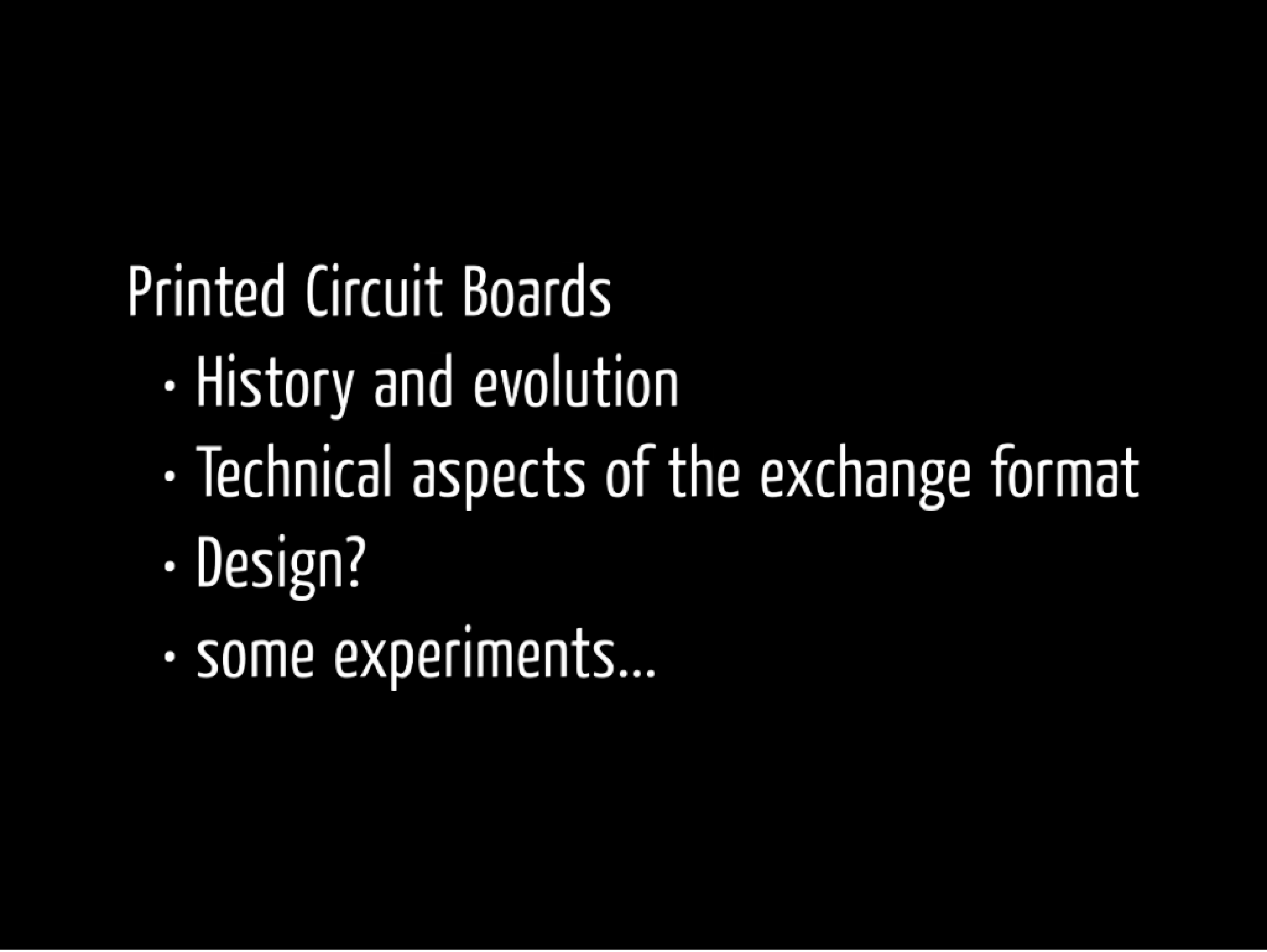
Fonts in PCB's

Design within technical limits

Simon Budig, Libre Graphics Meeting 2014

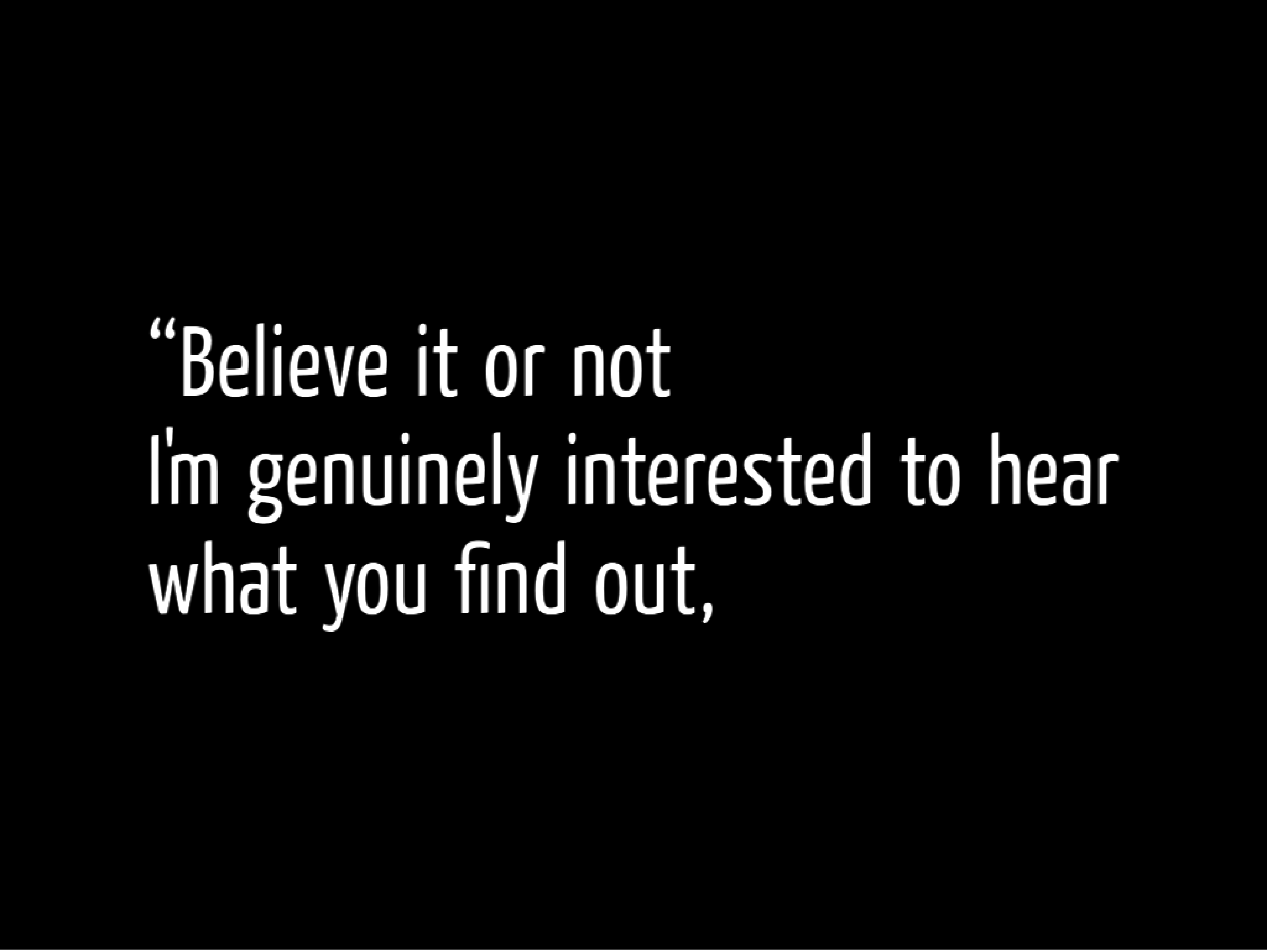
We're going to have a look at graphical and typographical elements on Printed Circuit Boards.



Printed Circuit Boards

- History and evolution
- Technical aspects of the exchange format
- Design?
- some experiments...

I want to present an overview on the history of PCBs, with attention to their emergent beauty. Drawing attention to shortcomings in the handling of artwork for PCBs I want to propose some alternative approaches and demonstrate some proof-of-concept experiments.



“Believe it or not
I'm genuinely interested to hear
what you find out,

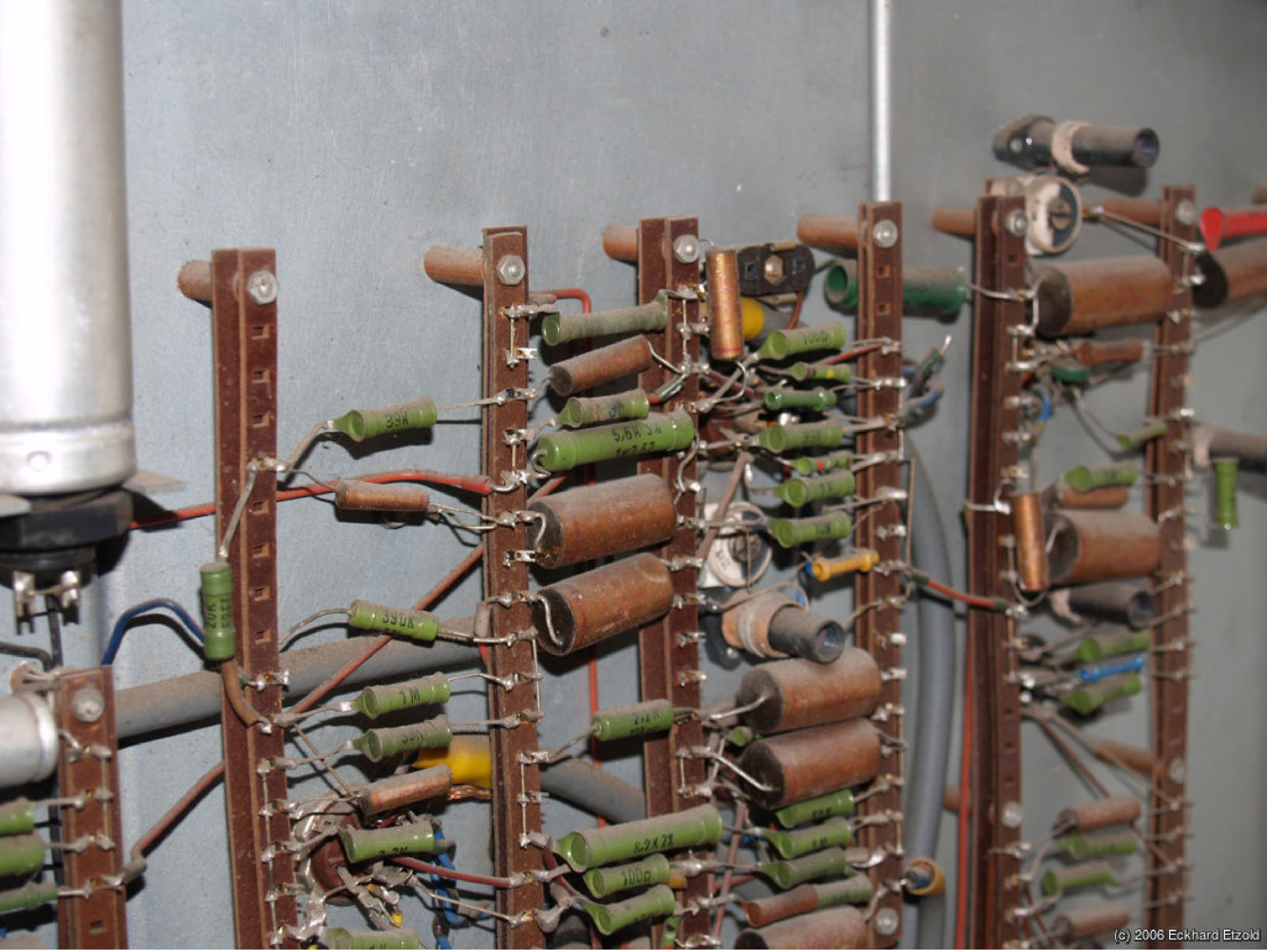
This is a quote by AndyC_772 which I encountered in a discussion on this topic in the EEVBlog.

