless Nickel

The nickel layer acts as a barrier to protect the copper from the solder, which could otherwise migrate into the copper and weaken the adhesion to the board's surface. Electroless nickel is used because it can be deposited evenly over the entire board

without the need for an electric current, making it ideal for complex boards with a lot of intricate detail.

Immersion Gold

The gold layer protects the nickel during storage and provides a surface that is resistant to oxidation, ensuring good solderability. It's deposited by a process known as immersion plating, where the PCB is dipped into a bath of gold salts. This gold layer is typically very thin, as it's only intended to protect the nickel layer beneath it and doesn't need to withstand soldering temperatures.

Advantages

ENIG has several advantages, including excellent surface planarity (making it ideal for PCBs with large BGA packages), good oxidation resistance, and a long shelf-life. However, it's also more expensive than other finishes like HASL (Hot Air Solder Leveling), and can sometimes suffer from what is known as "black pad syndrome," a defect where the gold and nickel layers separate, leading to poor solder joint integrity.

2.3.52.6 The PCB Build-Up

The term "PCB build-up" refers to the process and technique of adding layers to create a multilayer PCB (Printed Circuit Board). This term is often associated with high-density interconnect (HDI) PCBs, which are compact boards with a higher circuitry density than traditional PCBs.

In the context of PCB design and manufacturing, a build-up refers to the layer structure of a PCB. A simple two-layer PCB has a "1-2-1" build-up, which means there's one layer of insulating material (substrate), two layers of conducting material (usually copper), and another layer of insulating material.

For multilayer PCBs, the build-up gets more complex. For example, a four-layer PCB might have a "1-2-2-1" build-up. The numbers represent the sequence of conductive and insulating layers, starting from the top of the board and going to the bottom.

The complexity of PCB build-ups has increased with advancements in technology. Many modern electronics, such as smartphones and high-speed computers, use HDI PCBs that feature multiple layers and advanced build-up techniques.

Here are a few key terms related to PCB build-ups:

Core

This is the base material, usually an FR4 substrate with copper layers on both sides.

In the context of printed circuit boards (PCBs), the term "core" refers to the base material used in the construction of multilayer PCBs. A core is essentially a double-sided PCB (copper on both sides) with a substrate layer sandwiched between. The substrate layer is typically made from FR-4, a type of fiberglass-reinforced epoxy

laminate which is excellent for providing mechanical strength and is a good electrical insulator.

The copper on either side of the core serves as the conductive paths (or traces) for electrical signals. This copper is usually applied to the substrate in a very thin layer, which can be etched away in a pattern to form these traces.

In a multilayer PCB, several cores can be stacked together with additional layers of copper and insulating material (known as "prepreg") to achieve the desired number of conductive layers. Each core within the multilayer PCB is separated by a layer of this prepreg.

The cores in a PCB are usually specified by their thickness and the weight of the copper cladding. For example, a "1.6mm, 1oz copper" core has a substrate thickness of 1.6mm and a copper layer weight of 1oz per square foot. The thickness of the core can impact the rigidity and strength of the final PCB, while the weight of the copper affects the electrical characteristics of the board.

Prepreg

Prepreg, short for "pre-impregnated," is a type of material used in the manufacturing of multilayer printed circuit boards (PCBs). It's used as an insulating layer between copper layers and also to bind the layers together. It consists of fiberglass impregnated with resin that is partially cured, making it tacky and malleable under normal conditions.

In the context of PCB manufacturing, prepreg is used as an insulating layer between the conductive layers (or cores) of the PCB. When the PCB is subjected to heat and pressure during the lamination process, the resin in the prepreg flows, sticks to the cores, and then fully hardens, effectively gluing the different layers of the PCB together. The fully cured prepreg also serves as an excellent electrical insulator between the conductive layers of the PCB.

The thickness of the prepreg and the amount of resin it contains can be adjusted to control the final thickness and dielectric properties of the PCB. Several sheets of prepreg might be stacked together to achieve the desired thickness.

Like cores, prepregs are often specified by their thickness and resin content. For example, a common prepreg specification might be "7628," which refers to a type of heavy-weight fiberglass cloth. Other common prepreg types include "1080" (light-weight fiberglass cloth) and "2116" (medium-weight fiberglass cloth).

It's important to note that the handling of prepreg requires careful control over storage conditions (it must be stored in cool conditions to prevent premature curing) and manufacturing processes to ensure a reliable and high-quality PCB.

Sequential Build-up (SBU)

Understanding the build-up of a PCB is crucial for the PCB design process, as it impacts the electrical performance, signal integrity, mechanical strength, and manufacturing cost of the final product.

Sequential Build-Up (SBU) is a method used in the manufacturing of high-density interconnect (HDI) printed circuit boards (PCBs). As the name suggests, in the sequential build-up process, layers are added sequentially, one on top of another, to build a multilayer PCB.

Here's a basic overview of the SBU process:

Start with the Core: The process starts with a conventional double-sided or multilayer PCB as the core.

Add Insulating Layer: An insulating layer (typically prepreg) is laminated onto the core.

Add Copper Layer: A thin layer of copper is added on top of the insulating layer.

Pattern the Copper: The copper layer is then patterned using photolithography to create the necessary circuit traces.

Drill and Plate Vias: Microvias are then drilled (usually with a laser) through the insulating layer to the copper layer beneath. The vias are then plated to provide an electrical connection between layers.

Repeat as Necessary: Steps 2-5 are repeated for each additional layer. In most HDI boards, several layers of copper and insulation are added.

Final Lamination: Once all layers have been added, the board goes through a final lamination process to ensure all layers are properly bonded together.

The advantage of the SBU process is that it allows for the creation of very high-density PCBs, with multiple layers of fine circuit traces and small vias. This makes it possible to create smaller, lighter, and more complex PCBs, which is why SBU is commonly used in devices like smartphones and high-speed computers.

However, SBU is more complex and costly than traditional PCB manufacturing processes. It requires specialized equipment (like laser drilling machines) and careful process control to achieve reliable results. It's also important to work with a PCB manufacturer that has experience with SBU to ensure the best results.

2.3.52.7 Acceptability of Printed Boards : IPC-A-600

Acceptability of Printed BoardsThe IPC-A-600J is the latest revision of the IPC-A-600 standard, titled "Acceptability of Printed Boards". The IPC (Institute of Printed Circuits) is a global trade association that sets standards for the electronic interconnect industry.
The IPC-A-600 standard is widely recognized and used in the electronics industry. It defines three classes of acceptance criteria for visually inspecting rigid printed circuit boards (PCBs) to ensure their manufacturing quality. The three classes defined in this standard are:

Class 1

General Electronic Products - includes products suitable for applications where the major requirement is the function of the completed assembly.

Class 2

Dedicated Service Electronic Products - includes products where continued performance and extended life are required, and for which uninterrupted service is desired but not critical.

Class 3

High Performance Electronic Products - includes products where continued high performance or performance-on-demand is critical, equipment downtime cannot be tolerated, end-use environment may be exceptionally harsh, and the equipment must function when required (such as life support systems and flight control systems).

The standard covers many aspects of PCB quality, including base material surface and subsurface conditions, solder resist coverage, conductor width and spacing, annular ring, hole size and location, and more. It's a crucial reference for PCB manufacturers, assemblers, and qality control teams.

Remember, for the most accurate and up-to-date information, it's always best to refer to the IPC website or the actual IPC-A-600J document.

2.3.53 PCB Manufacturers

PCB manufacturers, or printed circuit board manufacturers, are companies that specialize in producing printed circuit boards. These are the fundamental components of virtually all electronic devices. A printed circuit board (PCB) mechanically supports and electrically connects electronic components using conductive tracks, pads, and other features etched from copper sheets laminated onto a non-conductive substrate.

PCB manufacturers' responsibilities typically include:

Design

Although the design is often provided by the customer or developed in collaboration with the customer, manufacturers may have in-house design services. This process involves laying out the circuit design on the board in a way that optimizes the use of space while minimizing potential interference.

Fabrication

This is the primary role of a PCB manufacturer. It involves creating the PCB by etching the design onto a copper-clad laminate, then drilling holes for component leads and adding a solder mask and silkscreen layer.

Assembly

Some PCB manufacturers also offer assembly services (these are often referred to as "turnkey" services). This involves soldering the components onto the board according to the provided Bill of Materials (BOM).

Testing

After the PCBs are assembled, manufacturers often perform tests to ensure the boards function as intended. This could involve visual inspection, automated optical inspection (AOI), x-ray inspection, In-Circuit Test (ICT), or functional tests.

PCB manufacturers serve a wide range of industries, including consumer electronics, telecommunications, aerospace, automotive, medical devices, and military and defense. They can produce different types of PCBs, including single-sided, double-sided, and multi-layer PCBs, as well as more specialized types like flexible and rigid-flex PCBs, depending on the requirements of the design.

The choice of a PCB manufacturer can significantly impact the quality, cost, and timely delivery of the finished product, so it's essential to select a manufacturer who can meet the specific needs of a given project.

2.3.53.1 PCB Manufacturer's Capabilities

The capabilities of a PCB (Printed Circuit Board) manufacturer can greatly impact the design and performance of a PCB. Different manufacturers may have different technical capabilities, depending on their equipment, experience, and focus area. Here are some key aspects of a manufacturer's capabilities that can affect a PCB:

Layer Count

Some manufacturers may be equipped to produce boards with many layers, while others might be limited to fewer layers. This can impact the complexity of the design that can be implemented.

Minimum Trace Width/Spacing

This defines the smallest possible width of a copper track and the smallest gap between two tracks that the manufacturer can reliably fabricate. More advanced manufacturers might be capable of producing smaller traces and spaces, which can allow for higher-density designs.

Hole Sizes

The smallest and largest hole sizes that a manufacturer can drill can influence the design, particularly regarding the use of through-hole components and vias.

Materials: Manufacturers may specialize in certain types of materials (e.g., standard FR-4, high-frequency materials, flexible materials, etc.). The material used can impact the electrical performance and mechanical properties of the PCB.

Surface Finishes

Different manufacturers may offer different surface finishes (e.g., HASL, ENIG, OSP, etc.), which can impact solderability, durability, and cost.

Tolerances

This refers to the degree of variation that is acceptable in the manufacturing process. Higher precision manufacturers can offer tighter tolerances, which can allow for more accurate and reliable designs.

Impedance Control

For high-frequency or high-speed designs, the manufacturer needs to be able to control the characteristic impedance of the traces. This requires specific manufacturing capabilities and expertise.

Special Features

Some manufacturers may offer special features like blind and buried vias, controlled depth drilling, heavy copper layers, etc., which can offer more flexibility in design.

Quality Control

The manufacturer's quality control processes and certifications (e.g., ISO 9001, IPC standards compliance, etc.) can impact the reliability and quality of the final product.

Turnaround Time

Different manufacturers may have different lead times for manufacturing and delivery, depending on their process efficiency and workload.

When selecting a manufacturer, it's important to consider these capabilities in relation to the requirements of your specific PCB design. A complex design might require a more capable (and potentially more expensive) manufacturer, while a simpler design might be able to be produced more cost-effectively by a less advanced manufacturer. Always consult with potential manufacturers to understand their capabilities and to ensure they can meet the needs of your design.

2.3.53.2 PCB Manufacturers in Africa

The printed circuit board (PCB) industry in Africa is not as developed as in other regions such as Asia, North America, and Europe. However, there are a few PCB manufacturers offering their services. Here are some of them:

Bosco Printed Circuits (Pty) Ltd.

Based in South Africa, Bosco Printed Circuits is one of the largest producers of PCBs in Africa. They cater to both local and international customers and offer a wide range of services, including single-sided, double-sided, and multilayer PCB production.

PNC Circuits

PNC Circuits, also in South Africa, provides a broad range of PCB manufacturing services. They cater to various sectors, including telecommunications, military, mining, and industrial electronics.

Trax Interconnect

Trax, located in Cape Town, South Africa, provides high-quality PCBs for local and international customers. They offer a variety of services, including PCB design, manufacturing, and assembly.

Elmatica

Elmatica is not an African company—it's based in Norway—but it has expanded its presence in South Africa. Elmatica provides a wide array of services, from early involvement in the PCB design stage to the actual delivery of the board.

When choosing a PCB manufacturer, it's essential to consider the specific needs of your project, including complexity, volume, budget, and turnaround time. The manufacturer's reputation, quality control processes, and customer service are also important considerations. As always, thorough research and consultation with potential manufacturers are recommended to ensure they can meet your project's requirements. Please note that capabilities, reputation, and other specifics could have changed post-September 2021.

2.3.53.3 PCB Manufacturers in Canada

Canada has several notable PCB manufacturers known for their quality standards and capabilities. Here are some of them:

AP Circuits

Based in Alberta, AP Circuits offers quick-turn PCB prototyping services. They're known for their fast turnaround times and competitive prices.

Candor Industries

Candor Industries is a unique PCB manufacturer because they use a completely additive process that's more environmentally friendly than traditional PCB manufacturing methods. They offer a wide variety of services, including multi-layer, flexible, and rigid-flex PCBs.

Circuit Center

Circuit Center provides a full range of PCB manufacturing services, including design, fabrication, and assembly. They specialize in quick-turn and prototype services.

Global Electronics Services

Global Electronics Services offer a wide range of PCB manufacturing services and are capable of handling both small-scale and large-scale production runs.

Pcb.ca (PCB Prototypes)

Located in Markham, Ontario, pcb.ca offers a range of PCB services, from design to manufacturing and assembly. They cater to a wide range of needs, from prototype to production quantities.

Circoflex

Circoflex is a well-known manufacturer for quick turn-around of high reliability PCBs. They also provide services in flexible and rigid-flex PCBs.

Remember, the best PCB manufacturer for your needs will depend on several factors, including the complexity of your design, your budget, and your desired delivery time. Always be sure to thoroughly research potential manufacturers and consult with them to ensure they can meet your project's requirements. As with any industry, capabilities, reputation, and other specifics could have changed post-September 2021.

2.3.53.4 PCB Manufacturers in China

China is home to a significant number of PCB manufacturers, given its status as a global hub for electronics manufacturing. Many of these manufacturers offer competitive pricing and advanced manufacturing capabilities. Here are a few well-known PCB manufacturers in China:

Shenzhen Zhongxinhua Electronics Co., Ltd.

Zhongxinhua has over 20 years of experience producing PCBs, and specializes in single-sided, double-sided and multi-layer PCBs.

RayMing Technology

Also known as RayPCB, this company offers a wide range of PCB manufacturing services, including for advanced multi-layer designs. They are capable of fast turnarounds and offer assembly services as well.

PCBWay

PCBWay offers a broad range of PCB services, including PCB prototyping, assembly, and advanced PCB production. They are known for their competitive pricing and fast delivery times.

JLCPCB

JLCPCB is one of the largest PCB manufacturers in China, known for its cost-effective services. They offer multi-layer PCBs, flexible PCBs, and also provide assembly services.

King Credie Technology

Specializing in advanced PCBs, King Credie can produce high-density interconnect (HDI) boards, rigid-flex boards, and more.

Golden Triangle Group

With over 30 years of experience, Golden Triangle Group manufactures a variety of PCBs including multi-layer, double-sided, high frequency, and metal substrate boards.

Elecrow

Elecrow offers PCB manufacturing and assembly services, and they are popular with makers and smaller scale electronics enthusiasts.

Seeed Studio

While they're more of a maker's hub, Seeed Studio offers Fusion PCB manufacture and assembly service that caters to prototypes and small-batch printed circuit board production.

Remember that the choice of manufacturer should be based on several factors including but not limited to cost, quality, capabilities, delivery times, and reliability. It's always recommended to do thorough research, read reviews, and if possible, get referrals before choosing a manufacturer. Also, keep in mind that the capabilities of these manufacturers might have changed.

2.3.53.5 PCB Manufacturers in Europe

There are several notable PCB manufacturers in Europe, known for their quality standards and technological capabilities. Here are some of them:

Eurocircuits

Based in Belgium, Eurocircuits is one of the most well-known PCB manufacturers in Europe. They specialize in prototype and small-series PCB manufacturing, and they have a user-friendly online interface for quoting and order placement.

Beta LAYOUT

Based in Germany, Beta LAYOUT offers PCB manufacturing, assembly, and even 3D printing services. They're also known for their RFID technology.

NCAB Group

NCAB is a global company with strong presence in Europe. They have extensive experience in PCB production and emphasize high-quality products.

ELTEC

Based in Italy, ELTEC has been in the PCB industry since 1981. They offer a wide range of services from prototype to mass production and from simple to complex multilayer PCBs.

Würth Elektronik

Located in Germany, Würth Elektronik is a reputable PCB manufacturer with a broad range of capabilities including flexible, rigid, and semi-flexible PCBs.

Multi-CB (Multi Circuit Boards Ltd.)

Multi-CB, based in the UK and Germany, offers a wide range of PCBs and related services. They have a strong focus on high-tech boards, like HDI and RF PCBs.

ELECONT

ELECONT, located in Spain, provides a variety of PCB manufacturing services with capabilities ranging from standard single-sided PCBs to multilayer PCBs.

Keep in mind that the best choice of manufacturer depends on your specific project needs, such as the required turnaround time, the complexity of the PCB design, the desired order volume, and your budget. Always be sure to thoroughly research and consult with potential manufacturers to ensure they can meet your project's requirements. Please note that the mentioned companies' capabilities and reputation could have changed.

2.3.53.6 PCB Manufacturers in the India

India has several notable PCB manufacturers that are known for their quality and diverse capabilities. Here are a few of them:

AT&S India Pvt. Ltd.

AT&S is a leading global manufacturer of high-end printed circuit boards. The company's India division is one of the top PCB manufacturers in the country, known for high-quality and high-precision PCBs.

Epitome Components Ltd.

This is one of India's leading electronic components manufacturing companies. They offer a variety of PCBs, including single-sided, double-sided, and multilayer.

Genus Electrotech Ltd.

Genus Electrotech is one of the fastest growing electronics companies in India. They provide a wide range of services including PCB manufacturing, with capabilities spanning from prototype to high volume production.

Circuit Systems India Ltd.

Based in Gujarat, Circuit Systems offers a broad range of PCBs, including single-sided, double-sided, metal clad, and multilayer PCBs.

Shogini Technoarts Pvt. Ltd.

With over 40 years of experience, Shogini Technoarts is a well-known PCB manufacturer in India. They cater to various industries such as automotive, telecommunications, and defense.

Ascent Circuits Pvt. Ltd.

Based in Karnataka, Ascent Circuits offers a wide range of PCB manufacturing services. Their offerings span from single-sided PCBs to complex multilayer designs.

When choosing a PCB manufacturer, it's essential to consider factors such as your specific project needs, the manufacturer's capabilities, their quality control processes, and their reputation for reliability. Always perform thorough research and consult with potential manufacturers to ensure they can meet your project's requirements. Note that capabilities, reputation, and other specifics could have changed.

2.3.53.7 PCB Manufacturers in the Middle East

There are several PCB manufacturers in the Middle East known for their quality standards and capabilities. Here are some of them:

MEPPI (Middle East Printed Circuits)

Based in Egypt, MEPPI is one of the oldest PCB manufacturers in the Middle East. They offer a wide range of services, including design, manufacturing, and assembly of PCBs.

Elco Group

Located in Israel, Elco Group is a global provider of manufacturing services for PCBs. They serve a variety of industries, including medical, defense, aerospace, and industrial sectors.

PCS Circuits

PCS Circuits is based in Israel and provides a range of PCB manufacturing services. They specialize in prototype production and small series manufacturing.

PCB Technologies

Also based in Israel, PCB Technologies offers a comprehensive range of PCB-related services, including design, fabrication, and assembly. They cater to diverse industries such as medical, defense, aerospace, and telecommunications.

Petratec International Ltd.

Petratec, another Israeli company, offers a wide range of services, including PCB design, fabrication, assembly, and components procurement.

When choosing a PCB manufacturer, consider factors such as your specific project needs, the manufacturer's capabilities, their quality control processes, and their reputation for reliability. Always perform thorough research and consult with potential manufacturers to ensure they can meet your project's requirements. As with any industry, capabilities, reputation, and other specifics could have changed.

