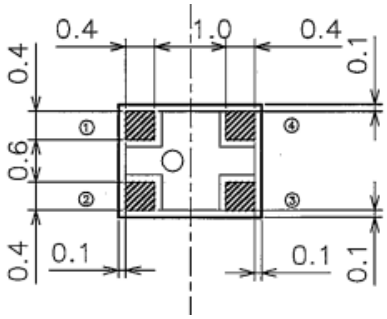
adafruit learning system

KTOWN's Ultimate Creating Parts in Eagle Tutorial

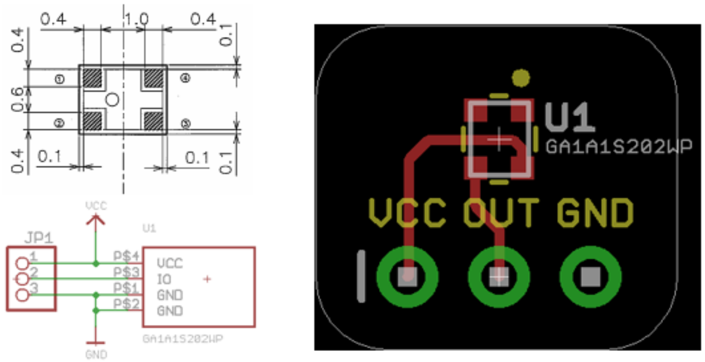
Created by Kevin Townsend



Last updated on 2013-09-11 02:00:38 PM EDT

Guide Contents

Introduction



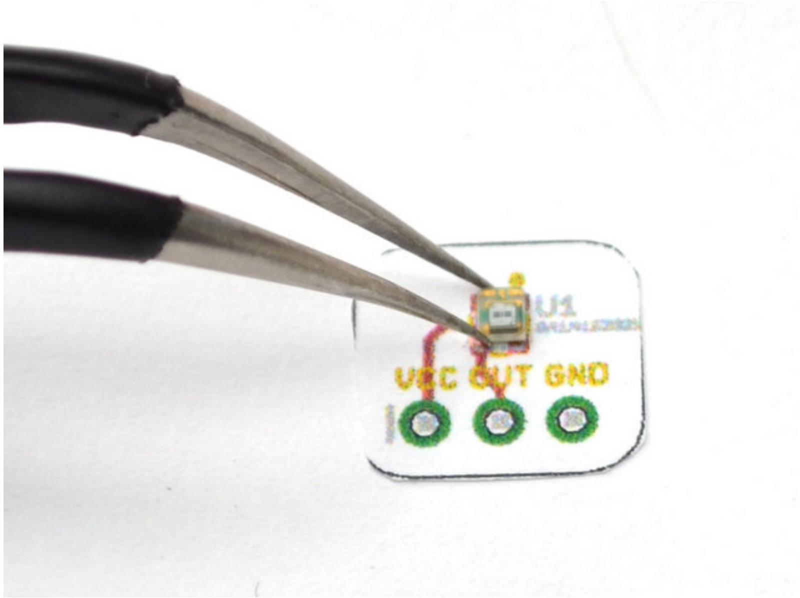
Designing reliable, manufacturable footprints and PCBs is the life-blood of any hardware company, and an essential skill for any maker.

While there are literally dozens of options out there, both commercial and open source, we use Eagle at Adafruit for all of our PCB design. It's just kind of become second nature to hammer out boards and parts in it, and once a tool becomes 'invisible' to you, it's really hard to be persuaded to change.

Like everyone, we've had to plumb the depths of Google, dig into obscure posts from long dead servers, and sometimes just fire off a board and see what happens to learn the ins and outs of Eagle.

We hope you can get a head start doing things the right way yourself with this tutorial, though, and avoid some of the head scratching we had to go through ourselves!

So ... grab your mouse, fire up Eagle, and follow along as we create a simple footprint from start to finish!



What You'll Need

- A copy of Cadsoft Eagle (http://adafru.it/c9i) (Eagle 6.3.0 was used in this tutorial)
- The datasheet for your part (we'll use the GA1A1S202WP Analog Light Sensor (http://adafru.it/c9k) here)
- A bit of attention to detail (http://adafru.it/c9m)

Understanding Parts in Eagle

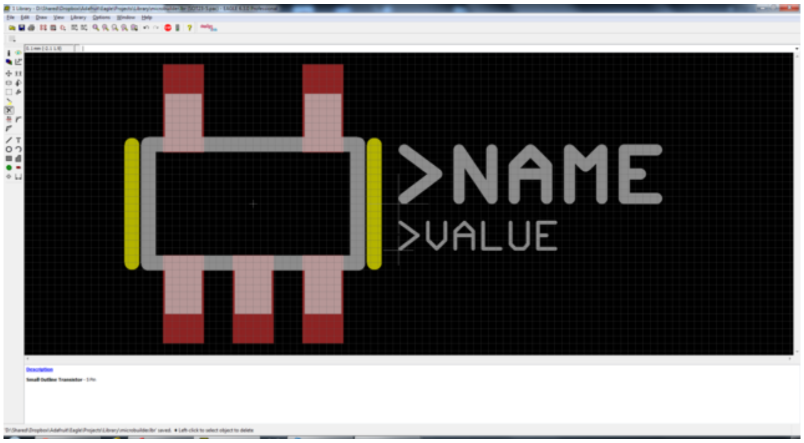
While every CAD package does this slightly differently, one of the first things you'll need to wrap your head around in Eagle before you can add a new 'part' to your library is that there isn't just one piece to your part.

If you open up your library -- just right-click on the library in Eagle and select '**Open**' -- you'll see that Eagle breaks parts down into three separate 'chunks':

Packages

This is the representation of your physical part -- where you design the actual metal contacts, specify whether the pads should have solder paste applied to them, lay out the physical dimensions of the part, perhaps put down an indicator to let users know where pin 1 is, etc.

This is what is commonly referred to as the 'footprint'. It might be a standard package like SOT23-5, TQFP48, or 0603, or the 'footprint' may be a one off design, such as a special connector with drill holes in a specific location, etc



Description

Small Outline Transistor - 5 Pin

'D:\Shared\Dropbox\Adafruit\Eagle\Projects\Library\microbuilder.lbr' saved. ♦ Left-click to select object to delete

Symbols

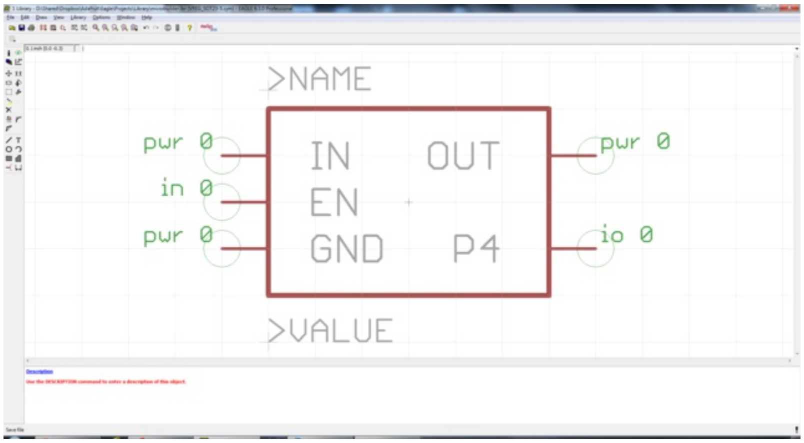
'Symbols' in Eagle are the visual or conceptual representation of your packages, and are what will get placed on your schematic to be hooked up with all the other parts.

The symbol contains a representation of each 'pin' or contacts on the package, usually along with a descriptive label to say what that pin does (ex. 'TXD', 'GND', 'USBD+', etc.).

The 'symbol' is the bridge that connects your schematic to the physical PCB. There is also a rich visual 'language' around symbols that allows engineers to communicate functionality and intent, such as knowing that a two pin blob is a resistor, a capacitor, a diode, or a fuse.

It's very important to provide basic information (such as clear pin names) here, and respect commonly-defined visual 'terms' so that people understand what you are trying to say. There are two main families of symbols, one predominantly used in the US and the other predominantly used in Europe and elsewhere, but either is fine as long as you follow one of them:

- Some common 'IEEE' (US) schematic symbols (http://adafru.it/c9o)
- Some common 'IEC' (EU) schematic symbols (http://adafru.it/c9q)



Description

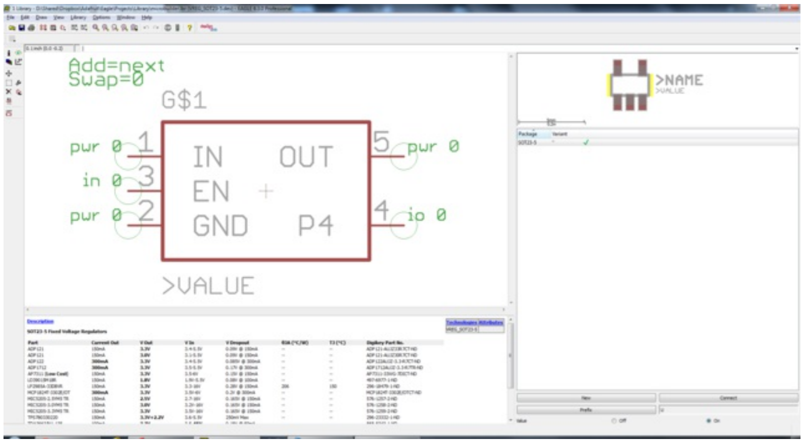
Use the DESCRIPTION command to enter a description of this object.

Devices

Devices in Eagle are where the 'Package' (or Packages!) are combined with the selected 'Symbol', and you indicate which pad on your footprint or package should be connected to which pin on your symbol.

You normally only have a single 'symbol' (some exception exist for very large and complex parts), but you can have multiple 'packages'. For example, we can define a standard 2-pin symbol for a resistor, and create a new 'RESISTOR' device. We can place our resistor symbol in the new device, and then add 10 or more different physical packages that we can use with that symbol (for example 0402, 0603, 0805, 1206, various through hole parts, etc.).

The advantage of separating packages and symbols is that we're allow to reuse common parts in this manner, and use existing packages across multiple parts. You will probably reuse the SOT23-5 package across a wide variety of devices, for example, from Voltage regulators to battery chargers to small sensors.



Description

SOT23-5 Fixed Voltage Regulators

Part	Current Out	V Out	V In	V Dropout	θJA (°C/W)	TJ (°C)	Digikey Part No.
ADP121	150mA	3.3V	3.4-5.5V	0.09V @ 150mA	--	--	ADP121-AUJZ33R7CT-ND
ADP121	150mA	3.0V	3.1-5.5V	0.09V @ 150mA	--	--	ADP121-AUJZ30R7CT-ND
ADP122	300mA	3.3V	3.4-5.5V	0.085V @ 300mA	--	--	ADP122AUJZ-3.3-R7CT-ND
ADP1712	300mA	3.3V	3.5-5.5V	0.17V @ 300mA	--	--	ADP1712AUJZ-3.3-R7TR-ND
AP7311 (Low Cost)	150mA	3.3V	3.5-6V	0.15V @ 150mA	--	--	AP7311-33WG-7DICT-ND
LD39015M18R	150mA	1.8V	1.9V-5.5V	0.08V @ 100mA	--	--	497-6977-1-ND
LP2985A-33DBVR	150mA	3.3V	3.3-16V	0.28V @ 150mA	206	150	296-18479-1-ND
MCP1824T-3302E/OT	300mA	3.3V	3.5V-6V	0.2V @ 300mA	--	--	MCP1824T-3302E/OTCT-ND
MIC5205-2.5YM5 TR	150mA	2.5V	2.7-16V	0.165V @ 150mA	--	--	576-1257-2-ND
MIC5205-3.0YM5 TR	150mA	3.0V	3.2V-16V	0.165V @ 150mA	--	--	576-1258-2-ND
MIC5205-3.3YM5 TR	150mA	3.3V	3.5V-16V	0.165V @ 150mA	--	--	576-1259-2-ND
TPS780330220	150mA	3.3V+2.2V	3.6-5.5V	250mV Max	--	--	296-23332-1-ND
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

Technologies	Attributes
VREG_SOT23-5	

Package	Variant
SOT23-5	

Note: As with schematic symbols, there is also a 'language' around devices names, where certain types of devices are commonly referred to via a letter followed by a unique number (for example 'X1' for a connector, 'R15' for a resitor, 'C7' for a capacitor, etc.). The Device is where you specify these letter identifiers in Eagle, but we will cover this later when we create a new device.

Creating a Package

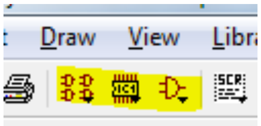
Creating packages is usually your first step making a footprint, and it's also the most critical. Any mistakes here will mean your part won't fit, or your board may be unmanufacturable, so you if you're going to put some extra effort in somewhere, this is where you want it to be!

Before You Get Started

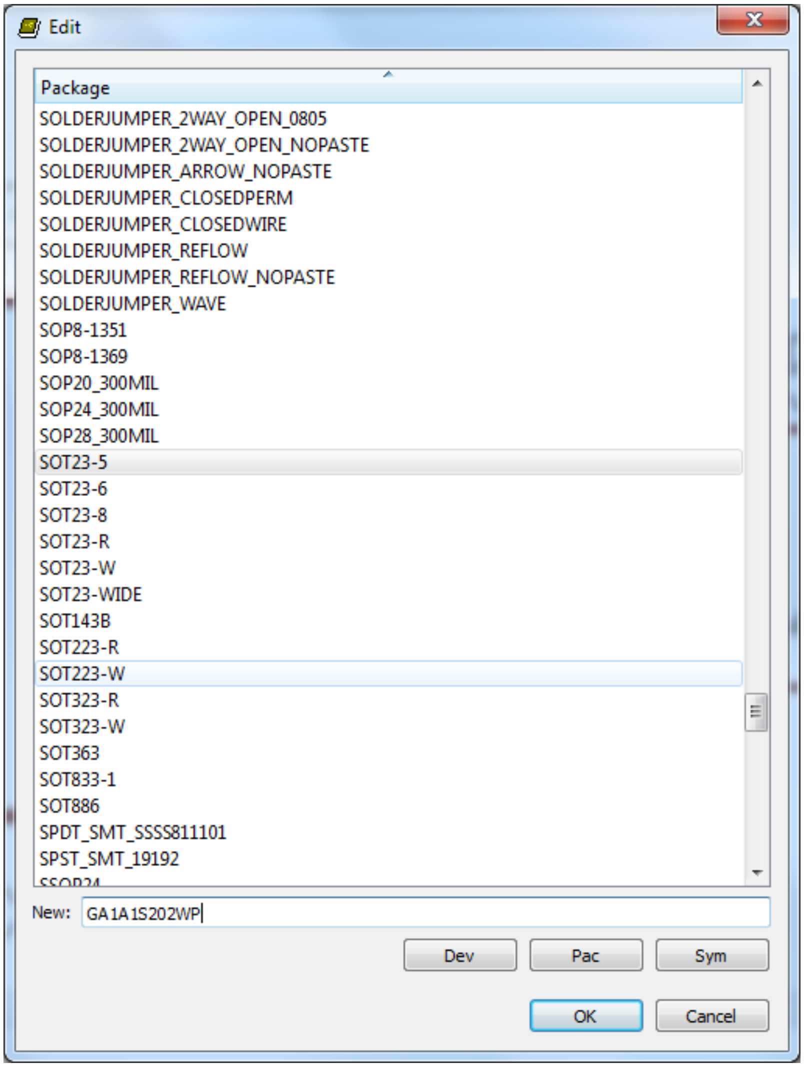
We'll be creating a footprint for the GA1A1S202WP Light Sensor (http://adafru.it/c9k) in this tutorial, since it's a nice easy part, and I just happen to have to make a breakout for this today. Make sure you have the datasheet opened so that you can follow along throughout this tutorial. The package details are found on page six of the PDF, marked as 4/15 in the footer.

1: Creating a New Package

Once you've opened up your library in the editor (just right-click on any library and select the 'Open....' option), you can create a new 'device', 'package' or 'symbol' via the three dedicated icons in the top left hand corner of your Eagle toolbar (shown in the same order below, and highlighted for convenience sake):



You can either select an existing package and edit it (just double-click an existing item in the list), or create a new package by entering a unique name in the '**New:**' textbox towards the bottom:



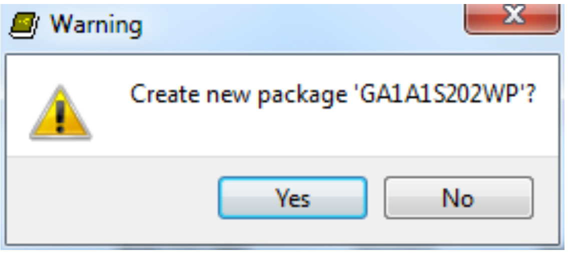
SOLDERJUMPER_2WAY_OPEN_0805
SOLDERJUMPER_2WAY_OPEN_NOPASTE
SOLDERJUMPER_ARROW_NOPASTE
SOLDERJUMPER_CLOSEDPERM
SOLDERJUMPER_CLOSEDWIRE
SOLDERJUMPER_REFLOW
SOLDERJUMPER_REFLOW_NOPASTE
SOLDERJUMPER_WAVE
SOP8-1351
SOP8-1369
SOP20_300MIL
SOP24_300MIL
SOP28_300MIL
SOT23-5
SOT23-6
SOT23-8
SOT23-R
SOT23-W
SOT23-WIDE
SOT143B
SOT223-R
SOT223-W
SOT323-R
SOT323-W
SOT363
SOT833-1
SOT886
SPDT_SMT_SSSS811101
SPST_SMT_19192
SSOP24

New: GA1A1S202WP

Give it a sensible name that you'll remember if it's a one off part, or the standard name if it's a common package (see all the SOT varieties above).

In this particular case, we"re going to call this footprint '**GA1A1S202WP**' since it's a unique part, so we'll enter the part name in the '**New:**' textbox, and then **click OK**.

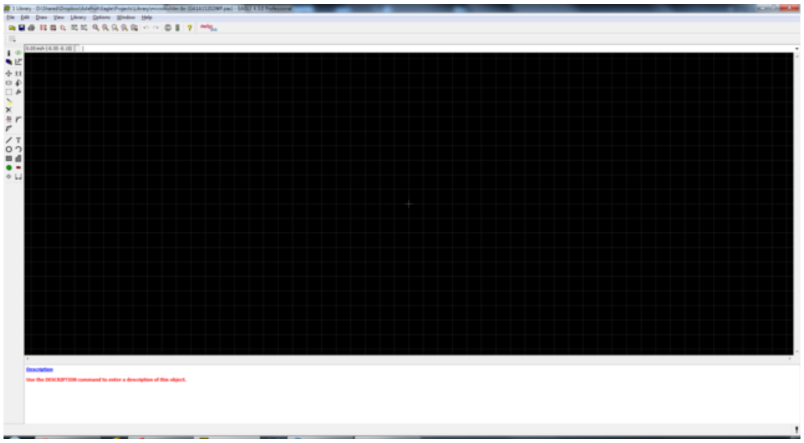
After clicking OK you'll see a very sensible little dialogue box asking if you're sure you want to accept the consequences of such a monumental decision. Go ahead and click '**Yes**':



Create new package 'GA1A1S202WP'?

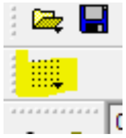
Welcome to the Package Editor

Congratulations ... you now have an empty package named 'GA1A1S202WP'. You should see something similar to the screen below, which is the standard package editor view:

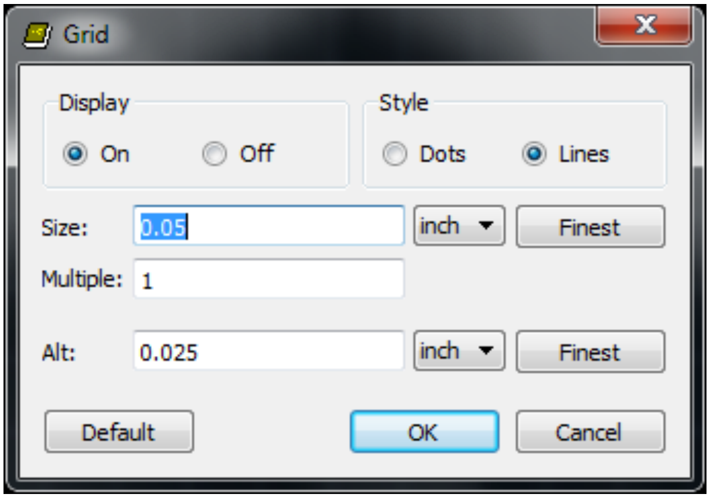


... Now Meet Your Eagle BFF

You see that little dot-dottly-doo icon in the rop-right hand corner there? Ya ... that unassuming little 32x32 pixel icon just below the folder: **It's the secret to reliable, manufacturable designs in Eagle!** Get to know it and love it, since you'll be spending a lot of time together with it:



Click that little icon and you'll see a dialogue box similar to the following:



Grid

Display: ● On ○ Off

Style: ○ Dots ● Lines

Size: 0.05 inch

Multiple: 1

Alt: 0.025 inch

This dialogue box controls the 'grid' in Eagle, and determines where you lines and shapes snap to when you're drawing stuff. This is extremely useful when designing parts, because you can adjust the grid to the exact dimensions you need, draw a precise line along one axis at that position, or continually copy and paste parts at a specific pitch (like a pad on a QFP package every 0.5mm, of every 0.65mm on a QFN IC, etc.).

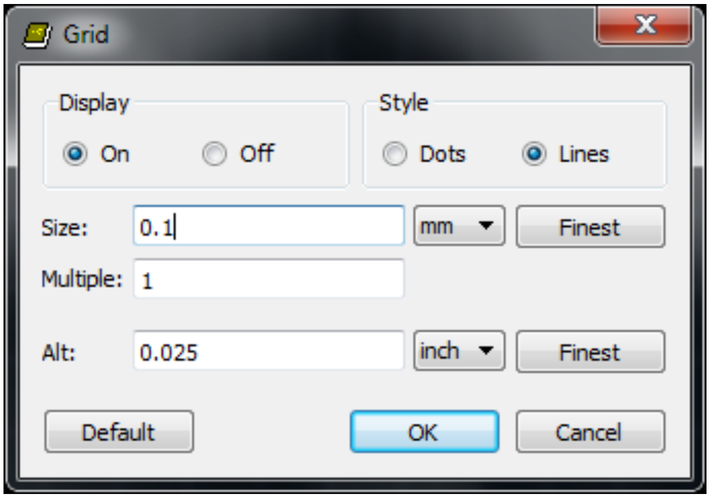
Learning to use the grid to your advantage is the key to quick but accurate footprint design.

There are several values here, but the only one that really matters is '**Size**' and the unit dropdown box just beside it.