but I'm afraid I think you may be looking for ideas and inspiration in a pretty dry, empty place.

Sorry.”

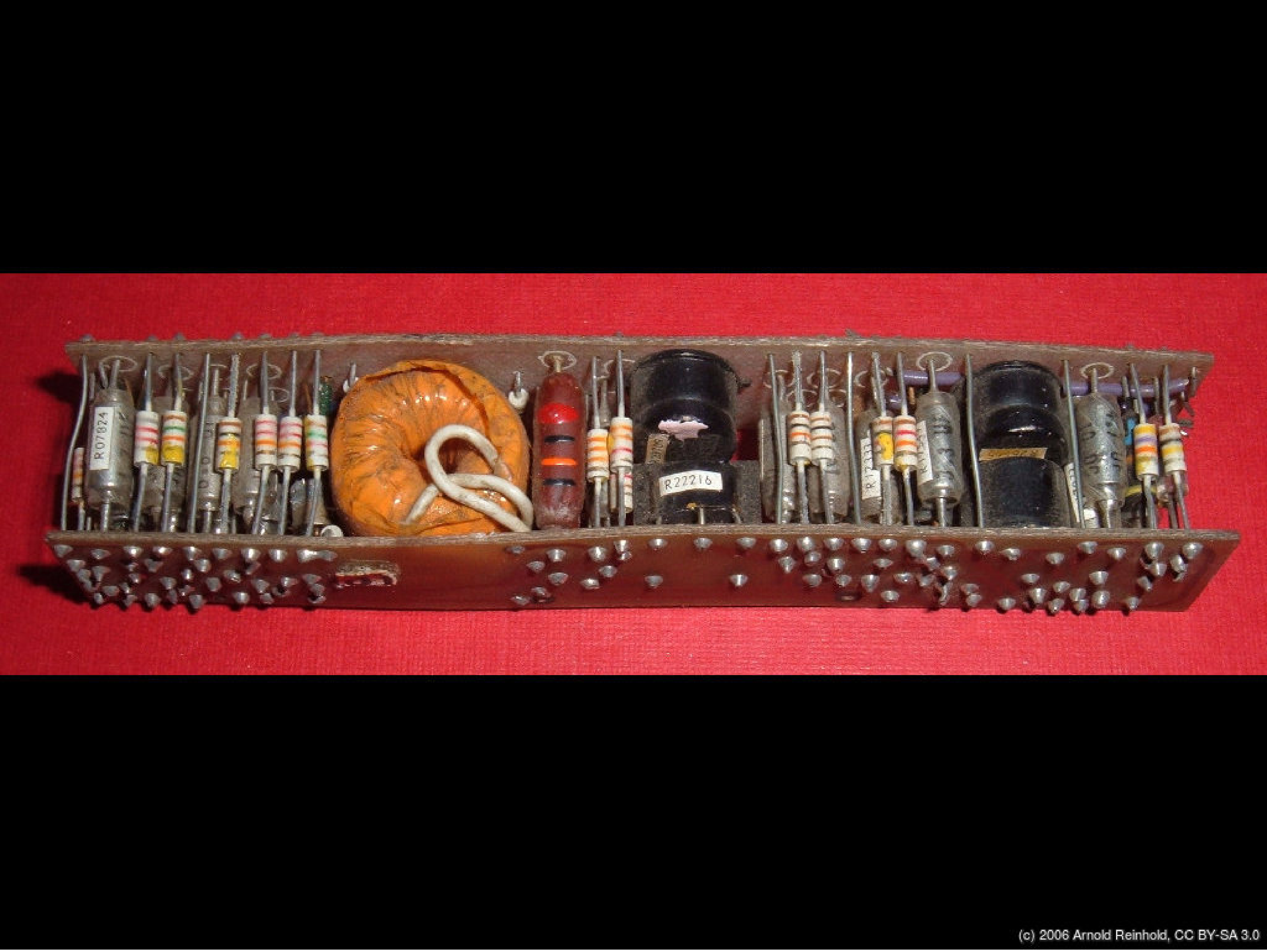
He is focussing on functionality, which doesn't necessarily lend itself to beauty. However, I believe he is blinded by routine: even when not done explicitly, PCBs are full of implicit patterns and have a specific charm.

Some enthusiasts and hobbyists outdo themselves, investing vast amounts of energy into the design of a PCB.



This is a circuit in an experimental colour monitor from 1963. It is obvious, how this kind of construction demands a lot of dexterity from the manufacturer. PCBs are intended to solve this problem.