2.3.53.8 PCB Manufacturers in the South America

There are several notable PCB manufacturers in South America. While the PCB industry in South America may not be as extensive as in regions like Asia or North America, there are still capable manufacturers present. Here are a few:

Circuibras

Circuibras is based in Brazil and is a significant player in the Brazilian PCB industry. They produce a variety of PCB types, including single-sided, double-sided, and multilayer boards.

CIPSA Circuits

Based in Barcelona and with a plant in Brazil, CIPSA is a leading European manufacturer of PCBs. They have a strong presence in the South American market and offer a wide range of PCB manufacturing services.

Printed Circuits Electronic Industrial S.A. (PCEISA)

Located in Colombia, PCEISA offers both PCB manufacturing and assembly services, including prototype and volume production capabilities.

Tecnoface

Also based in Brazil, Tecnoface specializes in PCB fabrication and assembly. They offer a wide variety of PCBs, from simple to complex designs.

Microcirtec

Based in Argentina, Microcirtec manufactures a range of PCBs, including single-sided, double-sided, and multilayer PCBs.

When choosing a PCB manufacturer, it's crucial to consider the specific needs of your project, including complexity, volume, budget, and turnaround time. The manufacturer's reputation, quality control processes, and customer service are also important considerations.

Please note that it's always recommended to do a comprehensive research and consultation before making a selection, and capabilities, reputation, and other specifics could have changed.

2.3.53.9 PCB Manufacturers in the UK

The UK has several notable PCB manufacturers known for their quality standards and advanced capabilities. Here are some of them:

European Circuits

European Circuits provides a wide range of PCB services including design, manufacturing, and assembly. They service a diverse range of sectors including aerospace, automotive, and telecommunications.

Kingfield Electronics

Kingfield Electronics offers full turnkey solutions for PCB assembly and manufacturing. They have been serving various industries for over 30 years.

B.E.C. Manufacturing

B.E.C. is well-known for delivering a range of services, including PCB assembly, prototyping, and testing. They cater to a variety of sectors like medical, industrial, and consumer electronics.

Multi-CB (Multi Circuit Boards Ltd.)

Multi-CB, based in Germany and the UK, provides a broad range of PCBs and related services. They have a strong focus on high-tech boards, including HDI and RF PCBs.

Hi5 Electronics

Based in Rochdale, Hi5 Electronics is a leading manufacturer of PCBs. They offer a variety of services, including prototype PCB assembly and manufacture.

Quantum CAD

Not just a PCB manufacturer, Quantum CAD offers a comprehensive suite of design services, including PCB design, layout, and data management.

Elite7 Group

Based in Surrey, Elite7 offers PCB manufacturing, SMT equipment, and various assembly services. They have extensive experience and a strong customer base in the UK.

Remember, the best PCB manufacturer for you will depend on your specific requirements, such as design complexity, order volume, turnaround time, and budget. When selecting a PCB manufacturer, ensure that you perform due diligence to verify they can meet your project's specifications. As with any industry, capabilities, reputation, and other specifics could have changed.

2.3.53.10 PCB Manufacturers in the USA

There are several well-regarded PCB manufacturers in the United States. These companies can offer high-quality PCBs, good customer service, and often have advanced capabilities for complex designs. Here are a few notable examples:

Advanced Circuits (4PCB)

Based in Colorado, Advanced Circuits is one of the most popular PCB manufacturers in the US. They offer a wide range of services, including quick-turn PCB prototypes and large-scale production.

Sunstone Circuits

Located in Oregon, Sunstone Circuits is known for their user-friendly online order process and reliable customer service. They offer a range of services from prototype to production volumes.

Sierra Circuits

Based in California, Sierra Circuits offers advanced PCB manufacturing capabilities, including HDI and flex PCBs. They're known for their high-quality products and excellent customer service.

San Francisco Circuits (SFC)

SFC specializes in quick turn and specialty PCB fabrication and design services. They have expertise in areas like RF/Microwave, flex, and rigid-flex circuits.

Epec Engineered Technologies

With over 65 years of experience, Epec offers a wide range of custom fabricated electronics including PCBs. They're a good choice for both prototype and high-volume production.

PCB Universe

Based in Washington, PCB Universe offers a wide range of PCB fabrication and assembly services. They have capabilities for a variety of specialty materials and advanced technologies.

Imagineering

This Illinois-based company offers both PCB manufacturing and assembly services. They're known for competitive prices and quick turn services.

Screaming Circuits

Oregon-based Screaming Circuits specializes in fast and high-quality PCB assembly, particularly for prototype and small-volume orders.

While American manufacturers often have higher prices than counterparts in countries like China, they may offer advantages in terms of delivery times, ease of communication, and adherence to high-quality standards. As with selecting any manufacturer, it's important to consider the specific requirements of your project and to thoroughly research potential manufacturers to ensure they can meet your needs. Please note that capabilities, reputation, and other specifics could have changed.

2.3.54 PCB Testing

PCB testing refers to the process of evaluating and verifying the functionality, quality, and reliability of printed circuit boards (PCBs). Testing is a critical stage in the PCB manufacturing and assembly process to ensure that the boards meet the required specifications and standards. PCB testing helps identify and rectify potential issues, defects, or errors before the PCBs are deployed in electronic devices or systems. There are several types of PCB testing, including:

Visual Inspection: Visual inspection is the initial step, where PCBs are visually examined for any obvious defects, such as missing components, soldering issues, or physical damage.

Automated Optical Inspection (AOI): AOI is a more advanced inspection method that uses cameras and software to automatically inspect PCBs for defects in solder joints, component placement, and other issues with high accuracy and speed.

In-Circuit Testing (ICT): ICT is a functional test that checks the electrical connections and verifies the performance of individual components on the PCB. This test is usually performed by a specialized test fixture that interfaces with the PCB's test points.

Boundary Scan Testing (JTAG): Boundary scan testing is a method for testing digital circuits on PCBs using the Joint Test Action Group (JTAG) interface. It allows for the testing of interconnected components and detecting faults in non-accessible areas.

Functional Testing: Functional testing involves testing the entire PCB or its sections by simulating real operating conditions and measuring the response to different inputs. This test verifies the overall functionality of the PCB in its intended application.

In-Circuit Emulation (ICE): ICE is used for testing complex digital circuits and microcontrollers. It involves using a special emulator to execute instructions and check the behavior of the circuit.

Thermal Testing: Thermal testing subjects the PCB to temperature extremes to evaluate its performance under varying thermal conditions and to identify potential thermal issues.

Environmental Testing: Environmental testing involves exposing the PCB to extreme conditions like humidity, vibration, and temperature to assess its robustness in harsh environments.

Reliability Testing: Reliability testing is conducted to determine the PCB's long-term performance and endurance under stress conditions to ensure its reliability over time.

Electrical Testing: General electrical testing includes checking for shorts, open circuits, continuity, impedance, and other electrical parameters using specialized test equipment.

X-ray Inspection: X-ray inspection is used to examine the internal structure of PCBs, especially for complex multilayer PCBs, to identify issues like misalignment, voids in solder joints, and delamination.

Each of these testing methods has its advantages and is used at different stages of the PCB manufacturing and assembly process to ensure that the final PCBs meet the required quality standards and specifications.

2.3.54.1 PCB Visual Inspection

PCB visual inspection is an essential step in the quality control process for printed circuit boards (PCBs). It involves carefully examining the PCBs visually to identify any visible defects, inconsistencies, or issues that may have occurred during the manufacturing or assembly process. Visual inspection is usually conducted at various stages of PCB production, including after fabrication, component placement, and soldering processes. Here are the key aspects of PCB visual inspection:

After PCB Fabrication: Once the PCBs are fabricated, they undergo an initial visual inspection to check for any irregularities in the board's surface, such as scratches, cuts, or copper peel-off. The presence of any damage or imperfections can affect the overall performance and reliability of the PCB.

Component Placement Inspection: After the components are placed on the PCB during the assembly process, a visual inspection is carried out to verify the correct orientation and alignment of the components. It ensures that the components are correctly positioned on the PCB according to the design specifications.

Solder Joint Inspection: After soldering, a critical aspect of visual inspection involves examining the solder joints between the components and the PCB. The inspector checks for solder bridges, open joints, solder voids, and any other defects that may affect the electrical connectivity or reliability of the PCB.

Silkscreen and Markings: The silkscreen layer, which includes component outlines, reference designators, part numbers, and logos, is examined to ensure that all necessary markings are present and accurate. This information aids in the assembly and maintenance of the PCB.

Alignment and Placement of Stencil: In surface-mount technology (SMT) assembly, the alignment and placement of the solder paste stencil are checked to ensure proper solder paste deposition on the PCB pads before component placement.

AOI (Automated Optical Inspection): In modern PCB assembly, automated optical inspection (AOI) machines are commonly used for more precise and efficient visual inspection. AOI uses cameras and algorithms to automatically detect defects and anomalies in the PCB, including missing components, skewed components, and oldering issues.

Quality Control Checks: Visual inspection is part of the overall quality control process, which may include various other inspections, such as electrical testing, in-circuit testing, and functional testing, depending on the complexity and requirements of the PCB.

The aim of PCB visual inspection is to catch any visible defects and errors early in the manufacturing process, allowing corrective actions to be taken promptly. It ensures that the PCBs meet the required quality standards and reduces the risk of faulty or unreliable products reaching the end-users. Visual inspection is particularly critical for high-reliability applications, such as aerospace, medical, and automotive electronics, where PCB failures can have severe consequences.

2.3.54.2 PCB Automated Optical Inspection

PCB Automated Optical Inspection (AOI) is an advanced and automated inspection process used in the quality control of printed circuit boards (PCBs). AOI involves using specialized equipment, such as AOI machines, equipped with high-resolution cameras and sophisticated image processing algorithms to automatically inspect PCBs for defects, errors, and inconsistencies. This inspection technique offers several benefits over manual visual inspection, including increased speed, accuracy, and repeatability.

Here's how PCB Automated Optical Inspection works:

Image Acquisition: The PCB is placed on the AOI machine's inspection stage, and high-resolution cameras capture detailed images of the PCB's surface. The cameras

can capture images from different angles, allowing for a comprehensive view of the PCB.

Image Processing: The AOI machine's software processes the captured images using advanced algorithms to identify various features, such as components, solder joints, and traces.

Defect Detection: The software compares the captured images with the expected reference images or the original PCB design data. It looks for defects such as missing components, misaligned components, tombstoning, solder bridges, insufficient solder, solder voids, and other anomalies.

Component Verification: AOI can verify the presence, type, and orientation of components on the PCB to ensure accurate assembly.

Inspection Criteria and Tolerance Settings: The AOI software allows for customizable inspection criteria and tolerance settings, enabling users to define acceptable defect limits based on specific design requirements and industry standards.

Automated Defect Classification: Defects identified by AOI are classified based on their severity. Critical defects that may affect functionality or reliability are flagged for immediate attention, while minor defects can be logged for further analysis.

Reporting and Traceability: AOI machines provide detailed inspection reports, including images of defects and their locations on the PCB. This information aids in troubleshooting and corrective actions. Additionally, AOI can be integrated into manufacturing databases, providing traceability for each inspected PCB.

Advantages of PCB Automated Optical Inspection

- **Speed and Efficiency:** AOI machines can inspect PCBs much faster than manual inspection, improving production throughput and efficiency.
- **Accuracy and Consistency:** AOI eliminates human error, ensuring consistent and accurate inspection results across all PCBs.
- **Real-time Feedback:** AOI provides immediate feedback, allowing manufacturers to take corrective actions promptly, reducing the risk of producing defective PCBs.
- **Increased Inspection Coverage:** AOI can thoroughly inspect dense and complex PCBs with multiple layers, ensuring comprehensive defect detection.
- **Cost-Effectiveness:** Although AOI machines represent an initial investment, they can lead to cost savings in the long run by reducing rework, scrap, and product recalls.
- **Non-destructive Testing:** AOI is a non-destructive testing method that does not damage the PCB during inspection.

PCB Automated Optical Inspection is a valuable tool for PCB manufacturers to enhance quality control, increase productivity, and deliver reliable and high-quality PCBs to their customers.

2.3.54.3 PCB In-Circuit Testing

PCB In-Circuit Testing (ICT) is a type of functional testing used to verify the electrical connectivity and functionality of individual components and circuitry on a printed circuit board (PCB). It is a valuable testing method used during the manufacturing process to ensure the quality and reliability of PCBs before they are integrated into electronic devices or systems. ICT is typically performed using specialized test fixtures and equipment designed for this purpose.

Here's how PCB In-Circuit Testing works:

Test Fixture Preparation: A custom test fixture is designed to interface with the specific PCB being tested. The test fixture contains spring-loaded probes or pogo pins that make contact with designated test points on the PCB. These test points are typically exposed pads or vias that allow access to the circuit nodes for testing.

Test Program Generation: A test program is developed based on the PCB design and the intended functionality. This test program contains a series of test vectors and instructions to be executed during the ICT process.

Board Loading: The PCB is placed in the test fixture, and the test fixture securely holds the PCB in place to ensure accurate and consistent contact with the test points.

Electrical Test Execution: The ICT equipment applies test signals to specific nodes on the PCB and measures the electrical responses to determine the functionality of various components, traces, and connections. The test program may include checks for short circuits, open circuits, incorrect component values, incorrect component placements, and other electrical anomalies.

Functional Verification: In addition to testing individual components and connections, ICT can also involve functional verification of specific circuit features and subsystems on the PCB to ensure that they operate as intended.

Data Collection and Analysis: The ICT equipment collects test data and compares the measured responses to expected values based on the test program. The test results are then analyzed to identify any faults or defects.

Fault Diagnosis and Reporting: If any faults or failures are detected, the ICT equipment provides diagnostic information to assist in locating the root cause of the

issues. This information helps in the troubleshooting and corrective action process. Test reports are generated, documenting the test results and any identified defects.

Advantages of PCB In-Circuit Testing

- **High Test Coverage:** ICT provides comprehensive test coverage, allowing for the detection of a wide range of electrical faults, including subtle defects that may not be visible during visual inspection.
- **Speed and Efficiency:** ICT can quickly test multiple circuit nodes simultaneously, making it an efficient method for high-volume PCB production.
- **Accuracy and Reproducibility:** ICT offers high accuracy and reproducibility in test results, ensuring consistent and reliable testing across multiple PCBs.
- **Non-destructive Testing:** ICT is a non-destructive testing method that does not damage the PCB during the testing process.
- **Component Verification:** ICT can verify the correct value and orientation of individual components on the PCB.
-

PCB In-Circuit Testing is a valuable quality assurance tool that helps ensure the functionality and reliability of PCBs before they are integrated into electronic products. By identifying and addressing defects early in the manufacturing process, ICT contributes to the production of high-quality PCBs and minimizes the risk of faulty or unreliable electronic devices.

2.3.54.4 PCB Boundary Scan Testing

PCB Boundary Scan Testing, also known as JTAG (Joint Test Action Group) testing, is a specialized method for testing and verifying the interconnections and functionality of digital integrated circuits on a printed circuit board (PCB). This testing technique is widely used for PCBs that incorporate complex digital devices, such as microcontrollers, FPGAs (Field-Programmable Gate Arrays), and other boundary scan-capable components.

Boundary Scan Testing utilizes the standardized JTAG interface, which allows for testing and debugging of digital circuits and components on the PCB without the need for physical access to each individual pin. The JTAG interface provides a way to access the internal circuitry of boundary scan-capable components, facilitating a comprehensive and non-intrusive test.

Here's how PCB Boundary Scan Testing works:

Boundary Scan Registers: Boundary Scan Testing relies on boundary scan registers (BSR) built into boundary scan-capable devices. These registers are used to shift data in and out of the components via the JTAG interface.

JTAG TAP Controller: The Test Access Port (TAP) controller manages the flow of test data in and out of the JTAG chain. It acts as a state machine controlling the JTAG interface's operation during testing.

JTAG Chain: The JTAG chain connects boundary scan-capable components on the PCB in series, forming a scan path. Each component in the chain has its own boundary scan register.

Scan Data Loading: During PCB Boundary Scan Testing, test data is serially loaded into the boundary scan registers of the connected components via the JTAG interface.

Boundary Scan Test Operation: Once the test data is loaded, the JTAG TAP controller shifts the test data through the boundary scan registers, applying test patterns to the connected components. These test patterns can be used to check for stuck-at faults, transition faults, and other digital circuit defects.

Observation and Result Capture: The test responses from the components are shifted out through the JTAG chain and analyzed to determine if the expected behavior matches the actual behavior of the components. Test results are captured and evaluated.

Debugging and Troubleshooting: PCB Boundary Scan Testing also provides powerful debugging and troubleshooting capabilities. It allows engineers to isolate faults to specific components and identify the root cause of issues.

Advantages of PCB Boundary Scan Testing

- **Non-intrusive Testing:** PCB Boundary Scan Testing is non-intrusive, meaning it does not require physical access to individual pins for testing. This is particularly beneficial for components with limited external access, such as BGA (Ball Grid Array) packages.
- **Comprehensive Test Coverage:** Boundary Scan Testing provides comprehensive test coverage for complex digital circuits, including interconnections and components that may not be reachable by other testing methods.
- **Efficient Test Execution:** PCB Boundary Scan Testing can be executed efficiently and quickly, making it suitable for high-volume production environments.
- **Debugging Support:** The JTAG interface supports in-circuit debugging, which is valuable during the development and prototyping stages of PCB design.
- Boundary Scan Testing is widely used in the electronics industry, especially in applications where digital circuits play a significant role. It has become an essential tool for ensuring the quality and reliability of PCBs that contain complex digital components.

2.3.54.5 PCB Functional Testing

PCB functional testing is a comprehensive testing process that verifies the overall functionality and performance of a printed circuit board (PCB) by subjecting it to simulated real-world operating conditions. This type of testing is designed to ensure that the PCB meets its intended design specifications and performs as expected in its intended application.

Functional testing involves evaluating the PCB's behavior when exposed to various inputs, stimuli, or signals to assess its responses and outputs. The purpose of functional testing is to verify that all the components, circuits, and connections on the PCB work correctly together to achieve the desired functionality.

Here's how PCB functional testing works:

Test Environment Setup: A test environment is set up, which includes the necessary equipment, test instruments, and test fixtures to simulate the conditions under which the PCB will operate.

Test Stimuli Generation: Test stimuli, such as electrical signals, data inputs, or control signals, are applied to the PCB's inputs to simulate the real-world operating conditions.

Measurement and Observation: The outputs of the PCB are measured and observed to determine whether they meet the expected responses, performance criteria, and functionality defined in the design specifications.

Functional Verification: The PCB's response to various stimuli is compared against the expected results, and its functional behavior is verified against the intended design.

Functional Test Coverage: Functional testing aims to achieve high test coverage by exercising all the critical functions, interfaces, and modes of operation of the PCB.

Automated Functional Testing: In modern PCB manufacturing, automated testing equipment and software are often used to streamline the functional testing process. Automated testing can help improve efficiency, repeatability, and consistency in testing multiple PCBs.

Boundary Cases and Stress Testing: Functional testing may include subjecting the PCB to boundary cases and stress testing to evaluate its performance under extreme or unusual operating conditions.

Advantages of PCB Functional Testing

- **Comprehensive Validation:** Functional testing provides a thorough evaluation of the PCB's functionality, ensuring that it performs as intended in real-world scenarios.

- **Early Defect Detection:** Functional testing can detect defects and issues that may not be apparent during other testing methods, such as visual inspection or in-circuit testing.
- **Real-world Performance Assessment:** By simulating real-world operating conditions, functional testing offers a more accurate assessment of the PCB's performance in its intended application.
- **Quality Assurance:** Functional testing plays a crucial role in quality assurance, helping to deliver reliable and high-quality PCBs to end-users.
- **Regulatory Compliance:** Functional testing is often required to meet industry standards, regulations, and certification requirements.

Functional testing is an integral part of the PCB manufacturing and assembly process. It ensures that the final product meets the design specifications, performs as expected, and meets the required quality standards, contributing to the overall success and reliability of electronic devices and systems.

2.3.54.6 PCB In-Circuit Emulation

PCB In-Circuit Emulation (ICE) is a powerful and specialized testing and debugging technique used to analyze and troubleshoot complex digital circuits, microcontrollers, and other programmable devices on a printed circuit board (PCB). It allows engineers to observe and interact with the internal operation of the digital components in real-time without disrupting the normal functioning of the PCB or the device under test.

