8 Introduction to Delayering and Reversing PCBs

by Joe Grand

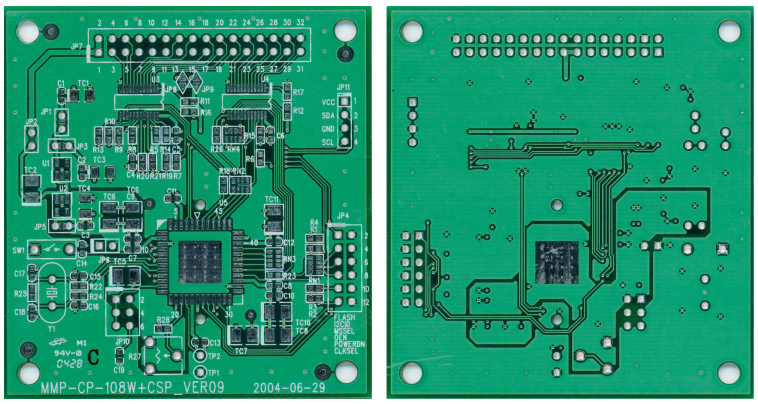

Figure 19: Our example PCB in its unmodified state. If only it knew the suffering that it was about to endure.

Figure 20: Sandpaper at work. You can see the copper of inner layer 2 starting to peek out from underneath the top substrate.

Printed Circuit Boards (PCBs) form the physical carrier for and provide electrical pathways between electronic components. They are created with layers of thin copper (conductive) foil laminated to insulating (non-conductive) layers. By accessing and imaging each individual copper layer of a PCB, it is possible to recreate the PCB layout. If the component types (and values, ideally) are known, you'll also be able to derive the schematic (a simplified, visual representation of the device's electronic design) or a desired portion thereof.

"Why bother?" you might ask. Maybe you want to understand how a particular product works, locate specific connections on the board (like JTAG or UART), clone the design, or figure out where you can modify

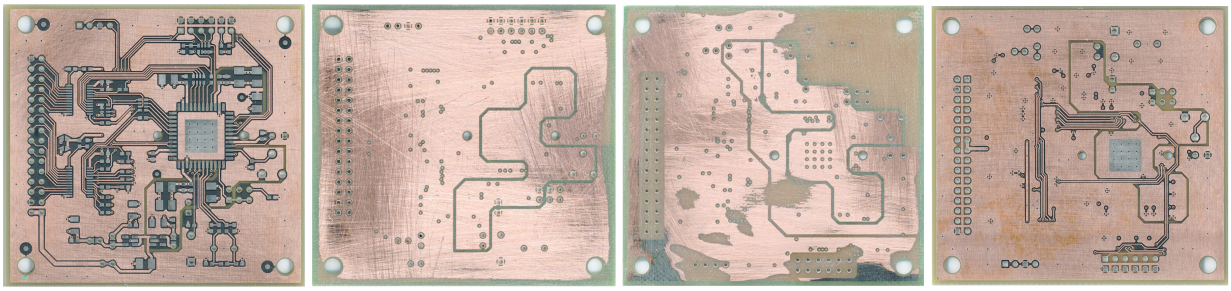

Figure 21: The four exposed layers of our example PCB.

it to inject malicious functionality. The techniques provided in this article might not be groundbreaking to those skilled in the hardware arts, but will serve as a resource for folks interested in meandering down the path of PCB reverse engineering.

8.1 Delayering

The first phase of the process is to obtain an image of each layer of the target circuit board. There are a variety of possible techniques, including low-tech, off-the-shelf solutions and those requiring expensive equipment and skilled operators. Some methods are destructive, meaning you'll never see your PCB again when you're done, and some are non-destructive, meaning the PCB will remain intact and unharmed. For now, we're going to focus on manual abrasion using sandpaper, which will destroy your board layer-by-layer, but is also the simplest and most accessible.