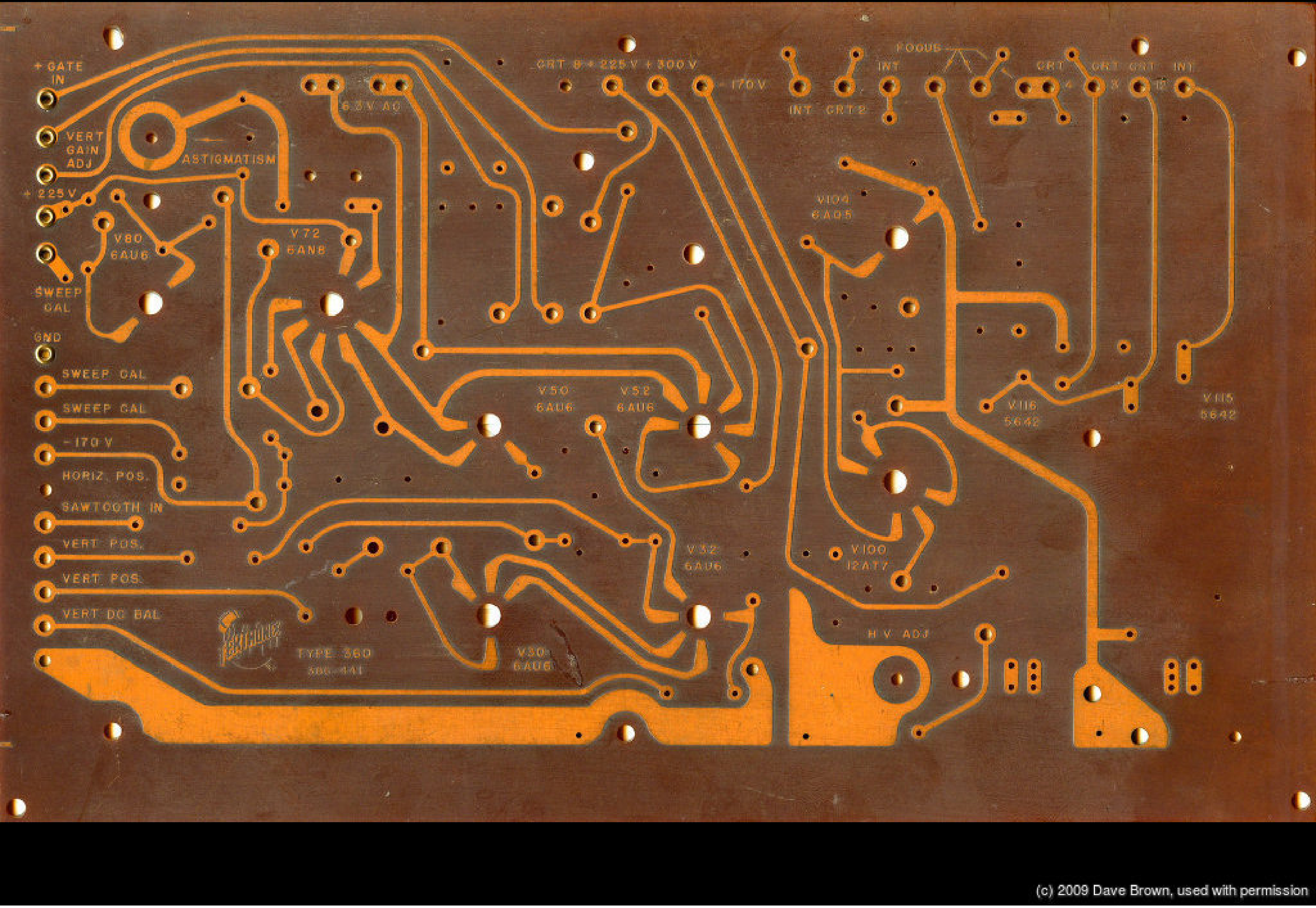
Photo (c) 2006 Eckhard Etzold



(c) 2006 Arnold Reinhold, CC BY-SA 3.0

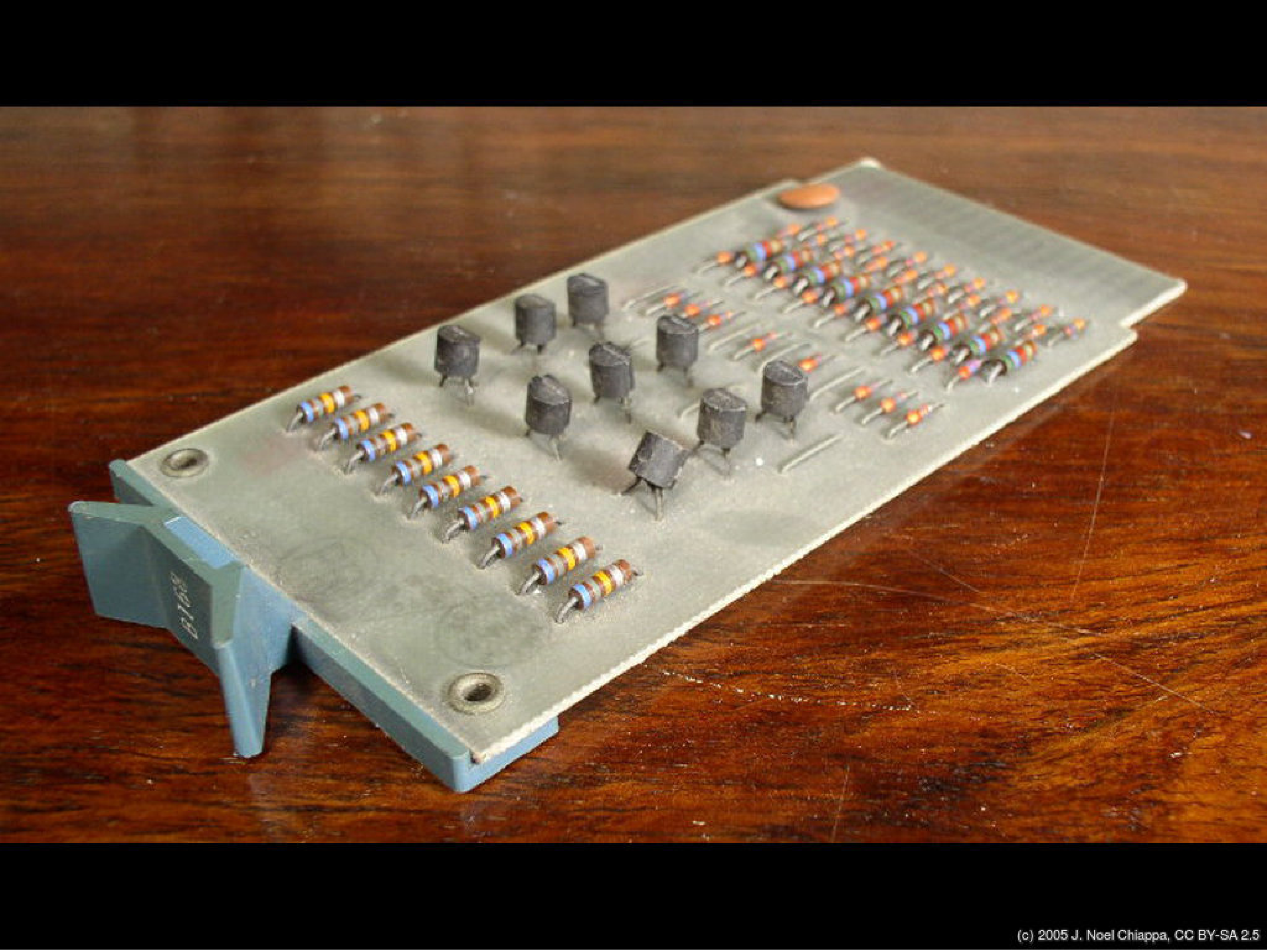
Cordwoodcircuits are another construction method. It is unclear to me how the connections between the components were done, there might be copper traces on the boards.

Photo (c) 2006 Arnold Reinhold, CC BY-SA 3.0



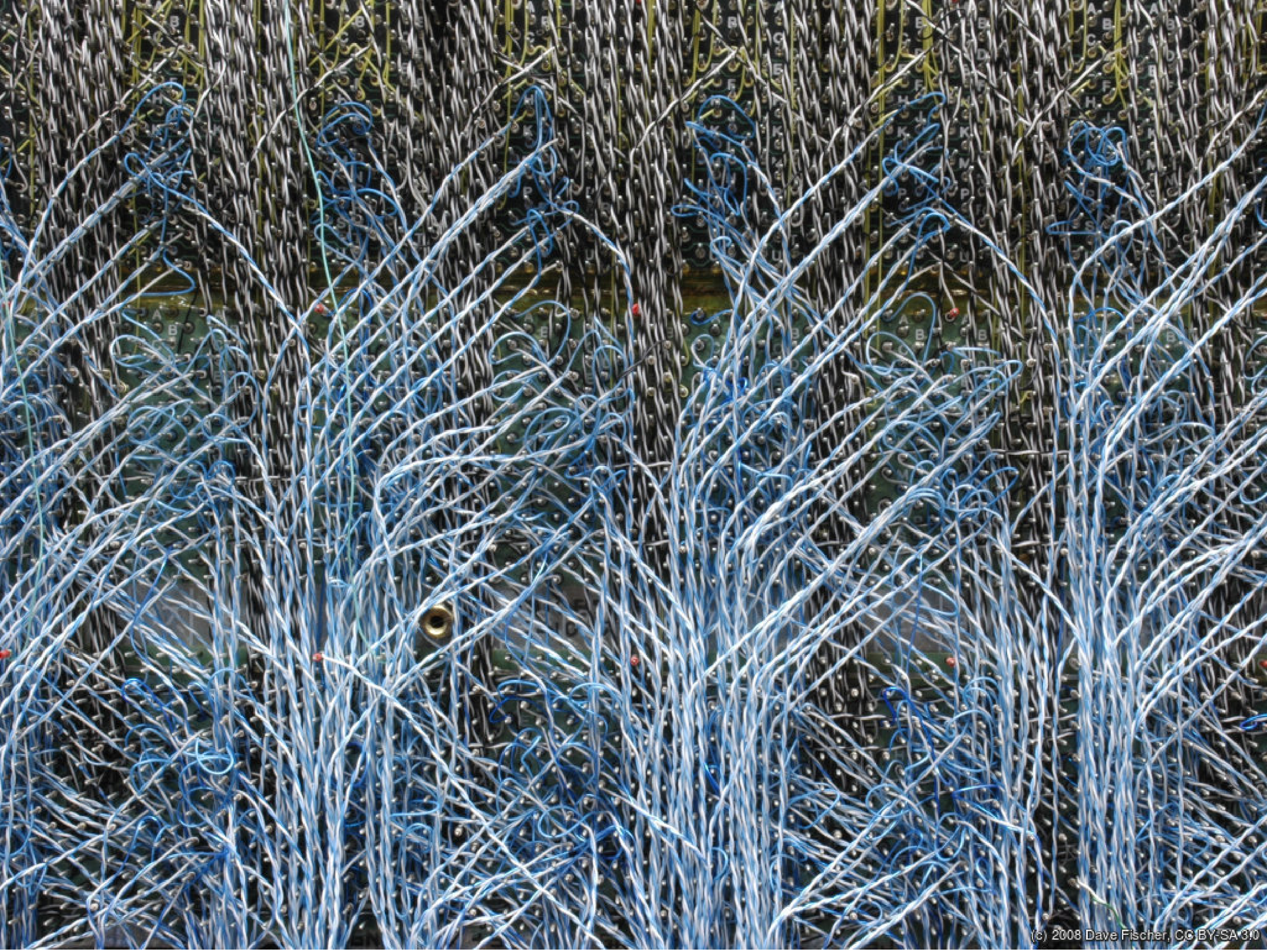
This is an experimental PCB by Tektronix. Produced before 1966 it demonstrates the efforts to improve reliability and the production process. Note the logo artwork.

Photo (c) 2009 Dave Brown, used with permission



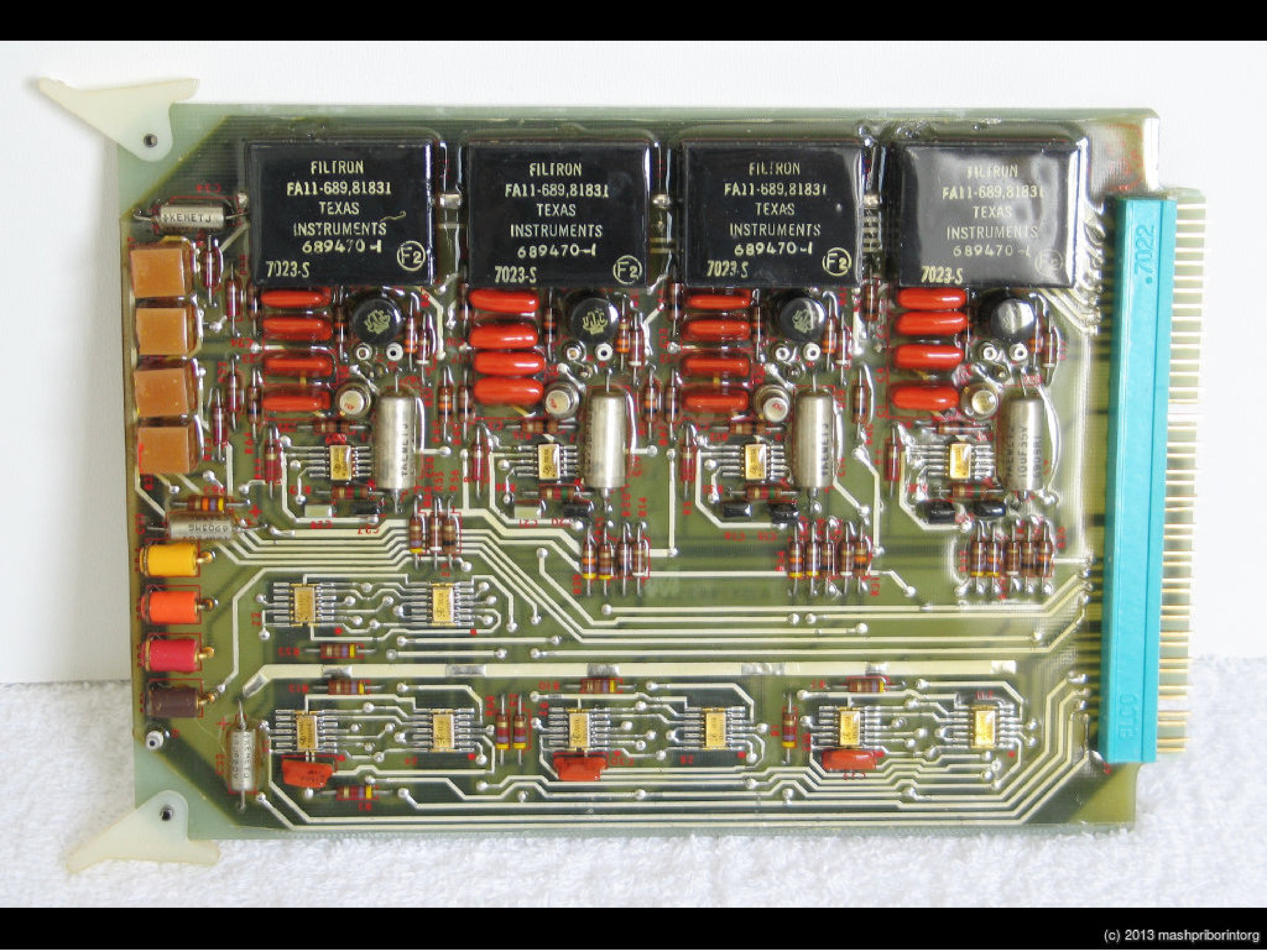
Computer module of a PDP-10 (KA-10) from 1971. This technology was called “Flip Chip Technology”.