- **Size** sets the distance between each line or 'snap point' on the grid. Every new line you draw or pad you place or copy/paste will be centered on a multiple of this size. If you have a 0.65mm pitch QFN package, you place one pad at the right position, set the grid to 0.65mm, and then copy and paste the rest of the row
- **Units Drop-Down**: Use this to select between common engineering units (mm, inch, mil, etc.). In generall, I always use mm since every modern datasheet is based on the metric system (and yes, that's a good thing).

For the GA1A1S202WP lets go ahead and set the grid to something small like **0.1mm**:

1. Set '**Size**' to **0.1**
2. Select '**mm**' in the drop-down box right beside it
3. Make sure the '**Display**' radio button is set to 'On' as well so that you can see your grid
4. Click '**OK**'



Grid

Display: ● On ○ Off

Style: ○ Dots ● Lines

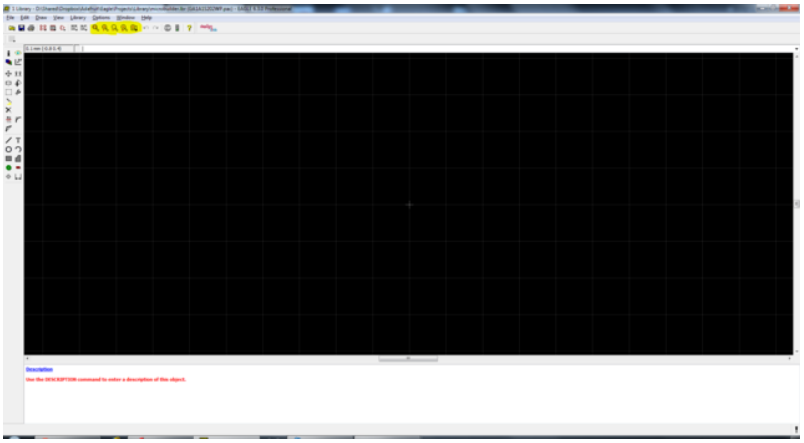
Size: 0.1 mm

Multiple: 1

Alt: 0.025 inch

After clicking OK, you should see a much tighter grid, with a small white '+' marking the 0,0 position in the middle of the screen:

Zoom in a bit using the scroll wheel on your mouse of the zoom icons (highlighted in the image below) until you get something resembling the zoom level below (click the image to see the grid):



Description

Use the DESCRIPTION command to enter a description of this object.

Note: If the grid is too small relative to your current 'zoom' level, the grid won't be visible and you'll ge a black screen with the center position marker. You can scroll in and out with the scroll wheel in your mouse, or or the zoom icons in the toolbar (various magnifying glasses)

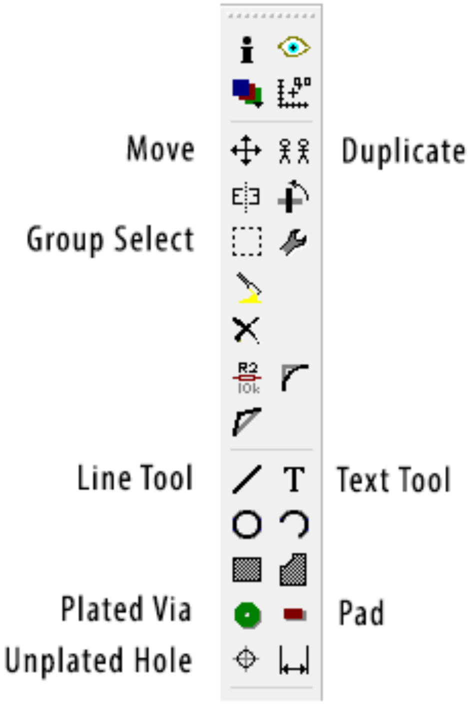
Creating Your First Pad

With time you'll find your own workflow for creating parts. I like to start with a part outline, and then add the pads on, but for convenience sake we'll start right away with some pads, and then worry about the rest later since we'll need to explain a few concepts before we can design the rest of the package.

You'll want to get familiar with the toolbar on the left-hand side of Eagle, since it contains icons for the most common commands (and some pretty uncommon ones as well):

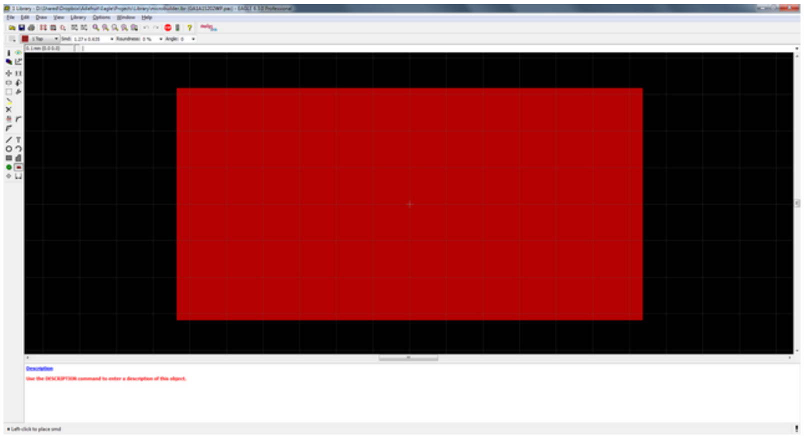
> Eagle is extremely keyboard and mouse heavy (in a good way!), and once you get efficient with Eagle you'll find yourself using keyboard commands for everything, or using your mouse shortcuts like the scroll-wheel to zoom, right-click to rotate, etc. What feels slow at first becomes second-nature when you learn the shorcuts.

You can place your mouse over the icons to see a label of what they do, but the most important icons for now are shown below:



The different tools will come with time, but for now click the red block marked 'Pad' above (it's actually called '**Smd**'). This will cause your mouse to have a red surface-mount pad that you can place anywhere you want by clicking inside the package.

> To stop using the 'Smd' function, just select a different icon in the left-hand toolbar, such as the 'move' command



Description

Use the DESCRIPTION command to enter a description of this object.

Essential Eagle Skills

Some things to note working with pads and parts in Eagle:

- **Zooming in and out:** If you have a scroll wheel on your mouse, scroll to zoom in and out. You'll do a lot of this in Eagle, so it's good to get used to it when placing and updating parts and working with PCBs, or zooming in on details regions where you need to do some work.
- **Rotating parts and pads:** Right-click the active pad to rotate. You'll do a lot of this 'select-then-rotate' when placing parts on your PCB or designing your packages.
- **Use that grid:** Notice that the pad moves based on the grid that we defined previously
- Also notice that **pads rotate and are place around their center point**. This is important when placing parts, since you need to calculate and enter it's exact position, which is based on the center of your pad.

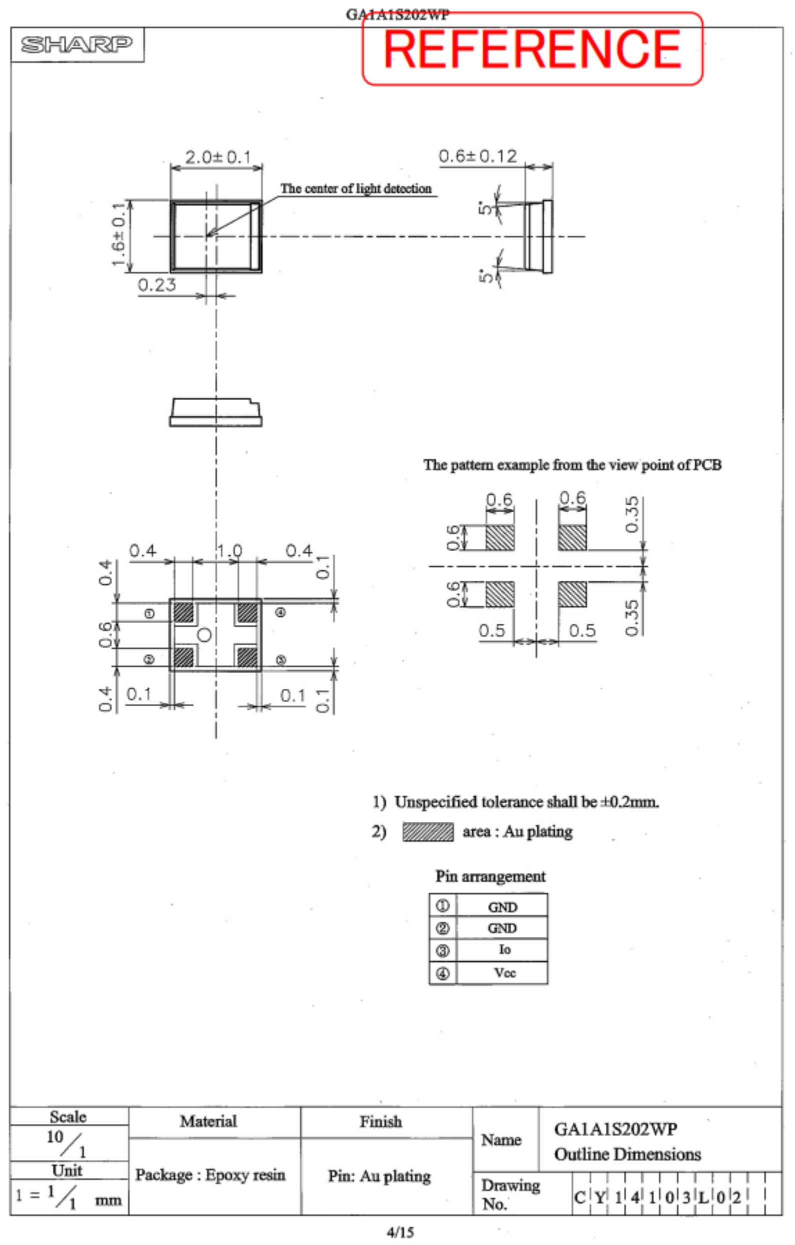
Setting Your Pad Size

Correctly sizing and placing your pads is the most critical part of the footprint design process, so you'll want to be sure you understand this section very well!

Now that you know how to create a pad, it's time to learn how to properly size them and position them.

Before we can go any further, you'll need to pull out the GA1A1S202WP datasheet (http://adafru.it/c9k) (or whatever other datasheet), and find the package outline. It's usually on the last couple pages of any datasheet, but in the case of this part it's on page 6 of the PDF.

For convenience sake, we've included a screenshot of the package outline below:



1) Unspecified tolerance shall be ±0.2mm.

2) ▨ area : Au plating

Pin arrangement

①	GND
②	GND
③	Io
④	Vcc

Scale	Material	Finish	Name	GA1A1S202WP Outline Dimensions
10/1				
Unit	Package : Epoxy resin	Pin: Au plating	Drawing No.	CY14103L02
1 = 1/1 mm				

4/15

Everything that we need to know to create our package is on this page, and it's important to become familiar with how these package drawings work.

There are two important elements that we need to take into account create a package:

1. The **package 'dimensions'**, which is the physical outline of the package. We'll come back to this one later.
2. The **pad locations and sizes**, which are the metal pads we will creates where to 'leads' or 'contacts' on your part will get soldered to your PCB.

Sizing Your Pad(s)

The first step we'll do to actually start creating a package for our sensor, is to start laying down the pads in the right size and location.

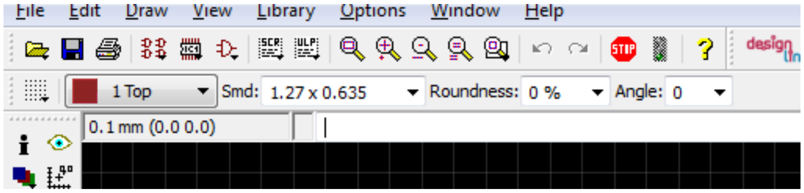
There are four pads on this part, represented by the four dark, hatched squares in the diagram.

> Most manufacturers include a 'recommended footprint pattern', showing the suggested size for your pads. Be sure to follow these suggestions as closesly as possible when they are present.

Looking at the recommended footprint, we can see that this part has four identical pads, and **the manufacturer suggests a pad size of 0.6 x 0.6mm**.

To create a 0.6 x 0.6mm pad, we need to click the 'pad placement' tool we discovered in the previous page (the '**Smd**' tool in the left-hand toolbar), and then adjust the size of the pad.

Once you've select the **Smd** tool, a new toolbar will appear at the top of your display with the settings for the new pad(s):



1 Top
Smd: 1.27 x 0.635
Roundness: 0 %
Angle: 0
0.1 mm (0.0 0.0)

Adjusting the size is easy.

The toolbar contains four items, but we only need to worry about two of them here:

- **Layer Selection**: The first drop-down box (shown with '**1 Top**' here) indicates on which 'layer' our pad should be placed. We'll explain layers later, but for now leave this at '1', which is the top copper layer, and where we want to place our pads 99% of the time.
- **Smd Size**: The second box (labelled '**Smd:**') is the most important one here. This is where we indicate the size of our new pad. It's currently 1.27 x 0.635.

To change our pad size to 0.6mm wide by 0.6mm high, we just type '**0.6 x 0.6**' in the '**Smd:**' textbox.

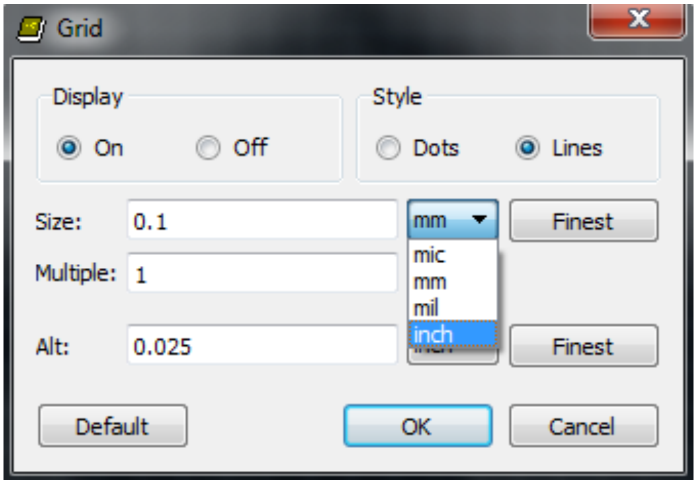
Essential Eagle Skills

Side Note on Units and Sizes in Eagle

Do you remember your Eagle BFF from a previous page, the **Grid Dialogue**? You'll see one reason why this little icon is so important here.

Our pad size in the screenshot above is set to 1.27 wide by 0.635 high, but there are no units specified!

This is because the 'units' in Eagle are controlled via the grid settings, and we are free to switch back and forth between mm, mil, or any other supported unit, and the numbers will be adjusted accordingly.


Grid

Display: ● On ○ Off

Style: ○ Dots ● Lines

Size: 0.1 mm

Multiple: 1

Alt: 0.025 inch

We see 1.27 x 0.635 now because we are on a '**0.1 mm grid**'. If you click the grid icon again (dialogue box shown above), you can change the grid units to 'inch', which would cause the default Smd size to be displayed as 0.05 x 0.025, and all of the other numbers in Eagle to be adjust to inches (or mils, etc.).

For now, keep these in mm, though, since 99% of parts today are specified in mm.

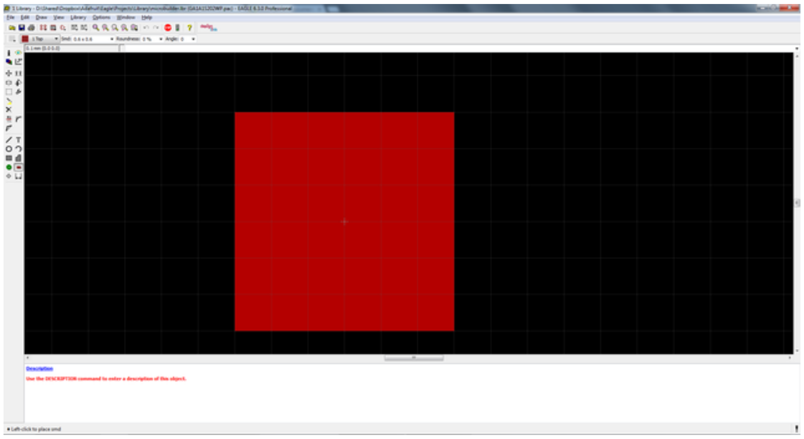
This is true of any tool in Eagle, and it's very important to understand this or you can end up with a part in inches when you needed a part in mm!

Placing Your Pads

Now that we've specified the size of our new pad (0.6 x 0.6 mm in this case), it's time to place it in our package.

This is where you really need to have a bit of attention to detail, since pad size and placement is critical to creating a functional and manufacturable footprint!

Selecting the '**Smd**' tool (the red box in bottom of the the left-hand toolbar), and adjusting the 'Smd Size' to 0.6 x 0.6 on a mm grid, we should see something similar to the following if you zoom in or out a bit:



Description

Use the DESCRIPTION command to enter a description of this object.

Left-click to place smd

If you zoom in, you can see that this pad is indeed 0.6mm x 0.6mm thanks to the 0.1mm grid lines we've set. The pad should align perfectly with six 'squares' in each direction on the grid.

You can also see that the pad moves relative to it's 'center' position, so if we place the pad at 0 x 0, it would actually extend 0.3mm out in every direction.

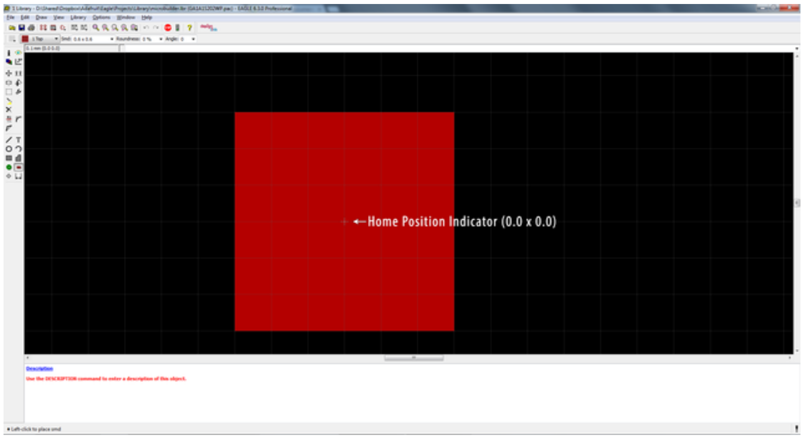
This is very important to understand when placing pads. A 0.6x0.6mm pad placed at 0 x 0 would actually start at X -0.3mm and Y -0.3mm and end at X 0.3mm and Y 0.3mm!

You need to fully understand this, since the only way to create good footprints is to manually calculate the exact position for your pads, and you need to enter the position data relative to the center of the pad.

Place Your First Pad

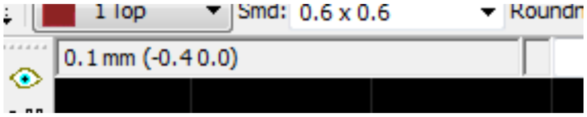
Generally the easiest way to start placing your pads, it to place a single pad at the 0 x 0 position, which we'll refer to as the 'home' or 'zero' position.

Place a 0.6 x 0.6 mm pad at the home position by selecting the Smd tool, setting the size, and then simply clicking the mouse at 0 x 0, which is indicated by a small white '+' on your grid:



You can also see the current position on the grid via the little text box in the upper left-hand corner of Eagle, which will update as you move your move around.

We can see in the image below that we are on a 0.1 (mm) grid, and the mouse is currently at -0.4 x 0.0:



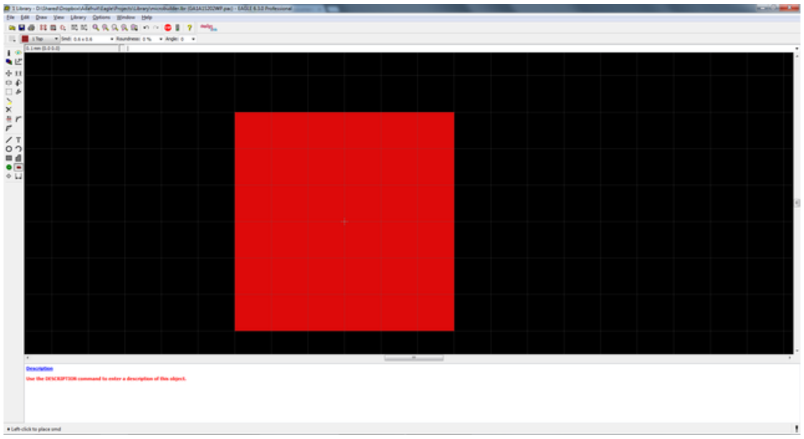
1 Top

Smd: 0.6 x 0.6

Roundn

0.1 mm (-0.4 0.0)

For now, go back to the 0.0 x 0.0 position, though, and click the mouse button once, which should create a new 0.6 x 0.6mm pad centered at 0 x 0:



Essential Eagle Skills

It's extremely important that all of your parts are designed around a sensible home or center position, preferably the mechanical center of the part!

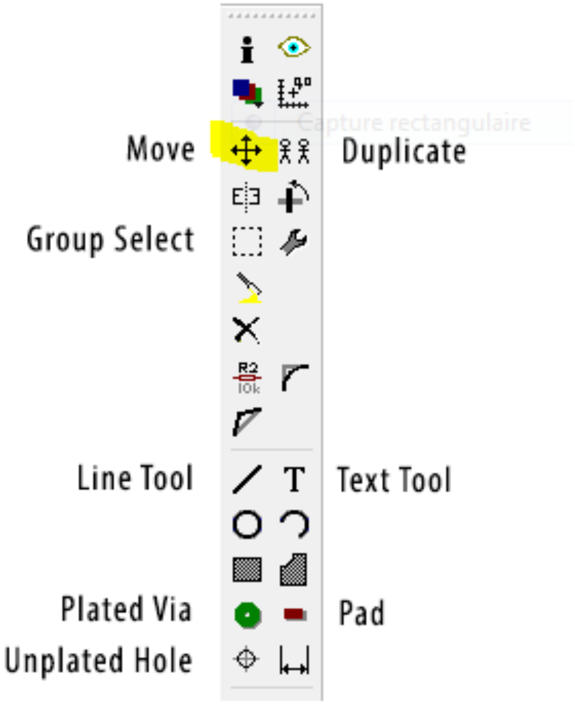
There are a number of reasons for this, but one of the key reasons is that parts are 'rotated' around their home position. By laying out your package around the mechanical center of the device, you can rotate your parts on your PCB without making a huge mess of the rest of the layout.

Accurately Positioning Pads

Now that you've placed your first pad, lets position it!

You need to **deselect** the 'Smd' tool or you'll keep adding new pads every time you click somewhere.