ICE involves the use of specialized hardware, such as an In-Circuit Emulator or an In-Circuit Debugger, which interfaces directly with the target device's pins on the PCB. This hardware allows engineers to load custom firmware or test programs into the device, observe the internal states and signals, and control the device's operation while it is operating in its actual circuit environment.

Here's how PCB In-Circuit Emulation works:

Connection to Target Device: The In-Circuit Emulator (ICE) or Debugger is connected to the target device (e.g., microcontroller) on the PCB using dedicated test probes or pins. These connections provide access to the internal circuitry of the device.

Firmware Loading: Custom firmware or test programs are loaded into the target device using the ICE hardware. This custom code may contain additional debugging features, diagnostic routines, or test sequences.

Real-time Interaction: With the ICE hardware connected and the custom firmware loaded, the engineer can interact with the target device in real-time. This includes pausing the execution of the device, stepping through instructions, and examining the internal register states, memory contents, and I/O signals.

Breakpoints and Triggers: Engineers can set breakpoints and triggers to halt the device's operation when specific conditions are met. This feature helps identify and analyze specific events or issues during the device's execution.

Data and Signal Capture: ICE allows for capturing and analyzing data and signals during the device's operation. This capability is valuable for monitoring the behavior of the device and detecting issues that might be challenging to observe with traditional testing methods.

Debugging and Troubleshooting: Engineers can use ICE to debug and troubleshoot issues in the target device's firmware or hardware. It enables them to identify software bugs, verify the correct functioning of hardware peripherals, and analyze timing-related problems.

Advantages of PCB In-Circuit Emulation

- **Real-time Analysis:** ICE provides real-time access to the internal states of the target device, enabling engineers to analyze its behavior in real-world conditions.
- **Non-intrusive Testing:** ICE is non-intrusive, meaning it does not disrupt the normal operation of the device or the PCB, allowing for accurate debugging and testing.
- **High-level Debugging:** ICE provides detailed insight into the target device's operation, allowing engineers to perform high-level debugging and optimize code execution.
- **Complex System Analysis:** ICE is particularly useful for analyzing complex digital systems, including multi-core processors or systems with intricate timing requirements.
- **Interactive Development:** Engineers can interactively develop, test, and refine their code or firmware while the device is operating in its actual environment.

In-Circuit Emulation is a valuable tool for debugging and validating complex digital circuits and microcontrollers. It helps engineers identify and resolve issues quickly, reducing development time and improving the quality and reliability of electronic products.

2.3.54.7 PCB Thermal Testing

PCB thermal testing, also known as thermal analysis or thermal profiling, is the process of evaluating and analyzing the temperature distribution and thermal behavior of a printed circuit board (PCB) during operation. This testing is essential to ensure that the PCB can dissipate heat effectively and maintain safe operating temperatures for all components.

Thermal testing helps identify potential hotspots, thermal gradients, and areas of concern on the PCB, which can affect the performance, reliability, and lifespan of electronic devices. It is particularly critical for PCBs used in applications with high-power components, high-frequency circuits, or in environments with challenging thermal conditions.

Here's how PCB thermal testing is typically conducted:

Thermal Sensors and Probes: Temperature sensors and probes, such as thermocouples or infrared sensors, are attached to specific locations on the PCB to measure the temperature during operation.

Test Setup: The PCB is placed inside an environmental chamber or test fixture that allows controlled thermal conditions. The test setup may simulate the actual operating environment or expose the PCB to temperature extremes for stress testing.

Temperature Profiling: During testing, the PCB is powered on and subjected to the intended operating conditions or specific thermal scenarios. The temperature sensors continuously measure the temperature at different points on the PCB.

Data Logging and Analysis: The temperature data is logged and recorded over a period of time to capture the thermal behavior of the PCB. The recorded data is then analyzed to identify hotspots, areas with insufficient heat dissipation, or any temperature-related issues.

Thermal Imaging: Infrared thermography or thermal imaging cameras are often used to visualize the temperature distribution across the entire PCB surface. This helps to quickly identify regions with abnormal temperature patterns.

Simulation and Modeling: In addition to physical testing, thermal analysis can also be performed using computer simulations and modeling techniques. Software tools can predict the thermal behavior of the PCB based on the design, material properties, and operating conditions.

Thermal Design Optimization: Based on the results of thermal testing and analysis, design changes or improvements may be implemented to optimize the PCB's thermal performance. This can include adjusting the placement of heat-generating components, optimizing copper traces for better heat dissipation, and adding thermal vias or heat sinks as needed.

Advantages of PCB Thermal Testing

- **Reliability and Longevity:** Thermal testing helps ensure that the PCB can operate within safe temperature ranges, reducing the risk of component failures and extending the life of electronic devices.
- **Quality Assurance:** By analyzing the thermal behavior of the PCB, manufacturers can validate that the design meets thermal specifications and performance requirements.
- **Troubleshooting:** Thermal testing can identify thermal-related issues early in the development process, allowing engineers to address and resolve them before the product is released.
- **Design Optimization:** The insights gained from thermal testing enable designers to make informed decisions and optimize the PCB layout and materials for improved thermal performance.
- **Compliance with Standards:** Many industries have specific thermal requirements and standards that must be met. Thermal testing helps ensure compliance with these standards.

Thermal testing is an essential aspect of PCB design and validation, especially for applications that involve high-power or high-temperature conditions. By understanding the PCB's thermal behavior, designers can make informed decisions to enhance the reliability and performance of electronic products.

2.3.54.8 PCB Environmental Testing

PCB environmental testing, also known as environmental stress testing or reliability testing, involves subjecting printed circuit boards (PCBs) to various environmental conditions to assess their performance, reliability, and robustness in challenging operating environments. The testing is conducted to ensure that the PCBs can withstand the stresses associated with real-world conditions and meet industry standards and regulations.

Environmental testing is especially crucial for PCBs used in applications exposed to harsh conditions, such as aerospace, automotive, military, industrial, and outdoor electronic systems. The testing process helps identify potential weaknesses, failure modes, and performance limitations of the PCB under different environmental stressors.

Here are some common types of PCB environmental testing:

- **Temperature Testing:** Thermal testing subjects the PCB to temperature extremes, both high and low, to evaluate its performance under varying temperature conditions. This testing ensures that the PCB can operate reliably within the specified temperature range.

- **Humidity Testing:** Humidity testing evaluates the PCB's resistance to moisture and humidity. It can help identify potential issues like corrosion, electrical leakage, and material degradation due to exposure to humid environments.
- **Thermal Cycling:** Thermal cycling involves subjecting the PCB to repetitive temperature changes, simulating the thermal stresses that occur during real-world use. This testing helps detect any weaknesses or fatigue-related failures.
- **Vibration Testing:** Vibration testing assesses the PCB's ability to withstand mechanical vibrations and shocks that may occur during transportation or operation. It helps identify potential solder joint failures, component dislodgment, and mechanical weaknesses.
- **Shock Testing:** Shock testing evaluates the PCB's resistance to mechanical shocks and impacts. It simulates the effects of drops, impacts, or sudden changes in motion that the PCB may experience in its intended application.
- **Salt Spray Testing:** Salt spray testing exposes the PCB to a salt-laden atmosphere to assess its resistance to corrosion, which is particularly important for PCBs used in marine or coastal environments.
- **Dust and Contaminant Testing:** Dust and contaminant testing evaluate the PCB's resilience against dust, dirt, and other contaminants that may accumulate in the operating environment.
- **Fungus Resistance Testing:** Fungus resistance testing assesses the PCB's ability to withstand fungal growth in humid or damp environments.
- **EMI/EMC Testing:** Electromagnetic interference (EMI) and electromagnetic compatibility (EMC) testing evaluate the PCB's susceptibility to electromagnetic interference and its ability to operate without causing interference to other components or systems.
- *Altitude Testing:* Altitude testing assesses the PCB's performance under low-pressure conditions, as it might encounter in high-altitude applications or during transportation by air.

The data collected during environmental testing is analyzed to identify any issues and to validate whether the PCB design meets the specified environmental requirements and industry standards. Any necessary design modifications are made to improve the PCB's reliability and performance under the intended operating conditions. By conducting environmental testing, manufacturers can ensure that their PCBs are reliable, durable, and able to withstand the environmental challenges they may encounter during their lifespan.

2.3.54.9 PCB Reliability Testing

PCB reliability testing, also known as reliability assessment or reliability qualification, is a systematic and comprehensive process used to evaluate the long-term performance, durability, and robustness of printed circuit boards (PCBs). The primary goal of reliability testing is to ensure that the PCBs meet their intended design specifications and can operate reliably and consistently throughout their expected lifespan in real-world applications.

Reliability testing involves subjecting the PCBs to various stressors and accelerated aging conditions to identify potential failure modes and assess their impact on the PCB's performance. The testing process is crucial for high-reliability applications, such as aerospace, automotive, medical, and industrial electronics, where the failure of a PCB can have serious consequences.

Here are some common types of PCB reliability testing:

- **Temperature Cycling:** Temperature cycling involves repeatedly subjecting the PCB to temperature changes between high and low extremes. This testing simulates the thermal stresses that the PCB may experience during operation and helps identify potential solder joint failures, material degradation, and reliability issues related to thermal expansion and contraction.
- **High-Temperature Operating Life (HTOL):** HTOL testing exposes the PCB to an elevated temperature for an extended period to assess its long-term performance under high-temperature conditions. It helps identify potential early failures and assess the PCB's stability and reliability.
- **Highly Accelerated Life Testing (HALT):** HALT combines various environmental stressors, such as temperature, vibration, and rapid thermal cycling, to accelerate the aging process of the PCB. HALT testing is used to quickly identify weak points and potential failure modes.
- **Highly Accelerated Stress Screening (HASS):** HASS is a subset of HALT, used in production testing to identify manufacturing defects and ensure the reliability of the final product.
- **Mechanical Shock Testing:** Mechanical shock testing evaluates the PCB's ability to withstand sudden impacts and shocks, simulating conditions that may occur during transportation or field use.
- **Vibration Testing:** Vibration testing assesses the PCB's resistance to mechanical vibrations and assesses the reliability of solder joints, components, and mechanical structures.
- **Humidity Testing:** Humidity testing evaluates the PCB's resistance to moisture and humidity, which can cause corrosion and lead to electrical failures.
- **Salt Spray Testing:** Salt spray testing assesses the PCB's resistance to corrosion in marine or coastal environments.

- **Thermal Shock Testing:** Thermal shock testing subjects the PCB to rapid temperature changes to evaluate its ability to withstand sudden thermal transitions.
- **EMI/EMC Testing:** Electromagnetic interference (EMI) and electromagnetic compatibility (EMC) testing assess the PCB's susceptibility to electromagnetic interference and its ability to operate without causing interference to other components or systems.

Reliability testing is a critical step in the PCB manufacturing and development process. It helps identify potential weaknesses and failure modes, allowing design improvements to be made before the PCBs are deployed in critical applications. By conducting comprehensive reliability testing, manufacturers can deliver high-quality and reliable PCBs that meet the demanding requirements of modern electronics.

2.3.54.10 PCB Electrical Testing

PCB electrical testing is a fundamental and essential step in the manufacturing and quality control process of printed circuit boards (PCBs). It involves checking the electrical characteristics, connectivity, and performance of the PCB to ensure that it meets design specifications and functions correctly. Electrical testing is typically performed at various stages of the PCB production, including after fabrication, assembly, and final testing before the PCBs are deployed in electronic devices.

Here are some common types of PCB electrical testing:

- **Continuity Testing:** Continuity testing checks for the presence of electrical connections between different points on the PCB. It verifies that there are no open circuits or unintended breaks in the conductive paths.
- **Short Circuit Testing:** Short circuit testing is performed to detect any unintended electrical connections between different traces or conductive paths on the PCB. It helps identify potential soldering defects or issues with insulation.
- **Functional Testing:** Functional testing verifies the overall functionality of the PCB in its intended application. The PCB is powered on and tested under simulated operating conditions to ensure that all components, circuits, and interfaces work as expected.
- **Voltage and Current Measurements:** Electrical testing involves measuring voltage and current at specific test points on the PCB to ensure that the electrical parameters are within the specified tolerances.
- **Impedance Testing:** In high-frequency applications or controlled impedance designs, impedance testing is performed to verify that the characteristic impedance of transmission lines matches the design specifications.

- **Signal Integrity Testing:** Signal integrity testing evaluates the quality of high-speed signals, looking for issues such as signal reflections, overshoot, undershoot, and noise that could affect the performance of the PCB.
- **Power Integrity Testing:** Power integrity testing analyzes the power delivery network on the PCB, ensuring that it can provide stable and sufficient power to all components without causing voltage fluctuations.
- **Electromagnetic Compatibility (EMC) Testing:** EMC testing checks for the PCB's susceptibility to electromagnetic interference and its ability to function without causing interference to other components or systems.
- **In-Circuit Testing (ICT):** ICT is a type of functional testing that checks the electrical connections and verifies the performance of individual components on the PCB using specialized test fixtures and equipment.
- **Boundary Scan Testing (JTAG):** Boundary scan testing uses the JTAG interface to test interconnected components and detect faults in non-accessible areas.

Automated test equipment (ATE), such as flying probe testers, bed-of-nails testers, and automated optical inspection (AOI) machines, are commonly used for electrical testing to improve efficiency and accuracy.

By conducting thorough electrical testing, manufacturers can identify and rectify any electrical defects or issues early in the production process, ensuring that the final PCBs meet the required quality standards and specifications. Proper electrical testing contributes to the production of reliable and high-performance electronic products.

2.3.54.11 PCB X-ray Inspection

PCB X-ray inspection, also known as X-ray imaging or X-ray testing, is a non-destructive inspection method used to examine the internal features and structures of printed circuit boards (PCBs). This inspection technique is particularly valuable for inspecting complex and densely populated PCBs with hidden or inaccessible components, such as those using surface-mount technology (SMT) or ball grid array (BGA) packages.

In PCB X-ray inspection, X-rays are used to penetrate the PCB's materials, including the solder joints, components, and traces. The X-rays interact with different materials and generate images that can reveal various defects, such as solder joint quality, component placement, and internal connectivity issues.

Here's how PCB X-ray inspection works:

X-ray Equipment Setup: A PCB X-ray inspection system consists of an X-ray source, a detector, and a control unit. The X-ray source emits a controlled beam of X-rays, and the detector captures the X-rays that pass through the PCB.

PCB Preparation: The PCB is securely positioned on the inspection stage, ensuring proper alignment with the X-ray source and detector.

X-ray Exposure: The X-ray source is activated, and the X-rays pass through the PCB. The X-rays are absorbed or scattered differently by various materials, depending on their densities. Components, traces, and solder joints appear as dark or light areas on the resulting X-ray image.

Image Acquisition: The X-ray detector captures the X-ray image, which is displayed on a monitor for analysis.

Defect Detection: The X-ray image is carefully examined for various defects, such as solder voids, solder bridges, insufficient solder, tombstoning, and open joints. The X-ray inspection can also reveal issues with component alignment, hidden cracks, and other internal defects that may not be visible through traditional visual inspection.

Failure Analysis: If any defects are detected, further analysis may be performed to determine the root cause of the issues.

Advantages of PCB X-ray Inspection

- **Non-destructive:** PCB X-ray inspection is non-destructive, meaning it does not damage the PCB or its components during the inspection process.
- **High Inspection Coverage:** X-ray inspection provides a comprehensive view of the internal structures, enabling the inspection of hidden or non-visible areas of the PCB.
- **Complex Component Inspection:** X-ray inspection is especially useful for inspecting BGA packages, where visual inspection is not possible due to the components' hidden nature.
- **Speed and Efficiency:** X-ray inspection can quickly and efficiently inspect multiple PCBs, making it suitable for high-volume production environments.
- **Inspection of Solder Joints:** X-ray inspection can accurately assess the quality of solder joints, which is critical for the reliability of electronic assemblies.

PCB X-ray inspection is a valuable tool for quality control and assurance, especially for PCBs used in high-reliability applications. It helps identify and rectify defects early in the manufacturing process, ensuring that the PCBs meet the required quality standards and performance criteria.

2.3.55 The Institute for Printed Circuits

The IPC (Institute for Printed Circuits) is a global trade association that develops standards, training programs, and certification programs for the electronics industry.

IPC is the global association that helps OEMs, EMS, PCB manufacturers, cable and wire harness manufacturers and electronics industry suppliers build electronics better. IPC members strengthen their bottom line and build more reliable, high-

quality products through proven standards, certification, education and training, thought leadership, advocacy, innovative solutions and industry intelligence.

IPC standards are widely used in the design, manufacturing, and assembly of printed circuit boards (PCBs). IPC standards are updated regularly to reflect changes in technology and industry best practices. They are used by PCB designers, manufacturers, assemblers, and test engineers to ensure that PCBs are designed and manufactured to the highest standards of quality and reliability.

Here are some examples of IPC standards related to PCB design:

IPC-2221: Generic Standard on Printed Board Design

This standard provides guidelines and design rules for the design of printed circuit boards. It covers topics such as board size, layer stackup, trace width, spacing, and other factors that affect the electrical performance and manufacturability of the board.

IPC-7351: Generic Requirements for Surface Mount Design and Land Pattern Standard

This standard provides guidelines for the design of surface mount land patterns for components. It covers topics such as pad size, shape, and placement, and it provides guidance on how to ensure proper solder joint formation.

IPC-6012: Qualification and Performance Specification for Rigid Printed Boards

This standard defines the requirements for the manufacture of rigid printed circuit boards. It covers topics such as material specifications, electrical performance, and testing requirements.

IPC-2223: Sectional Design Standard for Flexible Printed Boards

This standard provides guidelines for the design of flexible printed circuit boards. It covers topics such as materials, layer stackup, trace routing, and termination.

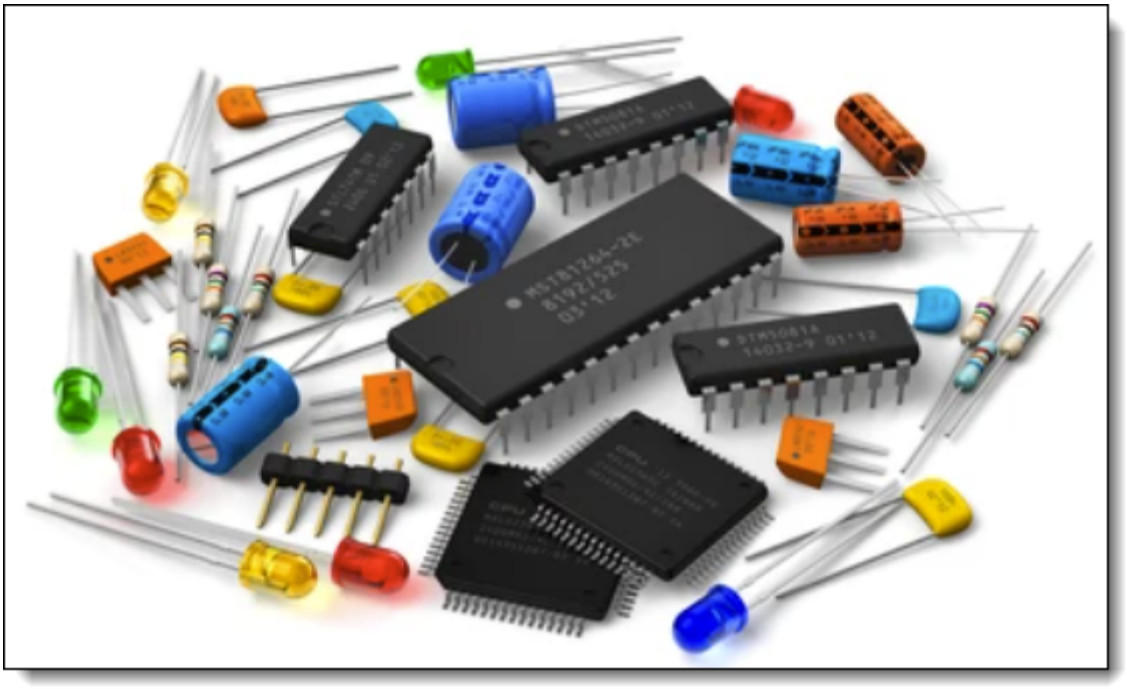
IPC-4101: Specification for Base Materials for Rigid and Multilayer Printed Boards

This standard defines the requirements for the materials used in the manufacture of printed circuit boards. It covers topics such as material properties, testing methods, and performance requirements.