Photo (c) 2005 J. Noel Chiappa, CC BY-SA 2.5



However, non-PCB solutions still were prevalent. This backplane of a PDP-10 main processor (KL-10) from 1975 is produced in semi-automatic wire-wrap technology with twisted pair wiring. The wiring instructions were stored on punched cards or tape.

(c) 2008 Dave Fischer, CC BY-SA 3.0

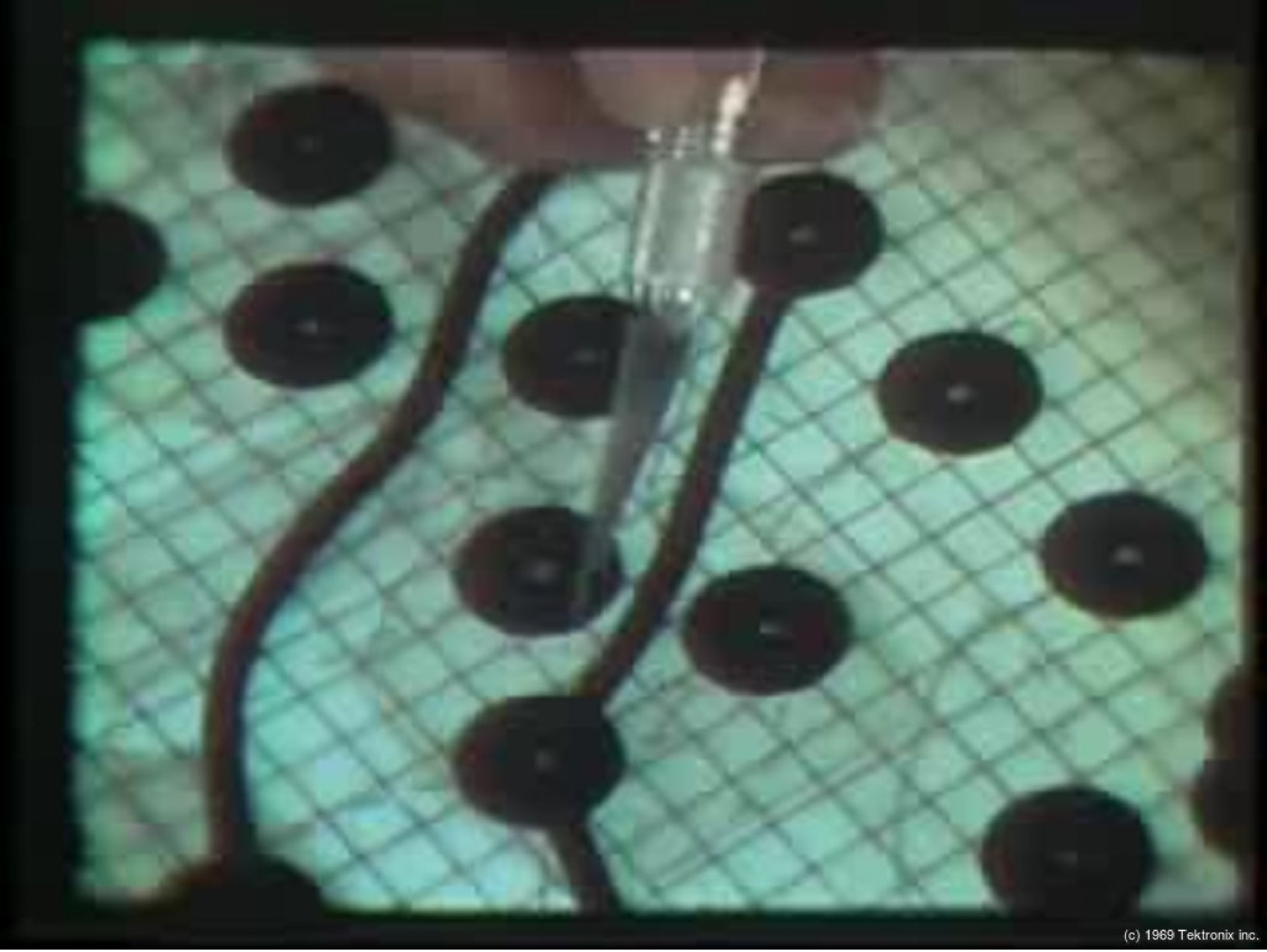


This PCB from 1969 looks astonishingly modern.

Photo (c) 2013 mashpriborintorg, used with permission

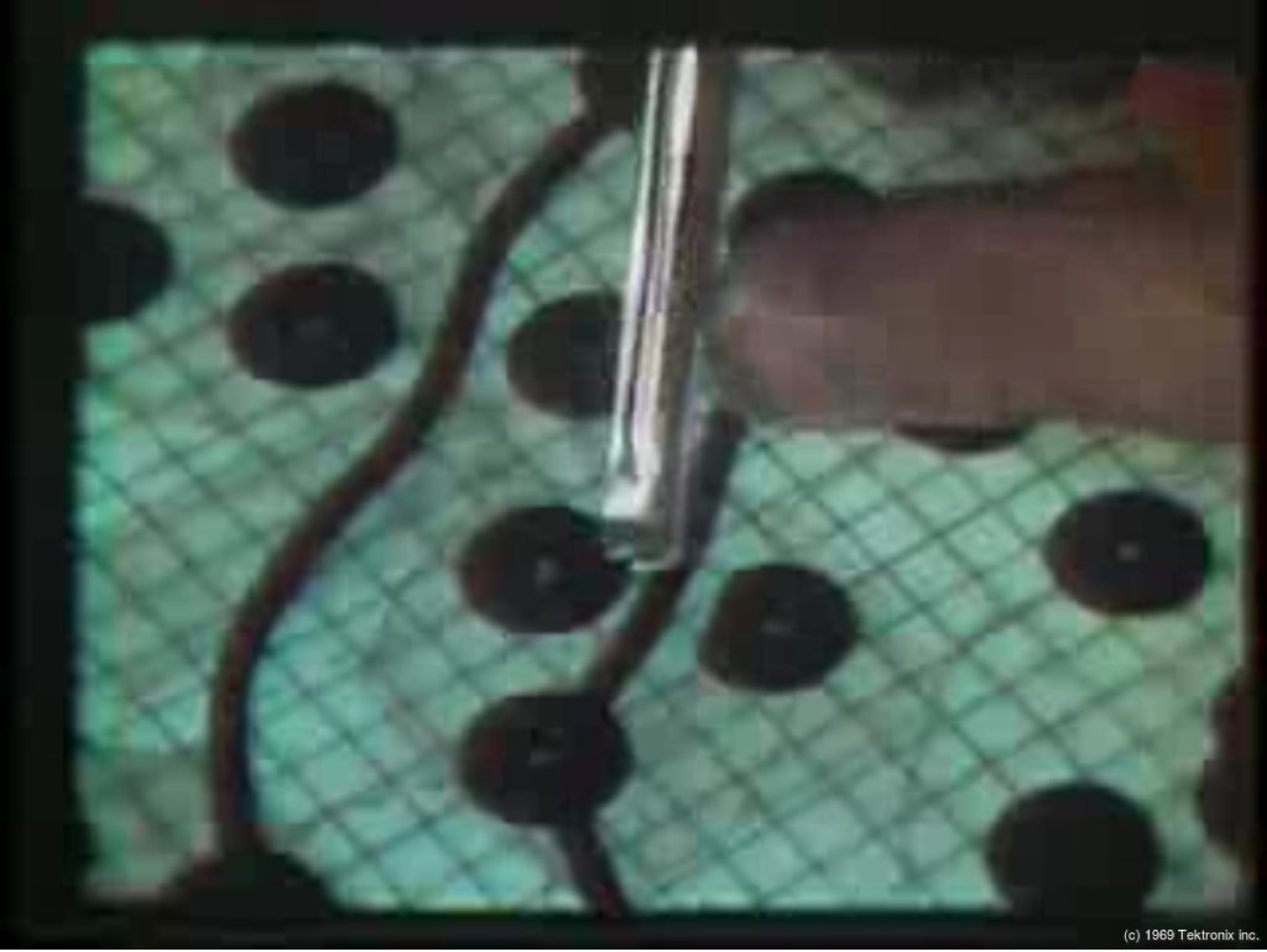
At that time producing PCBs was done by sticking symbols and lines to a sheet of photographic film. Usually this was done in a 2:1 or 4:1 scale.

captured frame (c) 1969 Tektronix inc.



Here the stickers get trimmed to increase the distance between different electrical signals.

captured frame (c) 1969 Tektronix inc.