Go back to the left-hand toolbar, and click an action like 'move' (highlighted in yellow below) that doesn't create something new:

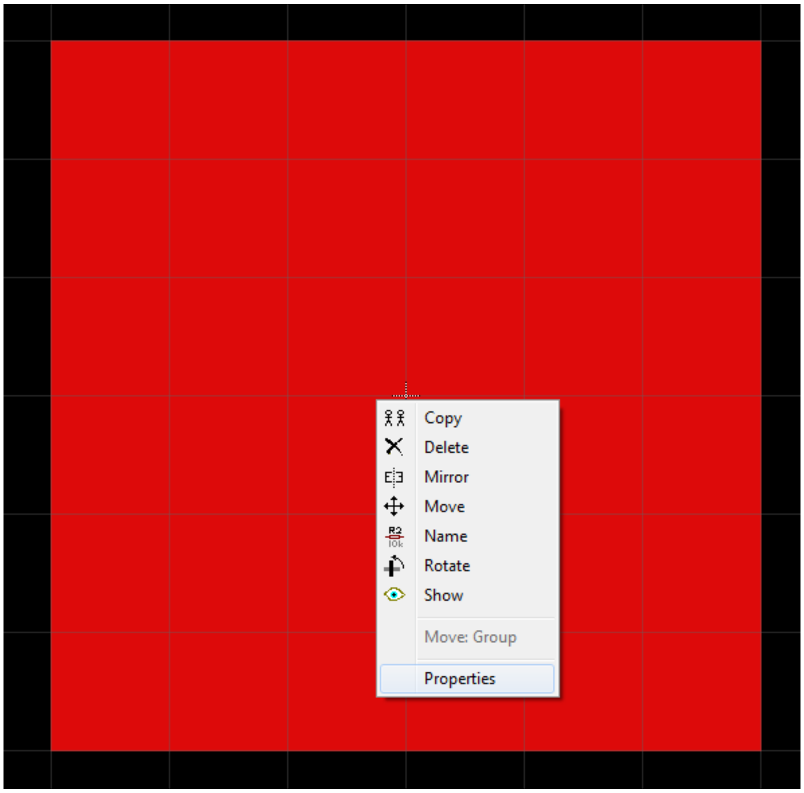


Meet Your Other Eagle BFF: The Properties Dialogue Box!

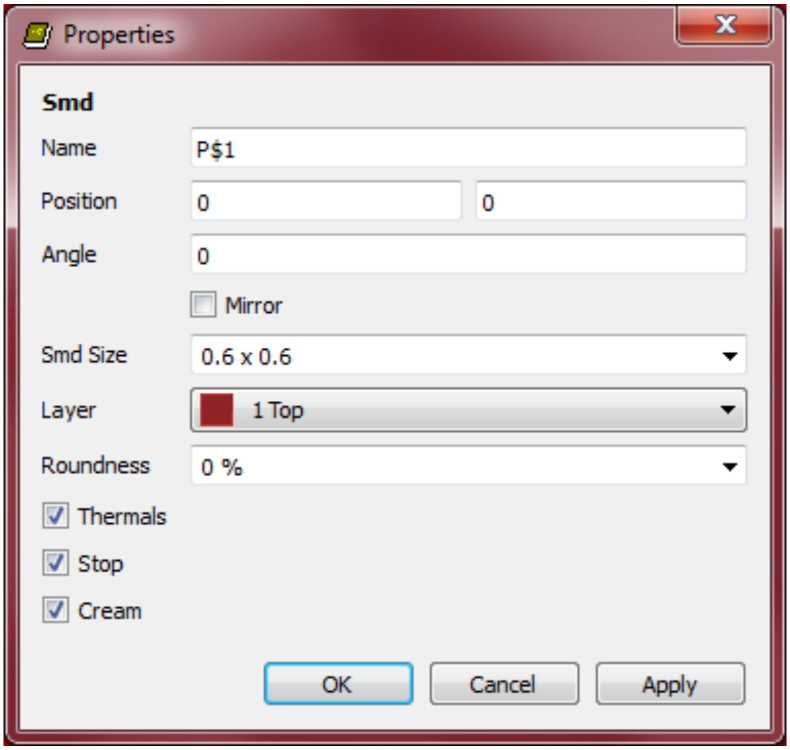
The best way to precisely position your pads is to manually enter their exact X + Y position (relative to center/home/zero) in the properties dialogue box.

You can also tweek the length and width of the pad and a bunch of other important settings here, but we'll cover some of that later.

Now that you've deselected the Smd tool (by selecting 'move' or something similar), move your mouse over your pad, **right-click on it** and select '**Properties**' from the popup menu:



You should see a dialogue box similar to the following:



Properties

Smd

Name: P$1

Position: 0 0

Angle: 0

☐ Mirror

Smd Size: 0.6 x 0.6

Layer: 1 Top

Roundness: 0 %

☑ Thermals

☑ Stop

☑ Cream

The properties dialogue box is extremely important, and you'll find yourself using this again and again throughout the part and PCB design process, so get acquainted with it now!

You can see that we have a textbox labelled '**Position:**'. This is where we can precisely place our pad exactly where it needs to be, using simple co-ordinates relative to the center/home/zero position.

Which brings us to our next logical question ... "How do we know exactly where to position our pads?!?"

Determine Pad Position Precisely

Going back to the datasheet we mentioned earlier, we can see that there are four 0.6 x 0.6 mm pads in our 2.0 x 1.6 mm rectangular footprint.

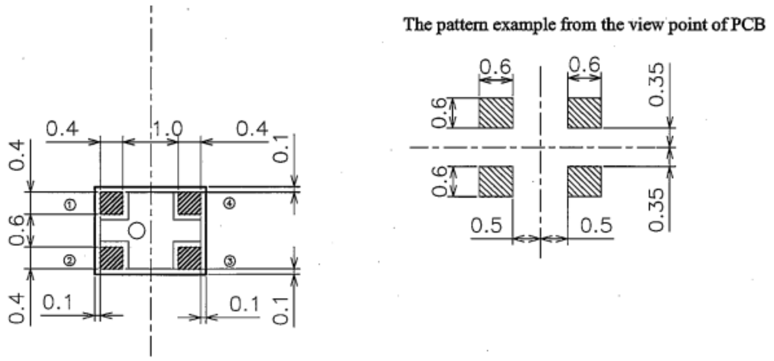
This is where you always need to do a bit of math, and calculate positions and sizes based on the data present in the datasheet.

Manufacturers will rarely display every possible value in the datasheet, but there is almost always enough information to infer the numbers you need with a bit of simple division, addition and subtraction.

The first pad that we'll place is the one in the top left-hand corner, which is marked as 'pin 1' in the datasheet:

> It will save time to start with pin one and then place the pads in the right numeric order, since Eagle will auto-increment the names to follow that same order

You can infer the exact center position of pad one by looking at the numbers available in the two illustrations below:



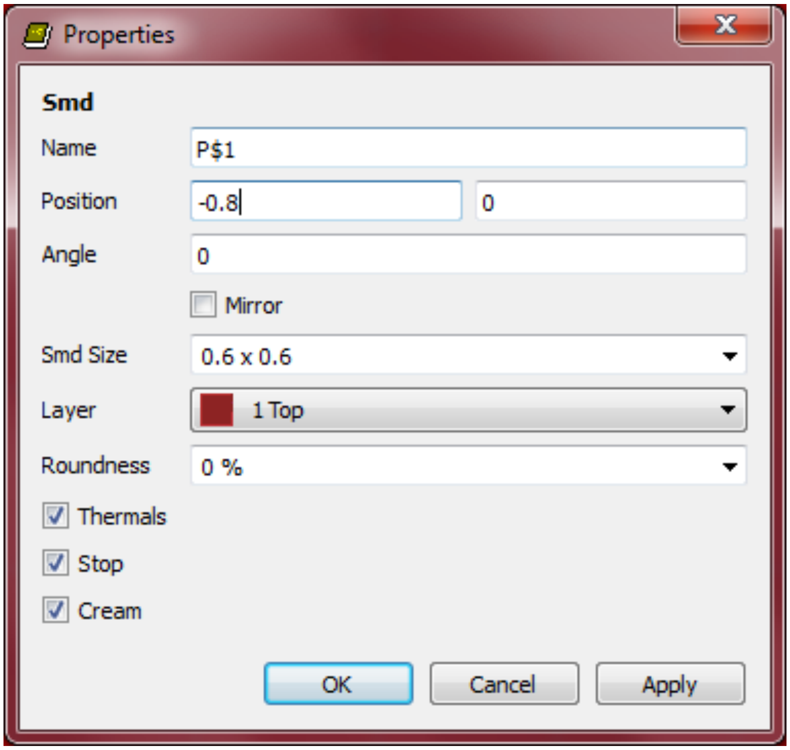
Looking at the recommended footprint in the image above (the image to the right), we can see that there is exactly 0.5mm distance between the right-hand edge of pad one (the top left pad) and the central '0' position on the X axis. There is also 0.35mm distance between the bottom edge of pad one and the '0' position on the Y axis.

Calculate the Pad's X Position

Since we know that pads in eagle as placed relative their center point, it's easy to calculate the exact 'X' position for this pad:

- Find the distance from the center of our part (the zero position indicated by the dash-dot lines) to the right-hand edge of our pad (0.5mm)
- Calculate the distance from the right-hand edge of our pad to it's center position, which is the pad width / 2 (0.6mm / 2 = 0.3mm)
- Add them together (0.5mm space + 0.3mm to the center of our pad), and we know that **our pad should be centered at -0.8mm on the X axis**.
- Now enter '**-0.8**' in the first textbox beside 'Position' in our properties dialogue:

NOTE: We enter a 'negative' value on the X axis since we want to position the pad 0.8mm before the zero or home position of our part ... entering a positive value would position it 0.8mm after the home position on the X axis


Properties

Smd

Name: P$1

Position: -0.8 | 0

Angle: 0

□ Mirror

Smd Size: 0.6 x 0.6

Layer: 1 Top

Roundness: 0 %

☑ Thermals

☑ Stop

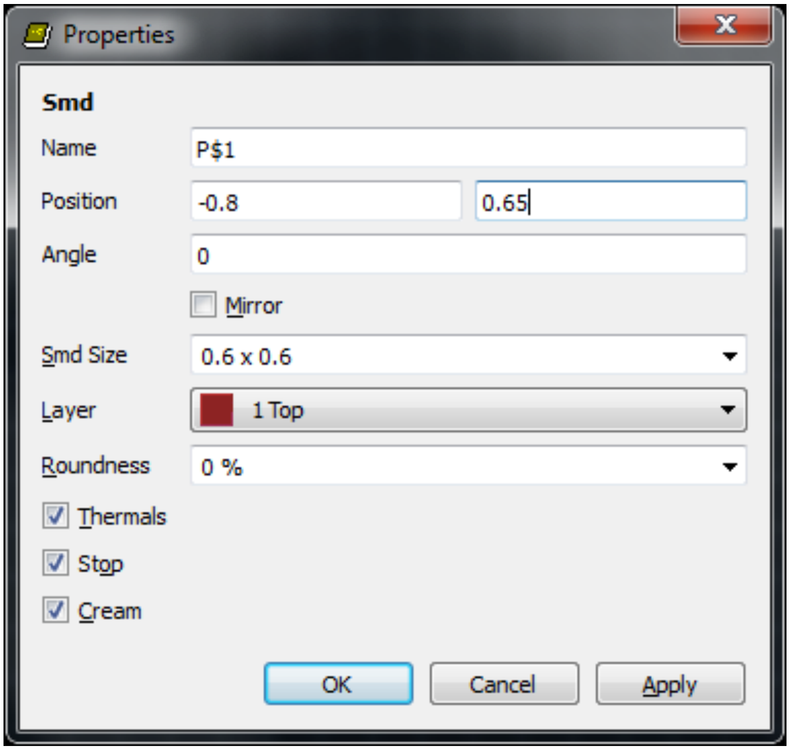
☑ Cream

Calculate the Pad's Y Position

Similarly, we can calculate the center Y position for the pad by taking the 0.35mm from the zero position to the bottom edge of our pad, plus 1/2 the pad height, meaning 0.35mm space + 0.3mm to pad center, = **0.65mm on the Y axis**.

Enter this value in the second box beside 'Position:' to set the Y location:

In this case, Y is a positive value since we want to place it 'above' the home or zero position of our part. To place it below the zero position on the Y axis we would enter -0.65mm. Everything is always positioned relative the center or home position!



Properties

Smd

Name: P$1

Position: -0.8 | 0.65

Angle: 0

☐ Mirror

Smd Size: 0.6 x 0.6

Layer: 🟥 1 Top

Roundness: 0 %

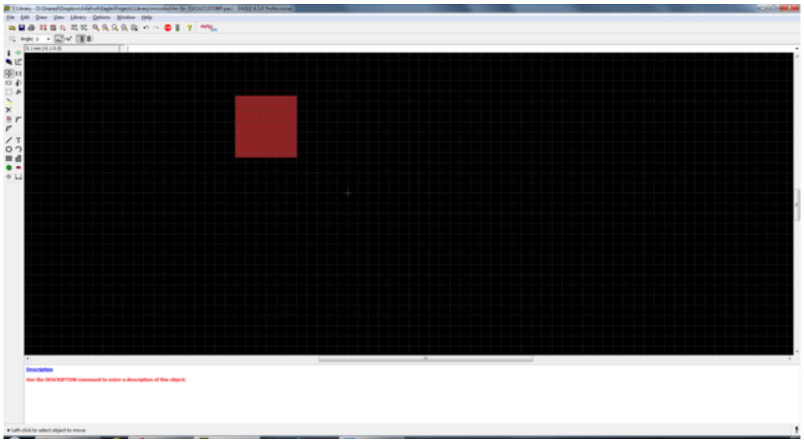
☑ Thermals

☑ Stop

☑ Cream

Now click '**OK**' and the pad should be centered at -0.8mm to the left and 0.65 mm to the top relative to the 'home' position for our package, mirroring the position of pad one in our datasheet:

> You might need to zoom out a bit to see your pad, using either the scroll wheel of the 'zoom' icons in the top toolbar



Description

Use the DESCRIPTION command to enter a description of this object.

Left-click to select object to move

That's it! You've now accurately sized, placed and positioned your first pad, and from here on in it's just a lot of rinse and repeat!

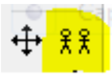
Copying Pads

Now that we know how to create, size and accurately position pads, we can go ahead and add the rest of our pads to the package.

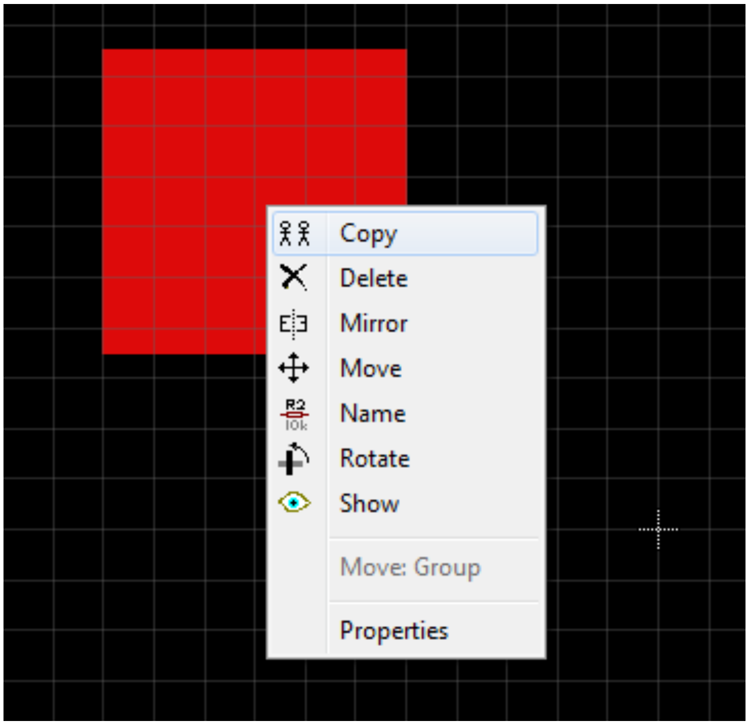
You can of course continue to do things the same way we describe in previous pages, but if you are placing a lot of identically sized and shape pads and you know the distance between the center points on either axis, there's a quicker way to do things:

Cloning Pads with the Copy Tool

The '**Copy**' tool -- shown as a creepy human cloning icon in the left-hand toolbar -- is your friend:



... or you can access it by right-clicking the pad or object you want to clone, and then selecting the 'copy' option from the popup menu:



This tool lets you make a copy of your pad (or whatever else), and keep all it's current settings (size, layer selection, etc.).

Just click somewhere else on your package to place a copy of it in approximately the right position. You can then precisely update the position the way we did with the first pad, via the **Properties** dialogue.

Copy and Place Pads in the Right Order!

Eagle automatically assigns new pad names based on the current pad name, so you should place the pads in the same order as they are numbered in the datasheet!

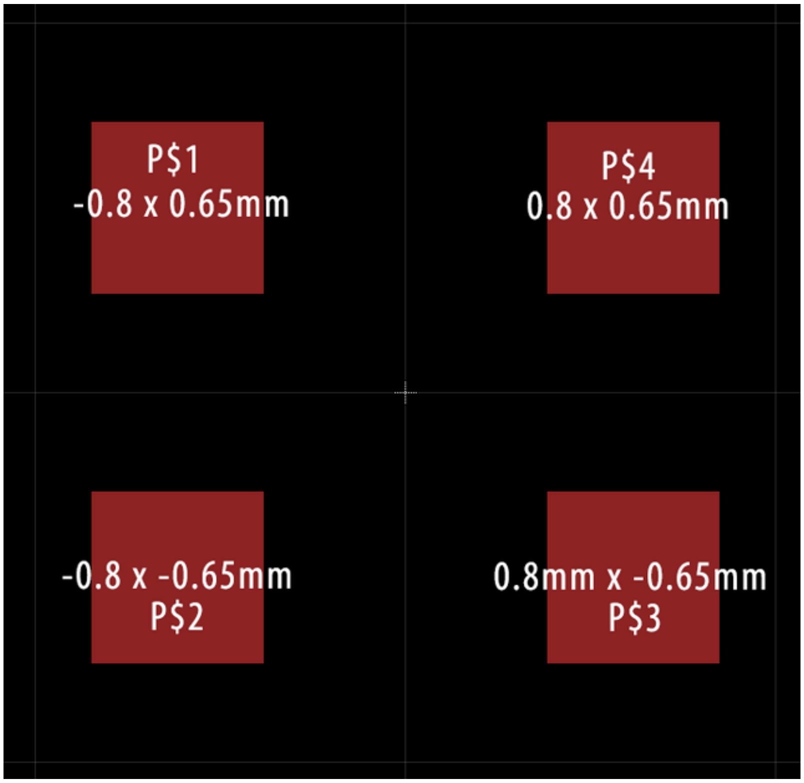
The first pad you placed will be named P$1, so the first time you copy this pad (or next time you create a new pad), it will be named P$2 (etc., etc.).

Take advantage of this time saving feature by placing your pads in the same order they are labelled on your package, counter-clockwise from pad 1 in the upper-left hand corner in this case: pin 1, then 2, then 3, then 4.

This will save you time later since you won't have to go in and rename parts manually, which can be annoying since you need to avoid duplicate names during the rename process.

Results

With a bit of cloning effort, you should end up with the following. We've added in the exact co-ordinates and names just for reference sake:



Essential Eagle Skills

Take Advantage of that Grid Tool!

What's great about having a Best Friend? You can call them up anytime to help you out when you have some particularly dull, monotonous task to do like moving!

If you have to copy or place a lot of identical pads at a known distance (such as an entire row of pads on an LQFP part), the best way to do things is the accurately place the first part in the row, and then use the **grid** to your advantage!

If you know that the center of the pads are consistently 0.5mm apart (often referred to as 0.5mm 'pitch'), go into the grid Dialogue and set it to 0.5, then copy and paste away. Every part will then snap to the next 0.5mm spot, and you can quickly place a whole row of parts accurately!

Setting Pad Names

The final step that you need to do creating the pads for your footprint is to make sure that they have appropriate 'names' -- specifically, that the pad names match the pin/pad numbers in the datasheet.

This is important because later on we will need to connect these 'pads' to our schematic symbol, and if we don't have the right names for our pads you can easily end up connecting something improperly and you'll either have a useless PCB or some potentially nasty and time consuming rework to do when it does arrive!

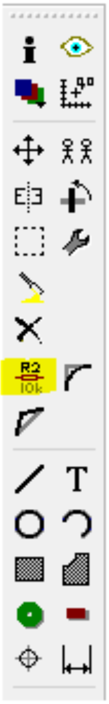
> A few extra minutes double-checking every pad and part of your footprint can save you hours or debugging and rework later, and more than a little bit of money on PCBs!

If you followed the advice in the previous pages, you 'created' the pads in the same order that they are listed in the datasheet, counter-clockwise starting with pad 1 in the upper-left-hand corner, and they should have the right numbers associated with them.

Just in case, though, you should always double-check your pin names before proceeding since this is an easy mistake to avoid, but one with significant consequence if we mess something up here.

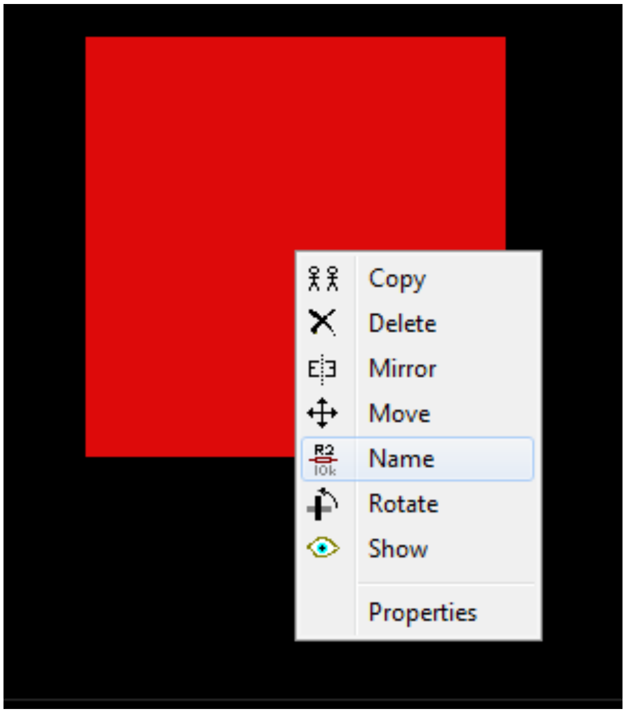
Verifying/Changing Pad Names

To view or modify the pad name, you can select the '**name**' tool in the left-hand toolbar, shown highlighted in yellow below:



Just select this tool, then click on any pad and it's name will be displayed, and you can update it if necessary.