2.4 Part Design

Printed Circuit Boards (PCBs) are essential components of nearly all modern electronic devices. They provide a physical base for mounting and interconnecting electronic components.

Various parts are used in PCBs, including:



A Selection of Parts

- **Resistors** These control the electric current flowing through a circuit by offering resistance.
- **Capacitors:** Capacitors store and release energy. They're essential for filtering noise, tuning resonant circuits, and storing energy as needed.
- **Inductors:** These components can store energy in a magnetic field when electric current flows through them. They're used in a variety of applications, including in filters and transformers.
- **Diodes:** Diodes allow current to flow in one direction but not the other. They are often used for converting AC to DC power (rectification), among other applications.
- **Transistors:** These can amplify and switch electronic signals and electrical power.
- **Integrated Circuits (ICs):** These are a set of electronic circuits on one small flat piece (or "chip") of semiconductor material, usually silicon. ICs can serve functions such as microprocessors, memory, or even complete digital systems.
- **Connectors:** These provide a path for electrical power or signals to flow into and out of the PCB.
- **LEDs:** Light Emitting Diodes provide visual feedback or illumination.

- **Switches:** These allow for manual control over the electric circuit.
- **Quartz Crystal Oscillators:** They generate an electrical signal with a precise frequency. Used for clock signals in microprocessors and to stabilize frequencies for radio transmitters and receivers.
- **Potentiometers:** These are three-terminal resistors with a sliding or rotating contact that forms an adjustable voltage divider.
- **Fuses:** This is a protective device that melts or breaks a circuit when the current flow exceeds a particular limit.

These components, among others, are soldered onto the PCB to create a complete electronic circuit. The specific parts used can vary widely depending on the intended function of the PCB.

2.4.1 What are electronic parts?

Electronic parts are physical components that make up an electronic system or device. They are typically passive or active components that are used to manipulate and control the flow of electrical currents and signals.

Passive electronic parts include resistors, capacitors, inductors, diodes, and transformers. These components do not require a power source to operate, and they are used to control the flow of electrical signals, store electrical energy, and filter or amplify signals.

Active electronic parts include transistors, integrated circuits (ICs), and microcontrollers. These components require a power source to operate, and they are used to amplify, control, and manipulate electrical signals.

Other types of electronic parts include connectors, switches, sensors, relays, and motors. Connectors are used to establish electrical connections between different parts of a system, while switches are used to turn circuits on and off. Sensors are used to detect physical changes or conditions and translate them into electrical signals, and relays are used to switch high voltage or current circuits on and off. Motors are used to convert electrical energy into mechanical energy.

Overall, electronic parts are the building blocks of electronic systems, and their correct selection, integration, and operation are critical to the performance and reliability of electronic devices and equipment.

2.4.2 Passive Parts

Passive electronic parts are components that do not require a power source to operate and can only control or store electrical signals. These parts typically use materials such as ceramic, plastic, or metal to manipulate the flow of electricity.

Examples of passive electronic parts include:

Resistors

These are components that resist the flow of electrical current. They are used in a wide range of electronic circuits to limit current, divide voltage, or control timing.

PCB resistors, often referred to as surface mount resistors or SMD resistors, are electronic components that limit or regulate the flow of electrical current in an electronic circuit.

Resistors are fundamental components in electronics and are found in almost all electronic devices. They're used in a wide range of applications, including setting the gain of an amplifier, dividing voltage, carrying out basic logic operations, and many other functions.

SMD resistors are designed to be mounted on the surface of a PCB. Unlike older, through-hole resistors, which have leads that pass through the board and are soldered on the other side, SMD resistors are soldered directly onto pads on the surface of the board.

SMD resistors come in various standard sizes. The size is usually denoted by a three- or four-digit number, such as 0603, 0805, or 1206. The first two digits represent the length and the last two digits represent the width, both in hundredths of an inch. For example, an 0603 resistor is 0.06 inches long and 0.03 inches wide.

The resistance value of a resistor is usually marked on its body using a numerical code. For SMD resistors, a common system is the EIA-96 code, which uses a three-digit number to represent the resistor's value and tolerance.

It's important to remember that resistors, like other electronic components, should be chosen and used according to the requirements of the specific circuit and application, taking into account factors like the required resistance, power rating, tolerance, temperature coefficient, and other parameters.

Capacitors

These are components that store electrical energy in an electric field. They are used in electronic circuits to smooth out voltage fluctuations, filter out noise, or block direct current.

Capacitors in a Printed Circuit Board (PCB) are electronic components that store and release electrical energy. They play a vital role in various aspects of electronic circuit design, including filtering, decoupling, energy storage, signal coupling or decoupling, electronic noise filtering, and tuning of some communication circuits.

Much like resistors, capacitors come in a variety of types and sizes, with the two most common being through-hole and surface-mount (SMD) capacitors.

Through-Hole Capacitors: These types of capacitors have long leads that can be threaded through the holes on the PCB and then soldered to pads on the other side. Through-hole capacitors were widely used in older electronic devices.

Surface-Mount Capacitors (SMD Capacitors): These types of capacitors are soldered directly onto the surface of the PCB, which makes them a more space-efficient choice. They are the most common type of capacitor found on modern PCBs. They come in standard sizes denoted by a three- or four-digit number, much like SMD resistors.

Capacitors vary widely in their specific properties, including their capacitance value (measured in farads), voltage rating, temperature stability, and equivalent series resistance (ESR). Different types of capacitors (e.g., ceramic, electrolytic, tantalum, etc.) have different properties, which makes them suitable for different applications.

The specific value of a capacitor is typically printed directly on the component, although for smaller SMD capacitors, this may not always be the case due to size limitations.

In a circuit, capacitors can be used for a variety of functions, such as smoothing in power supply applications, filtering in audio circuits, and for timing applications with resistors and capacitors forming RC (resistor-capacitor) circuits.

Inductors

These are components that store electrical energy in a magnetic field. They are used in electronic circuits to filter out high-frequency noise, control current, or store energy.

Inductors on a Printed Circuit Board (PCB) are passive electronic components that store energy in the form of a magnetic field when an electric current is passed through them. They're often used in filter circuits, power supply circuits, and RF (radio frequency) circuits, among other applications.

Inductors are characterized by their inductance value, which is measured in henries (H), and their current rating. The inductance value determines how much energy the inductor can store: the higher the inductance, the more energy stored for a given current. The current rating is the maximum amount of current the inductor can handle without overheating or becoming damaged.

Like resistors and capacitors, inductors can come in through-hole or surface mount (SMD) varieties:

Through-Hole Inductors: These inductors have leads that are inserted into holes on the PCB and soldered to pads on the other side. They're typically larger and used in applications where higher power handling and inductance values are needed.

Surface Mount Inductors (SMD Inductors): These inductors are soldered directly onto the surface of the PCB. They're smaller and used in applications where space is at a premium, like in modern, compact electronic devices.

The physical size and shape of an inductor can vary widely depending on its inductance value, current rating, and intended application. Some inductors are simple coils of wire, while others are wound around a magnetic core to increase their

inductance. There are also air core inductors, ferrite core inductors, and toroidal inductors, among others.

Inductors, because of their ability to oppose changes in current, are often used in circuits that need to smooth out changes in voltage or current, filter out certain frequencies, or store energy for later use. They're a key component in many types of electronic devices, from simple power supplies to complex RF transceivers.

Transformers

These are components that transfer electrical energy from one circuit to another through electromagnetic induction. They are used in electronic circuits to step up or step down voltage, isolate circuits, or match impedance.

Overall, passive electronic parts are critical components of electronic circuits and are used to control, store, or filter electrical signals in a wide range of applications.

Transformers in a Printed Circuit Board (PCB) are passive electrical devices that transfer electrical energy from one circuit to another through the process of electromagnetic induction. They are typically used to increase or decrease voltage levels between circuits, making them essential components in power supply designs.

Transformers consist of two or more coils of wire, known as windings, which are wrapped around a core, often made of ferrite or iron. When alternating current (AC) flows through one winding, it generates a magnetic field in the core, which then induces a voltage in the other winding. The ratio of turns between the primary (input) winding and the secondary (output) winding determines the voltage transformation.

PCB transformers come in a variety of types, including:

Through-Hole Transformers: These are typically larger and are soldered into holes on the PCB. They're used when larger power handling is required.

Surface Mount Transformers (SMD Transformers): These are designed to be soldered directly onto the PCB, making them a more space-efficient choice. They are often used in low-power applications where space saving is critical.

Planar Transformers: These are a type of SMD transformer that use PCB traces for their windings, which can offer better performance and integration but at a higher manufacturing cost.

Transformers used in PCBs can have a wide range of specifications depending on their intended application. They may be designed for different frequency ranges (from audio frequencies to RF), different power levels, different impedance matching requirements, and different isolation requirements, among other factors.

Transformers are used in a variety of applications including power supplies, audio systems, telecommunications, and RF circuits. It's important to choose the correct transformer for your specific application, taking into account factors such as power requirements, operating frequency, and the required voltage transformation ratio.

2.4.3 Active Parts

Active electronic parts are components that require a power source to operate and can control or amplify electrical signals. These parts typically use semiconductor materials, such as silicon, to manipulate the flow of electricity.

Examples of active electronic parts include:

Transistors

These are semiconductor devices that can amplify or switch electronic signals. They are used in a wide range of electronic circuits, including amplifiers, oscillators, and power controllers.

Transistors are semiconductor devices used to amplify or switch electronic signals and electrical power. They are one of the basic building blocks of modern electronic devices and are found in almost all electronic devices. In a Printed Circuit Board (PCB), transistors play critical roles in various circuit functions such as signal amplification, regulation, switching, signal modulation, and many others.

Transistors have three layers of semiconductor material and two pn junctions, hence they are sometimes referred to as a "double junction" device. The three parts of a transistor are the emitter, the base, and the collector.

There are two main types of transistors:

Bipolar Junction Transistors (BJTs): BJTs use both electron and hole charge carriers and are known for their amplification properties. They come in two types, NPN and PNP, which are distinguished by the direction of the current flow when in operation.

Field Effect Transistors (FETs): FETs control the electrical behavior of a device using an electric field. FETs come in many forms including Junction Gate Field-Effect Transistors (JFETs) and Metal Oxide Semiconductor Field-Effect Transistors (MOSFETs). They are widely used in applications like signal amplification and switching.

Transistors, like other electronic components, can be either through-hole or surface mount (SMD) and come in a variety of packages. The type of transistor and its specifications are typically printed on the body of the component, though some smaller SMD transistors might not have this due to their size.

When designing a PCB, it's important to consider the specifications of the transistor, including its maximum current, voltage ratings, gain, frequency response, and power dissipation. These parameters will depend on the application the transistor is being used for.

Diodes

These are components that allow current to flow in one direction while blocking it in the other. They are used in electronic circuits as rectifiers, voltage regulators, and signal demodulators.

Diodes are semiconductor devices used in electronic circuits that allow current to flow in one direction only. They have two terminals, an anode and a cathode. Current can flow from the anode to the cathode, but not in the other direction.

Diodes come in many forms and are used for a wide range of applications in a Printed Circuit Board (PCB). Here are some common types of diodes:

Rectifier Diodes: These are the most basic type of diode and are often used for converting alternating current (AC) to direct current (DC), a process known as rectification.

Schottky Diodes: These diodes have a lower forward voltage drop than regular diodes, which means they waste less energy. They are often used in power supply circuits.

Zener Diodes: These diodes can be designed to allow current to flow in the reverse direction when a specific set voltage (the Zener voltage) is reached. They're often used for voltage regulation.

Light Emitting Diodes (LEDs): These diodes emit light when current passes through them and are used for indication and lighting purposes.

Photodiodes: These diodes generate a current when they are exposed to light. They're used in various light sensing applications.

Diodes can come in both through-hole and surface-mount (SMD) varieties. The orientation of a diode when it's installed on a PCB is crucial, as installing it backwards will prevent the circuit from functioning properly. On schematics and on the diode itself, the cathode is usually marked with a line.

In terms of identifying diodes, the specific type and specifications of a diode are typically indicated by a printed code on the component's body. However, because of their small size, SMD diodes might not always have this code printed on them. Instead, you may need to refer to the manufacturer's datasheet or the PCB's bill of materials to identify them.

Integrated circuits (ICs)

These are miniaturized electronic circuits that can perform multiple functions. They are used in a wide range of electronic devices, including computers, smartphones, and medical equipment.

Integrated Circuits (ICs) are complex assemblies of transistors, diodes, resistors, and capacitors built onto a single piece of silicon, often referred to as a chip. These chips are encapsulated in a package that is then mounted onto a Printed Circuit Board (PCB). ICs are used to perform a wide range of functions in electronic devices.

ICs come in many different types and are used for a wide range of applications on a PCB, including:

Microprocessors and Microcontrollers: These are complex ICs that can perform computations and execute instructions. They are used as the brains of many types of electronic devices, from computers and smartphones to smaller devices like digital watches and microwave ovens.

Memory Chips: These ICs store data. Types of memory chips include RAM (Random Access Memory), ROM (Read Only Memory), and flash memory.

Analog ICs: These ICs process analog signals and can function as amplifiers, filters, and oscillators. Examples include operational amplifiers (op-amps) and voltage regulators.

Digital ICs: These ICs handle digital signals and include logic gates, flip-flops, counters, and shift registers.

Mixed Signal ICs: These ICs contain both analog and digital circuits. They are used in devices like smartphones, where they might handle both the digital processing of data and the analog processing of audio signals.

ICs can be either through-hole or surface mount. The specifications for an IC are usually found in its datasheet, which can often be found online. The datasheet will provide important details about the IC's function, pin configuration, recommended operating conditions, and other specifications.

ICs can come in a variety of packages, including Dual In-line Package (DIP), Small Outline IC (SOIC), Quad Flat Package (QFP), and Ball Grid Array (BGA), among others.

Selecting the correct IC for your application and ensuring that it is correctly implemented in your circuit design is a key aspect of PCB design. It's also important to ensure that the IC is correctly soldered to the PCB and that any necessary heat sinking or cooling is provided, as ICs can generate significant amounts of heat during operation.

Microcontrollers

These are specialized ICs that can control the operation of electronic systems. They are used in a wide range of applications, including automotive, industrial, and consumer electronics.

Microcontrollers are a type of integrated circuit (IC) that contain a processor core, memory, and programmable input/output peripherals. Essentially, they are a computer on a single chip designed to control electronic devices.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, remote controls, office machines, appliances, power tools, toys, and other embedded systems. They are the "brains" behind many smart devices and are ubiquitous in the modern world.

Most microcontrollers are designed to execute complex digital instructions for specific tasks. A single microcontroller can contain a processor, memory (both RAM and ROM), and input/output (I/O) interfaces for connecting with other components or devices.

Different microcontrollers offer different sets of peripherals like digital I/O pins, analog-to-digital converters (ADCs), digital-to-analog converters (DACs), UARTs (serial ports), I2C or SPI interfaces, pulse-width modulation (PWM) capabilities, and even built-in Wi-Fi or Bluetooth in some cases.

Microcontrollers are programmable, which means that they can be loaded with code that determines how they operate. This code can be written in a variety of programming languages, including assembly language, C, C++, and in some cases, higher-level languages like Python or JavaScript. The code is typically written on a separate computer, then loaded onto the microcontroller using a programmer or a serial interface.

Microcontrollers come in various packages, from traditional through-hole Dual In-line Package (DIP) to various Surface Mount Technology (SMT) packages like Quad Flat Package (QFP), Ball Grid Array (BGA), and many others.

Popular families of microcontrollers include the PIC series from Microchip, the AVR series from Atmel (including the ATmega series used in Arduino boards), the MSP430 series from Texas Instruments, and the ARM Cortex series from ARM Holdings, among others.

When designing with microcontrollers, it's crucial to consider factors such as processing power, memory size, power consumption, availability of needed peripherals, cost, and the availability of software tools and libraries to support development.

Operational amplifiers (op-amps)

These are high-gain voltage amplifiers that can be used to amplify and manipulate electronic signals. They are used in a wide range of electronic circuits, including filters, oscillators, and power controllers.

Operational amplifiers, often abbreviated as op-amps, are a type of integrated circuit (IC) that can amplify voltage. They are one of the most common types of ICs used in analog electronics and can be used in a wide variety of electronic devices.

Op-amps have two input pins and one output pin. The two input pins are labeled as the inverting input (usually denoted with a minus sign) and the non-inverting input (usually denoted with a plus sign). The output voltage of the op-amp is a multiplication of the difference between the voltages applied to the two input pins and the gain of the op-amp.

Op-amps are versatile devices and can be used in a wide variety of circuits, including amplifiers, filters, comparators, and oscillators, among others. The specific function of an op-amp in a circuit depends on how the circuit is designed.

There are many different types of op-amps, each with its own specifications and characteristics. Some are designed for general use, while others are designed for specific applications, such as audio amplification, precision measurements, or high-speed signal processing. The specifications of an op-amp, including its gain, bandwidth, noise performance, and power consumption, will vary depending on its design.

Op-amps can be either through-hole or surface-mount, and they come in a variety of packages, including Dual In-Line Package (DIP), Small Outline Integrated Circuit (SOIC), and many others.

When designing a circuit that uses an op-amp, it's important to select an op-amp that meets the requirements of the circuit in terms of its specifications. It's also important to design the circuit correctly to ensure that the op-amp operates in its linear region (where it can accurately amplify signals), unless it's being used in a non-linear application such as a comparator or an oscillator.

Overall, active electronic parts play a crucial role in modern electronic devices and equipment, and their correct selection and integration are critical to the performance and reliability of electronic systems.

2.4.4 Virtual Parts

In electronic design, a virtual part in a schematic refers to a component or device that is not physically present in the circuit but exists virtually to represent a specific function, placeholder, or abstraction. Virtual parts are often used in schematic diagrams to simplify complex designs, create hierarchical representations, or denote functionalities that are implemented at the system level.

Here are some common uses and examples of virtual parts in schematics:

Function Blocks: In complex designs, certain functions might be implemented as separate blocks that perform specific tasks but are not individual physical components. These functional blocks can be represented as virtual parts in the schematic.

Subcircuits: In hierarchical designs, subcircuits can be represented as virtual parts. These subcircuits might be designed on separate sheets or stored as separate libraries to be reused across different projects.

Software Components: In mixed-signal designs or systems that incorporate microcontrollers or digital signal processors, software components can be represented virtually to indicate their interaction with the hardware.

System-Level Abstractions: At the system level, certain components or subsystems may be abstracted to simplify the schematic and focus on higher-level connections and relationships.

Reference Components: Virtual parts can be used as placeholders for components that will be defined later in the design process or for components that are selected based on specific project requirements.

Simulation Models: Virtual parts can represent simulation models, such as behavioral models or models of external devices, to predict and analyze the system's behavior.

Custom Symbols: In some cases, designers may create custom symbols for a group of components or a specific function, treating them as virtual parts.

Using virtual parts in schematics allows designers to create more organized and manageable circuit representations, especially in large and complex designs. Virtual parts can help improve readability, simplify design reuse, and provide a higher-level perspective of the overall system.

While virtual parts are essential for schematic organization, it's crucial to ensure that their purpose is well-documented and understood by all team members involved in the design process. Proper documentation will prevent confusion and ensure that the virtual parts are correctly implemented during layout and manufacturing stages.

2.4.5 What are schematic symbols?

Schematic symbols are graphical symbols used in a schematic diagram to represent electronic components and their connections. These symbols are used to convey information about the function and properties of a component, as well as its electrical connections to other components in a circuit. Schematic symbols are standardized across the industry, meaning that a given symbol will typically represent the same component regardless of who created the schematic.

Schematic symbols can represent a wide variety of electronic components, including resistors, capacitors, inductors, diodes, transistors, integrated circuits, and many others. The symbols are typically designed to be simple and easy to recognize, while still conveying important information about the component's function and properties.