After designing the PCB the film gets photographically reduced. It is used to expose a photosensitive layer on top of the copper. Areas not exposed get protected from the etching process, forming the conductive structures on the PCB.

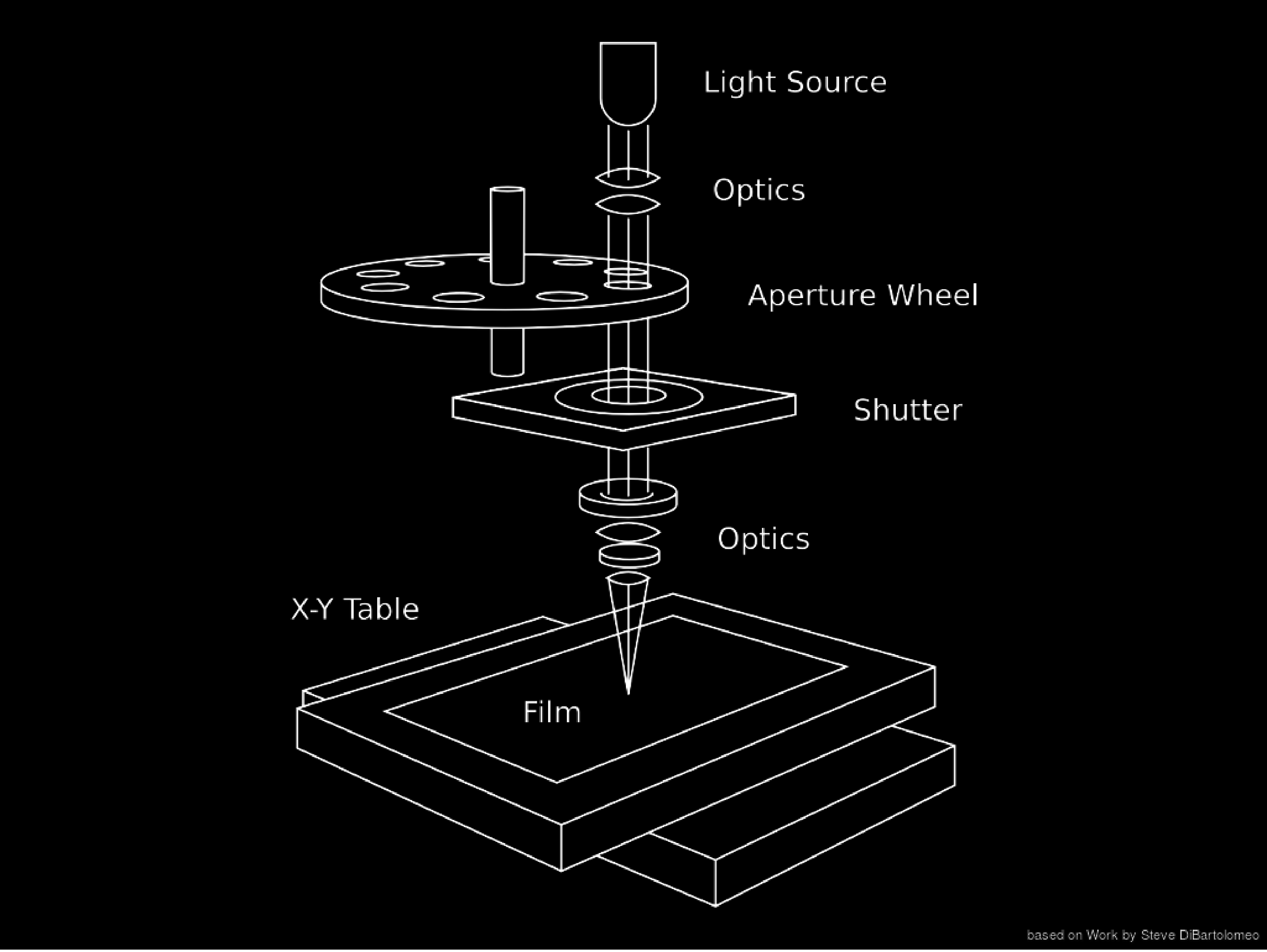
captured frame (c) 1969 Tektronix inc.

Holes get drilled using a pantograph drilling machine.

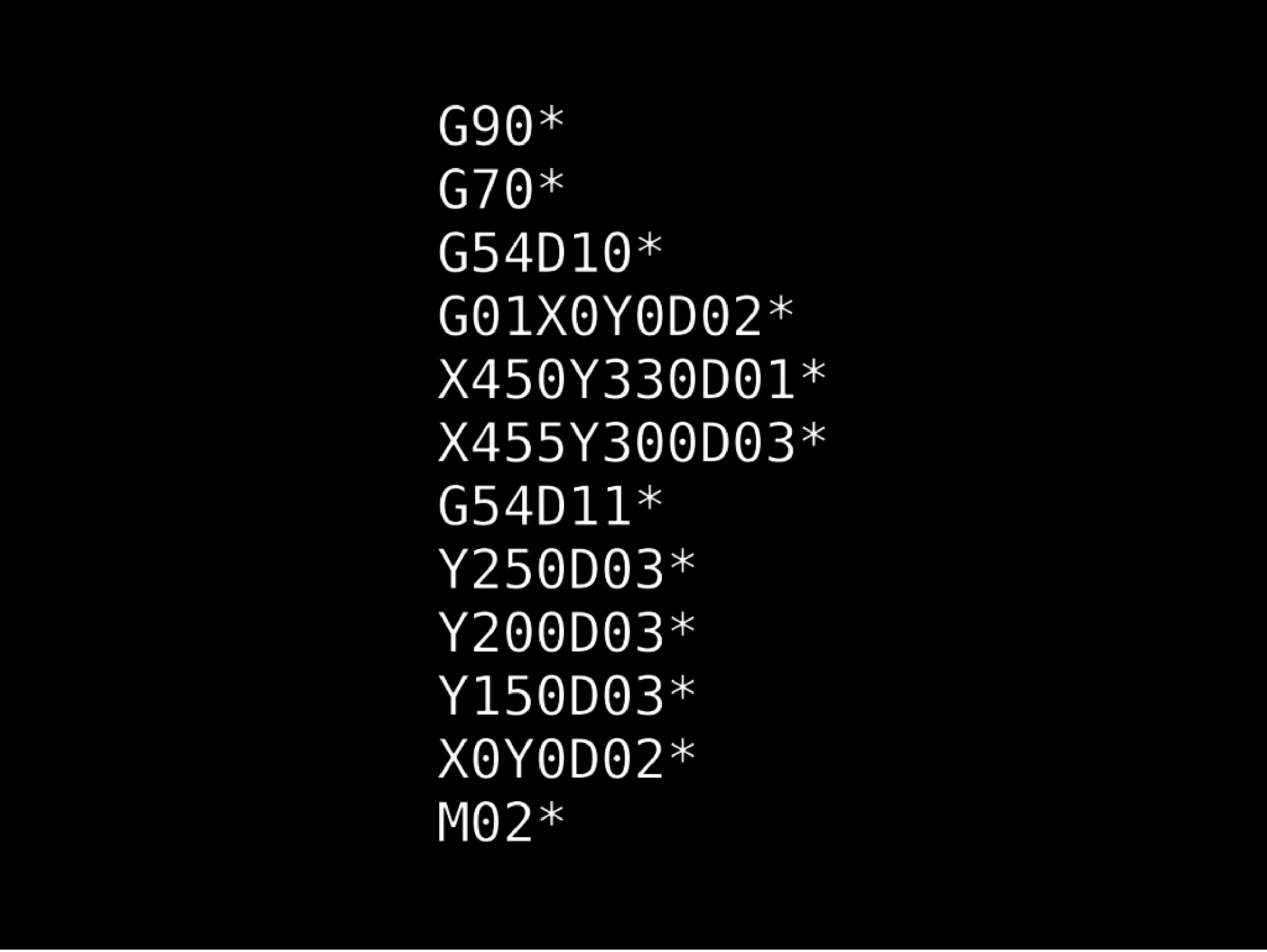
captured frame (c) 1969 Tektronix inc.

Also there already was automation: Drilling instructions on punched tape.

captured frame (c) 1969 Tektronix inc.