Alternatively, you can also right-click on the pad and select 'Name' from the popup menu:



A Note on Default Pad Names

By default Eagle will assign pad names like **P$1**, **P$2**, etc., incrementing the number by one each time. This is perfectly normal for Eagle users, and what you will see in most footpints.

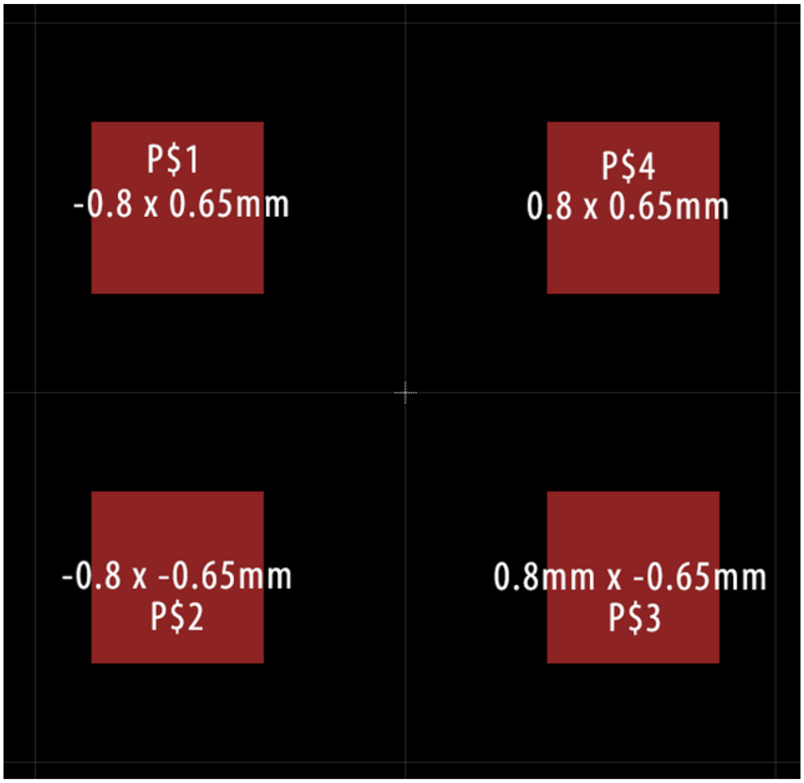
It's also acceptable to rename the pads to '1', '2', '3', though, instead of 'P$1', 'P$2, 'P$3'. The advantage is that this clutters up the schematics less, but it's purely a cosmetic change.

Any other change in names except to correct the pin numbers should be avoided unless you are very confident that you know what you are doing!

Be sure to following the pin numbers in the datasheet exactly, since this is what you will use to connect the pins later, and this is what anyone looking and your footprints will expect to see! Anything else is just asking for easily avoidable problems.

Correct Results

The pin names below are what you should have before proceeding, since this matches the pin names in the datasheet. If you don't get the same names, go back and correct them before continuing on to the next steps.



If you've made it this far, you actually have a fully functional package that you can connect up to any schematic symbol, and place it on your board.

Go ahead, pat yourself on the back! You've done the bare minimum of sizing and placing a set of appropriately named pads, and organizing them in a single package, which is the main requirement to move on to the next stage, creating a symbol for your schematic.

That said ... you'll thank yourself later for putting a bit of extra effort into things now. Accurate footprints are about more than a few pads and proper names.

An Introduction to Layers in Eagle

If you've made it this far, you actually have a fully functional package that you can connect up to any schematic symbol, and place it on your board.

Go ahead, pat yourself on the back! You've done the bare minimum of sizing and placing a set of appropriately named pads, and organizing them in a single package, which is the main requirement to move on to the next stage, creating a symbol for your schematic.

That said ... you'll thank yourself later for putting a bit of extra effort into things now. Accurate footprints are about more than a few pads and proper names.

Now it's time to add in some basic mechanical details, and later to make sure we have a way of identifying our parts once they get place on our PCB!

Before we can do that, though ... we need to make an important detour into something fundamental to working in Eagle ...

Decidedly Not Your Eagle BFF: An Introduction to Layers

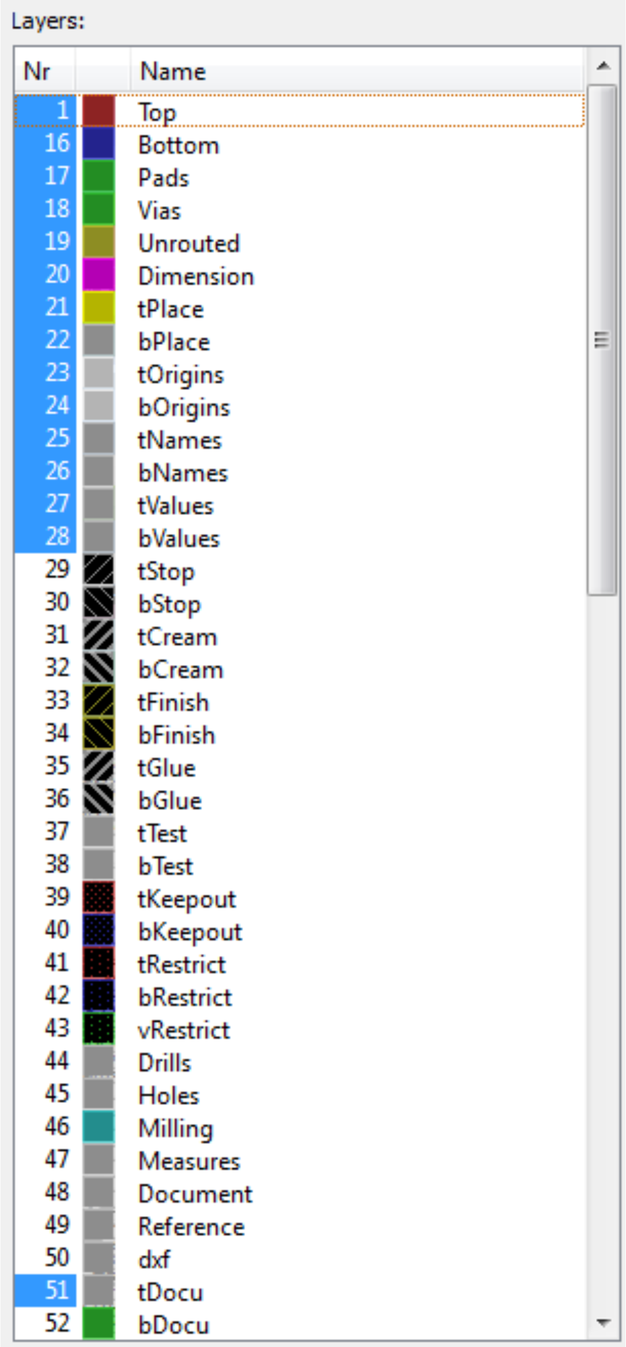
Layers are an essential part of Eagle. They're allow us to organize the many types of information that are required to make a PCB, and generate the documentation that we can share with other engineers or companies.

Layers are what allows us to generate the files that board houses can manufacture PCBs from.

That said, Layers is one of the more complicated things to wrap your head around if you're new to Eagle, so we'll try to explain the basics here.

Common Layers for Packages

Each 'layer' in Eagle has a dedicated number, and these layers are used to separate all the types of content that make up your PCB or your parts:



Layers:

Nr		Name
1	🟥	Top
16	🟦	Bottom
17	🟩	Pads
18	🟩	Vias
19	🟫	Unrouted
20	🟪	Dimension
21	🟨	tPlace
22	■	bPlace
23	■	tOrigins
24	■	bOrigins
25	■	tNames
26	■	bNames
27	■	tValues
28	■	bValues
29	▨	tStop
30	▧	bStop
31	▨	tCream
32	▧	bCream
33	▨	tFinish
34	▧	bFinish
35	▨	tGlue
36	▧	bGlue
37	■	tTest
38	■	bTest
39	▦	tKeepout
40	▦	bKeepout
41	▦	tRestrict
42	▦	bRestrict
43	▦	vRestrict
44	■	Drills
45	■	Holes
46	🟦	Milling
47	■	Measures
48	■	Document
49	■	Reference
50	■	dxf
51	■	tDocu
52	🟩	bDocu

These layers contain documentation details (layer 51), actual manufacturing info such as the 'paste layer' (layer 31), the copper layers for you pads (layer 1), etc.

Since each layer has it's own dedicated functions, it's important that you use the right layer for the right type of information!

Many tools in Eagle use layers for many different things, and thankfully they generally select the most appropriate layer by default, but for reference sake we generally the following 'layers' when designing packages:

- The **Smd tool** (to create pads) generally uses layer 1 to indicate which side of your PCB the pad should be placed on, though in very rare circumstances you may need to use layer 16 as well:

    - **Layer 1 (Top)** is used to draw pads on the top of the PCB
    - **Layer 16 (Bottom)** is used to draw pads on the bottom of the PCB
    - On multi-layer boards with the professional version of Eagle, you also have access to layers 2-15, but we won't worry about these for now

- The **Line Tool** can draw lines on any layer, but we'll see shortly there are two layers that are particularly important with this tool:

    - **Layer 21 (tPlace)**, which is used to draw lines that will be rendered as the silk screen on your PCBs (the printed text/lines/shapes we see)
    - **Layer t51 (tDocu)** is used for documentation purposes, such as drawing the mechanical dimensions of your part (more on that shortly!). Normally this layer isn't printed on the PCBs, but it's very important for documentation and for PCB design.

- When assigning **Names** and **Values** to packages (more on that shortly as well!), the following layers are used:

    - **Layer 25 (tNames)** is used to hold the unique 'names' for each part on your PCB (ex. C1, R5, X12, etc.)
    - **Layer 27 (tValues)** is used to hold the value for each part (such as 10K, 0.1µF, AT86RF212, etc.)

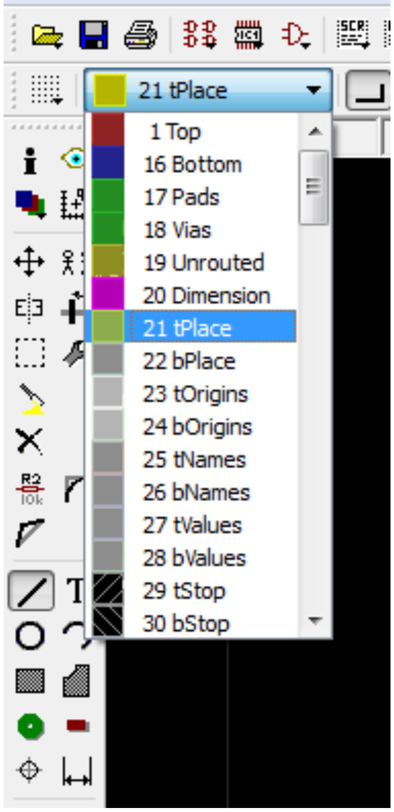
We'll cover these and other layers on an as-needed basis, and this will start to make sense once you work with them, but for now these are the main ones to familiarize yourself with.

Navigating Through Layers

The only important thing to know now, aside from having a general idea of what layers are, is how to switch between layers when you need to.

Any tool that works with different layers will expose the same '**Layer Selection**' drop-down box in the tool's command bar up in the top right-hand corner. You can see the drop-down box after selecting the line tool, for example, which will default to layer 21, which is the silk-screen for the top of your PCB.

We've extended the drop-down list out just for reference sake so that you can see some of the other layers you can select with this tool:



The best way to understand layers, though, is to start using them, so lets backtrack from this little side-route, and finish up the last little details of our new package!

Eagle is full of little shortcuts, and we can't cover them all here without overwhelming people, but one useful shortcut with the layer selection drop down is that you can select your tool (the line tool, for example), move your mouse over the layer selection box, and use the scroll wheel on your mouse to select layers. If you're switching between later 1 (top copper) and later 16 (bottom copper) this can save a lot of time.

Adding Dimension Details to your Package

Back from out layers sidetrack, let's finish off the package we started.

Now that we've placed and name our pads, the next step is adding some basic mechanical details on the part.

Adding some basic details on the physical size and shape of your parts lets you see exactly where the boundaries of your part are when you place it on your PCB. This is extremely useful since it allows you to visually avoid placing two parts in an overlapping position, or having your connectors hanging over the edge of a PCB, etc.!

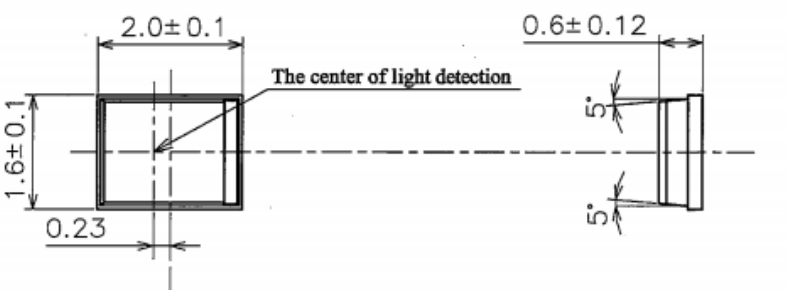
> Without this extra mechanical information you can very easily end up with a board you can't physically assemble, so once again ... a few minutes of effort and diligence now will save you many hours of headache later!

Drawing a Basic Package Outline

You don't need to go into a lot of detail with the dimension details, but as a minimum you should at least add a accurate rectangular package outline that represents the further dimensions of your part.

This is what you will use as a visual queue to make sure you don't have parts that overlap each other during assembly.

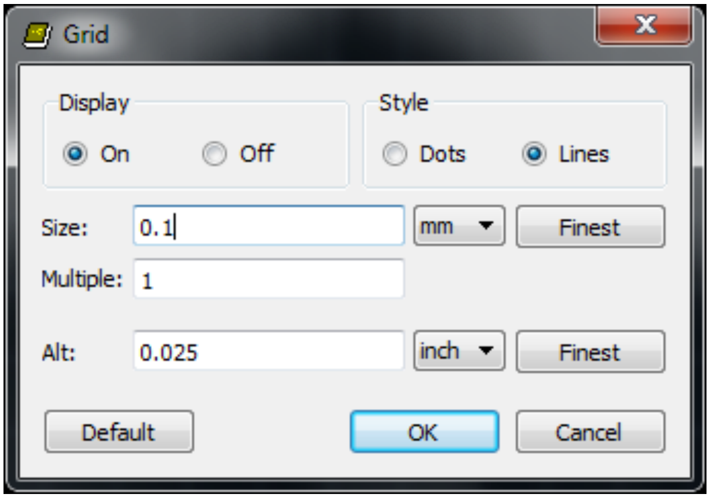
Looking back at our datasheet again, the basic part outline is easy to draw: it's a simple 2.0 x 1.6mm rectangle:



Step One: Check Your Grid

Before we start drawing the outline, let's do a quick double check to make sure the grid is set to something sensible.

Click the grid icon in the upper left-hand corner of Eagle. Since our outline is 2.0 x 1.6 mm, lets make sure the grid is at something like 0.1mm that works with both dimensions. Make sure the 'Display' radio button is set to 'On' as well:



Display: ● On ○ Off

Style: ○ Dots ● Lines

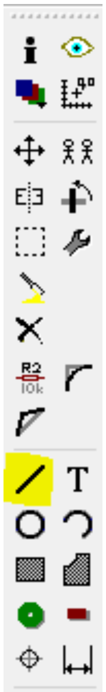
Size: 0.1 mm

Multiple: 1

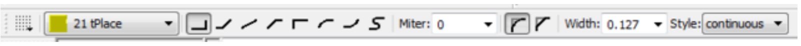
Alt: 0.025 inch

Step Two: Select the Wire Tool

Select the '**Wire**' tool in the left hand side of Eagle. It's a simple line, and this is exactly what it's used for ... to draw a line or a series of connected lines on whatever layer you select:



By default, the Wire tool will default to **Layer 21 (tPlace)**, which corresponds to the top silk screen layer on your PCB (where you would place outlines or text that you want printed on your PCB):



21 tPlace

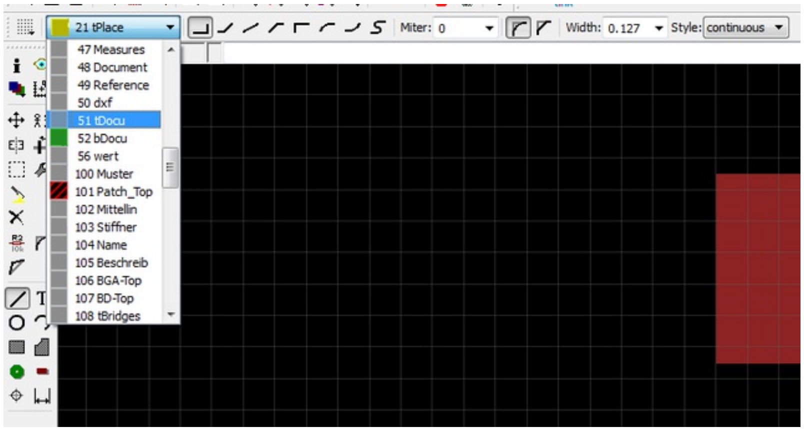
Miter: 0

Width: 0.127

Style: continuous

We'll use layer 21 shortly, and it's a very important layer, but part outlines normally aren't printed on the PCB, they're only important when doing your PCB layout or perhaps for documentation purposes layer to render high quality images of your PCB.

With the 'Wire' tool still selected, switch over to **Layer 51 (tDocu)**. The tDocu layer is used for documentation purposes (51 or tDocu for the top of your PCB, 52 or bDocu for the bottom of your PCB), and this is where you normally draw any extra mechanical information that might be useful during assembly.



21 tPlace

Miter: 0

Width: 0.127

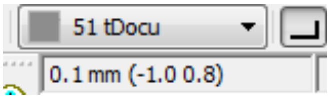
Style: continuous

Step Three: Draw Your Outline

Now that we've selected the 'Wire' tool, and we're working on the right layer, we can start drawing our outline!

Move your mouse over to the -1.0 x 0.8 position, which corresponds to the top left corner of our package. This is why it's useful to have the grid setup to something like 0.1mm units!

You can see your exact position by looking at the little box in the top left-hand corner of Eagle:

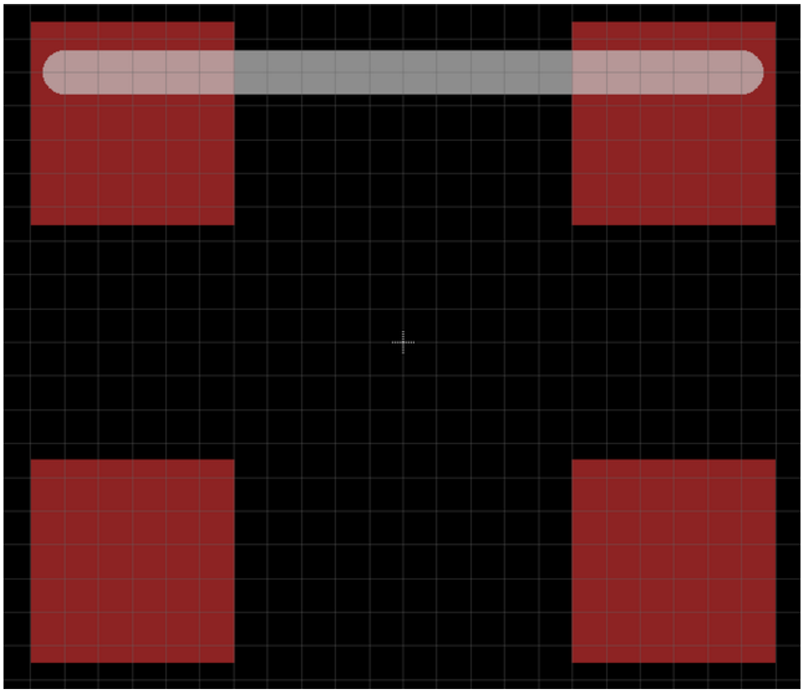


51 tDocu

0.1 mm (-1.0 0.8)

We'll start here since our part is 2.0 x 1.6mm, so -1.0mm x 0.8mm is 1/2 the package size, taking into account the fact that our part is layed out around it's mechanical center.

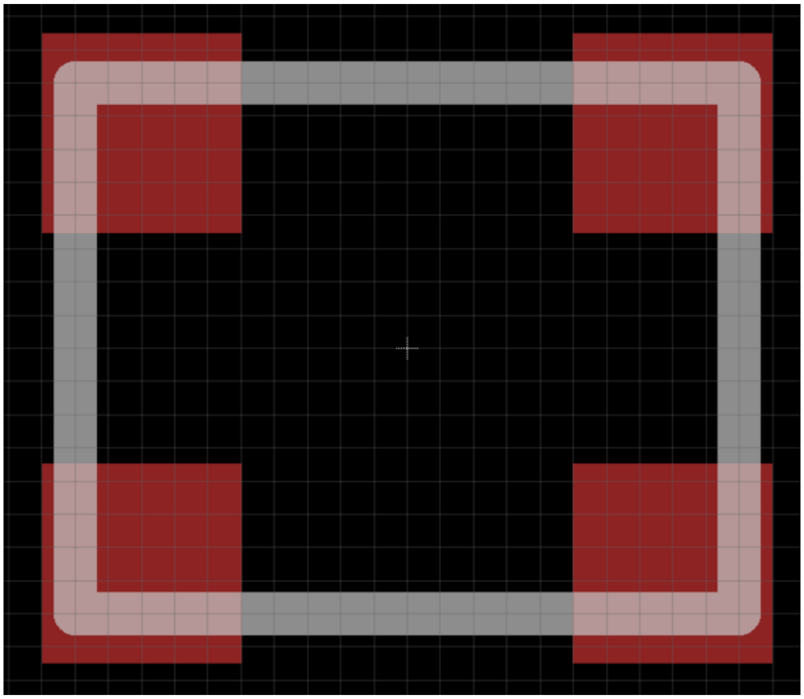
Click once at -1.0 x 0.8, and then move the mouse over to 1.0 x 0.8, which should results in something like the following image:



Now continue the outline:

- Draw a new line down to 1.0 x -0.8
- Then head over to -1.0 x -0.8
- The head back up to 1.0 x -0.8, being careful to click on out starting point
- Next, **press the escape key to stop using the 'Wire' tool!**

You should end up with something like this:

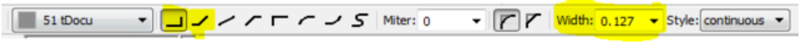


That's it! You now have an accurate mechanical outline of your part, so you know exactly what your PCB will look like when you start assembling the prototypes!

Essential Eagle Skills

You can adjust the width of the lines drawn with 'Wire' tool (or any other tool) in the dedicate toolbar at the top, as well as the way that lines are drawn (only right angles, right or 45° angles, any angle, etc.).

The most important icons are highlighted below, but feel free to explore them all since you'll need to get to know these as you work with Eagle:



51 tDocu

Miter: 0

Width: 0.127

Style: continuous

For now the default values are fine, though!