In addition to representing individual components, schematic symbols can also be used to represent connections between components, such as wires, buses, and signal paths. By using these symbols to create a schematic diagram, engineers and designers can create a clear and concise representation of the circuit, which can be used to guide the PCB design process and ensure the proper operation of the electronic device.

Printed design schematic symbols are used to depict the various components of an electrical or electronic system. They can range from simple shapes like circles and

squares to complex combinations of lines and curves. These symbols are used on circuit diagrams, PCB layouts and other design documents to represent physical elements such as resistors, capacitors, diodes, transistors, and integrated circuits. The symbols also represent connections between these components and the power source.

The use of printed circuit design schematic symbols has increased significantly in recent years due to advances in technology. With the advent of high-density integrated circuits (ICs) and more complex circuit designs, symbols have become more sophisticated and intricate. Some modern ICs contain hundreds of individual components that need to be clearly identified on a schematic diagram so that engineers can quickly comprehend their function within the overall system.

Symbols used to represent electronic components generally conform to industry standards laid out by organizations such as IEC (International Electrotechnical Commission). This ensures consistency across different schematics produced by different manufacturers, making it easier for engineers to read them without requiring additional explanation or knowledge about each symbol's specific meaning. For example; a resistor will always be represented with a zig-zag line whereas an LED will always be depicted with two arrows pointing in opposite directions indicating its light emitting properties.. This helps reduce confusion when decoding a circuit diagram which is particularly useful when troubleshooting issues in an electronic system.

Another aspect of printed design schematic symbols is their ability to communicate information beyond simply identifying a component's function but also its orientation on a board or relative position within the overall system. This can help engineers better understand the interaction between components and allow them make informed decisions during development or debugging stages. Symbols may show polarity or specific pin number assignments for components that require them, as well as highlighting connections or paths throughout the circuit diagram with dotted lines or arrows.

Overall printed design schematic symbols are essential for understanding how different electrical or electronic systems are wired together and how they interact with one another. Without clear visual representations for each component it would be difficult for engineers to accurately diagnose faults or identify potential areas for optimization within a given device or system architecture.

2.4.6 What are schematic symbol terminals?

Schematic symbol terminals, also known as pins or connectors, are the small metal contacts that appear on a schematic symbol of an electronic component. These terminals are used to represent the electrical connections between components in a circuit diagram or schematic.

Each terminal on a schematic symbol corresponds to a physical pin on the actual component. For example, a resistor symbol may have two terminals, which represent the two leads of the resistor. The terminals are usually labeled with letters, numbers, or other symbols that indicate the function or purpose of the connection.

When designing a circuit, the schematic symbol terminals are used to show how the component is connected to other components in the circuit. For example, the terminal on a transistor symbol that represents the emitter might be connected to the ground, while the collector terminal might be connected to the power supply.

It is important to correctly label the terminals on a schematic symbol, as this ensures that the circuit diagram accurately represents the physical connections of the components in the actual circuit.

2.4.7 Schematic Symbol Design

Schematic symbol design is the process of creating symbols that represent electronic components on a schematic diagram. A schematic diagram is a graphical representation of an electronic circuit, and it shows the connections between components and their electrical functions. Here are some key considerations when designing schematic symbols:

1. *Representation of the component:* The symbol should accurately represent the physical appearance and electrical function of the component. The symbol should include all necessary pins or connection points, as well as any other relevant information, such as part numbers, values, or ratings.
2. **Consistency:** Symbols should be consistent in appearance and function across all circuits and designs. This makes it easier for designers to quickly identify and use components, and it reduces the risk of errors or confusion.
3. **Clarity and simplicity:** Symbols should be easy to understand and interpret. They should be simple and unambiguous, with clear and concise labels and annotations.
4. **Standardization:** Symbols should conform to recognized standards, such as those established by the Institute of Electrical and Electronics Engineers (IEEE) or the International Electrotechnical Commission (IEC). This ensures compatibility and interoperability with other designs and systems.
5. **Design rules:** Symbols should follow specific design rules and guidelines for readability, scalability, and manufacturability. These rules may include minimum line thicknesses, font sizes, and spacing requirements.

Schematic symbols are typically created using specialized software, such as EDA (electronic design automation) software. These tools provide a library of pre-made symbols for common components, as well as tools for creating custom symbols. It is

important to test schematic symbols for accuracy and functionality before using them in a design.

2.4.8 The Part's Datasheet

Electronic PCB (Printed Circuit Board) part datasheets are documents provided by manufacturers that contain detailed information about a specific component or part. These datasheets are crucial in the design and manufacturing of PCBs as they provide engineers with everything they need to know about the component.

Here are some of the details that you can typically find in a datasheet:

General Description: This section gives an overview of what the part is and what it does.

Key Features and Specifications: This includes electrical, mechanical, and thermal characteristics of the part, such as voltage, current, power, temperature ranges, etc.

Pin Configuration and Functions: Details about each pin of the component, its purpose, and how it should be connected in the circuit.

Functional Diagrams and Schematics: Diagrams to help better understand the internal structure and operation of the component.

Performance Graphs: Charts and graphs that show how the part behaves under different conditions.

Application Information: Suggestions and guidance on how to use the part, often with example circuits.

Package Information: Physical dimensions, layout, and packaging details. This information is crucial for PCB layout design.

Ordering Information: Details about different variants of the part and how to order them.

These datasheets are standardized to some extent, but there can be variations depending on the manufacturer and the type of component. The datasheet is an essential tool for engineers when designing and troubleshooting circuits, and is typically the first place to look when working with a new component.

The parts datasheets are normally Adobe PDF files.

Adobe PDF (Portable Document Format) is a file format developed by Adobe Systems that is used to represent documents in a manner independent of application software, hardware, and operating systems. Each PDF file encapsulates a complete description of a fixed-layout flat document, including the text, fonts, graphics, and other information needed to display it.

Key features of PDF files include:

Platform Independence: PDF files can be opened and viewed on any device (computer, smartphone, tablet) regardless of the operating system (Windows, macOS, Linux, Android, iOS), as long as a PDF reader software is installed.

Maintain Format: When you create a PDF file, it will look the same on any device, retaining its formatting, fonts, and layout. This is particularly useful for documents like forms, invoices, or any document where layout and presentation are important.

Security: PDF files can be secured with a password and can also have restrictions placed on them to prevent actions like copying, editing, or printing.

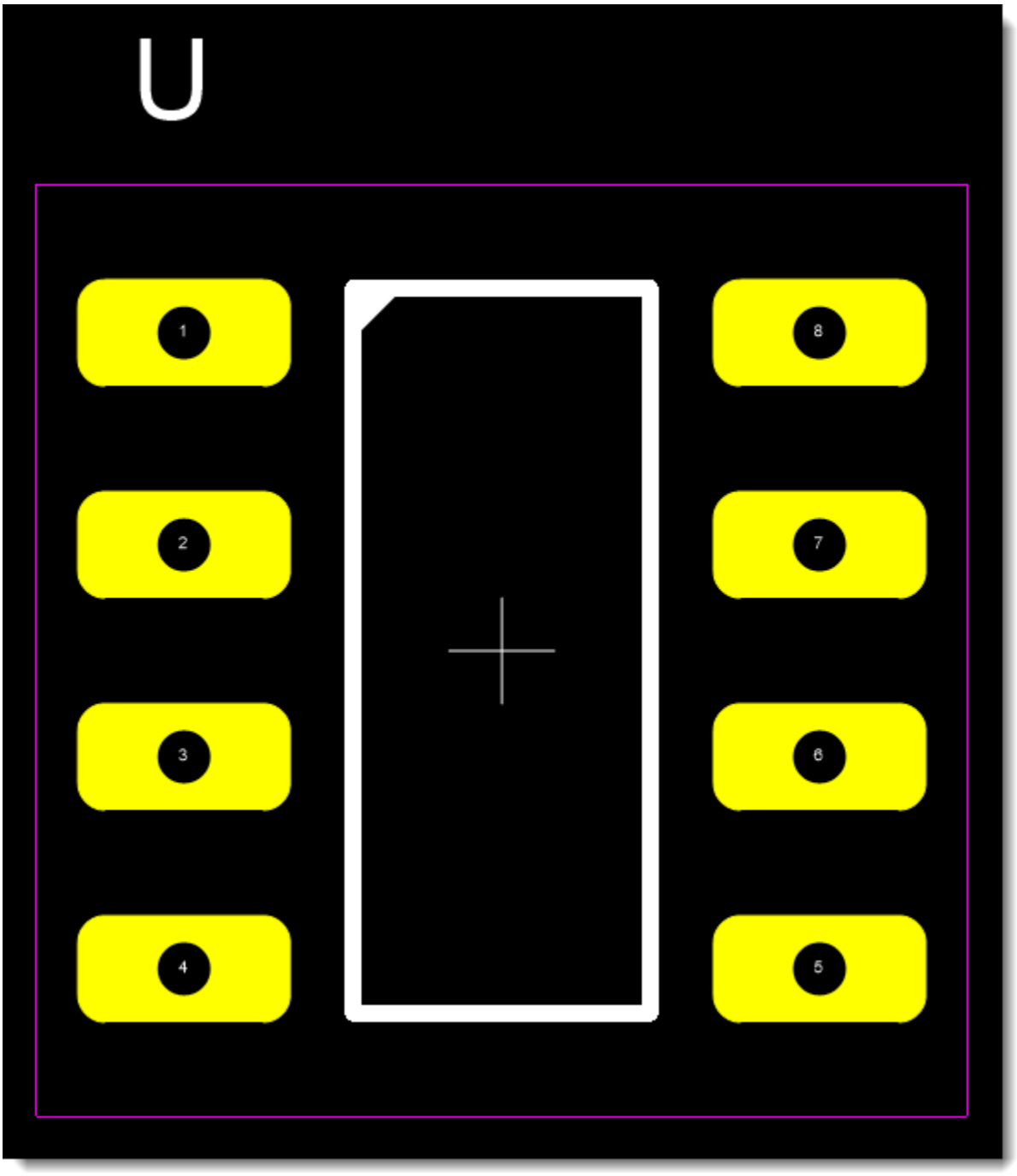
Searchable Text: The text within a PDF is searchable. This makes it easy to find specific information in large documents.

Hyperlinks, Multimedia, and Interactivity: PDFs can contain hyperlinks, can be integrated with multimedia (like audio and video files), and can include interactive elements like forms and buttons.

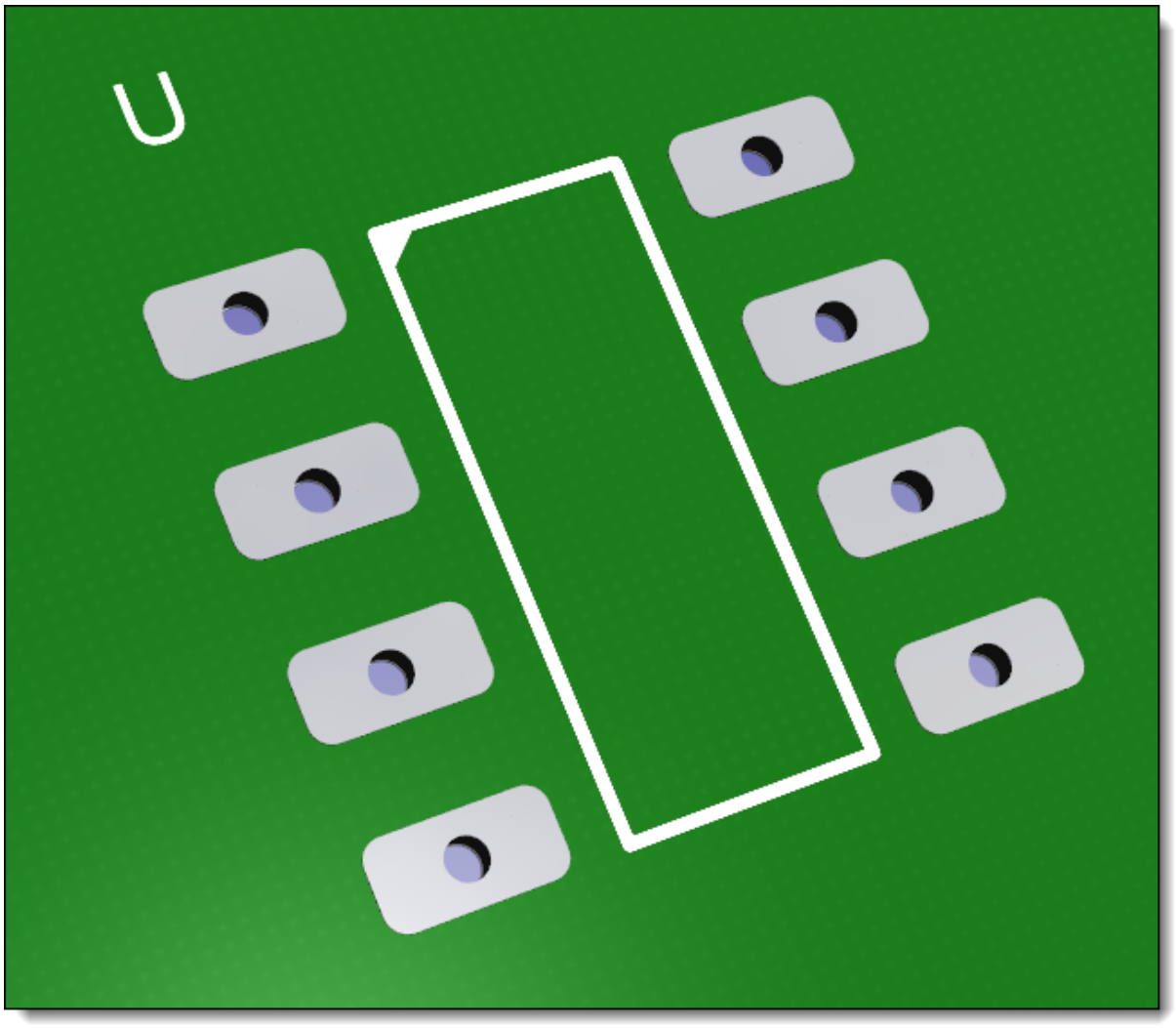
Compression: Large documents and files can be compressed into a PDF, making them easier to share and distribute.

2.4.9 What are footprints?

PCB land patterns, also known as footprints, are the physical layouts of copper pads and other features on a Printed Circuit Board (PCB) that are used to connect electronic components. These patterns are designed to match the physical dimensions and pin configurations of specific electronic components, and they are typically created using a specialized software tool.



PCB Footprint for a 8-Pin DIP



Footprint for a 8-Pin DIP Viewed in 3D

PCB land patterns are important because they help ensure that components can be securely mounted on the PCB and that their electrical connections are properly established. The size and shape of the copper pads, as well as their spacing and orientation, are all critical factors that must be carefully considered when designing the land pattern for a given component.

There are many different types of PCB land patterns, each of which is designed to match the specific requirements of a particular component. Some common types of land patterns include through-hole, surface-mount, and ball grid array (BGA) patterns. These patterns can be designed to accommodate a wide range of component sizes and configurations, making them an essential element of the PCB design process.

2.4.10 What are footprint land pattern pads?

Footprint, land pattern, and pads are all terms used in PCB design to refer to the physical layout of the copper pads and holes that are used to mount and connect electronic components to the board.

A footprint, also called a land pattern or package, is the set of copper pads and holes that define the physical layout and dimensions of an electronic component on a PCB. Footprints are designed to match the specific size, shape, and pin spacing of a particular component, and are created using specialized software tools.

Pads are the small, circular or rectangular copper areas on the surface of a PCB that connect the component's leads to the traces on the board. Pads can be round, oval, rectangular, or any other shape that is appropriate for the component being used.

The holes on a PCB are used to mount the component to the board using pins or through-hole soldering. The size and shape of the holes are determined by the size and shape of the component leads.

In summary, footprint, land pattern, and pads are all terms used in PCB design to describe the physical layout of the copper pads and holes that are used to mount and connect electronic components to the board. A footprint defines the specific size, shape, and pin spacing of a component, while pads are the small copper areas on the board that connect the component leads to the traces on the board.

2.4.11 What is a PCB silkscreen?

A PCB silkscreen is a layer of markings that is printed on the surface of a printed circuit board (PCB). The silkscreen layer contains text, symbols, and other graphical information that is used to identify and label the components, test points, and other features on the board.

The silkscreen layer is typically printed in white ink on top of the solder mask layer, which is the layer that covers most of the copper traces and pads on the board. The silkscreen layer is used to provide information about the board's components, such as the part numbers, values, and reference designators. It may also include other information such as logos, warnings, and instructions.

Some common silkscreen markings include:

Reference designators: These are letters and numbers that are used to identify the components on the board. For example, "R1" might indicate a resistor, while "C2" might indicate a capacitor.

Part numbers: These are the manufacturer's part numbers for the components on the board.

Test points: These are locations on the board where test equipment can be attached to measure voltages or currents.

Board name and revision: This identifies the name and revision number of the board.

Other markings: These may include logos, warning symbols, and other instructions or notes.

The silkscreen layer is an important part of the PCB design, as it helps to ensure that the board can be assembled and tested correctly. By providing clear and accurate information about the components and other features on the board, the silkscreen layer can help to prevent errors and ensure that the board functions as intended.

2.4.12 What is a PCB courtyard?

In printed circuit board (PCB) design, a courtyard is a reserved space on the board that is used for component placement and orientation. The courtyard is typically a rectangular or square area that is defined by a set of guidelines or dimensions provided by the PCB designer or manufacturer.

The purpose of the courtyard is to provide a clear and consistent space on the board for each component. By defining a standard size and shape for the courtyard, designers can ensure that each component is placed and oriented correctly on the board, and that there is adequate space for assembly and inspection.

The courtyard may also include other design elements, such as reference designators, component outlines, and polarity markings. These elements help to ensure that each component is correctly identified and placed on the board.

2.4.13 What are footprint reference designators?

Footprint reference designators are the unique labels used to identify and reference individual components on a printed circuit board (PCB). They are typically a combination of letters and numbers that identify the specific component and its location on the board.

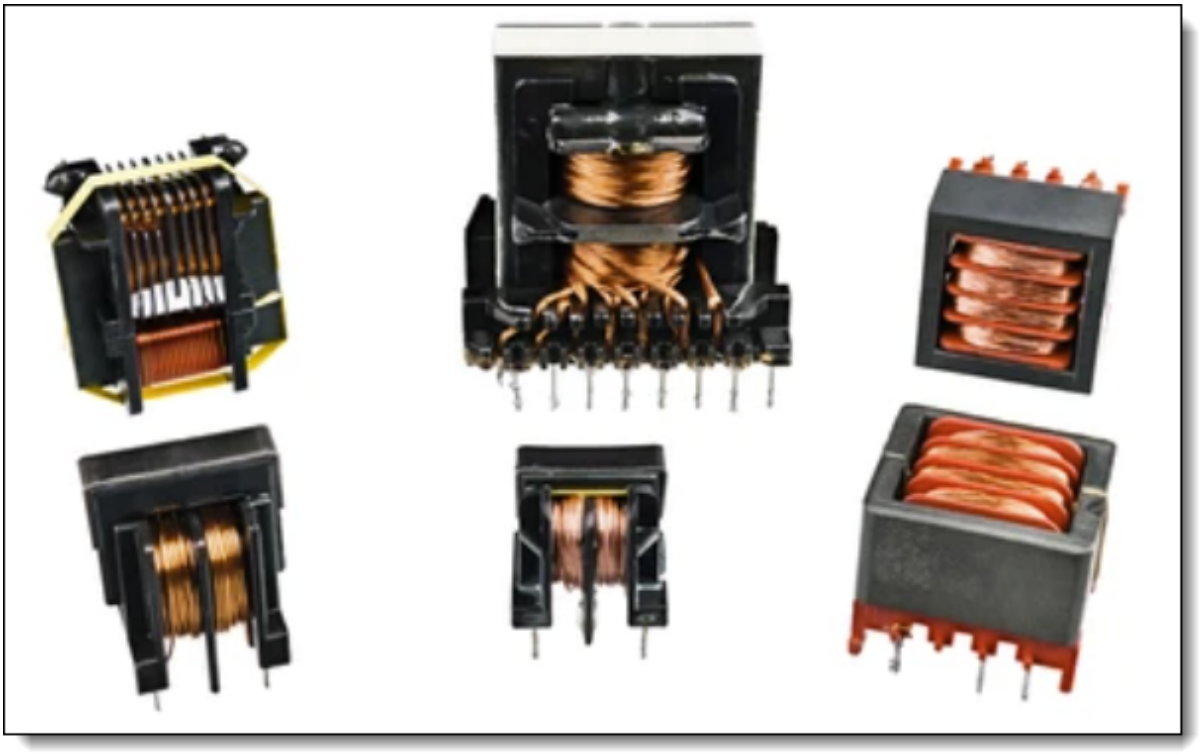
Footprint reference designators are assigned to components during the PCB design process. Each component is given a unique label that identifies its type and location on the board. For example, "R1" might indicate a resistor located in the top left corner of the board, while "C2" might indicate a capacitor located near the center of the board.