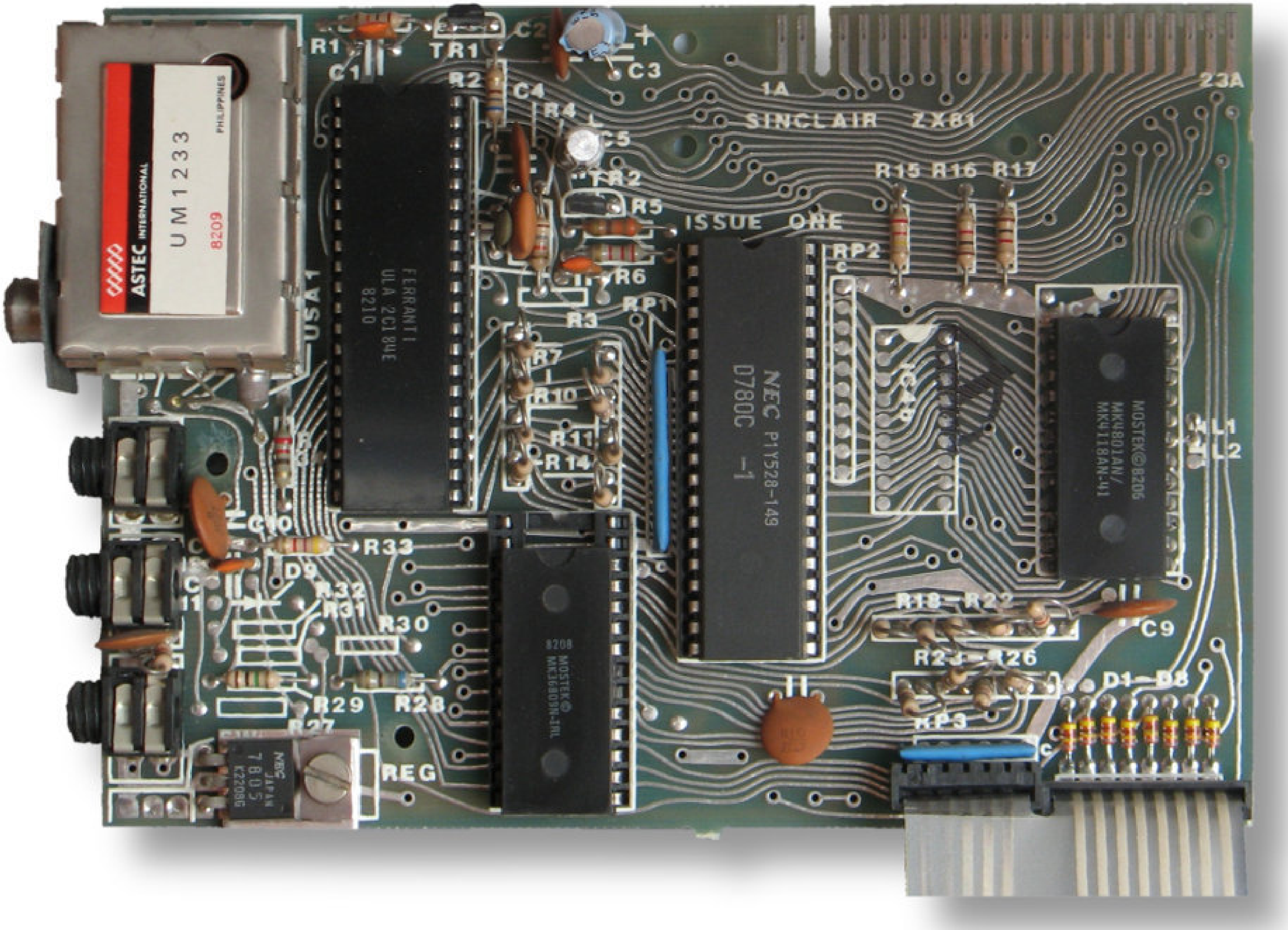


More automation: Photo plotters enable CAD for the PCB design. An aperture shape gets projected on photosensitive film, while moving in X- and Y-direction. Moving the light across the film results in traces, flashes produce specific shapes on the film. based on work by Steve DiBartolomeo



```
G90*
G70*
G54D10*
G01X0Y0D02*
X450Y330D01*
X455Y300D03*
G54D11*
Y250D03*
Y200D03*
Y150D03*
X0Y0D02*
M02*
```

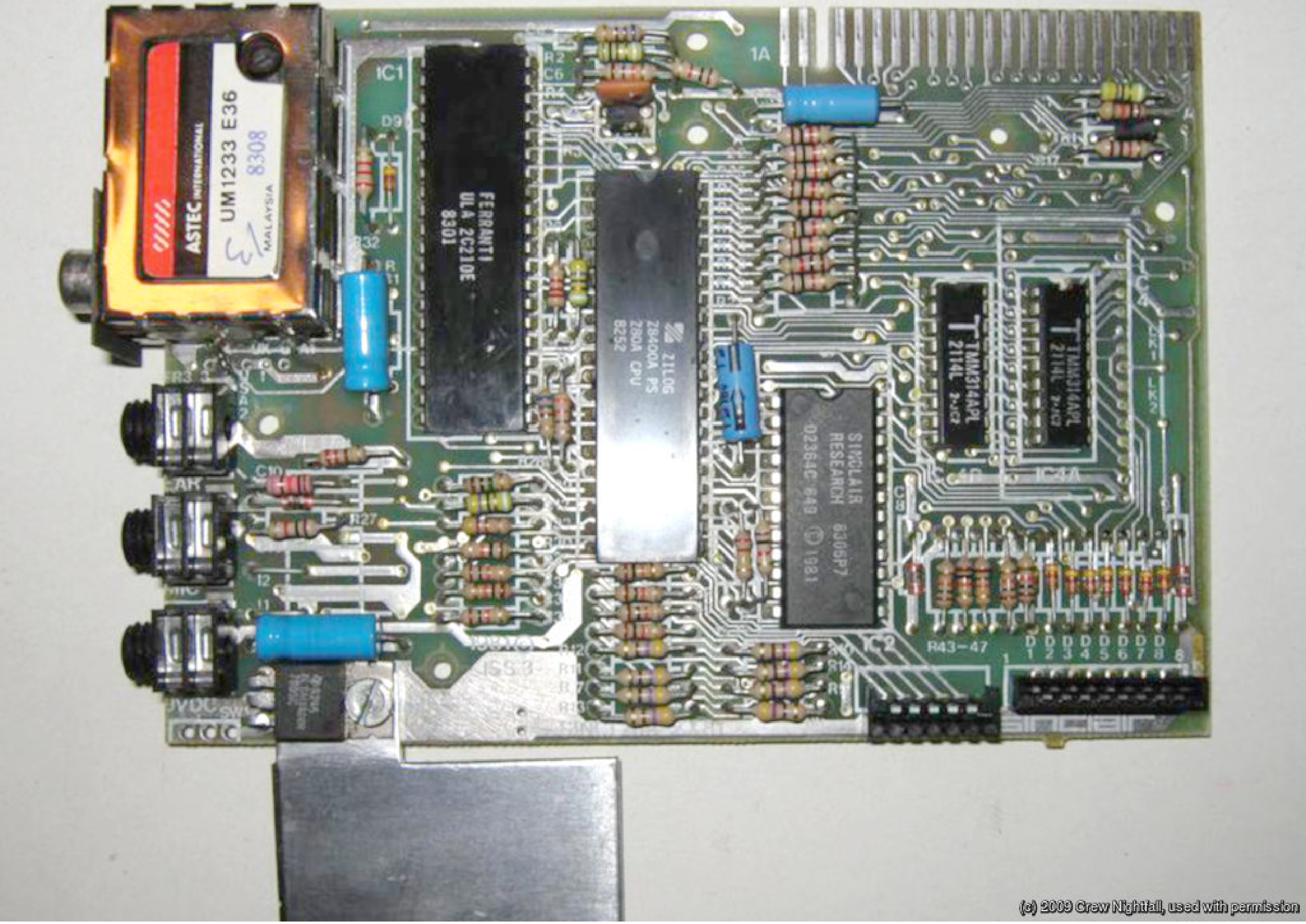
“Gerber” files (named after “Gerber Scientific”, a manufacturer of Photo plotters) today still are widely used to exchange PCB data. A whole ecosystem is built around them, this is not going to change soon. Graphical primitives are straight lines and circular segments, beziers are unavailable. This makes the design of “nice” fonts harder.



(c) Journey234, Public Domain

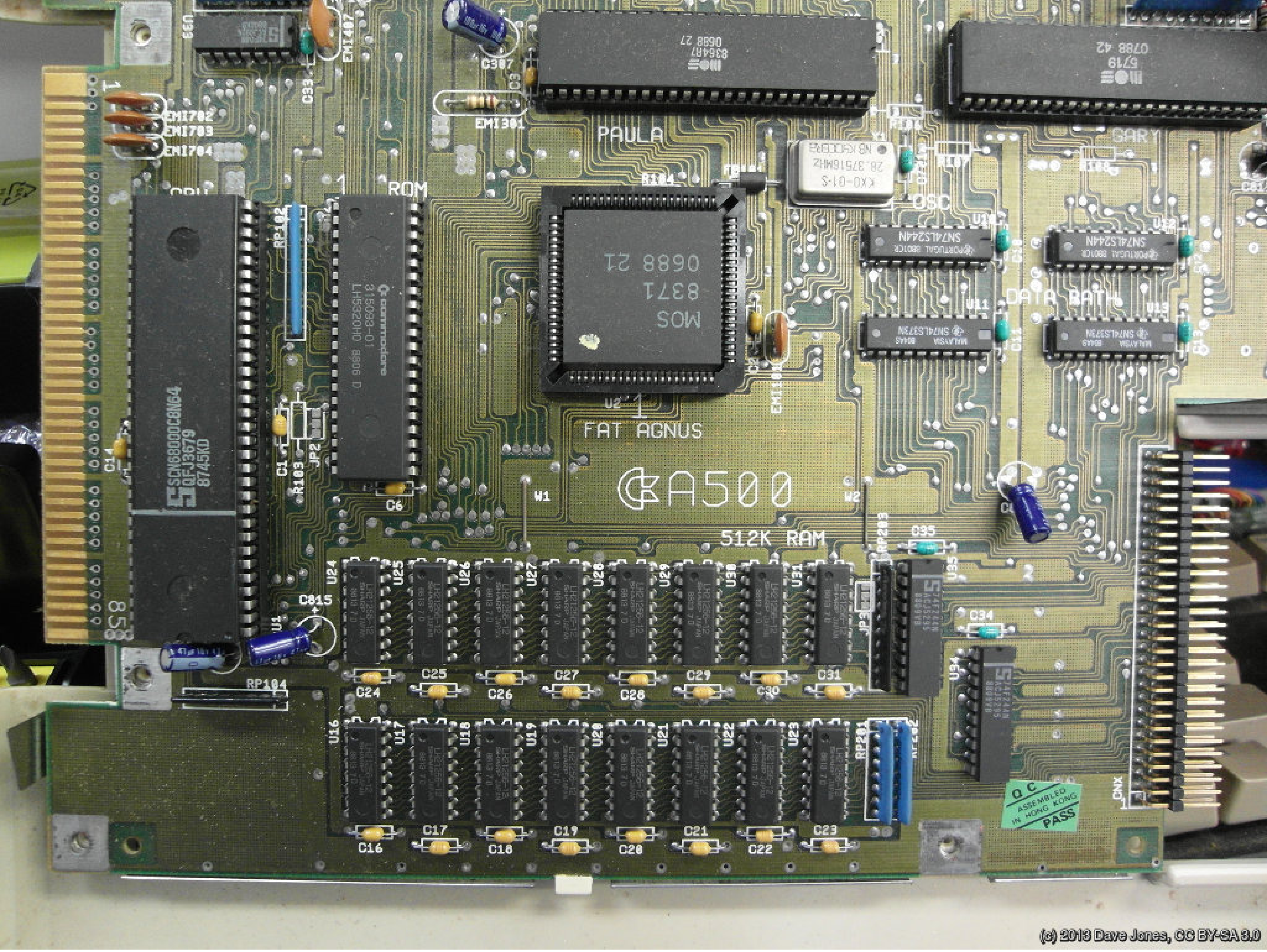
The impact of CAD on the industry can be seen pretty clearly in this example. This PCB from a Sinclair ZX81 from 1981 is done with manual tools. Have a look at the “smooth” and “organic” shapes used for the traces.