Adding a Silk Screen Outline to your Package

While not strictly necessary, adding an additional outline that goes on the silkscreen of your PCB can be very useful during placement, since it's a good visual indicator of where you should place the part.

This isn't a replacement for the dimension details we added in the last step -- The silk screen on your PCBs isn't very reliable since it often gets shifted a bit to the side during manufacturing -- but it's worth adding as a visual indicator for assembly.

The process is identical to the way we added our dimension details in the last step, but there are a few things to keep in mind here:

Tips on Creating Proper Silk Screen Outlines

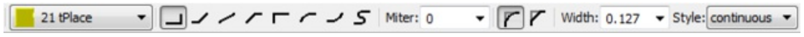
The following guidelines are a good starting point when adding a printer outline to your PCB:

- Always place the lines or text on **Layer 21 (tPlace)**!
- Be sure not to place any lines on top of the copper pads, since **you can't print silk screen on bare copper pads!**
- Create your outline slightly larger than the actual physical dimensions, so that you can still see the outline once the part is physically place (perhaps 0.1 - 0.2mm larger on each side)

Drawing Our Silk Screen Outline

The silk screen outline is drawing exactly the same way as out dimension details in the previous step ... using the 'Wire' tool.

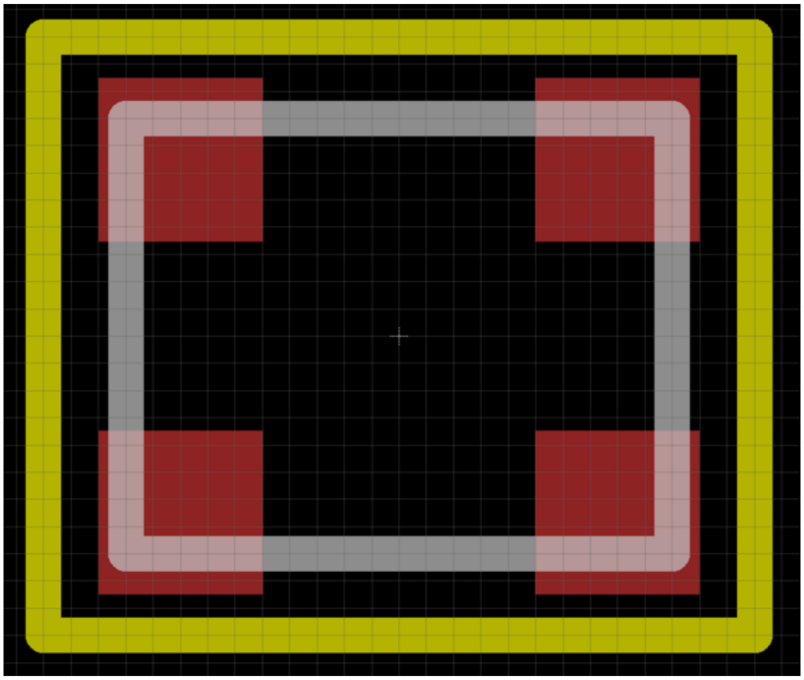
The important thing here is to make sure that you are on the right layer, **Layer 21 (tPlace):**



21 tPlace
Miter: 0
Width: 0.127
Style: continuous

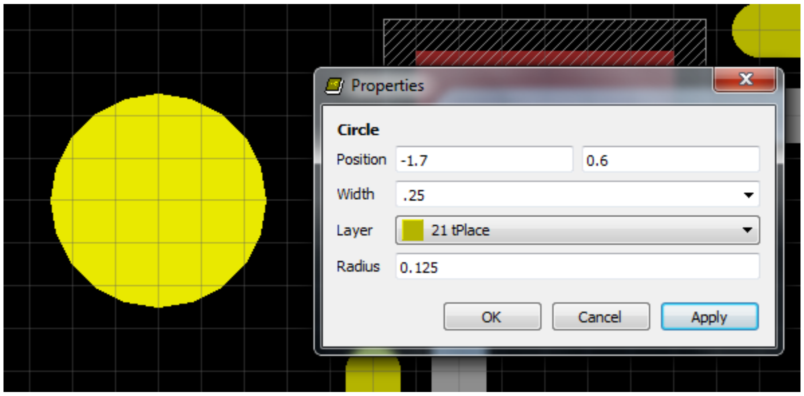
Once that's done, just draw a rectangle that's slightly larger than the mechanical dimensions of the board, making sure that **we don't place the lines over any copper pads!**

Something like this should be fine. It's a bit bigger than I would usually place, but this avoids any overlap on the pads, and everything is smaller looking than you think when you actually see it on a PCB:



Orientation Indicator

It's also important to indicate the location of pin 1 on our packages, to make sure they get placed in the right orientation. We can do this by selecting the '**circle**' tool and drawing a small circle on the silk screen layer beside pin one, as follows:



Properties

Circle

Position -1.7 0.6

Width .25

Layer 21 tPlace

Radius 0.125

You can tweek the '**Width**' and '**Radius**' settings until you get a solid circle about the size that you think its appropriate.

Adding >Name and >Value to the Package

The last step in our package (!!!) is to add two essential bits of meta data.

Every part that gets place on your schematic (and eventually your PCB) gets assigned a unique 'Name', which is used to distinguish parts during board assembly, or just to communicate basic information in an intelligent way.

Saying 'Change R23 on your board to a 10K resistor' is a whole lot less error prone than 'Change the third resistor down, just after the USB connector on the left hand-side your PCB to a 10K resistor'.

It's a good idea to have these names (R23, C14, U1), as well as the values (10K, 0.1µF, AT86RF212) visible on both our schematic and our PCBs. So how do we do that?

Meet tNames, tValues ...

Eagle has two dedicated layers to handle exactly this kind of information on our PCBs:

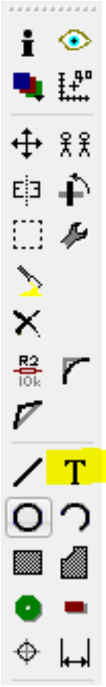
- **Layer 25 (tNames)**, which is used to store part names (R1, etc.) on the top layer of our PCBs
- **Layer 27 (tValues)**, which is used to store part values (10K, etc.) on the top layer of our PCBs

> Don't worry if you find all this layer stuff confusing. With time it becomes second nature, so don't get discouraged or overly frustrated if you have to keep looking back here to know what layer certain things should go on!

We'll use these two layers, plus the 'Text' tool to place two magic labels on your board, one that Eagle will automatically update with our part name, the other that Eagle will update with our part value.

... and the Text Tool Too!

So how do we place these magic layers? With the 'Text' tool, of course:

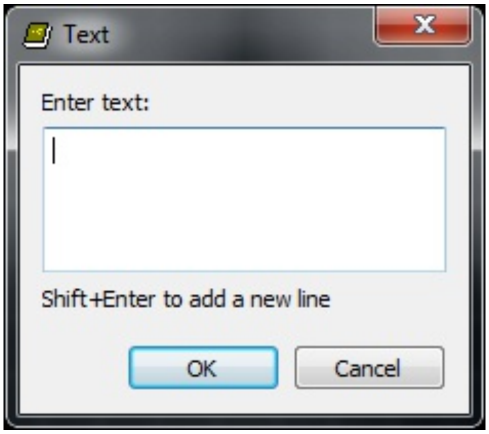


Place The >Name Placeholder

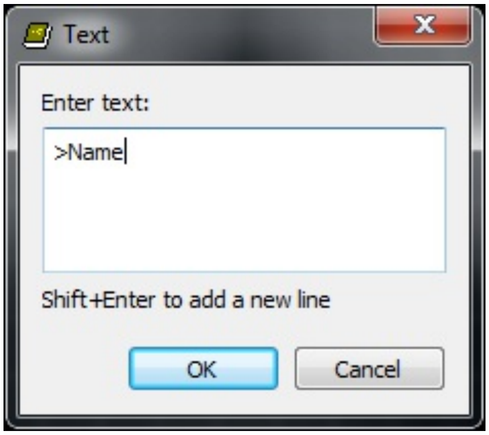
The first thing you need to do with any new package, is add a piece of text with '**>Name**' somewhere appropriate in your package.

> The '&gt;' character before name is a special character that lets Eagle know that this value should be dynamically updated by the program. If you just entered 'Name', Eagle will leave the label as is, which isn't very helpful during assembly since every part will have the same identifier on the PCB!

Click on the '**Text**' tool from the side toolbar, and you should see a popup Window similar to the image below:

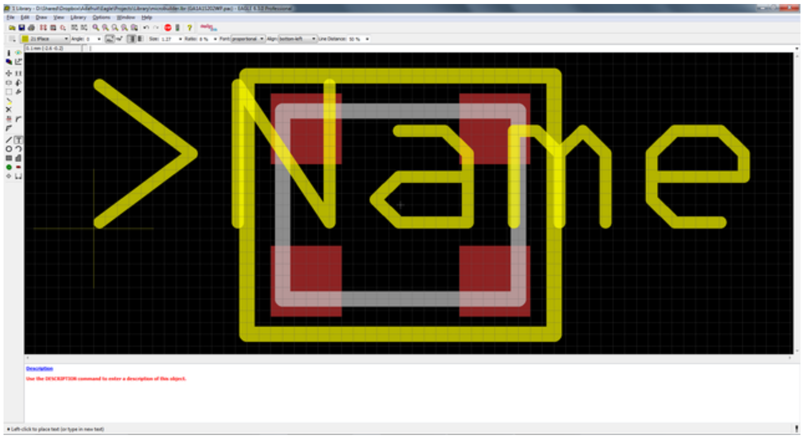


Now enter '**>Name**' into this text box and click the OK button:



Enter text: >Name

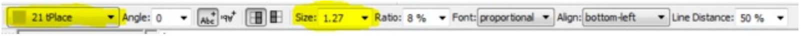
You should end up with something like this on your screen:



While the idea here is basically good, the default text is way too big for us, and it's also on the wrong layer since the 'Text' tool defaults to layer 21 (which is the top silkscreen, the same layer we used for the visual outline in the last step).

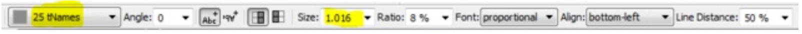
Adjustment Option 1: Use That Toolbar!

With the text still enabled (don't click anywhere yet and don't pressed escape!), you can either move your mouse up and adjust the current layer and text size in the 'Text' toolbar in the top of Eagle (highlighted below) ...



21 tPlace Angle: 0 Size: 1.27 Ratio: 8 % Font: proportional Align: bottom-left Line Distance: 50 %

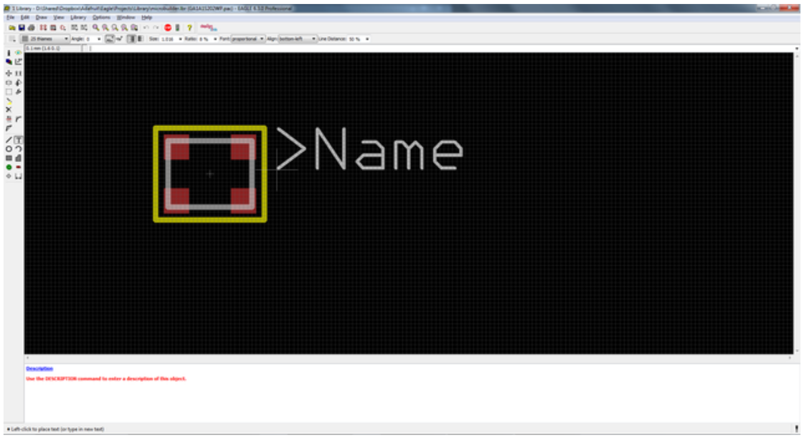
Which can be change to **layer 25**, with **size 1.016:**



25 tNames Angle: 0 Size: 1.016 Ratio: 8 % Font: proportional Align: bottom-left Line Distance: 50 %

Now the text should be a bit smaller, and on the right layer, and you can zoom out a bit with the scroll wheel on your mouse (or the zoom icons in the top toolbar), and place the '>Name' label to the side out our part, as follows:

The text should be a different color than the original yellow, since all layers in Eagle are color coded. If you've selected the right layer it should be a light gray color by default!



Click the mouse button to place the '>Name' label somewhere, and then **press the escape key twice**!

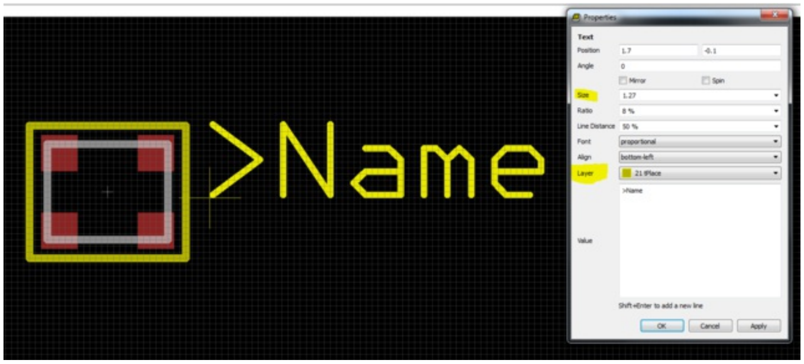
Pressing the escape key once will abort placing '>Name' and send you back to the text dialogue box, and pressing escape the second time will close the dialogue box and send us back to Eagle.

If you're not happy with the position of your text, just select the 'Move' tool and tweek the position a bit, but the position in the image above is generally good.

Adjustment Option Two: Eagle BFF #2, the Properties Window

Remember Eagle BFF #2? You can also use that to adjust all of these properties. Simply place the '>Name' text in an appropriate position on your package, and after pressing escape twice to get out of the text tool, **right-click on the label and select the 'Properties' menu item**.

You can adjust the layer and size of your text in the Properties Window, shown highlighted below:



Properties

Text

Position: 1.7 -0.1
Angle: 0
☐ Mirror ☐ Spin
Size: 1.27
Ratio: 8 %
Line Distance: 50 %
Font: proportional
Align: bottom-left
Layer: 21 tPlace
Value: >Name

Shift+Enter to add a new line

- Change the layer to **25 (tValues)** and the **Size to 1.016**, then click the OK button, and the color should change to gray, which is the default for layer 25 (as seen in final image in option one above).

That's it ... once last text addition, and we're (at long last) done with our package!

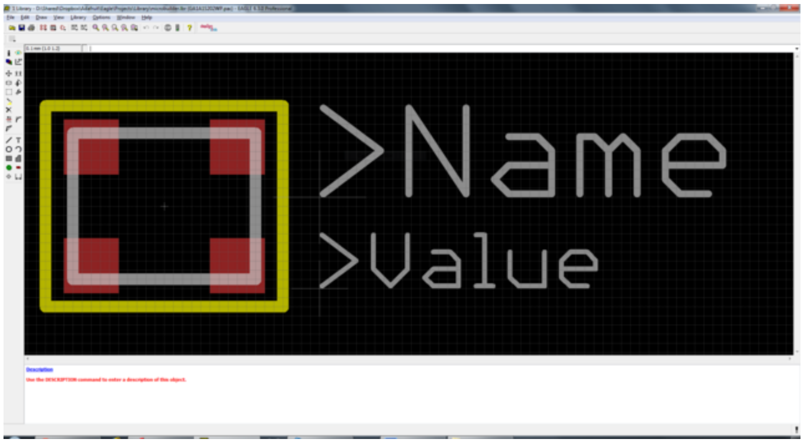
Place the >Value Placeholder

Similar to the >Name Placeholder, you want to place another text label with '>Value', which Eagle will automatically update to contain the parts value. Follow exactly the same process described above, with these two exceptions:

- **The >Value text should be place on Layer 27 (tValues)**
- **The size can be a bit smaller (perhaps 0.6096)**

Final Results

You should end up with something like this for your final results:



Use the DESCRIPTION command to enter a description of this object.

Essential Eagle Skills

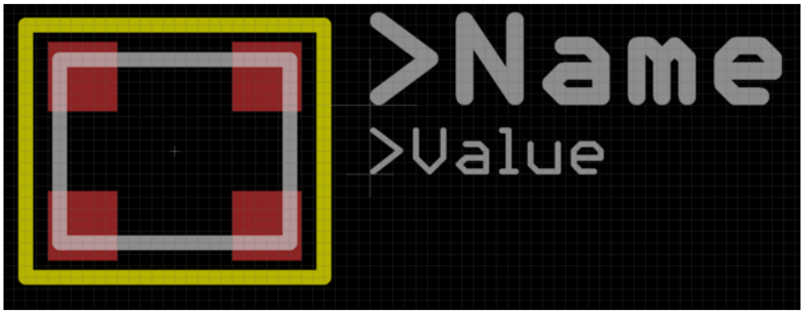
The exact size of your >NAME and >VALUE text is really a matter of personal taste, and with time you'll find your own way ... but you should at least be consistent about it.

Every part in my own library uses the following values (numbers shown on a mm grid!):

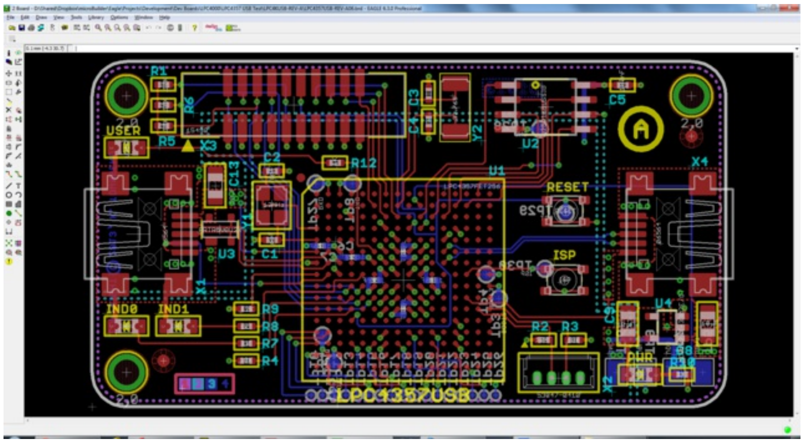
Name: Size = 0.8128, Ratio = 18%
Value: Size = 0.4064, Ratio = 10%

These are probably much smaller than many people may want -- the names aren't easily readable with a naked eye if you don't have good vision -- but I do a lot of boards with 0402 discretes, and tightly packed parts, so it's important to me to keep the labels small. During PCB layout I place the 'values' inside the parts so I know what values are related to what parts during assembly.

You can see this part with my own preferred label sizes below, but with enough patience you'll find your own style and figure out the values that work for you. For now, it's better to start bigger and simpler with the values use earlier in this tutorial:



Here's an example of a board using these sizes next to a few 0402 discretes, and you can see how I place the >Name beside the parts, and the >Value inside so that I know what's what during assembly. Names on the top layer are in blue, and names on the bottom layer are in gray:

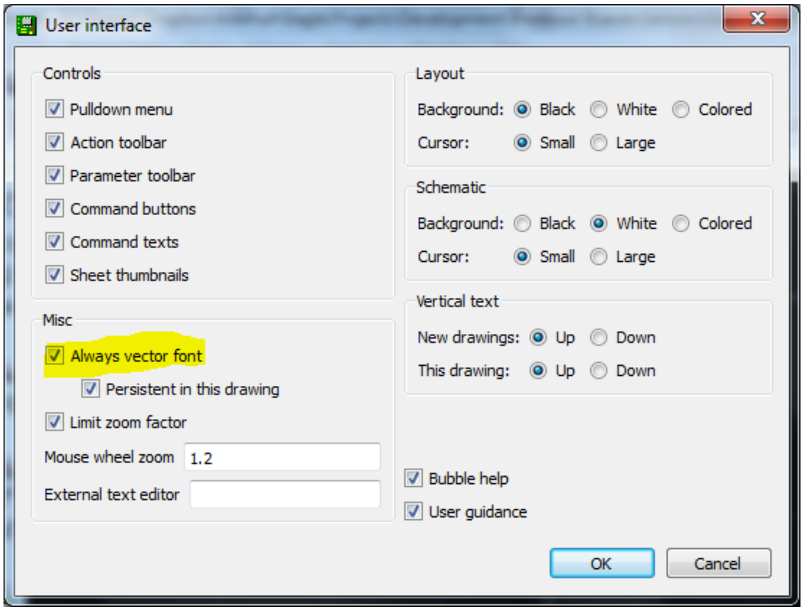


Pro Tip: You can move the Name and Value items around on your PCB via the 'Smash' tool. Just 'Smash' your parts on the PCB, and you'll be able to move and resize them right in the board editor!

Essential Eagle Skills: Use Vector Fonts!

It's important to make sure you have vector fonts enabled in Eagle.

By default, vector font **aren't** enabled, and not using them can make a mess of your Gerber files when it's time to go to production, so go to your '**Options > User Interface...**' menu item, and enable the checkbox below if it isn't already. Your board manufacturer will thank you for it:



User interface

Controls

☑ Pulldown menu
☑ Action toolbar
☑ Parameter toolbar
☑ Command buttons
☑ Command texts
☑ Sheet thumbnails

Misc

☑ Always vector font
☑ Persistent in this drawing
☑ Limit zoom factor

Mouse wheel zoom: 1.2

External text editor:

Layout

Background: ● Black ○ White ○ Colored

Cursor: ● Small ○ Large

Schematic

Background: ○ Black ● White ○ Colored

Cursor: ● Small ○ Large

Vertical text

New drawings: ● Up ○ Down

This drawing: ● Up ○ Down

☑ Bubble help
☑ User guidance

The Symbol Editor

Now that we've successfully created a new package for the GA1A1S202WP, we have something we can place on our PCB and reliably manufacture.

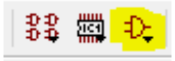
The next step is getting the part onto a schematic to connect it to other components and devices.

This is the purpose of 'symbols' in Eagle. Symbols are used as a visual representation of our physical part, including the individual pins or pads. They act as a conceptual bridge between the functional intent of our schematic and the physical or mechanical package itself.