The top and bottom of a PCB are usually coated in solder mask, a non-conductive layer that protects the PCB from dust and oxidation and provides access to copper areas on the board that are intended to be exposed. You'll want to remove the solder mask so you have unobstructed access to the underlying copper. To do so, attach the PCB to your work surface with a clamp or double-sided tape. Then, use 60 to 220 grit sandpaper in even strokes at light pressure across the entire board. Optionally, you can put spare PCBs of the same height as the target on either side to help maintain planar motion and even sanding pressure. Holding the sandpaper by hand will give you the best control. If you're prone to repetitive stress injuries, a tool such as a Norton Sheet Sander may serve you well.

Once you've exposed the copper, it's time to capture an image of the layer. If you have access to a flatbed scanner, use that. Otherwise, a point-and-shoot camera will work. (When using a camera instead of a scanner, be aware that you may need to rotate and lens-correct the resulting image to make it appear as planar and true-to-form as possible.)

To access the inner layers, the process is similar to removing the solder mask. For this step, you'll need harder pressure and more elbow grease to deal with removing the layer of insulating substrate, a fiberglass/epoxy weave.

Figure 19 shows the top and bottom of our example PCB in its unmodified state. This board is 4-layer, 62 mil thick, with trace widths ranging from 12 to 48 mil. Figure 20 shows PCB delayering in action. After you've successfully accessed and imaged each layer of the PCB, you should end up with a sequence similar to Figure 21.

8.2 Image processing

With your PCB layer images in hand, the next phase is to use an image processing/manipulation tool of your choice to adjust the images, create a stack-up of the layers, and configure the opacity of each so that you can see all copper features at once: footprints, traces, vias, and fills. Suitable programs include Adobe Photoshop, GIMP, and Paint.NET.

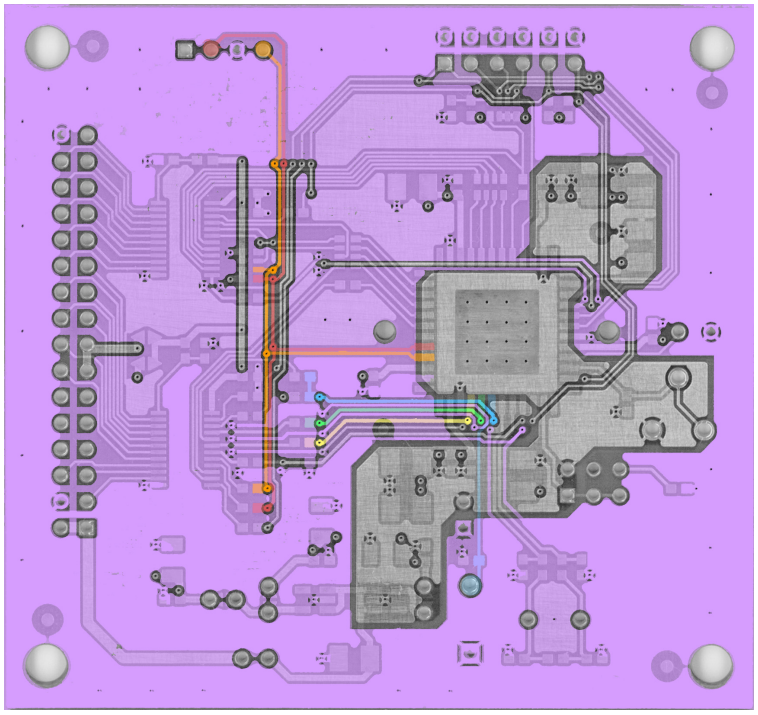

Figure 22: Layer stack-up of our example PCB. Layer opacity was adjusted to see through the board and arbitrary traces were colored using a flood fill.

The image processing tasks are as follows:

1. Rotate and mirror the images so they all have the same orientation. For reverse engineering purposes, you'll want a view of each layer as if you're looking down at it through the top of the board. This means that the bottom half of your image set will need to be flipped/mirrored vertically. Choose a feature of the PCB that exists on all layers, such as a mounting hole, test point, via, or through-hole footprint, and make sure that it's in the same position on the board in each of the images.