(c) Journey234, Public Domain



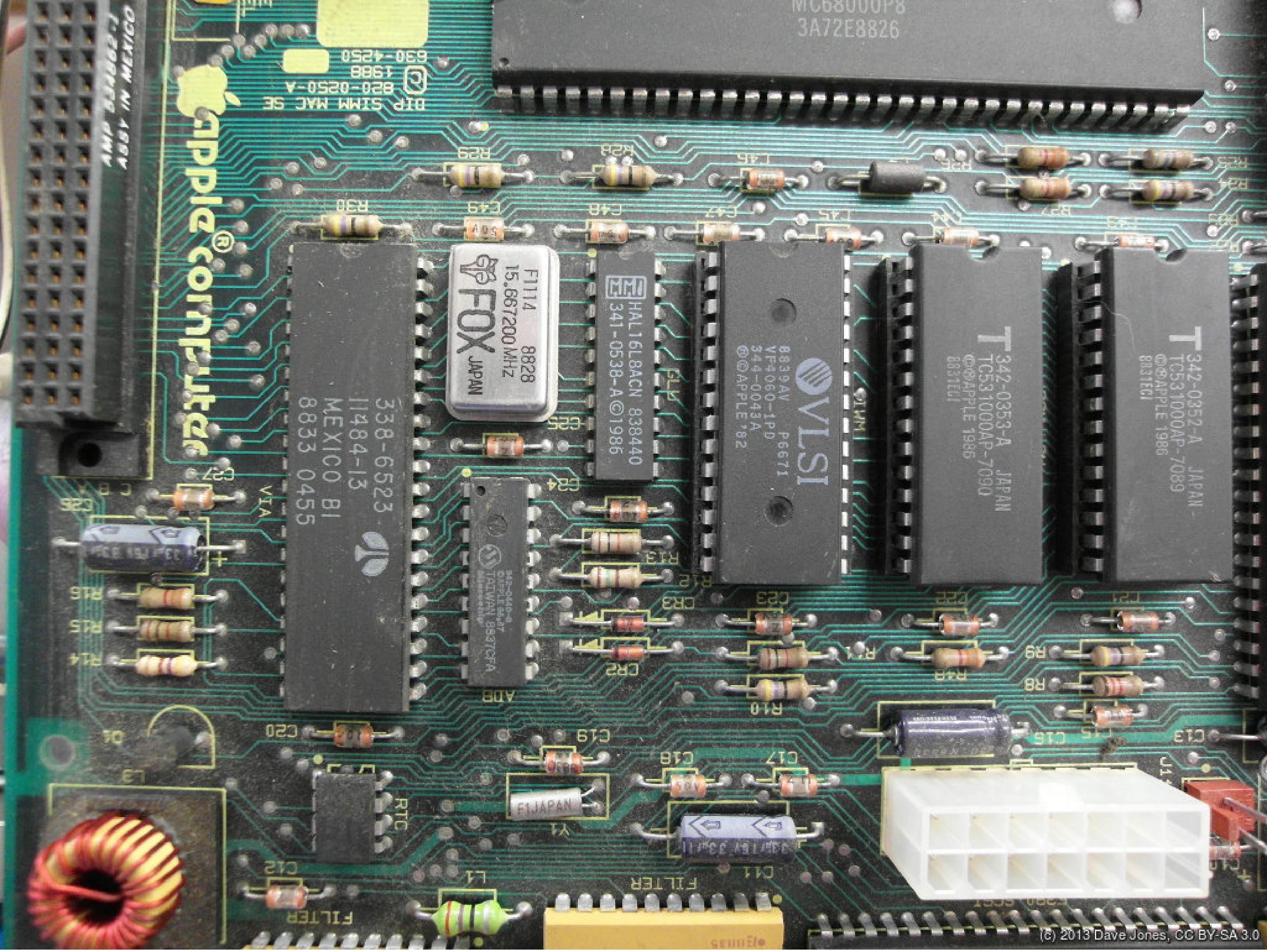
Two years later the ZX81 gets reissued and now the PCB design is done with CAD tools. Note how straight lines as well as the typical 45° angles we are familiar with in PCB design appear.

(c) 2009 Crew Nightfall, used with permission



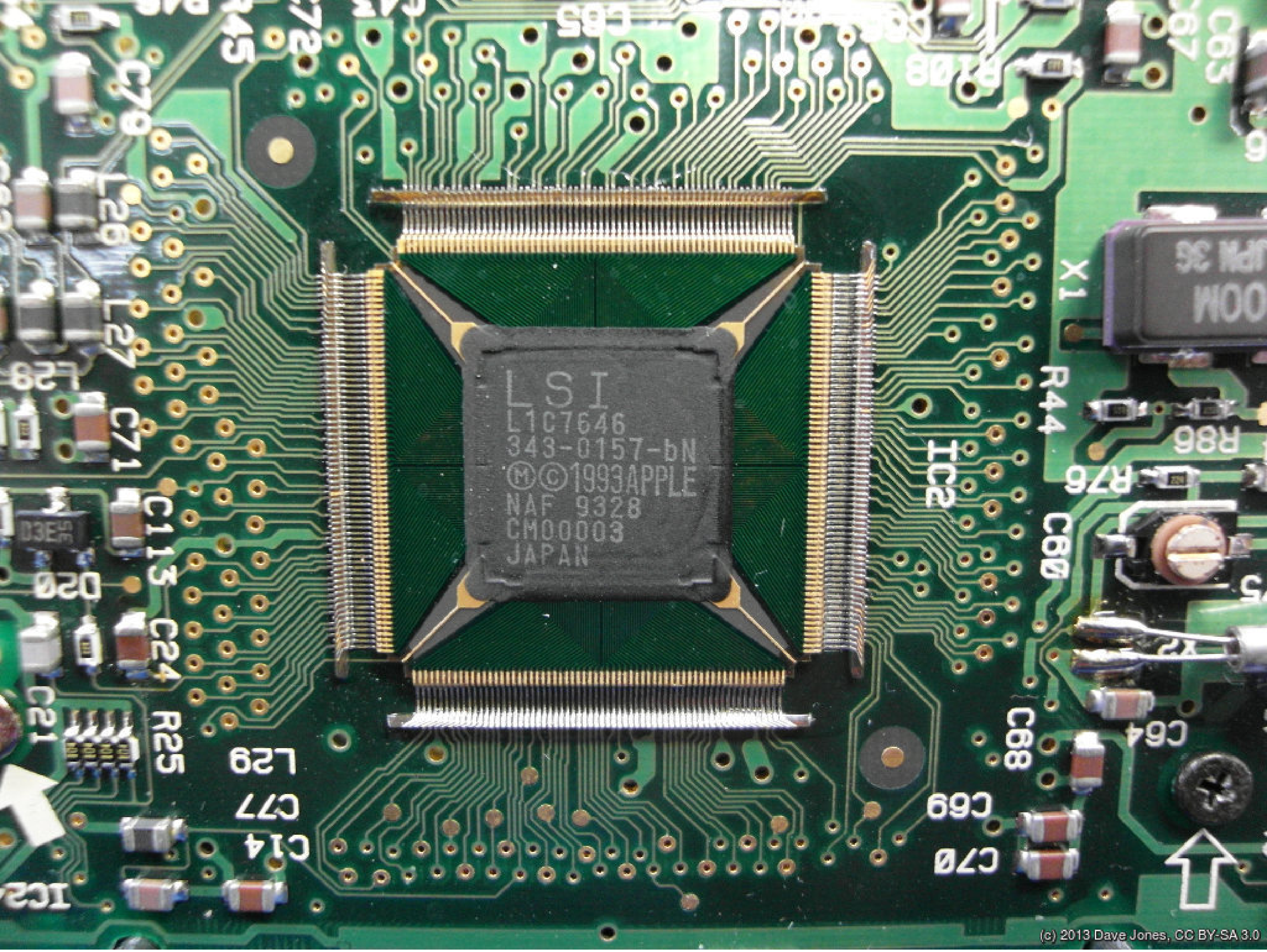
In the Amiga 500 we see CAD text: the letters are constructed from straight line segments. However, there are circular segments in the corners in the signal traces. Someone put effort into the Commodore logo.

(c) 2013 Dave Jones, CC BY-SA 3.0



An Apple Macintosh SE PCB from 1988. Again obviously CAD, this time with the typical 45° corners. Component parts are marked with CAD text on the silkscreen, but in this design the appearance of the company logo/branding has received attention.