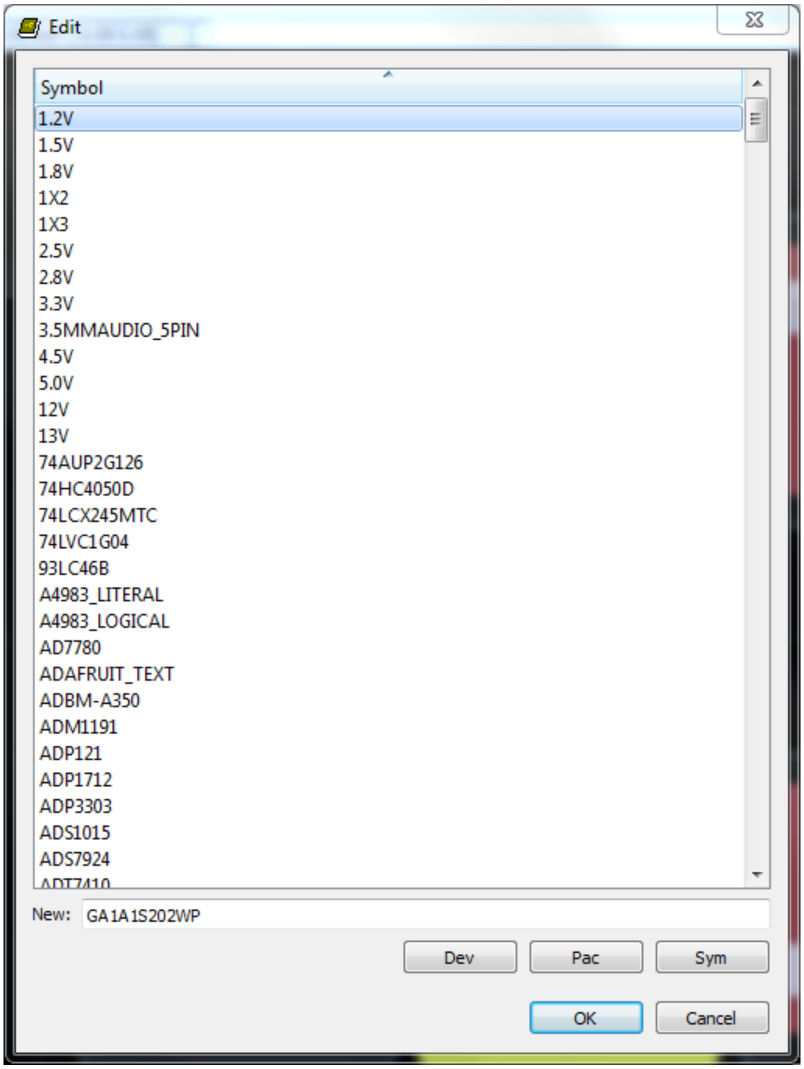
Symbols are useful as an abstraction layer since we can have one symbol, for example for a resistor, with many packages associated with it (0402, 0603, 0805, 1206, etc.). By separating the two, we can easily change the physical package used on our board without changing the entire schematic.

Creating a New Symbol

Creating a symbol is exactly the same as creating a package. Simply click the 'Symbol' icon at the top of your library editor ...



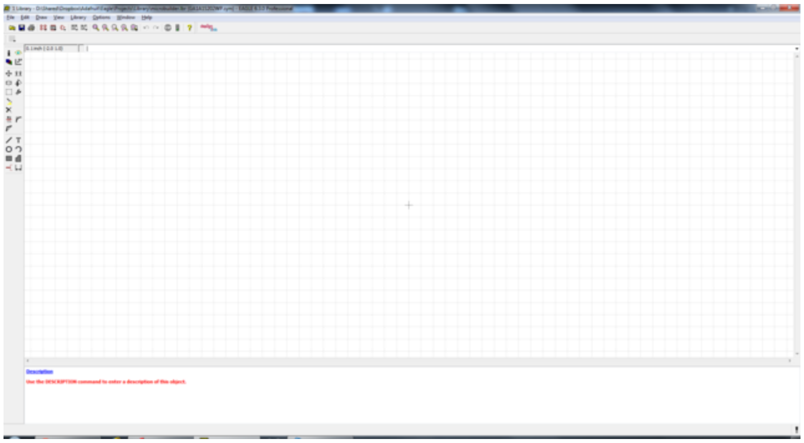
... then enter a unique name for your symbol ('GA1A1S202WP' is used below) and click '**OK**'.



1.2V
1.5V
1.8V
1X2
1X3
2.5V
2.8V
3.3V
3.5MMAUDIO_5PIN
4.5V
5.0V
12V
13V
74AUP2G126
74HC4050D
74LCX245MTC
74LVC1G04
93LC46B
A4983_LITERAL
A4983_LOGICAL
AD7780
ADAFRUIT_TEXT
ADBM-A350
ADM1191
ADP121
ADP1712
ADP3303
ADS1015
ADS7924
ADT7410

New: GA1A1S202WP

This will create a new symbol and open it up in the symbol editor, and you should see a screen similar to the following:



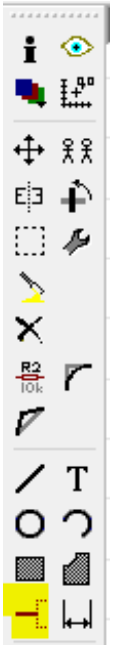
Description

Use the DESCRIPTION command to enter a description of this object.

Adding Pins to Your Symbol

The main purpose of symbols is to map the pins or pads on your part from your schematic to the physical package.

In order to do this, we use the 'Pin' tool, which can be found in the symbol toolbar on the left-hand side of the editor (highlighted in yellow below):

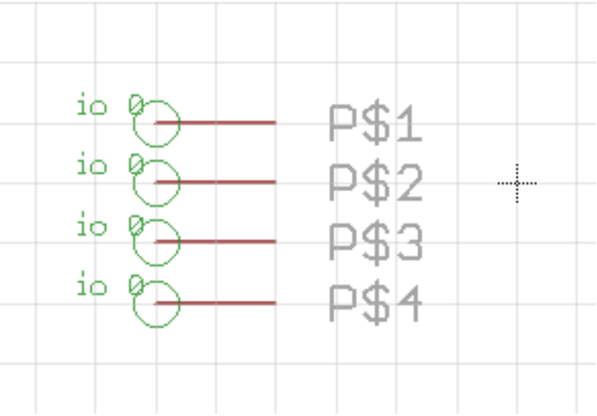


Adding Pins to Your Symbol

To place a pin in your symbol, simple select the 'Pin' tool, and then click anywhere on your symbol, which will add a new pin which we can configure later.

Since the GA1A1S202WP has four physical pads, we'll place fours pins in our symbol, once for each pad.

This should give you something similar to the following once placed:



You'll noticed that the pin tool is similar to the 'Smd' tool in the parts editor ... the pins are automatically assigned unique names, starting at P$1, and incrementing by one for each new pad.

Assigning Pin Names

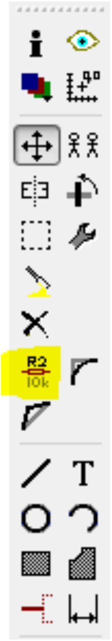
In order to make our schematics useful to end users, it's important to assign useful names to your pins.

Gnd pins should be assigned the name 'Gnd' (or sometimes VSS, depending on the naming conventions you want to use), and SPI pins should be assigned useful names like MISO, MOSI, SCK, and CS, etc.

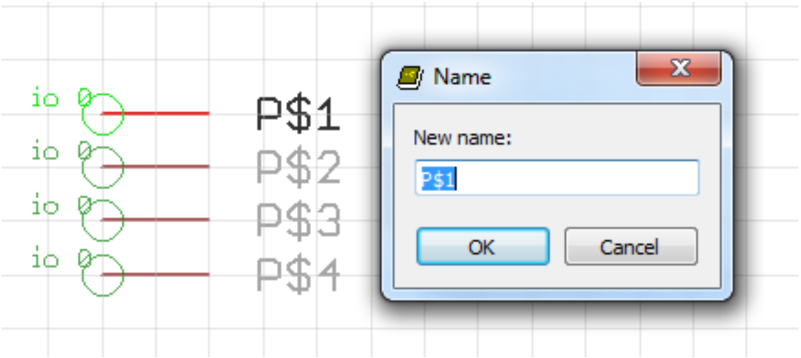
Most of the time, the datasheet will contain a schematic symbol we can copy, though, including pin names. The GA1A1S202WP datasheet doesn't contain one, but there are only four pins so it's easy to figure out a basic layout and pin names.

Renaming Pins

Renaming pins in the symbol editor is the same as renaming pads in the package editor ... you simply select the 'Name' tool from the toolbar (highlighted in yellow below) ...



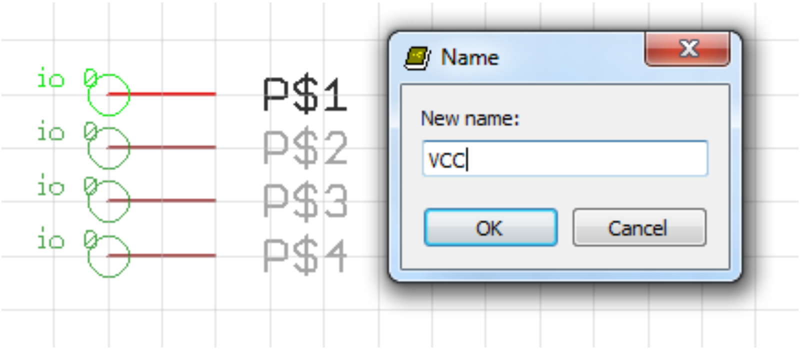
... then click on your pin, and assign a unique name to it via the popup dialogue box:



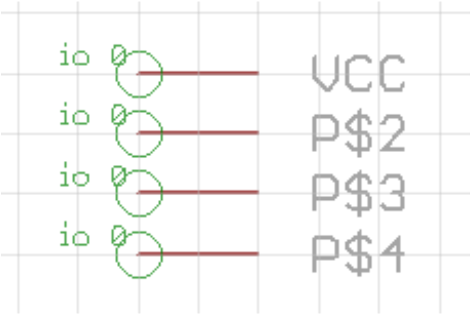
In this case, our device as the following four pins:

- Pin 1 = GND
- Pin 2 = GND
- Pin 3 = IO (Analog Output)
- Pin 4 = VCC (the power supply for this device)

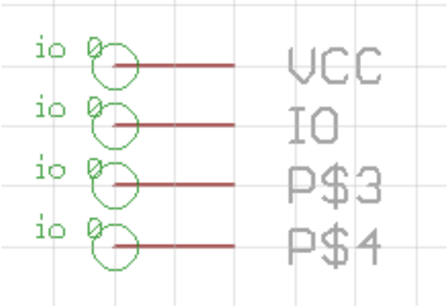
Lets start with VCC, renaming P$1 to 'VCC', which should give you the following results:



... and after clicking 'OK' ...



Now do the same for P$2, renaming it to 'IO', which will result in the following:



Shared Pin Names

This leaves us with the last two pins, both of which are 'Gnd', but it also introduces a problem.

Like pads on your package, **pins must all have unique names**!

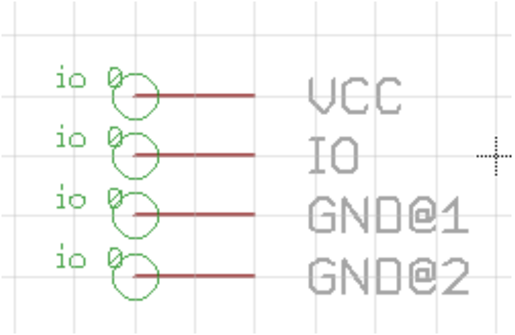
When you have several pins with the same name and functionality (which is often the case with 'Gnd', 'VCC, 'VDD', etc.), we can assign the common name followed by the '@' symbol, and a unique number.

When Eagle renders our symbol on the schematic the @1, @2, etc., part of the name will be hidden, and only the common name preceding the @ will be shown.

Try it out by assign the following names to the last two pins in our symbol:

- GND@1
- GND@2

which should give you something similar to the image below:



That's it! You have the absolute minimum requirement for your symbol, and we can move on to the final step in the part design process, the 'Device' ... but as with the package, it's worth spending a few more minutes and do things properly in our new symbol.

Drawing a Symbol Outline

While a set of pins is technically all that's required to pair a symbol to your physical package, schematics have a very rich visual language around them that engineers across the globe know and expect to see.

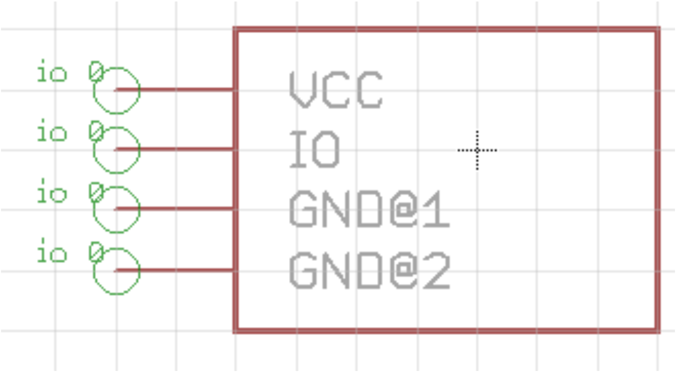
Every engineers knows the difference between a resistor and a capacitor on a schematic because there are a few agreed upon 'symbols' for these parts.

It's important to respect the expectation of other engineers and end users when making new schematic symbols. For common parts, try to follow existing conventions, and do a quick search online before making your own symbols.

Creating a Symbol Outline/Shape

We can use the '**Wire**' tool (which functions like the 'Line' tool that we used in the package editor) to draw a visual symbol for our part in the symbol editor.

The GA1A1S202WP is an extremely basic IC, and doesn't really match an standard four pin symbol, so we can simply draw a box with the four pins sticking out of it for our schematic symbol, similar to the following:



By default the 'Wire' tool will place the lines on layer 94 (Symbols), which is generally the layer you should use for this information.

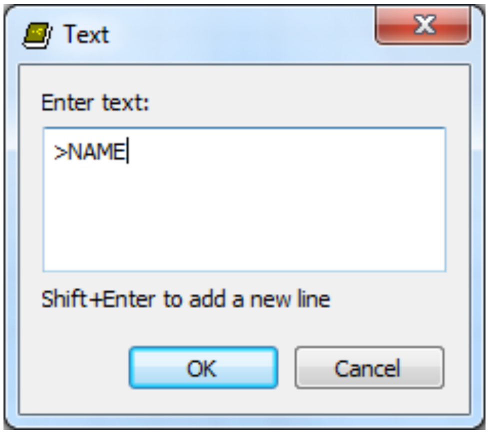
Adding >Name and >Value to Your Symbol

Similar to our package, it's important to add a '>Name' and '>Value' label on our symbol so that we can meaningfully identify parts, and display the part values on our schematic.

While it's helpful to drop 10 resistors on your schematic, it doesn't help anyone if you have no way to display what value those resistors should have!

Adding >NAME

To add >NAME to our symbol, simply select the '**Text**' tool in the left-hand toolbar, and we should get the following dialogue box, where we can enter our text:



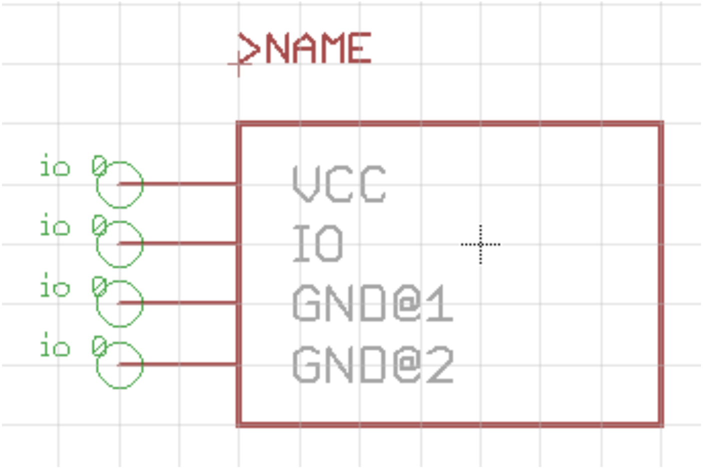
Text

Enter text:

>NAME

Shift+Enter to add a new line

After clicking the 'OK' button, we can place the text on top of our part outline, as follows:

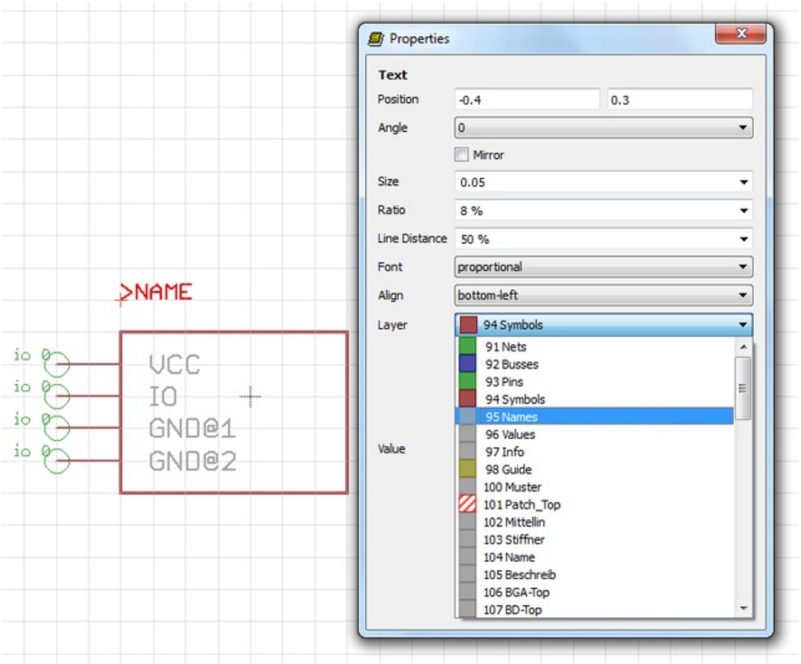


As with our package, though, the text should go on a layer dedicated to names, which in the case of our symbol is **Layer 95 (Names)**.

To change the layer of the '>Name' text, simply right-click on the text, and select the

'Properties' item at the bottom of the context menu.

From the 'Properties' dialogue box, changed the text layer to 95, as shown in the image below with the layer selection drop down expanded:



Properties

Text

Position -0.4 0.3

Angle 0

□ Mirror

Size 0.05

Ratio 8 %

Line Distance 50 %

Font proportional

Align bottom-left

Layer 94 Symbols

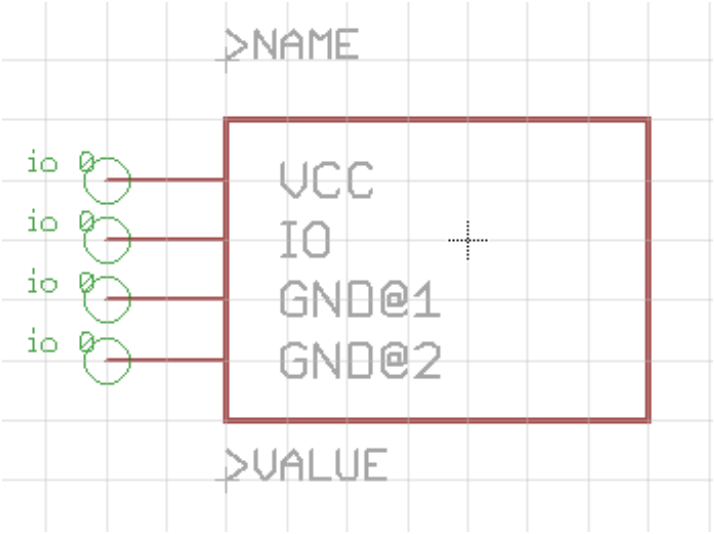
Value

If you did this correctly, the >NAME text should turn gray, to reflect the default color of layer 95!

Adding >VALUE

Adding the >VALUE label is identical to the way we added >NAME, except that >VALUE should be plaved on **Layer 96 (Values).**

After adding the text and placing it on the correct layer, you should end up with something resembling the following in your symbol editor:



Final Pin Adjustments

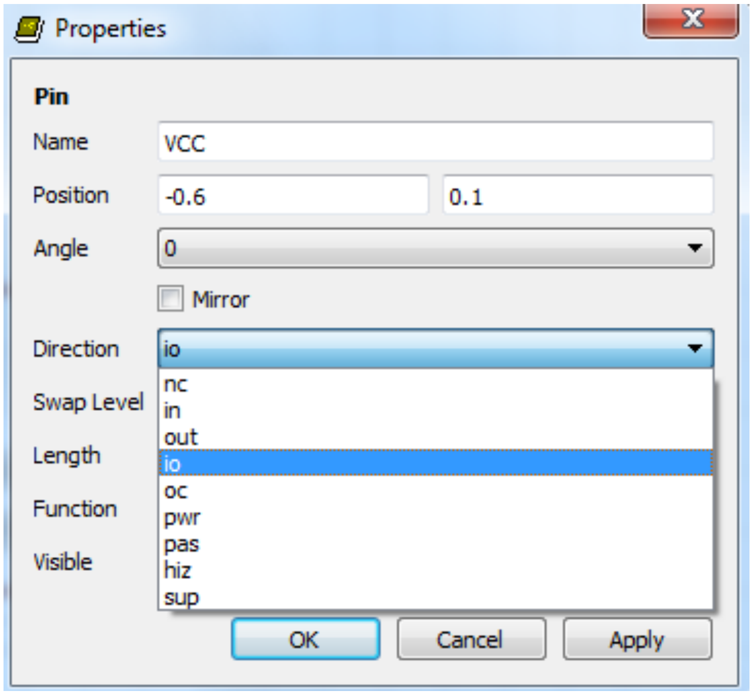
The final step in making a relatively complete symbol is adding a little bit of information about the pins themselves.

> This step isn't strictly necessary, but it can be useful in certain situations, and it's worth spending a minute or two adding this one last touch before moving on to the final step in the device design process.

Setting Pin Directions

It can be useful to say what direction or function a pin serves, particularly if you make use of some of the rules checking functionality in Eagle to validate your schematic and board files.

Eagle defines a number of possible pin directions, which we can see by right-clicking on any pin and selecting the properties window, and then expanding the '**Direction**' drop down list, as shown below:


Properties

Pin

Name: VCC

Position: -0.6 0.1

Angle: 0

□ Mirror

Direction: io

Not all of these direction are commonly used, but the important ones that you should be concerned with are:

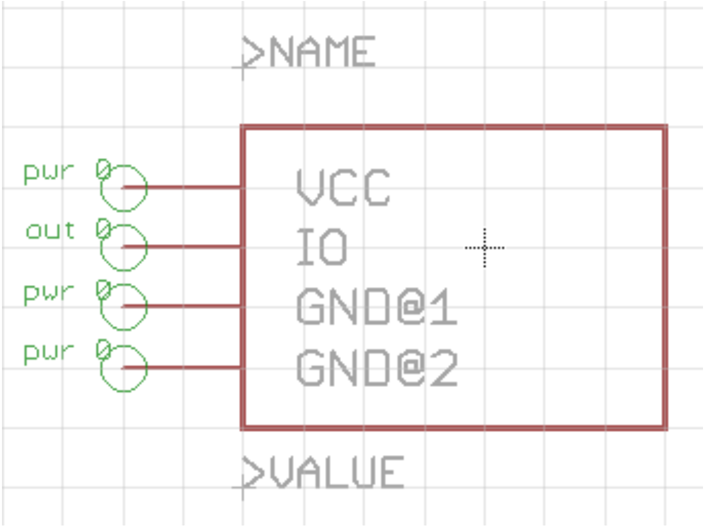
- **nc** - For 'Not Connected', which is used to indicate that this pin should not be connected to anything on your schematic. Assigning a pin as 'NC' will allow Eagle to warn us if you mistakenly connect something this this pin.