2. Adjust the images so the copper features on each layer are easily distinguishable from the underlying substrate. The exact adjustments you need to perform will vary depending on the quality of your deconstruction process and resulting images. At a minimum, you'll want to remove unnecessary features, adjust brightness/contrast, and desaturate to shades of grey or convert to black and white.

3. Merge the images into a single file, to create a stack-up of the layers, by placing each one on its own layer within your image processing tool. Set the opacity of each layer to 50% as a starting point, while leaving the bottom layer at 100%. This will let you see through the layers enough to identify the PCB features on each. Make sure that drill holes and other through-hole features match across the entire board surface. You may need to make small rotational or minor scaling adjustments to exactly align the layers.

8.3 Reverse engineering

The goal of this phase is to determine how components are physically interconnected on the board by visually following the copper, assisted by your image processing tool. If you want to make use of the information you glean from these efforts, you may want to have a modicum of electronics knowledge.

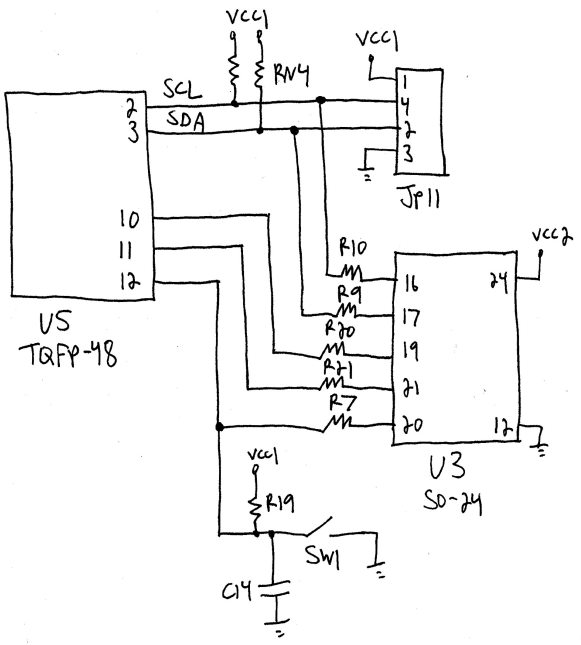

Figure 23: Schematic based on the colored signals of Figure 22. This kind of visual representation is much easier than a collection of PCB layer images.

To begin, identify the major component footprints on the board and pick a starting location on one of them. If component part numbers are known, obtain their associated data sheets for details about the component, its pinout, and pin functionality. Then, prepare yourself for a lot of repetition.

With your image processing tool, enable and disable the layers as needed while using a flood fill to set the color of the desired trace and anything it's in contact with. You'll find yourself hopping between the various layers and zooming in and out as you follow the trace around and through the board. Draw a schematic as you go, adding to it each time you finish coloring a route. Keep in mind that the PCB silkscreen often contains reference designators, part numbers, component values, and other useful information that you can incorporate into your schematic. A board's physical characteristics and actual layout can also be very important aspects of the design, but we'll ignore them for now. Repeat these steps until every trace is accounted for.

Figure 22 shows a working view of my PCB layer stack-up with a few arbitrarily selected connections traced and colored. Figure 23 shows the resulting schematic.

If you want to see a true master of signal tracing, watch any of Chris Tarnovsky's chip hacking presentations from Black Hat or DEFCON. For a different approach to PCB reverse engineering, take a look at Throbscottle's Instructable.

8.4 Next steps

As you might now be aware, the current state of PCB reverse engineering is a manual, time consuming, and often difficult task. The obvious progression of this work is to automate as much of the process as possible. I've started developing a toolkit to assist in recreating a complete schematic based on a collection of PCB layer images. Imagine Karsten Nohl, Starbug, and Martin Schobert's degate or Adam Laurie's rompar, but for circuit boards. I, for one, am excited about the possibilities.