The reference designator is typically printed on the silkscreen layer of the PCB, which is the layer of text and symbols that is printed on the surface of the board. The silkscreen layer helps to identify and label the components on the board, and is an important part of the PCB design process.

In summary, footprint reference designators are the unique labels used to identify and reference individual components on a printed circuit board. They are assigned during the PCB design process, and are typically printed on the silkscreen layer of the board. By using reference designators, designers and manufacturers can ensure that each component is correctly identified and located on the board.

2.4.14 PCB Transformers

A PCB (Printed Circuit Board) transformer is a particular type of electrical transformer that is designed to be mounted directly onto a circuit board. These transformers are used to step-up, step-down, or isolate electrical voltages within the circuit board.



PCB Transformers

Transformers operate on the principle of electromagnetic induction. They consist of two or more coils of wire (called windings) wrapped around a core made of ferromagnetic material. By varying the number of turns in the windings, transformers can increase or decrease the voltage.

Here's a bit more about different types of PCB transformers:

Step-up transformers

These transformers are used to increase the voltage from the primary winding to the secondary winding. They have more turns on the secondary winding than the primary.

A step-up transformer is a type of electrical transformer that increases the voltage from the primary coil (input) to the secondary coil (output). It's called a "step-up" transformer because it steps up, or increases, the voltage while decreasing the current. The power (the product of voltage and current) remains the same, neglecting losses due to the transformer's efficiency.

The transformation ratio of a transformer, i.e., the ratio of the output voltage to the input voltage, is directly proportional to the ratio of the number of turns in the secondary coil to the number of turns in the primary coil. Therefore, a step-up transformer will have more turns of wire on its secondary coil than its primary coil.

Step-up transformers are commonly used in several applications, including:

Power transmission: In electrical power grids, step-up transformers are used at the power station to increase the voltage for transmission. High voltage allows for efficient long-distance transmission because it reduces the amount of energy lost as heat due to resistance in the wires.

Electronics Devices: They are also used in various electronic devices to increase voltage as required by specific circuits or components.

X-ray and CT machines: These medical devices use step-up transformers to produce the high voltages necessary to generate X-rays.

Remember that while step-up transformers increase voltage, they also decrease current. This is due to the conservation of energy (assuming ideal conditions with no energy losses), which in the context of transformers is often referred to as "impedance matching". The product of voltage and current (which is power) stays the same in both the primary and secondary coils.

Step-down transformers

These transformers decrease the voltage from the primary winding to the secondary winding. They have fewer turns on the secondary winding than the primary.

A step-down transformer is a type of electrical transformer that reduces the voltage from the primary coil (input) to the secondary coil (output). As the name suggests, it "steps down" the voltage. While it reduces the voltage, it increases the current. The power (the product of voltage and current) remains the same, assuming an ideal transformer with no energy losses.

The transformation ratio of a transformer, i.e., the ratio of the output voltage to the input voltage, is directly proportional to the ratio of the number of turns in the secondary coil to the number of turns in the primary coil. Therefore, a step-down transformer will have fewer turns of wire on its secondary coil than its primary coil.

Step-down transformers are used in a variety of applications, including:

Power distribution: In electrical power grids, step-down transformers are used to reduce the high transmission voltages down to safer levels that can be used in homes and businesses.

Electronics Devices: They are also used in various electronic devices to decrease voltage to levels suitable for specific circuits or components.

Chargers: Many chargers for devices like laptops and smartphones include a step-down transformer to reduce the mains voltage to a level suitable for the device.

Remember that while step-down transformers decrease voltage, they increase current. This is due to the conservation of energy (assuming ideal conditions with no energy losses), which in the context of transformers is often referred to as "impedance matching". The product of voltage and current (which is power) stays the same in both the primary and secondary coils.

Isolation transformers

These transformers have an equal number of turns on the primary and secondary windings, so they maintain the same voltage. However, they are used to electrically isolate different parts of the circuit, which can help to improve safety and reduce noise.

PCB transformers are commonly used in power supplies, audio systems, and many other applications. They must be designed to meet the specific power, voltage, and space requirements of the PCB. Also, due to their proximity to other components on the board, heat dissipation and electromagnetic interference are important factors to consider in their design and placement.

An isolation transformer is a type of transformer used to transfer electrical power from a source of alternating current (AC) power to some equipment or device while isolating the powered device from the power source, usually for safety reasons.

The primary and secondary windings of an isolation transformer have the same number of turns, meaning the voltage in and out is the same (assuming ideal conditions). But the key feature of an isolation transformer is the isolation it provides - the primary (input) and secondary (output) sides are not directly connected.

Isolation transformers have several important uses:

Safety: By providing galvanic isolation (a lack of direct electrical connectivity), an isolation transformer prevents the possibility of a shock if a person touches a live part of the circuit.

Noise reduction: They can reduce electrical noise and spikes, which can be useful in sensitive electronics like audio and medical equipment.

Preventing ground loops: A ground loop is an unwanted current that flows in a conductor connecting two points that are supposed to be at the same potential but are actually at different potentials. This can cause interference or noise in audio or data signals. An isolation transformer can help prevent ground loops.

Testing equipment: They are used in oscilloscopes and other test equipment to prevent any unintentional creation of a ground path.

In summary, isolation transformers are a critical tool for both safety and the effective functioning of many types of electronic equipment.

2.4.15 PCB Potentiometers

A PCB Transformer

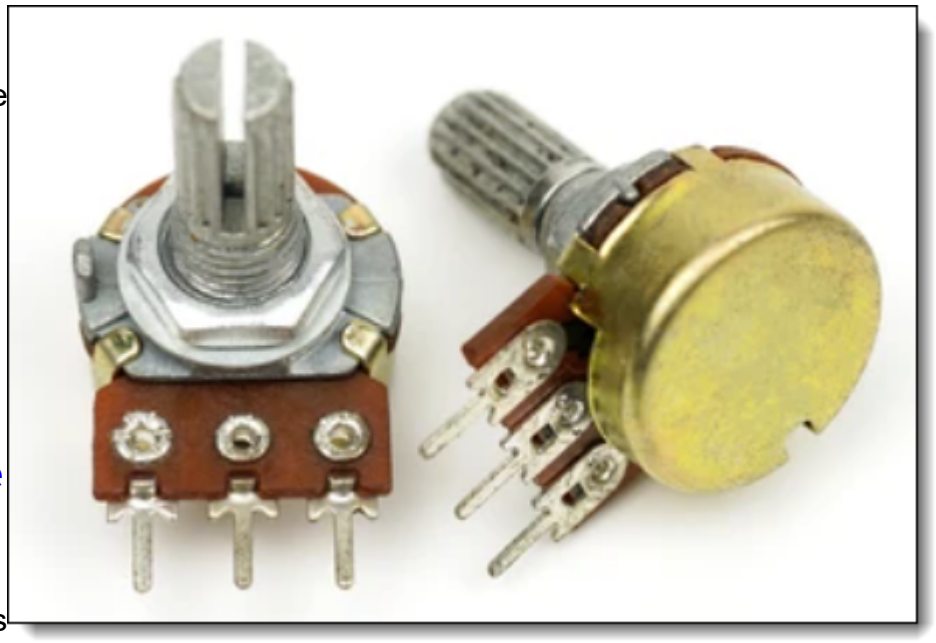
Potentiometers, often referred to as pots, are a type of resistor with a third terminal that forms an adjustable voltage divider. They are used to measure electromotive force or to control electrical devices such as volume controls on audio equipment. Potentiometers can be used as a rheostat (two-terminal variable resistor), or as a true potentiometer (three-terminal device).

When it comes to Printed Circuit Boards (PCBs), PCB potentiometers are simply potentiometers designed to be mounted on these boards. They come in many different forms, sizes, and complexities, but they all function by changing the resistance value in a circuit.

The two most common types are:

Rotary Potentiometers

These have a rotating shaft that you turn to change resistance. They are often used in applications such as audio volume control or light dimmer switches.



Two PCB Transformers

Slide or Linear Potentiometers

These have a slider which moves along a linear path to change resistance. They are often used in mixing boards and light controls.

Adjustments to the potentiometer modify the flowing current, enabling fine-tuning of the connected electrical devices. For instance, a potentiometer might be used on a PCB inside a radio to adjust the volume, or in a light fixture to adjust the brightness of the light. The specific design and use of a PCB potentiometer can vary widely depending on its intended application.

2.4.16 USB PCB Sockets

USB PCB sockets, also known as USB connectors or USB ports, are components used in electronic devices to provide a standardized interface for connecting and transferring data between devices. These sockets are typically mounted on printed circuit boards (PCBs) and are designed to accept USB cables, allowing for the easy connection of various peripherals and devices.

There are several types of USB connectors, each with its own specifications and use cases. Some common USB connector types include:

USB Type-A Sockets

USB Type-A sockets are one of the most common types of USB connectors used in electronic devices. They are typically found on computers, laptops, USB hubs, and many other devices. USB Type-A sockets are used to connect a wide range of peripherals, such as keyboards, mice, printers, external hard drives, and USB flash drives, to a host device.

Here are some key features and characteristics of USB Type-A sockets:

- **Physical Design:** USB Type-A sockets have a rectangular shape with four pins inside. They are designed for a single-directional connection, which means they have a specific orientation that must be aligned correctly for insertion.
- **Data Transfer:** USB Type-A sockets support various USB standards, including USB 2.0, USB 3.0 (also known as USB 3.1 Gen 1), USB 3.1 (also known as USB 3.1 Gen 2), and USB 3.2. The data transfer rates vary based on the USB version. For instance, USB 2.0 offers lower data transfer speeds compared to USB 3.0 and later versions.
- **Charging:** USB Type-A sockets can also be used for charging devices, providing power to recharge smartphones, tablets, and other gadgets. The charging capabilities depend on the USB standard and the power delivery specifications of the host device.
- **Compatibility:** USB Type-A connectors are not reversible, meaning they have a specific orientation for insertion. This can sometimes lead to frustration when trying to plug in a device in the dark or without looking.
- **Variants:** USB Type-A connectors come in two variants: Standard-A and Mini-A. Standard-A is the larger, rectangular connector found on most computers and laptops. Mini-A is a smaller variant that was used in some early devices but is less common today.
- **Mechanical Durability:** USB Type-A connectors are known for their robust mechanical design, which makes them suitable for frequent insertion and removal.

When incorporating USB Type-A sockets into a PCB design, it's important to follow recommended design guidelines to ensure proper signal integrity and reliability. This includes considerations for PCB layout, signal routing, grounding, and ESD protection to minimize potential issues with data transfer and connectivity.

It's worth noting that USB technology continues to evolve, and while USB Type-A sockets remain widely used, newer USB connector types like USB Type-C offer additional features, such as reversible orientation and higher data transfer rates. As a result, some modern devices are transitioning to USB Type-C for improved user experience and versatility.

USB Type-A sockets

USB Type-B Sockets

USB Type-B sockets are another common type of USB connector used in electronic devices. They are typically found on peripherals like printers, scanners, external hard drives, and other devices that need to be connected to a computer or host device. USB Type-B sockets are used to establish data communication and, in some cases, power delivery between the peripheral device and the host.

Here are some key features and characteristics of USB Type-B sockets:

Physical Design: USB Type-B sockets come in a variety of shapes and sizes, but they generally have a square or trapezoidal shape with a unique design of pins inside. They are designed to fit corresponding USB Type-B plugs and cables.

- **Variants:** There are several variants of USB Type-B connectors, including Standard-B, Mini-B, and Micro-B. Each variant is designed for specific use cases and device sizes.
  - **Standard-B:** This is the larger version of the Type-B connector and is commonly used in printers, scanners, and other larger peripherals.
  - **Mini-B:** Mini-B connectors are smaller and were commonly used in portable devices such as digital cameras, MP3 players, and mobile phones (before the advent of USB Type-C).
  - **Micro-B:** Micro-B connectors are even smaller and were introduced to accommodate the trend toward smaller and thinner devices like smartphones and tablets.
- **Data Transfer:** USB Type-B sockets support various USB standards, including USB 2.0, USB 3.0 (also known as USB 3.1 Gen 1), USB 3.1 (also known as USB 3.1 Gen 2), and USB 3.2. The data transfer rates depend on the USB version and the specific device's capabilities.
- **Charging:** USB Type-B connectors are also used for charging in some cases, especially for devices that don't require high power delivery. However, newer devices and standards (such as USB Power Delivery) often use USB Type-C connectors for enhanced charging capabilities.
- **Orientation:** Like USB Type-A, USB Type-B connectors are not reversible and require correct alignment for insertion.
- **Mechanical Durability:** USB Type-B connectors are designed to withstand frequent insertion and removal, making them suitable for devices that may be connected and disconnected frequently.

When designing PCBs with USB Type-B sockets, it's important to consider factors such as proper PCB layout, signal integrity, grounding, and ESD protection to ensure reliable data transfer and connectivity. Designers should follow recommended guidelines provided by USB standards organizations to ensure the best performance and compatibility with various devices.

It's worth noting that USB technology has evolved, and newer connector types like USB Type-C offer additional features, including reversible orientation and enhanced power delivery, which can be advantageous in certain applications.

USB Type-C Sockets

USB Type-C sockets are a modern and versatile type of USB connector that has gained widespread adoption in recent years. USB Type-C connectors offer several benefits over previous USB connector types, such as reversible orientation, faster

data transfer rates, increased power delivery, and support for various protocols beyond USB, such as Thunderbolt 3 and DisplayPort.

Here are key features and characteristics of USB Type-C sockets:

- **Reversible Design:** One of the most notable features of USB Type-C is its reversible design, which means that the connector can be inserted in either orientation, eliminating the frustration of trying to insert the connector the "right" way.
- **Size and Compatibility:** USB Type-C connectors are smaller and more compact compared to previous USB connector types like Type-A and Type-B. This compact size makes them suitable for a wide range of devices, including smartphones, laptops, tablets, and accessories.
- **Data Transfer:** USB Type-C supports various USB standards, including USB 2.0, USB 3.0 (also known as USB 3.1 Gen 1), USB 3.1 (also known as USB 3.1 Gen 2), and USB 3.2. These standards offer faster data transfer rates compared to older USB versions.
- **Power Delivery:** USB Type-C supports USB Power Delivery (USB PD), which allows for higher power delivery and faster charging. This feature is particularly useful for charging laptops, smartphones, and other power-hungry devices.
- **Alternate Modes:** USB Type-C connectors can support alternate modes beyond USB data transfer and charging. For example, they can carry DisplayPort or HDMI video signals, allowing devices to output video directly through the USB-C port.
- **Thunderbolt 3 Compatibility:** Many USB Type-C connectors also support Thunderbolt 3 technology, which enables even faster data transfer rates and the ability to connect multiple high-resolution displays, external GPUs, and other high-performance peripherals.
- **Mechanical Durability:** USB Type-C connectors are designed to withstand repeated insertion and removal, making them suitable for devices that require frequent connectivity changes.
- **Adapter Compatibility:** USB Type-C to legacy adapters are available, allowing devices with USB Type-C ports to connect to older USB devices using Type-A, Type-B, or other connectors.

When incorporating USB Type-C sockets into a PCB design, it's essential to follow recommended design guidelines to ensure proper signal integrity, power delivery, and compatibility. This includes considerations for PCB layout, routing, grounding, and ESD protection.

USB Type-C has become the standard for many modern devices, including smartphones, laptops, tablets, and accessories, due to its versatility and enhanced capabilities. As a result, USB Type-C is becoming increasingly prevalent across various industries and applications.

USB Mini and Micro Connectors

These smaller connectors are commonly used in compact devices like smartphones, digital cameras, and portable hard drives.

Designing PCBs with USB Sockets

When designing PCBs with USB sockets, it's important to consider factors such as the type of USB connector, its mechanical dimensions, placement on the PCB, and electrical considerations like signal integrity and power delivery. USB connectors should be placed following design guidelines to ensure proper functionality and reliable connections.

It's worth noting that USB technology and standards have evolved over time, and new versions with improved data transfer rates and power delivery capabilities have been introduced. Designers should be aware of the specific USB standard (such as USB 2.0, USB 3.0, USB 3.1, USB 3.2, and USB 4) that their chosen connector supports and ensure copatibility with the intended usage scenario.

When working with USB PCB sockets, it's advisable to refer to the official USB specifications and guidelines provided by organizations such as the USB Implementers Forum (USB-IF) to ensure proper implementation and compatibility with various device

2.4.17 PCB Switches

Here are some common types of PCB switches:

Tactile Switches

PCB tactile switches are a type of electronic switch used on printed circuit boards (PCBs) that provide tactile feedback when pressed. Tactile switches are commonly used in various electronic devices and applications where a user interface is required. They offer a distinct physical sensation and audible click when activated, which helps users confirm that a button press has been registered.

Here are some key features and considerations related to PCB tactile switches:

- **Tactile Feedback:** Tactile switches are designed to give users a tactile sensation when pressed. This feedback helps users know that the switch has been activated, which can be particularly important in applications where precise input or control is required.
- **Actuation Force:** Tactile switches have a specified actuation force, which is the amount of force required to press the switch and trigger its operation. The actuation force can vary between different switch models and is an important consideration for the user experience.

- **Travel Distance:** Travel distance refers to how far the switch moves when pressed. Tactile switches typically have a relatively short travel distance, contributing to their quick and responsive feel.
- **Operating Life:** The operating life of a tactile switch refers to the number of times it can be pressed before its performance starts to degrade. This is an important consideration for applications that require long-lasting and reliable switches.
- **Contact Configuration:** Tactile switches come in different contact configurations, including single-pole single-throw (SPST) and single-pole double-throw (SPDT). SPST switches make or break a single circuit, while SPDT switches can connect one of two circuits.
- **Mounting Style:** Tactile switches can have various mounting styles, including surface mount (SMD) and through-hole (THD) mounting. The choice of mounting style depends on the PCB design and manufacturing process.
- **Termination Type:** Tactile switches can have different termination types, such as solder terminals, compliant pins, or surface mount pads, which determine how they are soldered onto the PCB.
- **Applications:** Tactile switches are used in a wide range of applications, including consumer electronics (keyboards, remote controls, game controllers), industrial equipment, medical devices, automotive controls, and more.

When incorporating PCB tactile switches into a design, considerations should be given to factors such as switch placement, orientation, mechanical layout, actuation force, and durability. It's important to ensure that the switches are properly integrated into the PCB layout and that their characteristics meet the requirements of the intended application.

As with any component selection for a PCB, it's recommended to refer to the manufacturer's datasheets and guidelines to ensure proper usage and performance of tactile switches.

Push Button Switches

PCB push button switches, also known simply as push buttons or momentary switches, are electronic components used on printed circuit boards (PCBs) to create a momentary electrical connection when they are pressed. These switches are widely used in various applications to provide user input, control functions, or trigger specific actions.

Here are key features and considerations related to PCB push button switches:

- **Momentary Action:** Push button switches are designed for momentary action, meaning they are activated only when they are pressed and return to their original position when released. This makes them suitable for functions like toggling on/off, triggering events, or sending short commands.