- **in** - For input pins
- **out** - For output pins
- **io** - Pin is bi-directional (input and output) * **This is the default pin direction!**
- **pwr** - For VCC, VDD, VSS, GND, VBUS, VIN, VBAT, and similar 'power' pins
- **pas** - For pins on passive parts like resistors and capacitors (no often used creating custom parts)

To set the pin direction, just right-click on every pin and select the 'Properties' option, then set the direction in the properties dialogue box. In the case of the GA1A1S202WP, we would set the direction as follows:

- Set VSS, GND@1 and GND@2 to '**pwr**'
- Set IO to '**out**', since this is actually the analog output on our device

This will give you something resembling the following on our updated symbol:

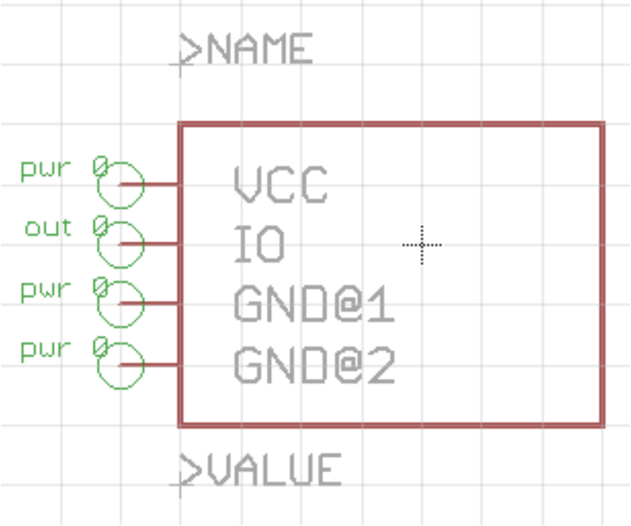


Adjusting Pin Length (Optional)

This is purely a cosmetic step, but if you wish you can also make the pins larger or smaller. My own preference is to set the pins to '**short**' (the default, shown in the image above, is '**middle**').

If you want to, you can change the length in the properties window. Whatever size you decide to work with, though, you should try to be consistent about keeping the same length throughout your library.

You can see the results of setting the pin length to short in the image below:



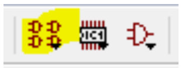
That's it! We've made a complete schematic symbol representing our part, and we're ready to move on to the final step in the part design process, connecting out symbol and our package in the device editor!

The Device Editor

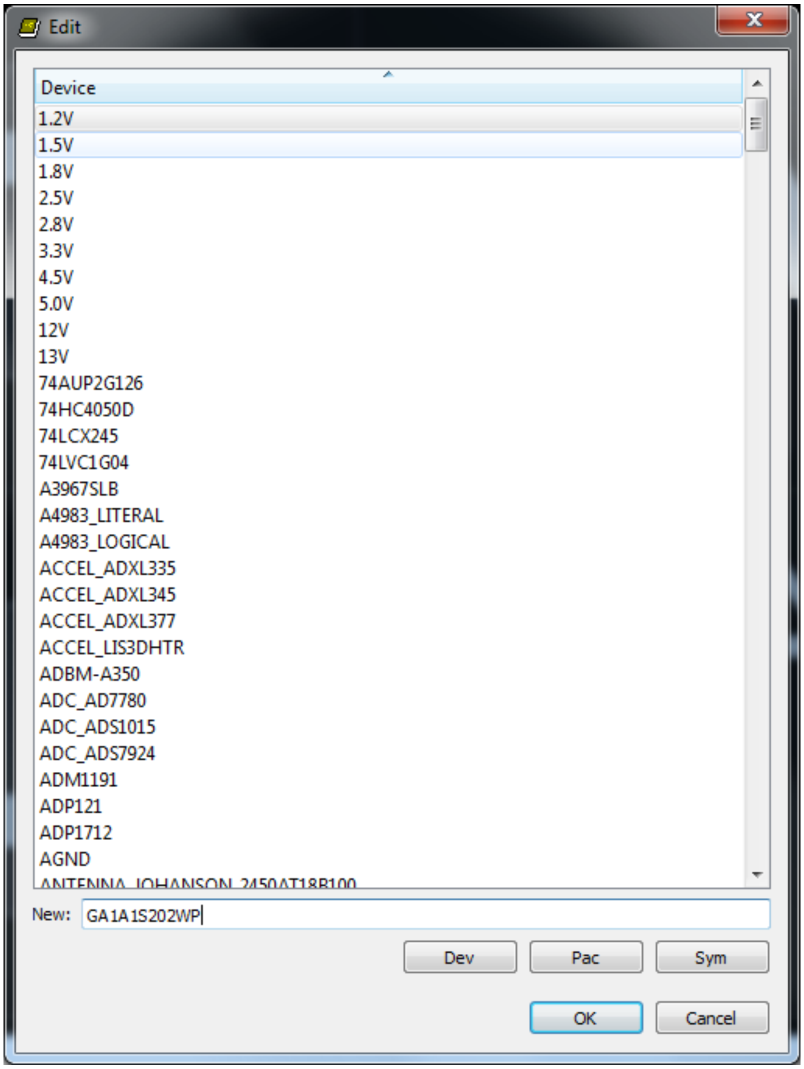
The last step designing a part in Eagle is to combine our package (or packages) with out symbol (or very rarely several symbols), which results in something called a 'device' in Eagle.

Creating a New Device

Creating a device is done exactly the same way we created a new package and symbol: Click on the dedicated 'Device' icon in the library editor toolbar ...



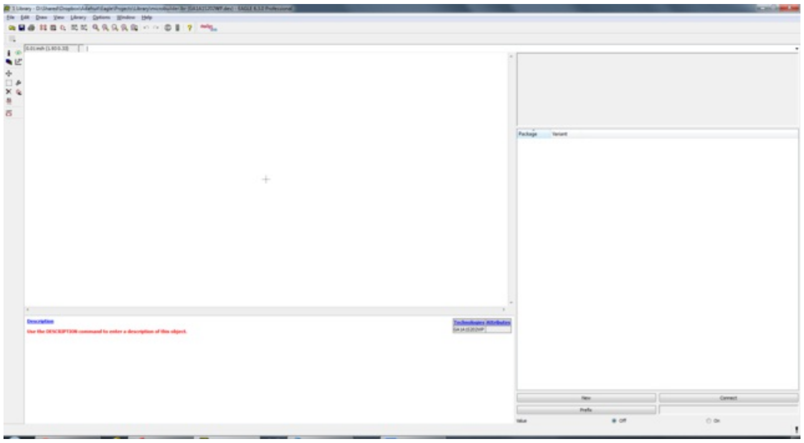
... and enter a unique name for our device ('GA1A1S202WP' in this case):



Device
1.2V
1.5V
1.8V
2.5V
2.8V
3.3V
4.5V
5.0V
12V
13V
74AUP2G126
74HC4050D
74LCX245
74LVC1G04
A3967SLB
A4983_LITERAL
A4983_LOGICAL
ACCEL_ADXL335
ACCEL_ADXL345
ACCEL_ADXL377
ACCEL_LIS3DHTR
ADBM-A350
ADC_AD7780
ADC_ADS1015
ADC_ADS7924
ADM1191
ADP121
ADP1712
AGND
ANTENNA_JOHANSON_2450AT18B100

New: GA1A1S202WP

This will open up the device editor with our new device, and you should see something similar to the following screen:



Description

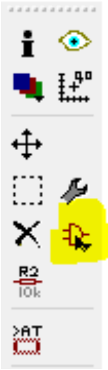
Use the DESCRIPTION command to enter a description of this object.

Technologies Attributes

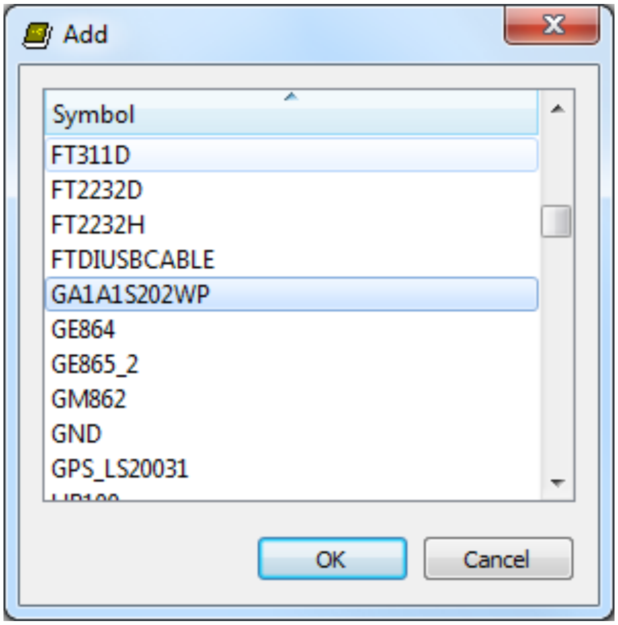
Add a Symbol to our Device

The first step in creating a new device is to add our symbol, which is what will get display on our schematic.

To do this, simple click the '**Add**' icon in the device toolbar on the left-hand side of the editor...



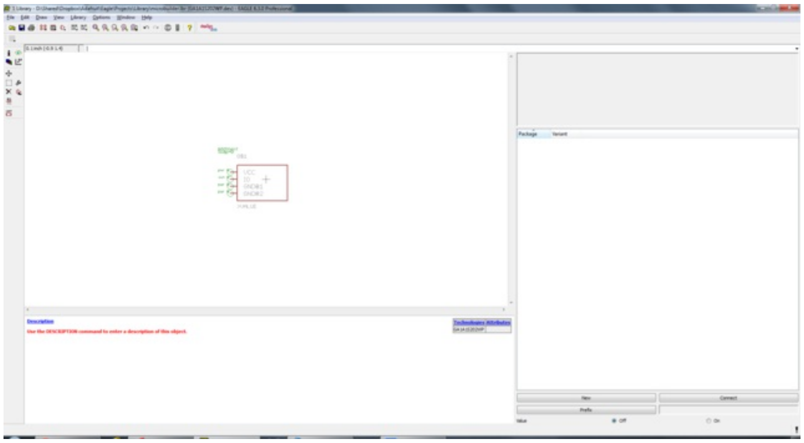
... and then select the right symbol from the dialogue box that pops up (also named 'GA1A1S202WP'):



Symbol

FT311D
FT2232D
FT2232H
FTDIUSBCABLE
GA1A1S202WP
GE864
GE865_2
GM862
GND
GPS_LS20031

After clicking the 'OK' button, add the symbol to the top-left hand box, as seen in the image below:

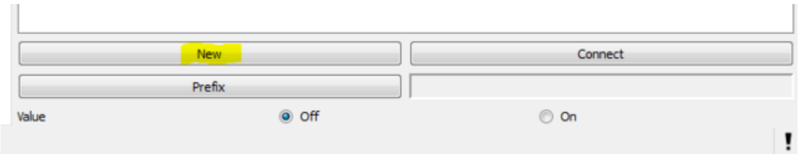


Use the DESCRIPTION command to enter a description of this object.

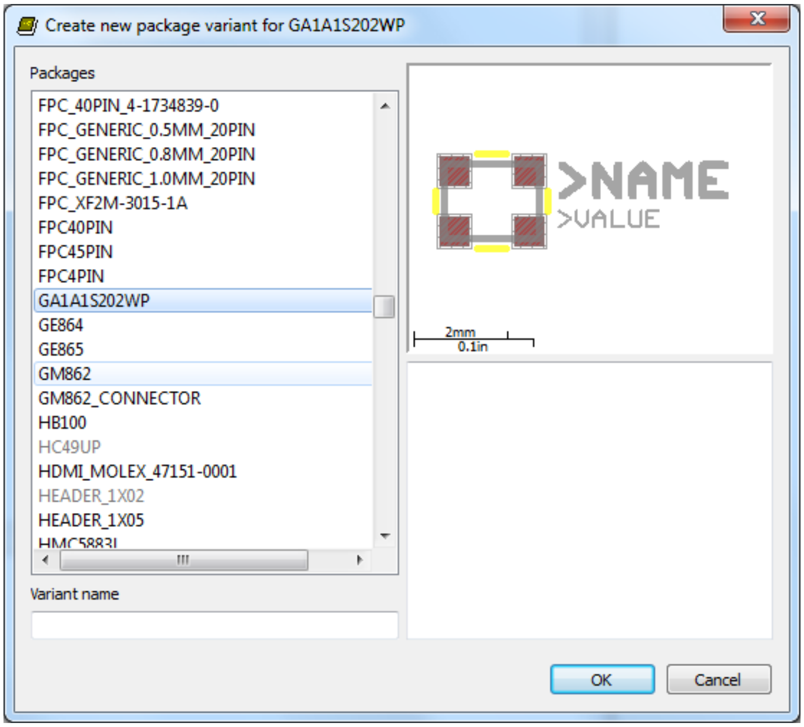
Add a Package to our Device

The next step is adding one or more packages to our new device.

To add a package, click the '**New**' button in the bottom left-hand corner of our device editor ...



... and then select the right package from the popup dialogue box:

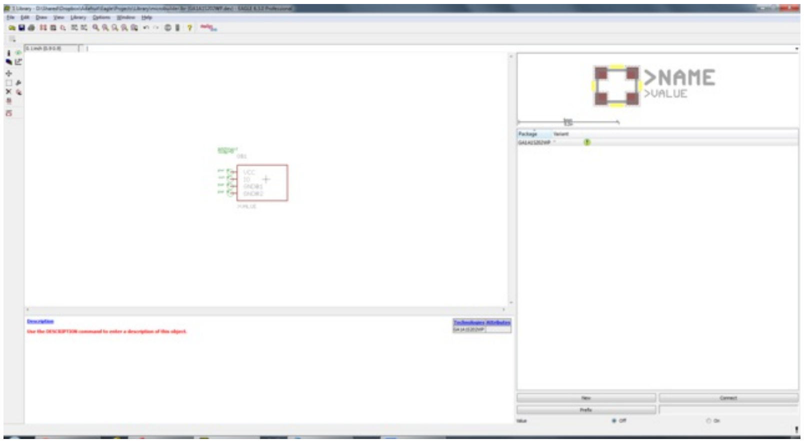

Create new package variant for GA1A1S202WP

Packages

FPC_40PIN_4-1734839-0
FPC_GENERIC_0.5MM_20PIN
FPC_GENERIC_0.8MM_20PIN
FPC_GENERIC_1.0MM_20PIN
FPC_XF2M-3015-1A
FPC40PIN
FPC45PIN
FPC4PIN
GA1A1S202WP
GE864
GE865
GM862
GM862_CONNECTOR
HB100
HC49UP
HDMI_MOLEX_47151-0001
HEADER_1X02
HEADER_1X05
HMC5883L

Variant name

Click the 'OK' button, and the package should get added to the package list on the right-hand side of the device editor:



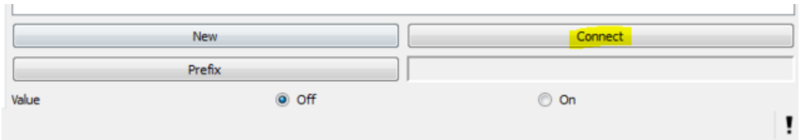
Description

Use the DESCRIPTION command to enter a description of this object.

Associate Symbols and Packages

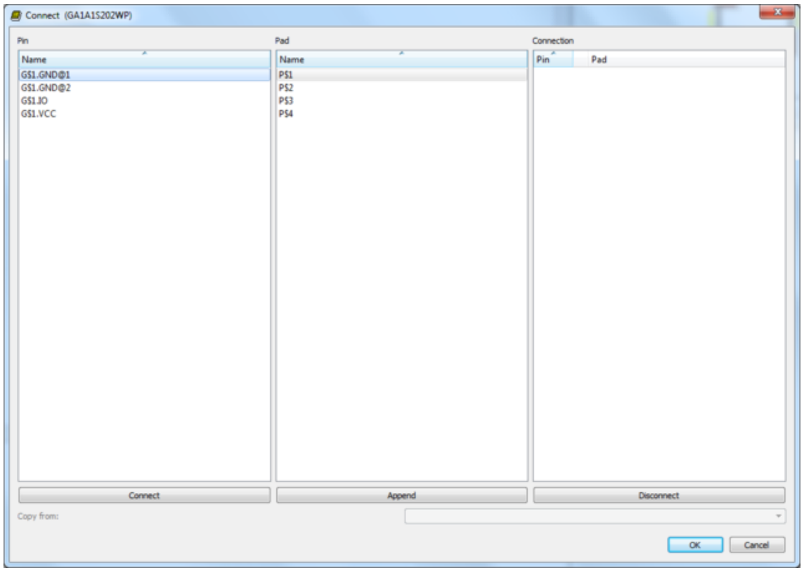
The next step is to create an association between the pins that we created on our symbol, and the pads that we created in our package.

We do this by selecting our package from the package list, and then using the '**Connect**' button just beneath the list:



Connecting Pins and Pads

The connect button should bring up a dialogue box similar to the following:


Connect (GA1A1S202WP)

Pin	Pad	Connection
Name	Name	Pin Pad
G$1.GND@1	P$1	
G$1.GND@2	P$2	
G$1.IO	P$3	
G$1.VCC	P$4	

This dialogue box is used to map pins on our schematic symbol with the physical pads on our package.

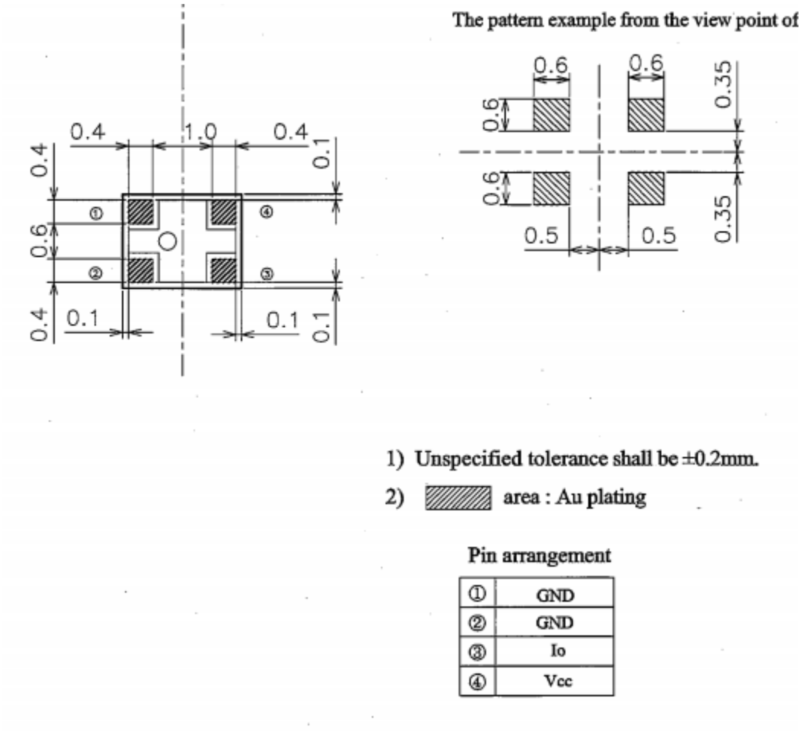
The symbol pins are listed on the left-hand column, the middle column lists the pads in our package, and the final column is used to show 'connected' pins and pads.

Connecting Your First Pin and Pad

To connect a pin and a pad, simply select your pin, then select the appropriate pad, and click the '**Connect**' button.

It's extremely important to pay attention when connecting pins and pads. Always double-check your results against the datasheet since mistakes are easy to make here, and they may have catastrophic consequences on the PCB (unintended shorts, etc.)

Opening up our datasheet, we can see that the GA1A1S202WP has the following pin layout:



1) Unspecified tolerance shall be ±0.2mm.

2) ▨ area : Au plating

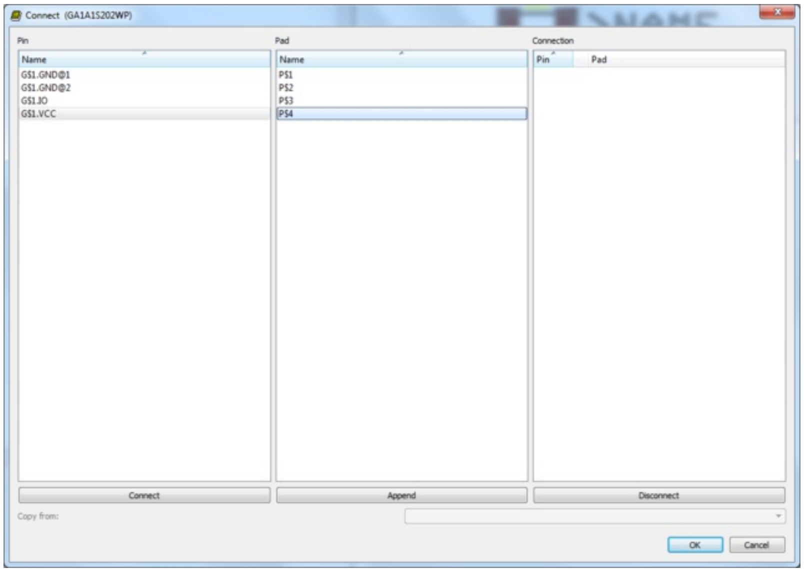
Pin arrangement

①	GND
②	GND
③	Io
④	Vcc

We can see looking at the pin numbers the following associations:

- Pins 1 and 2 (as indicated in the datasheet footprint) are GND
- Pin 3 is IO
- Pin 4 is VCC

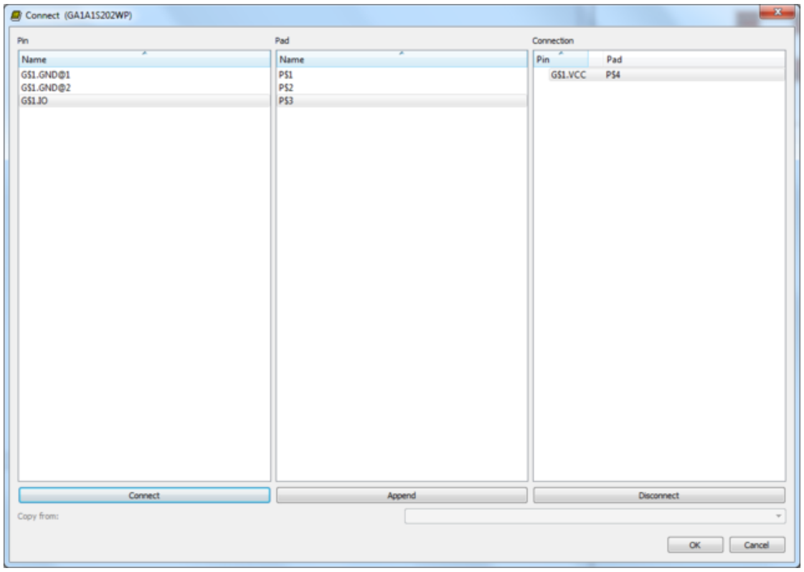
Starting at the bottom and moving up, we would connect Pin 4 (VCC) by selecting VCC in the pin list, and P$4 in our pad list ...