- **Contact Configuration:** Push button switches typically have a single-pole, single-throw (SPST) contact configuration. This means they have one set of contacts that either make or break a single electrical circuit.
- **Actuation Force:** Push button switches have a specified actuation force, which is the amount of force required to press the button and activate the switch. Actuation force can vary between different switch models and affects the user experience.
- **Travel Distance:** Push button switches have a travel distance, which is the distance the button travels when pressed. This distance is relatively short, contributing to the responsive feel of the switch.
- **Housing and Button Design:** Push button switches come in various shapes, sizes, and styles. The design of the housing and button can impact the aesthetics, ergonomics, and ease of use of the switch.
- **Mounting Style:** Push button switches can be surface mount (SMD) or through-hole (THD) components, depending on the PCB design and manufacturing process.
- **Termination Type:** Push button switches can have different termination types, such as solder terminals or surface mount pads, which determine how they are soldered onto the PCB.
- **Applications:** Push button switches are used in a wide range of applications, including consumer electronics (remote controls, appliances, toys), industrial equipment, medical devices, automotive controls, and more.
- **Colors and Markings:** Push button switches often come in different colors, and they may have markings or symbols to indicate their function or status.
- **Protection and Sealing:** Some push button switches come with protection against dust, moisture, and contaminants, making them suitable for use in rugged or outdoor environments.

When integrating PCB push button switches into a design, considerations should be given to factors such as switch placement, orientation, mechanical layout, actuation force, durability, and the electrical connection. Proper design and layout are important to ensure reliable and consistent performance of the switches.

As with any component selection for a PCB, it's recommended to refer to the manufacturer's datasheets and guidelines to ensure proper usage and performance of push button switches.

Slide Switches

PCB slide switches, also known as slide switches or slider switches, are electronic components used on printed circuit boards (PCBs) to control the flow of electrical current by moving a slider or lever to different positions. These switches are commonly used for toggling between different states or modes in various electronic

devices and applications. Here's more detailed information about PCB slide switches:

Features and Characteristics

- **Toggle Functionality:** PCB slide switches offer a toggle function that allows users to change the position of the slider to open or close an electrical circuit. They provide a simple and mechanical way to change the state of a switch.
- **Contact Configuration:** Slide switches typically have a single-pole, double-throw (SPDT) contact configuration. This means they have one set of common terminals that can connect to one of two output terminals, enabling two possible circuit configurations.
- **Mechanical Design:** Slide switches consist of a housing with a slider or lever that can be moved horizontally to different positions. The slider's movement determines the state of the switch. They are designed for easy manual operation.
- **Mounting Styles:** PCB slide switches are available in both surface mount (SMD) and through-hole (THD) versions. The choice of mounting style depends on the PCB design and manufacturing process.
- **Termination Types:** Slide switches can have different termination types, such as solder terminals or surface mount pads, which determine how they are soldered onto the PCB.
- **Number of Positions:** Slide switches come in various configurations with different numbers of positions. Common configurations include single-pole single-throw (SPST), single-pole double-throw (SPDT), and multi-position switches.
- **Applications:** Slide switches are used in applications where a straightforward toggle function is required, such as power on/off controls, mode selection, setting different options, or changing circuit configurations.
- **Size and Form Factor:** Slide switches come in various sizes and form factors, making them suitable for different PCB layouts and designs.
- **Protection and Sealing:** Some slide switches come with protection against dust, moisture, and contaminants, making them suitable for use in rugged or outdoor environments.

Design Considerations

- Proper placement on the PCB is crucial to ensure easy user access and efficient operation.
- Mechanical layout should consider the travel distance, actuation force, and tactile feel of the switch.
- Electrical characteristics, including current and voltage ratings, must be matched to the application's requirements.

- Sufficient spacing and clearance should be provided between the switch and surrounding components to avoid interference.
- Manufacturer datasheets and guidelines should be consulted to ensure proper usage and performance of slide switches.

Slide switches are versatile components commonly used in various electronic devices, ranging from consumer electronics to industrial equipment. Their simple and reliable operation makes them valuable for applications that require user interaction and control.

Toggle Switches

Toggle switches have a lever that is flipped up or down to change the switch state. They are commonly used in applications where a stable switch position is desired, such as in electronic equipment and appliances.

PCB toggle switches are electronic components used on printed circuit boards (PCBs) to control the flow of electrical current by flipping a lever or toggle to different positions. These switches are widely used for on/off functions, mode selection, and other binary switching operations in various electronic devices and applications. Here's more detailed information about PCB toggle switches:

Features and Characteristics

- **Toggle Functionality:** PCB toggle switches provide a toggle function that allows users to change the position of the lever or toggle to open or close an electrical circuit. Once set in a position, the switch remains in that state until manually changed.
- **Contact Configuration:** Toggle switches come in various contact configurations, including single-pole, single-throw (SPST), single-pole, double-throw (SPDT), double-pole, single-throw (DPST), and double-pole, double-throw (DPDT). These configurations determine the number of circuits the switch can control and how they are interconnected.
- **Mechanical Design:** Toggle switches consist of a housing with a lever or toggle that can be flipped up or down. The lever's position determines the state of the switch, either open (off) or closed (on).
- **Mounting Styles:** PCB toggle switches are available in both surface mount (SMD) and through-hole (THD) versions. The choice of mounting style depends on the PCB design and manufacturing process.
- **Termination Types:** Toggle switches can have different termination types, such as solder terminals or surface mount pads, which determine how they are soldered onto the PCB.
- **Applications:** Toggle switches are commonly used for applications that require a stable switch position, such as power on/off controls, mode selection, and setting

different options. They are widely used in a variety of industries, including consumer electronics, industrial equipment, automotive, and more.

- **Size and Form Factor:** Toggle switches come in various sizes and form factors, making them suitable for different PCB layouts and designs.

Design Considerations:

Proper placement on the PCB is important to ensure easy user access and efficient operation.

Mechanical layout should consider the lever's travel distance, actuation force, and tactile feedback.

Electrical characteristics, including current and voltage ratings, must match the application's requirements.

Sufficient spacing and clearance should be provided around the switch to avoid interference.

Manufacturer datasheets and guidelines should be consulted to ensure proper usage and performance of toggle switches.

Toggle switches are durable and reliable components commonly used in scenarios where a clear, stable switch position is required. The manual flipping action makes them suitable for applications where users need to physically control the state of a circuit, and their straightforward operation makes them valuable for a wide range of electronic devices.

Rotary Switches

Rotary switches are designed to be rotated to different positions to establish different connections. They are often used to select from multiple options or settings

PCB rotary switches are electronic components used on printed circuit boards (PCBs) to control the flow of electrical current by rotating a knob or shaft to different positions. These switches are commonly used for selecting different options, settings, or modes in various electronic devices and applications. Here's more detailed information about PCB rotary switches:

Features and Characteristics

- **Rotary Functionality:** PCB rotary switches provide rotary or circular movement, allowing users to select from multiple positions or options by turning a knob or shaft.
- **Contact Configuration:** Rotary switches come in various contact configurations, including single-pole, multi-throw (SPnT) or multi-pole, multi-throw (MPnT). These configurations determine the number of circuits the switch can control and how they are interconnected.

- **Mechanical Design:** Rotary switches consist of a housing with a central shaft or knob that can be rotated to different positions. The knob's rotation determines the switch's state or configuration.
- **Number of Positions:** Rotary switches can have multiple positions, typically ranging from 2 to 12 or more. Each position corresponds to a specific circuit connection.
- **Mounting Styles:** PCB rotary switches are available in through-hole (THD) versions for traditional PCB assembly.
- **Termination Types:** Rotary switches have solder terminals or pins that are soldered onto the PCB.
- **Applications:** Rotary switches are used for applications where users need to select from a range of options, settings, or modes. They are commonly found in audio equipment, instruments, industrial machinery, and other devices where user interaction is required.
- **Size and Form Factor:** Rotary switches come in various sizes and form factors, accommodating different PCB layouts and designs.

Design Considerations

- Proper placement on the PCB is important to ensure convenient user access and smooth knob rotation.
- Mechanical layout should consider the knob's rotation angle, tactile feedback, and detents (clicks) if applicable.
- Electrical characteristics, including current and voltage ratings, must match the application's requirements.
- Sufficient spacing and clearance should be provided around the switch to allow for knob movement.
- Manufacturer datasheets and guidelines should be consulted to ensure proper usage and performance of rotary switches.

Rotary switches are valuable components for applications that require users to select from multiple options or settings. Their circular movement and clear position selection make them suitable for scenarios where precise control and user customization are important. As with any component, thorough consideration of the design and usage requirements is crucial to ensuring the desired functionality and reliability of PCB rotary switches.

DIP Switches

Dual Inline Package (DIP) switches are small, slider-style switches that are commonly used for setting configuration parameters on a PCB. They are often found on electronic boards where users or technicians need to change certain settings.

PCB DIP switches (Dual Inline Package switches) are electronic components used on printed circuit boards (PCBs) to set configuration parameters or control various functions. DIP switches consist of a row of tiny individual switches that can be set to either an "on" or "off" position, allowing users to customize the behavior of a device or circuit.

Here's more detailed information about PCB DIP switches:

Features and Characteristics

- **Binary Configuration:** Each switch in a DIP switch has two positions, typically labeled as "on" and "off." This binary configuration allows users to set a series of binary codes, which can be interpreted by the device or circuit to control specific features or settings.
- **Contact Configuration:** DIP switches can have various contact configurations, such as single-pole, single-throw (SPST) or single-pole, double-throw (SPDT). The choice of configuration depends on the specific application.
- **Mechanical Design:** DIP switches are typically arranged in a row or array and are enclosed within a package with a small lever or slider that can be moved to change the switch positions.
- **Number of Positions:** DIP switches come in various configurations with different numbers of positions, ranging from a few to more than a dozen.
- **Mounting Style:** PCB DIP switches are designed for through-hole (THD) mounting. They have leads or pins that are soldered onto the PCB.
- **Termination Type:** DIP switches have solder pins that are inserted through holes in the PCB and soldered in place.
- **Applications:** DIP switches are commonly used to set configuration options in electronic devices, such as selecting memory addresses, enabling/disabling certain features, setting modes, or calibrating parameters. They are also used for programming microcontrollers or configuring hardware interfaces.

Design Considerations

- Proper placement on the PCB is important to ensure accessibility for users and ease of setting the switches.
- The arrangement of the DIP switches should be well-laid out to prevent confusion and errors when configuring options.
- The mechanical layout should account for the lever or slider's movement and provide adequate spacing between switches.
- Electrical characteristics, such as current and voltage ratings, must match the application's requirements.

- Manufacturer datasheets and guidelines should be consulted to ensure proper usage and performance of DIP switches.

DIP switches offer a straightforward and reliable way to configure devices or circuits, allowing users to make custom settings without the need for additional software or interfaces. Their binary nature and physical presence make them valuable for applications where configuration needs to be set manually or where permanent settings are desired.

Membrane Switches

Membrane switches consist of a thin, flexible layer with conductive paths. When pressed, they make contact and complete a circuit. They are often used in applications where a sealed and low-profile switch design is needed, such as in consumer electronics and medical devices.

PCB membrane switches, also known simply as membrane switches, are specialized electronic components used on printed circuit boards (PCBs) to provide a user interface with a sealed and low-profile design. Membrane switches consist of multiple layers, including a flexible membrane with conductive paths and graphic overlays, that respond to touch or pressure to complete a circuit and trigger specific functions.

Here's more detailed information about PCB membrane switches:

Features and Characteristics

- **Sealed and Low-Profile Design:** PCB membrane switches are designed to be low-profile and sealed, making them suitable for applications where a flat and protected user interface is needed. The design prevents dust, moisture, and contaminants from entering the device.
- **Multilayer Structure:** Membrane switches typically consist of multiple layers, including a graphic overlay with printed symbols or labels, a spacer layer, and a bottom membrane layer with conductive traces. When the top layer is pressed, it makes contact with the bottom layer, completing a circuit.
- **Tactile Feedback:** Membrane switches can incorporate tactile feedback through embossing or dome-shaped buttons, giving users a tactile sensation when pressing the switch. This feedback helps users confirm that their input has been registered.
- **Customizable Graphics:** The graphic overlay layer of membrane switches can be customized with various designs, symbols, colors, and labels, allowing for branding, instructions, or visual cues.
- **Momentary Action:** Membrane switches offer momentary action, meaning they only complete the circuit while pressure is applied. Once the pressure is released, the circuit is broken.

- **Applications:** Membrane switches are used in a wide range of applications, including consumer electronics, appliances, medical devices, industrial control panels, automotive controls, and more.

Design Considerations

- Proper placement and alignment on the PCB are important to ensure accurate touch or pressure activation.
- Mechanical layout should account for tactile feedback and the desired force required for activation.
- The graphic overlay design should be clear, legible, and suit the application's requirements.
- Electrical characteristics, such as circuit layout and trace design, must match the intended functionality.
- Manufacturer specifications and guidelines should be followed to ensure proper usage and performance of membrane switches.

Membrane switches offer a sleek and durable user interface solution that is well-suited for environments where protection against external elements is important. Their customizable graphics and sealed design make them versatile components for various devices that require user interaction while maintaining a clean and functional appearance

Capacitive Touch Switches

Capacitive touch switches detect touch through changes in capacitance and are commonly used in touch-sensitive interfaces, such as touchscreens and touchpads.

PCB capacitive touch switches are electronic components used on printed circuit boards (PCBs) to detect touch or proximity without requiring physical pressure. These switches use the principles of capacitance to sense the presence of a conductive object, such as a human finger. Capacitive touch switches have become increasingly popular for creating modern and user-friendly interfaces in a wide range of electronic devices.

Here's more detailed information about PCB capacitive touch switches:

Features and Characteristics

- **Touch Sensing:** Capacitive touch switches detect touch or proximity by measuring changes in capacitance between the switch's electrode and a conductive object, such as a human finger. The user's touch alters the capacitance, triggering the switch.
- **No Moving Parts:** Capacitive touch switches do not have moving mechanical parts like traditional switches, making them highly durable and less susceptible to wear and tear.

- **Sensitivity and Precision:** Capacitive touch switches can be designed with different levels of sensitivity, allowing for precise detection of touch even through materials like glass or plastic overlays.
- **Customizable Interfaces:** The design of capacitive touch switches can be customized with various graphic overlays, symbols, and visual cues. This makes them versatile for creating user-friendly and visually appealing interfaces.
- **Gesture Recognition:** Advanced capacitive touch switches can support gesture recognition, enabling functions such as swiping, pinching, and rotating.
- **Multitouch:** Some capacitive touch switches can detect multiple touches simultaneously, enabling multitouch functionality.
- **Applications:** Capacitive touch switches are used in a wide range of applications, including smartphones, tablets, touchscreens, appliances, automotive controls, consumer electronics, industrial equipment, and more.

Design Considerations

- Proper placement on the PCB and design of the touch-sensitive area are important to ensure accurate touch detection.
- Mechanical layout should consider the desired touch sensitivity, size of the touch area, and potential interference from external factors.
- Electrical characteristics, such as sensor layout, trace design, and shielding, must be carefully considered for reliable touch detection.
- Manufacturer specifications and guidelines should be followed to ensure proper usage and performance of capacitive touch switches.

Capacitive touch switches offer a modern and intuitive way to interact with electronic devices, providing a sleek and responsive user interface. Their absence of physical buttons and moving parts contributes to their durability and resistance to wear over time. Capacitive touch technology has revolutionized the way we interact with technology, enhancing user experiences and enabling innovative functionalities.

When designing PCBs with switches, considerations must be given to factors such as switch type, mechanical layout, actuation force, durability, and electrical characteristics. Proper placement and design of switches on the PCB are crucial to ensure reliable and consistent performance. Additionally, factors like switch bounce (a phenomenon where a switch briefly toggles between on and off states) and EMI (electromagnetic interference) should be addressed in the design to avoid unwanted behavior and signal noise.

Ultimately, the choice of PCB switches depends on the specific requirements of the application, including user experience, environmental conditions, and desired functionality.

2.5 The Origin

In electronic design, the schematic coordinate origin refers to the reference point from which the X and Y coordinates of schematic elements, such as components, wires, and text, are measured. It is essentially the starting point of the coordinate system used within the schematic design.

The coordinate origin is typically denoted by a small crosshair or marker on the schematic canvas. When elements are placed on the schematic, their positions are defined by their X and Y coordinates relative to this origin.

The choice of the coordinate origin is arbitrary and does not affect the functionality of the schematic. However, it can be useful for ensuring consistency and ease of working on the design, especially when there are multiple sheets or when the design is transferred between different tools.

Some considerations regarding the schematic coordinate origin include:

- **Consistency:** In a multi-sheet design, it is helpful to use the same coordinate origin across all sheets. This consistency makes it easier to align elements between sheets and maintain a coherent layout.
- **Tool Compatibility:** When sharing schematics between different Electronic Design Automation (EDA) tools, it is essential to be aware of how each tool handles the coordinate origin. Some tools may use different default origins or have specific preferences for defining the origin.
- **Component Placement:** The choice of the coordinate origin can influence how components are placed and arranged within the schematic. Designers may find it more convenient to have the origin at a specific location that corresponds to a critical component or functional block.
- **Grid Alignment:** The coordinate origin often aligns with the grid lines, which can help with precise component placement and alignment to the grid.

Designers can typically adjust the schematic coordinate origin in their EDA software tools. This capability allows them to set the origin at a location that makes sense for their particular design needs. Additionally, he AutoTRAX DEX PCB Designer offers the ability to set a "snap" feature, such as "Snap to Grid" or "Snap to Guides," to help align elements to the grid or user-defined guides relative to the chosen coordinate origin.

2.6 Design Units

In PCB (Printed Circuit Board) design, various units of measurement are used to specify dimensions, distances, and other physical parameters. The choice of units depends on the design requirements, the region or country where the design is

being created, and the preferences of the designer or design team. The most common units of measurement used in PCB design are:

Inches (in)

Inches are a widely used unit of measurement in PCB design, especially in regions like the United States.

Millimeters (mm)

Millimeters are commonly used in PCB design, particularly in regions that follow the metric system, such as Europe and many parts of Asia.

Mils (thousandths of an inch)

Mils are commonly used for specifying very small distances, especially for trace widths and clearances in high-density PCB designs. One mil is equal to one-thousandth of an inch (0.001 inches).

Micrometers (µm)

Micrometers, also known as microns, are used for extremely fine measurements in PCB manufacturing, such as specifying the thickness of copper layers or the width of very narrow traces.

Designers and PCB manufacturers must be consistent in using units throughout the design process to avoid errors and ensure accurate fabrication of the PCB. Many Electronic Design Automation (EDA) software tools allow designers to set the preferred unit of measurement and automatically convert values when switching between units.

For example, a typical PCB design software might have options to choose units, such as inches or millimeters, as well as convert values between them seamlessly.

Designers should also follow industry standards and guidelines when specifying measurements to ensure that the PCB meets the required specifications and performs reliably in its intended application.

2.7 Snap to Grid

In electronic design software, "Snap to Grid" is a feature that allows schematic elements, such as components, wires, and text, to automatically align to a grid when they are moved or placed on the schematic canvas. The grid consists of regularly spaced intersecting horizontal and vertical lines, and the Snap to Grid feature ensures that schematic elements "snap" to these grid points or lines when their positions are adjusted.

The Snap to Grid feature offers several advantages in schematic design:

- **Alignment:** Snap to Grid helps align components and elements along straight horizontal and vertical lines, creating a neat and organized layout. This improves the visual appearance of the schematic and makes it easier to read and understand.
- **Consistency:** Snap to Grid ensures that components are placed at consistent intervals, which helps maintain uniform spacing between elements in the design. This consistency enhances the overall look of the schematic and reduces visual clutter.
- **Precision:** By snapping elements to the grid, designers can achieve precise placement of components and wires, avoiding unintentional overlapping or misalignments.
- **Ease of Editing:** When components are aligned to the grid, it becomes easier to move, copy, and modify parts of the schematic without disrupting the overall layout.
- **Spacing:** Snap to Grid can help enforce minimum spacing rules between components and wires, ensuring that there is sufficient clearance between elements to avoid electrical issues and errors during the PCB layout phase.
- **Grid Adjustment:** In most Electronic Design Automation (EDA) software tools, designers can adjust the grid settings, such as grid spacing and grid origin, to match their specific design requirements and preferences.
- While Snap to Grid is generally a useful feature, designers need to use it judiciously. In some cases, strict adherence to the grid may not be practical, especially when working on intricate parts of the schematic that require fine adjustments or custom alignment.