Pin: Name	Pad: Name
G$1.GND@1	P$1
G$1.GND@2	P$2
G$1.IO	P$3
G$1.VCC	P$4

Connection: Pin	Pad

... and clicking the '**Connect**' button, which will remove the pin and pad from the first two columns, and add it to the final column:



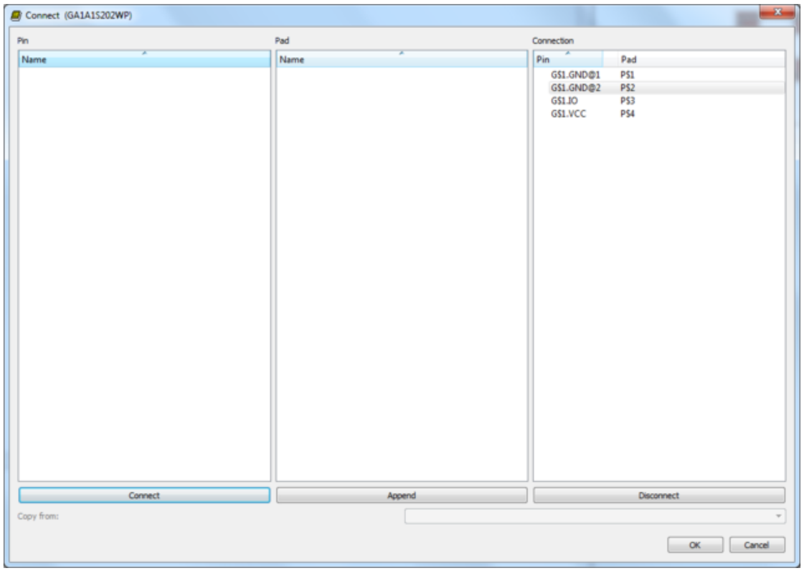
Pin: Name	Pad: Name
G$1.GND@1	P$1
G$1.GND@2	P$2
G$1.IO	P$3

Connection: Pin	Pad
G$1.VCC	P$4

If you made a mistake connecting a pin and a pad, just select the faulty connecting in the third column and click the 'Disconnect' button!

Final Results

Connecting the result of our pins should give you something similar to the following:



Pin	Pad
G$1.GND@1	P$1
G$1.GND@2	P$2
G$1.IO	P$3
G$1.VCC	P$4

Now just click the 'OK' button and we can add some finishing touches to our new device!

Adding a Name Prefix

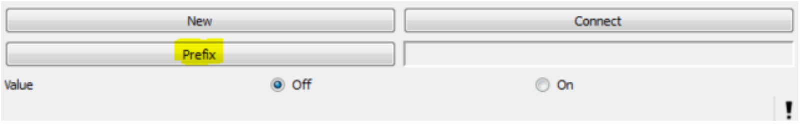
If you remember the '>NAME' text that we added to our packages (http://adafru.it/c9s) as well as our symbols (http://adafru.it/c9u), you'll see here why it's useful to make our schematics easier for other people to understand.

Adding a Prefix to the Device Name

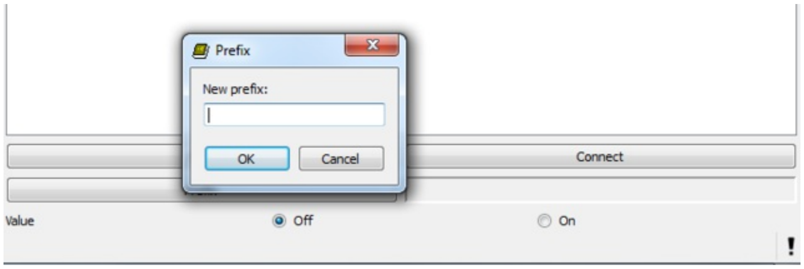
By default, Eagle will automatically assign a name to our device when we place it on our schematic, making sure that each part is numerically unique. It does this by looking for the '>NAME' text that we inserted in our part, updating it with a unique name.

This allows us to distinguish parts during the manufacturing stage, but we can make things a lot easier for other engineers, customers or manufacturing partners by adding an optional one or two letter prefix to these names.

To add this prefix, we use the '**Prefix**' button below the package list:



After clicking the 'Prefix' button, we're presented with a dialogue box where we can enter a short bit of text that will be inserted before any numeric values Eagle assigns to our part:



If we enter 'U' here, which is usually used to identify integrated circuits, our part would have a name like U1, U6, U14, with the numeric part generated by Eagle.

Adding a prefix is optional, but it's very highly recommended, since standard naming will make the assembly process much easier, and it will also make debugging HW easier if you need to communicate with other engineers or customers!

Standard Device Prefixes

As with schematic symbols, there is a certain tradition and expection built up around part names, and when possible you should adhere to these standard naming conventions for your parts. Its expected that resistors will be named Rx, capacitors will be named Cx, integrated circuits (such as our sensor here) will be names Ux, etc.

The Wikipedia entry on Electronic Symbol (http://adafru.it/c9w) contains a list of some of the most common prefixes, but when in doubt you can also have a look at some other schematics from trusted sources to see what prefix should be used with what device types.

Some of the most commonly used prefixes are shown below for convenience sake:

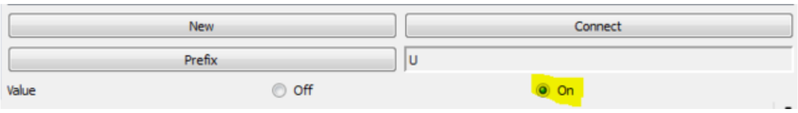
- C - capacitor
- D - diode (including LEDs!)
- J - header pins or 'jack' connector
- JP - jumper
- L - inductor
- M - motor
- Q - transistor/FETs
- R - resistor
- S - switch (TACT switch, etc.)
- TP - test point
- U - integrated circuit (sensors, MCUs, etc.)
- X - Wikipedia lists this last transducer, but it is often used to indicate connectors (USB, etc.)
- Y - Crystals or oscillators

Enabling the Value Field

The >VALUE field that we added to our package and symbol is also very useful, but it is disabled by default in the device editor (meaning that it can't be edited in the schematic via), and should generally be enabled so that the end user can update it if necessary.

Enabling Updates to the Value Field

To allow users to change the default 'value' for your new device, you need to set the 'Value' radio button to 'On' instead of the default 'Off':



U

Value ○ Off ● On

If you don't enable this option, all parts will use the device name as the default text for &gt;VALUE, so it's generally a good idea to enable it. There are often variations of packages, and enabling this option allows end users to specify the exact part they intended to use!

Adding a Description

The final (!!!) step is optional, but it's one last, nice little detail that a future version of you will appreciate as much as anyone else who uses your library: add some basic details about the part to your device's '**Description**' field.

What To Include in the Description

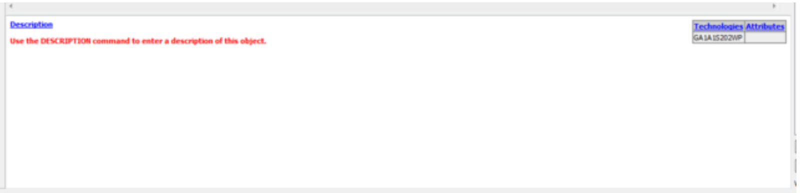
While it's entirely up to you, some information you might consider including, spending a few minutes adding key information from the datasheet now can save you a lot of time later when you need to make sure every part on your board meets certain tolerances like temperature range or minimum supply voltage.

As a rule of thumb, you should consider adding the following information in the description field:

- The **recommended supply voltage range** for the device (2.3-3.2V in the case of our sensor)
- The **operating temperature** for your device (-40°C to 85°C in this case)
- Any **key information about the sensor**, such as the output range for the GA1A1S202WP which is 3-55000 lux
- If you use the same one or two suppliers, you may also find it useful to include the **vendor-specific part numbers** as well, such as the part code for Digikey, Mouser, Farnell, etc.

Adding Description Data

To add some text to the description field, simply click the 'description' label that is in the bottom-most box by default ...

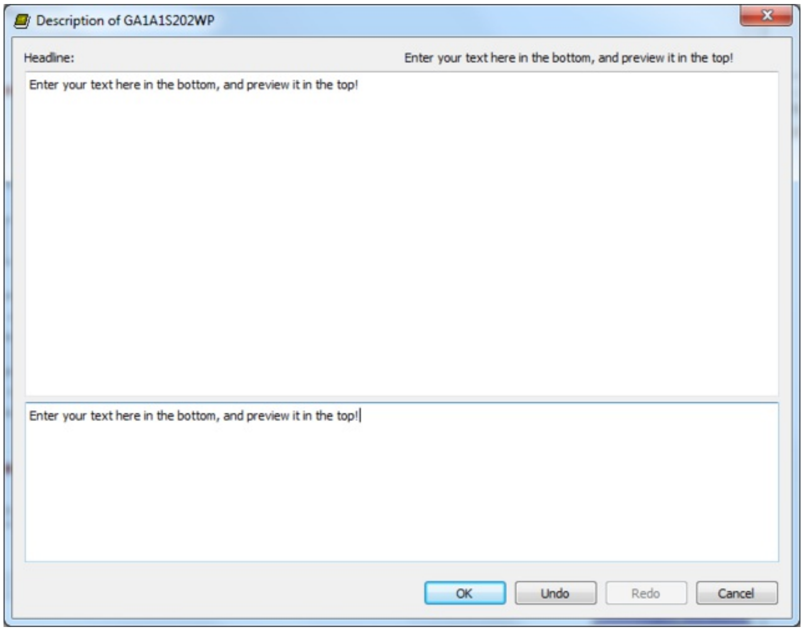


Description

Use the DESCRIPTION command to enter a description of this object.

Technologies	Attributes
GA1A1S202WP	

This will bring up a dialogue box where we can enter our description:



Description of GA1A1S202WP

Headline:

Enter your text here in the bottom, and preview it in the top!

Enter your text here in the bottom, and preview it in the top!

Enter your text here in the bottom, and preview it in the top!

You can use basic HTML formatting for your description, including common HTML tags like 'p' for paragraphs, 'b' for bold, 'ul' and 'li' for lists, etc.

Example

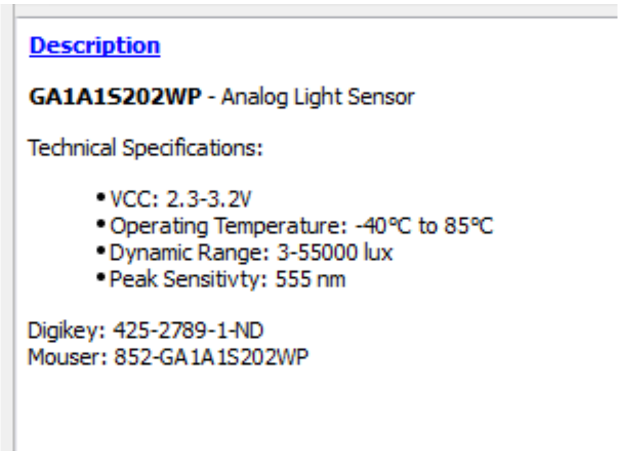
Here is the information included in my part description, taking advantage of some basic HTML tags:

```
<b>GA1A1S202WP</b> - Analog Light Sensor

<p>Technical Specifications:
<ul>
<li>VCC: 2.3-3.2V</li>
<li>Operating Temperature: -40°C to 85°C</li>
<li>Dynamic Range: 3-55000 lux</li>
<li>Peak Sensitivty: 555 nm</li>
</ul>
</p>

<p>Digikey: 425-2789-1-ND<br/>
Mouser: 852-GA1A1S202WP</p>
```

This results in the following description in the preview window and inside Eagle when we look at this device in detail:



Description

GA1A1S202WP - Analog Light Sensor

Technical Specifications:

- VCC: 2.3-3.2V
- Operating Temperature: -40°C to 85°C
- Dynamic Range: 3-55000 lux
- Peak Sensitivty: 555 nm

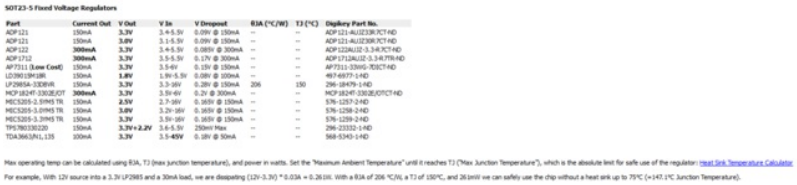
Digikey: 425-2789-1-ND
Mouser: 852-GA1A1S202WP

Essential Eagle Skills: Advanced Descriptions

The fact that we can use basic HTML tags, including tables, can be used to our advantage in Eagle for parts with many packages or many variants.

For example, I have literally dozens of SOT23-5 voltage regulators that I use on different boards, depending on the amount of current I need, the output voltage (or voltages!), etc.

To organize all of this information correctly, and allow me to quickly select the right part out of the dozens I have on hand, I created a table for the SOT23-5 VREG device with the key information below:



SOT23-5 Fixed Voltage Regulators

Part	Current Out	V Out	V In	V Dropout	θJA (°C/W)	TJ (°C)	Digikey Part No.
ADP121	150mA	**3.3V**	3.4-5.5V	0.09V @ 150mA	--	--	ADP121-AUJZ33R7CT-ND
ADP121	150mA	**3.0V**	3.1-5.5V	0.09V @ 150mA	--	--	ADP121-AUJZ30R7CT-ND
ADP122	**300mA**	**3.3V**	3.4-5.5V	0.085V @ 300mA	--	--	ADP122AUJZ-3.3-R7CT-ND
ADP1712	**300mA**	**3.3V**	3.5-5.5V	0.17V @ 300mA	--	--	ADP1712AUJZ-3.3-R7TR-ND
AP7311 **(Low Cost)**	150mA	**3.3V**	3.5-6V	0.15V @ 150mA	--	--	AP7311-33WG-7DICT-ND
LD39015M18R	150mA	**1.8V**	1.9V-5.5V	0.08V @ 100mA	--	--	497-6977-1-ND
LP2985A-33DBVR	150mA	**3.3V**	3.3-16V	0.28V @ 150mA	206	150	296-18479-1-ND
MCP1824T-3302E/OT	**300mA**	**3.3V**	3.5V-6V	0.2V @ 300mA	--	--	MCP1824T-3302E/OTCT-ND
MIC5205-2.5YM5 TR	150mA	**2.5V**	2.7-16V	0.165V @ 150mA	--	--	576-1257-2-ND
MIC5205-3.0YM5 TR	150mA	**3.0V**	3.2V-16V	0.165V @ 150mA	--	--	576-1258-2-ND
MIC5205-3.3YM5 TR	150mA	**3.3V**	3.5V-16V	0.165V @ 150mA	--	--	576-1259-2-ND
TPS780330220	150mA	**3.3V+2.2V**	3.6-5.5V	250mV Max	--	--	296-23332-1-ND
TDA3663/N1,135	100mA	**3.3V**	3.5-**45V**	0.18V @ 50mA	--	--	568-5343-1-ND

Max operating temp can be calculated using θJA, TJ (max junction temperature), and power in watts. Set the "Maximum Ambient Temperature" until it reaches TJ ("Max Junction Temperature"), which is the absolute limit for safe use of the regulator: Heat Sink Temperature Calculator

For example, With 12V source into a 3.3V LP2985 and a 30mA load, we are dissipating (12V-3.3V) * 0.03A = 0.261W. With a θJA of 206 °C/W, a TJ of 150°C, and 261mW we can safely use the chip without a heat sink up to 75°C (=147.1°C Junction Temperature).

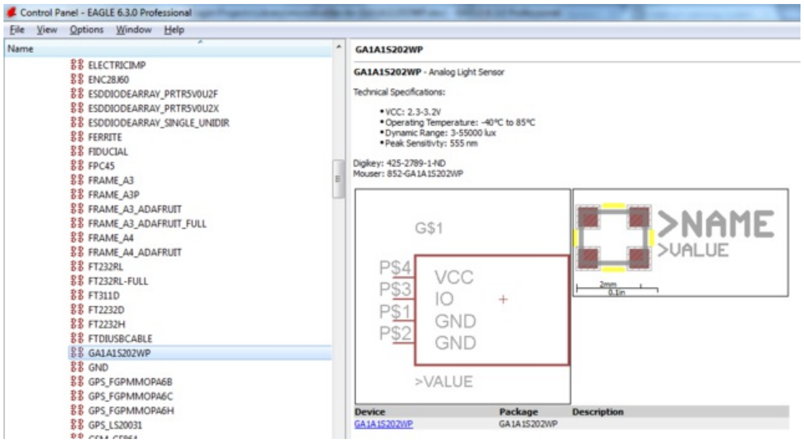
If you look closely, you'll see that I also added some additional information that may be useful to other people using these parts to calculate the max operating temperature for the regulator.

If this part is likely to be reused a lot, or if you had to look up some specialized information when selecting the right part, including a brief summary of that information in the Eagle device will often come in handy, and a future version of yourself will be very glad for the extra bit of effort you put into things while this information was still fresh in your head!

The End!

That's it! You've successfully designed a robust, reliable, ready to use device, and you can start using this part with confidence in your schematics and board layouts!

After saving your library, you should be able to see your new part (along with the descriptive text we added in the last step) by expanding the library in the main Eagle window, and selecting the appropriate part name from the treeview:



GA1A1S202WP

GA1A1S202WP - Analog Light Sensor

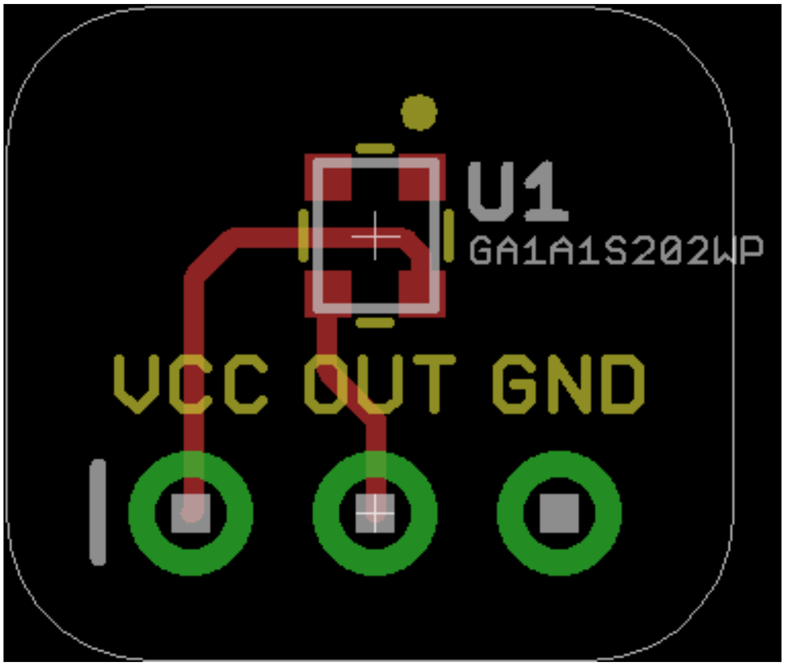
Technical Specifications:

- VCC: 2.3-3.2V
- Operating Temperature: -40°C to 85°C
- Dynamic Range: 3-55000 lux
- Peak Sensitivty: 555 nm

Digikey: 425-2789-1-ND
Mouser: 852-GA1A1S202WP

Device	Package	Description
GA1A1S202WP	GA1A1S202WP	

And here's out part in a PCB, along with the update labels, the physical part outline, the pin 1 indicator, etc.:



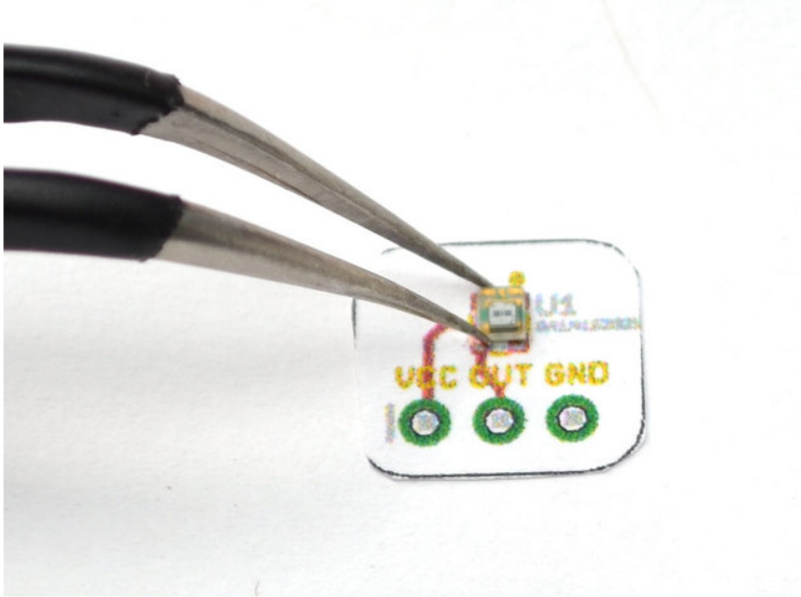
Note: The silkscreen on this part is slightly different than in the tutorial, since I needed to use this part in a very tight layout, and had to reduce the area taken up by the silkscreen.

It might have seemed like a Herculean effort doing this all the first time around, but I can promise this takes 10-15 minutes max once you get the hang of it.

But it's important to build up the right habits at the beginning, which is why we went into such detail in this tutorial. Learn to do things the right way now, and a future version of yourself will be very grateful, and you might even have a bit more money in your pocket avoiding a few board revisions due to poorly designed footprints!

1:1 Print Test

Before sending out to fabrication, be sure to do a 1:1 print test! Print out a paper copy of the PCB design and place the part over the pads, checking that the outline, pads and shape line up. A magnifying glass is handy here



Where to Now?

Like any complex software package, Eagle takes time, effort, and a certain amount of trial and error to master. We've covered the basics here in professional footprint design, but there's a lot more to be said about the subject.

But have a go at making more and more complex parts, be brave about putting your money where your mouth is having boards made, and most important ... learn from your mistakes! We've definitely made more than a few here, but that's the way you learn!

If you want to get all crazy on your footprints, you can also have a look at our other Eagle footprint learning guide: Creating Accurate Footprints in Eagle (http://adafru.it/c9y)!