The AutoTRAX DEX PCB Designer offers the Snap to Grid feature and allow users to enable or disable it as needed during the schematic design process. Additionally, some tools may provide other alignment options, such as Snap to Component, Snap to Angle, and Snap to Intersection, to further enhance the ease of creating schematics with precision and accuracy.

2.8 PCB Design Grids

n PCB (Printed Circuit Board) design, grids are regularly spaced horizontal and vertical lines used as a visual reference for component placement, trace routing, and other layout tasks. Grids help designers maintain consistent spacing, alignment, and organization throughout the PCB layout, leading to a cleaner and more professional design.

There are typically two types of grids used in PCB design:

- **Schematic Grid:** The schematic grid is used during the schematic capture phase of the design process. It helps with component placement, wire routing, and overall organization of the schematic diagram. The schematic grid is mainly for visual aid and does not directly impact the physical layout of the PCB.
- **Layout Grid:** The layout grid is used during the PCB layout phase. It defines the grid spacing for component pads, traces, vias, and other physical elements on the PCB. The layout grid is a critical part of PCB design as it directly affects the accuracy and alignment of components and traces on the PCB.

Benefits of using PCB design grids include:

- **Alignment and Consistency:** Grids help ensure that components and traces are consistently aligned, resulting in a neater and more organized layout.
- **Efficient Placement:** Grids facilitate efficient placement of components and routing of traces by snapping elements to grid intersections.
- **Avoid Overlapping:** Grids prevent overlapping of components and traces, which can lead to design errors and manufacturing issues.
- **Spacing Control:** The grid spacing allows designers to maintain specific component and trace clearances to meet design requirements.
- **Manufacturability:** An organized layout achieved through the use of grids helps ensure that the PCB can be manufactured accurately and efficiently.
- **Electrical Integrity:** Properly aligned components and traces can enhance signal integrity and reduce the likelihood of electrical issues.

Modern PCB design software provides options to set and customize both schematic and layout grids. Designers can adjust the grid settings, such as grid spacing, origin, and visibility, to suit their specific design requirements and preferences.

It is essential to use grids wisely and strike a balance between precision and flexibility. In some cases, components or traces may require fine adjustments that go beyond the grid spacing. In such situations, designers can temporarily disable grid snapping to achieve the desired placement.

2.9 Snap to Guides

In electronic design software, "Snap to Guides" is a feature that allows schematic elements, such as components, wires, and text, to automatically align to user-defined guides when they are moved or placed on the schematic canvas. Guides are horizontal and vertical lines that designers can position anywhere on the schematic to help with alignment and spacing.

The Snap to Guides feature offers several benefits in schematic design:

Custom Alignment: Snap to Guides allows designers to define custom alignment lines and use them as reference points for placing and arranging components. This enables precise alignment based on specific design requirements.

Consistency: By snapping elements to user-defined guides, designers can maintain consistent spacing and alignment throughout the schematic, creating a more organized and visually appealing design.

Design Accuracy: The feature ensures that components and wires align with critical points or reference lines, which is essential for accurate circuit connections and layout.

Ease of Editing: When elements are snapped to guides, it becomes easier to edit the schematic without unintentionally shifting components or wires out of place.

Alignment with Existing Components: Snap to Guides allows designers to align new components with existing components, ensuring proper connection and layout continuity.

Multiple Guides: Designers can create multiple guides to accommodate complex schematic layouts and intricate component arrangements.

Using Snap to Guides can be particularly beneficial when designing schematics with specific layout requirements, or when creating complex designs that demand precise alignment and spacing. It provides a level of flexibility not always achievable with traditional Snap to Grid functionality alone.

The AutoTRAX DEX PCB Designer provides the Snap to Guides feature, allowing designers to create and move guides as needed. Designers can toggle the feature on or off and position guides at strategic points on the canvas to align components and wires effectively. The ability to customize guides and align elements precisely simplifies the process of creating professional and accurate schematic diagrams.

2.10 Snap to Objects

"Snap to Objects" is a feature that enables schematic elements to automatically align or snap to nearby objects when they are moved or placed on the schematic canvas. The objects to which elements snap can include other components, wires, pads, or any other relevant elements in the design.

The Snap to Objects feature offers several advantages in schematic design:

- **Automatic Alignment:** When Snap to Objects is enabled, components and wires will automatically align with nearby objects, ensuring precise placement and alignment on the schematic.
- **Consistency:** The feature helps maintain consistent spacing between components and ensures that elements are neatly arranged on the canvas, enhancing the overall appearance of the schematic.

- **Avoid Overlapping:** Snap to Objects prevents overlapping of components, pads, or wires, which could lead to design errors or signal integrity issues.
- **Ease of Editing:** When elements snap to nearby objects, it becomes easier to move or edit parts of the schematic without disrupting the overall layout.
- **Accurate Connectivity:** By snapping wires to nearby pads or connection points, Snap to Objects ensures proper electrical connectivity between components.
- **Component Spacing:** The feature can assist in maintaining specific component spacing, especially for components that require a particular distance for thermal or mechanical considerations.
- **Routing Optimization:** Snap to Objects can help with routing efficiency by automatically aligning new traces with existing traces, pads, or components.

Designers can usually configure the Snap to Objects feature in electronic design software according to their specific needs and preferences. For example, designers may choose to enable or disable snapping to particular types of objects or adjust the snapping distance to control the proximity at which elements will snap to other objects.

While Snap to Objects is generally beneficial, designers should use it judiciously, especially in cases where precise control over element placement is required, or in complex designs where automatic alignment may interfere with specific layout requirements. In such situations, the feature can be temporarily disabled for precise manual placement.

By taking advantage of Snap to Objects, designers can create organized and visually appealing schematic layouts, simplifying the design process and improving the overall quality of the electronic design.

2.11 Aligning Objects

Aligning objects in Schematic/PCB design is a crucial aspect of creating a well-organized and visually appealing layout. Proper alignment ensures that components, traces, pads, and other elements are precisely positioned relative to each other, making the design easier to understand, manufacture, and troubleshoot. Most Electronic Design Automation (EDA) software tools offer various alignment options to help designers achieve accurate and consistent object alignment. Here are some common methods for aligning objects in PCB design:

Manual Alignment: Most PCB design tools allow you to manually select multiple objects and move them to align with each other. This method gives you fine control over object placement and alignment.

Snap to Grid: Enabling the "Snap to Grid" feature ensures that objects are automatically aligned with the grid lines. This feature is especially helpful when aligning objects along straight horizontal or vertical lines.

Distribute: The "Distribute" function in PCB design software allows you to evenly distribute selected objects horizontally or vertically. This is useful for creating uniform spacing between components or traces.

Centering: You can center objects with respect to the horizontal or vertical axis of the design canvas. Centering is especially useful for aligning components or groups of components within specific areas of the design.

Reference Points: Some tools allow you to set reference points on objects and align other objects to those reference points. This is helpful when aligning objects based on specific points rather than the object's boundaries.

Grouping: Grouping objects allows you to treat them as a single entity, making it easier to align and move multiple objects together while maintaining their relative positions.

Alignment Tools: Many PCB design tools provide dedicated alignment tools that offer various alignment options, such as aligning objects left, right, top, bottom, center, or distributing them evenly.

Using Guides: Creating guides on the canvas helps guide the alignment of objects. You can snap objects to guides to achieve precise alignment.

Keyboard Shortcuts: The AutoTRAX DEX PCB Designer offers keyboard shortcuts for alignment, making it quicker to align objects without needing to navigate through menus.

When aligning objects, it's important to consider the specific requirements of your design, such as component clearances, trace routing, and signal integrity. Careful alignment helps improve the overall quality and manufacturability of the PCB design.

2.12 Distributing Objects

Distributing objects in PCB design refers to the process of spacing multiple objects evenly along a specified axis, whether horizontally or vertically. Distributing objects ensures that they maintain a consistent and organized arrangement, which is crucial for achieving a professional and visually appealing PCB layout. Most Electronic Design Automation (EDA) software tools provide distribution features to help designers evenly space and align objects.

Here's how to distribute objects in PCB design:

- **Select Objects:** First, select the objects you want to distribute. These could be components, traces, pads, or any other elements you wish to space evenly.

- **Access Distribution Feature:** In your PCB design software, navigate to the menu or toolbar where the distribution features are located. These features might be named "Distribute," "Evenly Spaced," "Align and Distribute," or similar.
- **Choose Distribution Axis:** Specify whether you want to distribute the objects horizontally or vertically. Some tools might also offer options for distributing along both axes simultaneously.
- **Select Distribution Type:** Depending on the software, you may have options to distribute objects evenly between their outer edges, centers, or reference points.
- **Execute Distribution:** After selecting the desired distribution axis and type, execute the distribution command. The software will automatically adjust the spacing between the selected objects to achieve even distribution.
- **Fine-Tune Spacing:** Some tools allow you to fine-tune the spacing or alignment after distribution, ensuring that the arrangement suits your design requirements.
- **Apply Distribute Vertically or Distribute Horizontally:** Some software tools might provide separate commands for distributing objects along each axis. In this case, you would choose the appropriate command based on the alignment you need.
- **Keyboard Shortcuts**: The AutoTRAX DEX PCB Designer offers keyboard shortcuts to quickly access distribution features.

Distributing objects is especially useful when arranging components or traces in rows, columns, or grids. It helps maintain uniform spacing, which can improve the visual appearance of the layout and ensure that components are positioned within the required clearances.

When using the distribution feature, keep in mind any specific design constraints, such as minimum trace widths, component clearances, and signal integrity considerations. Distributing objects should contribute to an organized and optimized design while meeting all design requirements.

2.13 Dimensions

In electronic design, schematic dimension symbols are graphical representations used to annotate and indicate the physical dimensions, sizes, and spacing of components, PCB features, and other elements in a schematic diagram. These symbols help convey crucial information about the physical aspects of the circuit design and assist in ensuring accurate representation during the PCB layout phase.

Here are some common schematic dimension symbols:

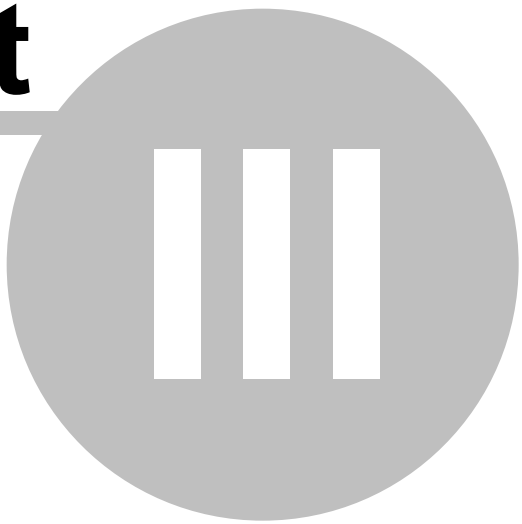
- **Horizontal and Vertical Lines with Arrows:** These symbols represent dimensions for the length or width of components or PCB features. The arrows at the ends of the lines indicate the direction of the dimension measurement.

- **Diameter Circle:** This symbol is used to indicate the diameter of circular components or holes, such as vias or mounting holes.
- **Radial Lines:** Radial lines are used to denote the diameter of a circular component or pad.
- **Horizontal and Vertical Lines with No Arrows:** These symbols are used for indicating distances or clearances between components, traces, or other objects on the schematic.
- **Double-Ended Arrow:** This symbol represents the distance between two points or the spacing between two objects.
- **Box or Rectangle with Dimension:** This symbol is used to enclose a component or an area and annotate its dimensions.
- **Grid-like Pattern:** A grid-like pattern with dimensions shown on the schematic is used to represent an array of components or pins.
- **Connector Pitch:** This symbol denotes the pitch or spacing between the pins or terminals of a connector.
- **Component Footprint Symbol:** Some schematic symbols for specific components may include a representation of the physical package or footprint size.
- **Reference Designator with Dimension:** Dimension symbols may be placed adjacent to a component reference designator to indicate its size.

Schematic dimension symbols are typically accompanied by numeric values to indicate the actual measurement in the required unit (e.g., millimeters or inches). The symbols are placed alongside the relevant components, pads, traces, or areas they refer to, providing clarity about the physical aspects of the circuit design.

When using schematic dimension symbols, it's essential to ensure accuracy and consistency to avoid errors during the PCB layout and manufacturing stages. Properly annotated dimensions contribute to the successful realization of the electronic design and aid in meeting specific design requirements and constraints.

Part



3 Downloads

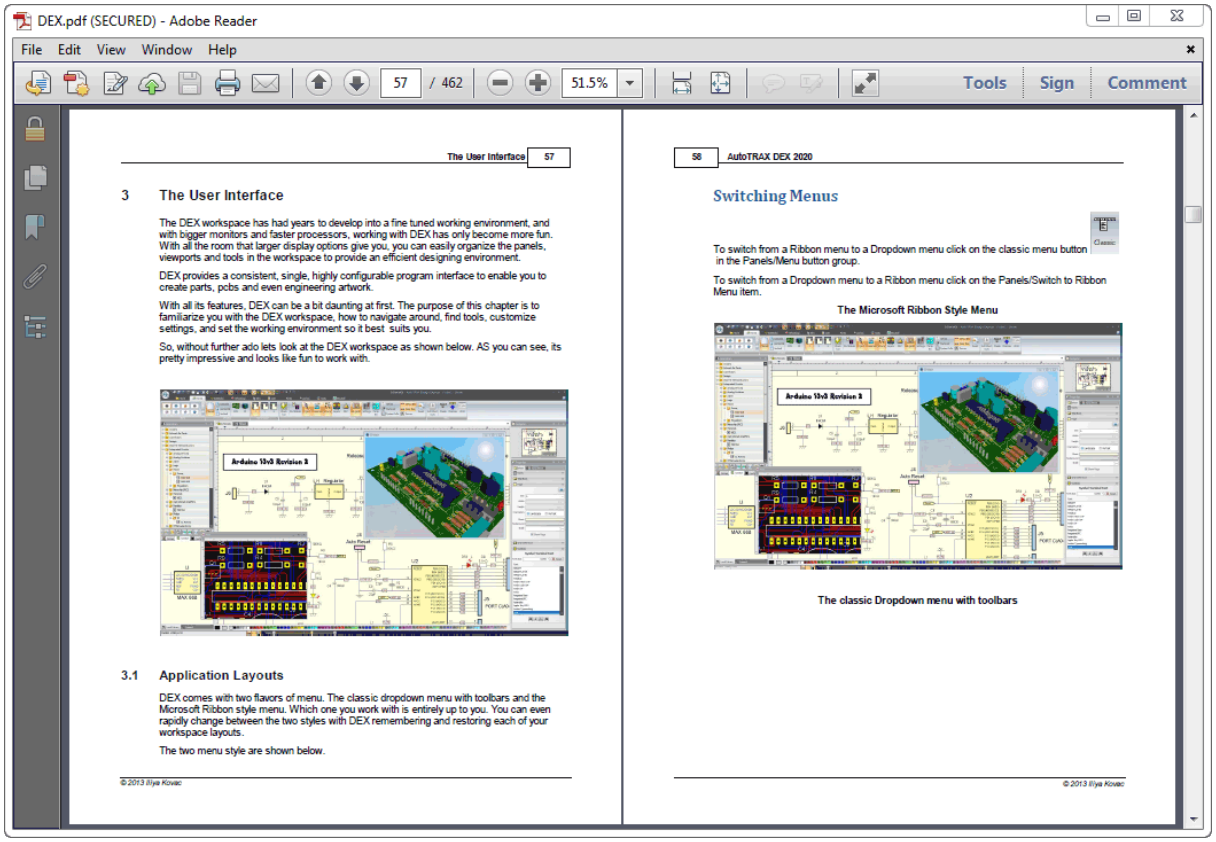
Download the AutoTRAX Progam

Download AutoTRAX now and experience 21st. century schematic capture and PCB design software at it's best. You will download the installer executable which will install the program. It also include an uninstaller.

Click here to download...

Download the Free Adobe PDF Book

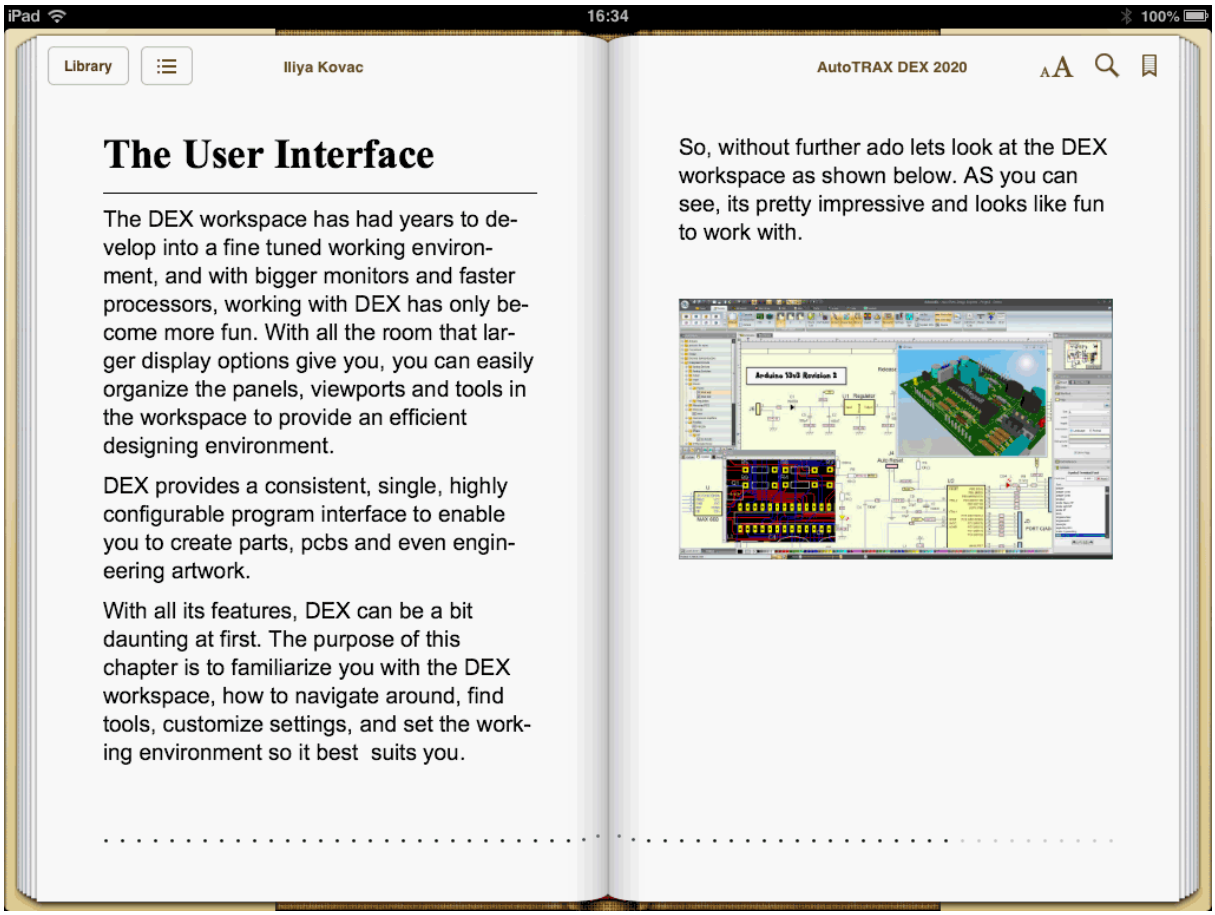
You can download this manual as a free Adobe PDF book from here...



Adobe PDF Book

Download the Free Apple ePub Book

You can download this manual as a free Apple ePub book from here...



Apple ePub Book

Part

4 YouTube Tutorial Videos

The AutoTRAX DEX PCB Designer YouTube Channel

[View the AutoTRAX DEX PCB Designer video channel...]

Index

- 3 -

- A -

- B -

- C -

- D -

- E -

- F -

- G -

- H -

- I -

- L -

- M -

- O -

- P -

- R -

- S -

- T -

- V -

- W -

Endnotes 2... (after index)