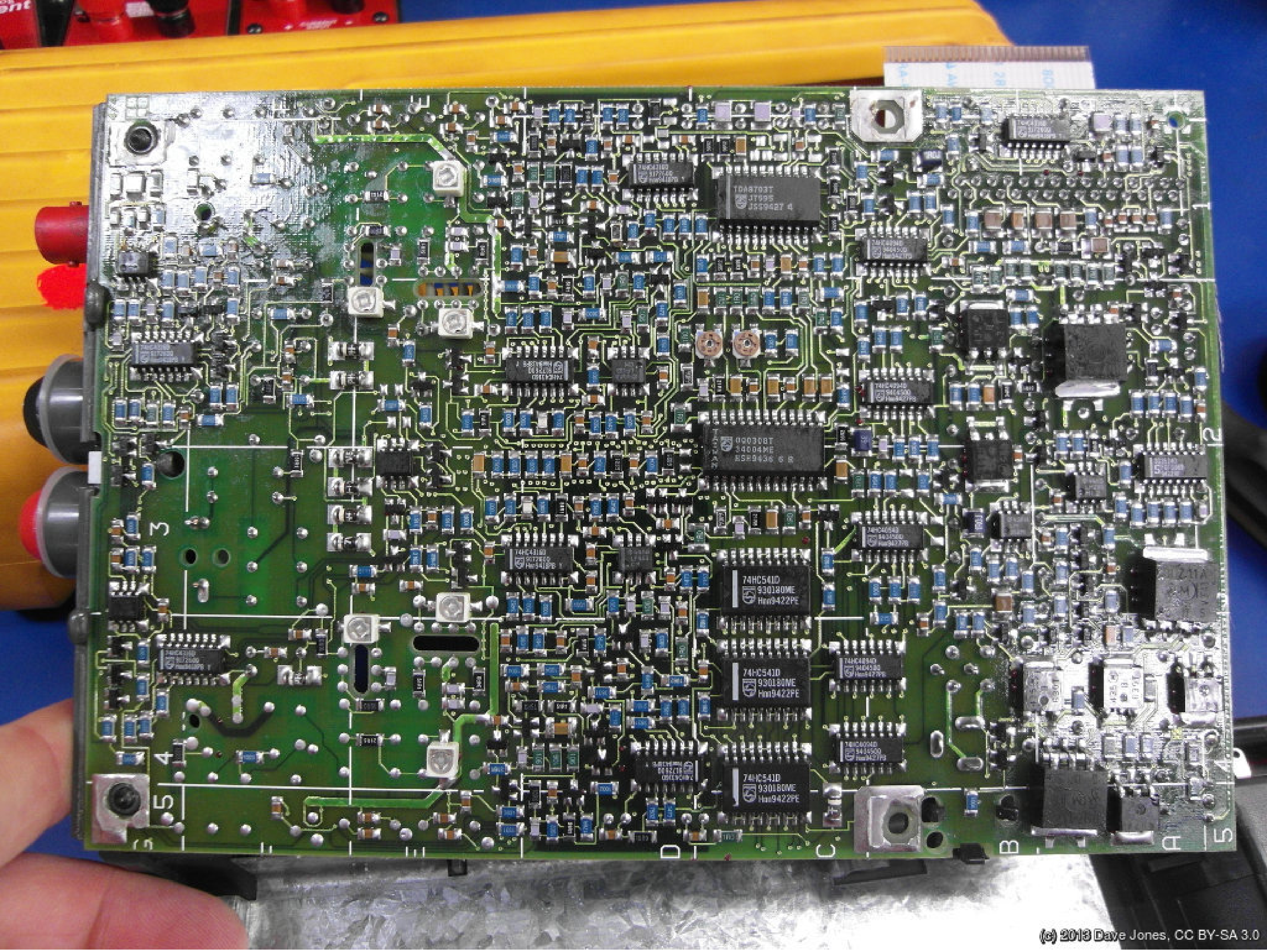
(c) 2013 Dave Jones, CC BY-SA 3.0



(c) 2013 Dave Jones, CC BY-SA 3.0

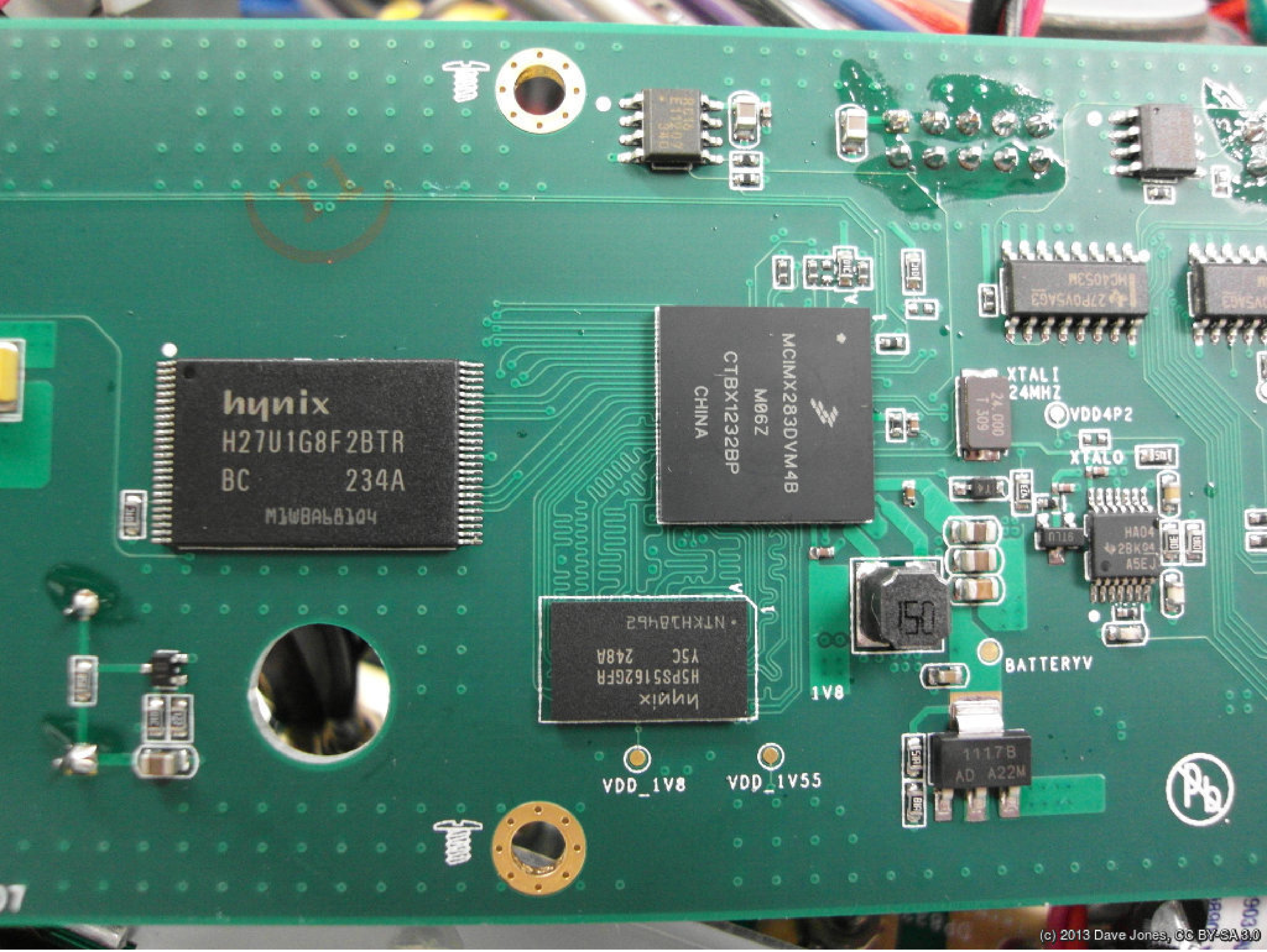
An Apple Newton Messagepad from 1993. Isn't this unbundling of chip connections just beautiful?

(c) 2013 Dave Jones, CC BY-SA 3.0



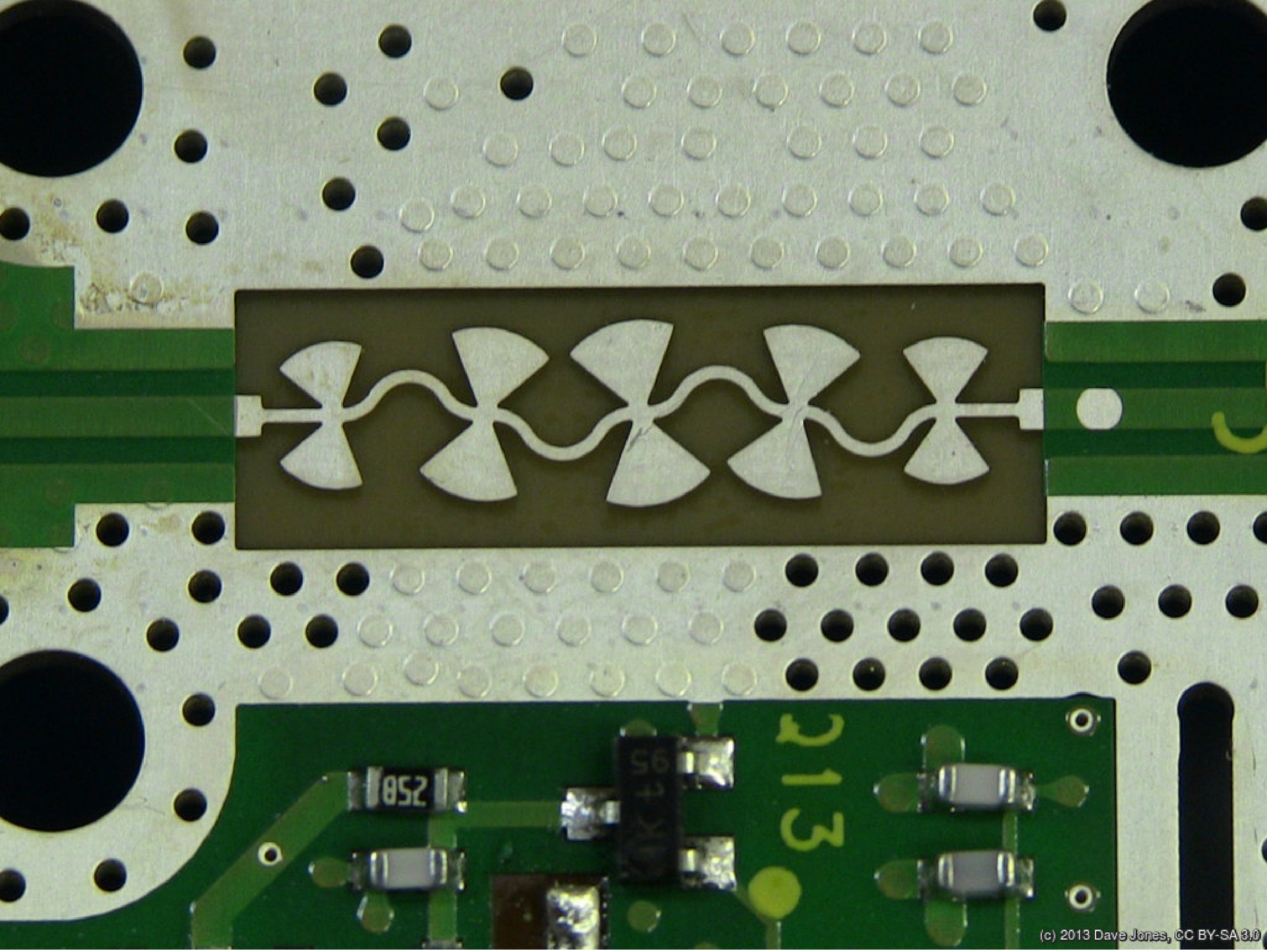
PCBs primarily serve technical purposes. They have to work on a technical level, aesthetics comes second, if at all. This is from a Fluke Scopemeter from 1994. One of the primary purposes obviously was to serve as a carrier for electronic parts...

(c) 2013 Dave Jones, CC BY-SA 3.0



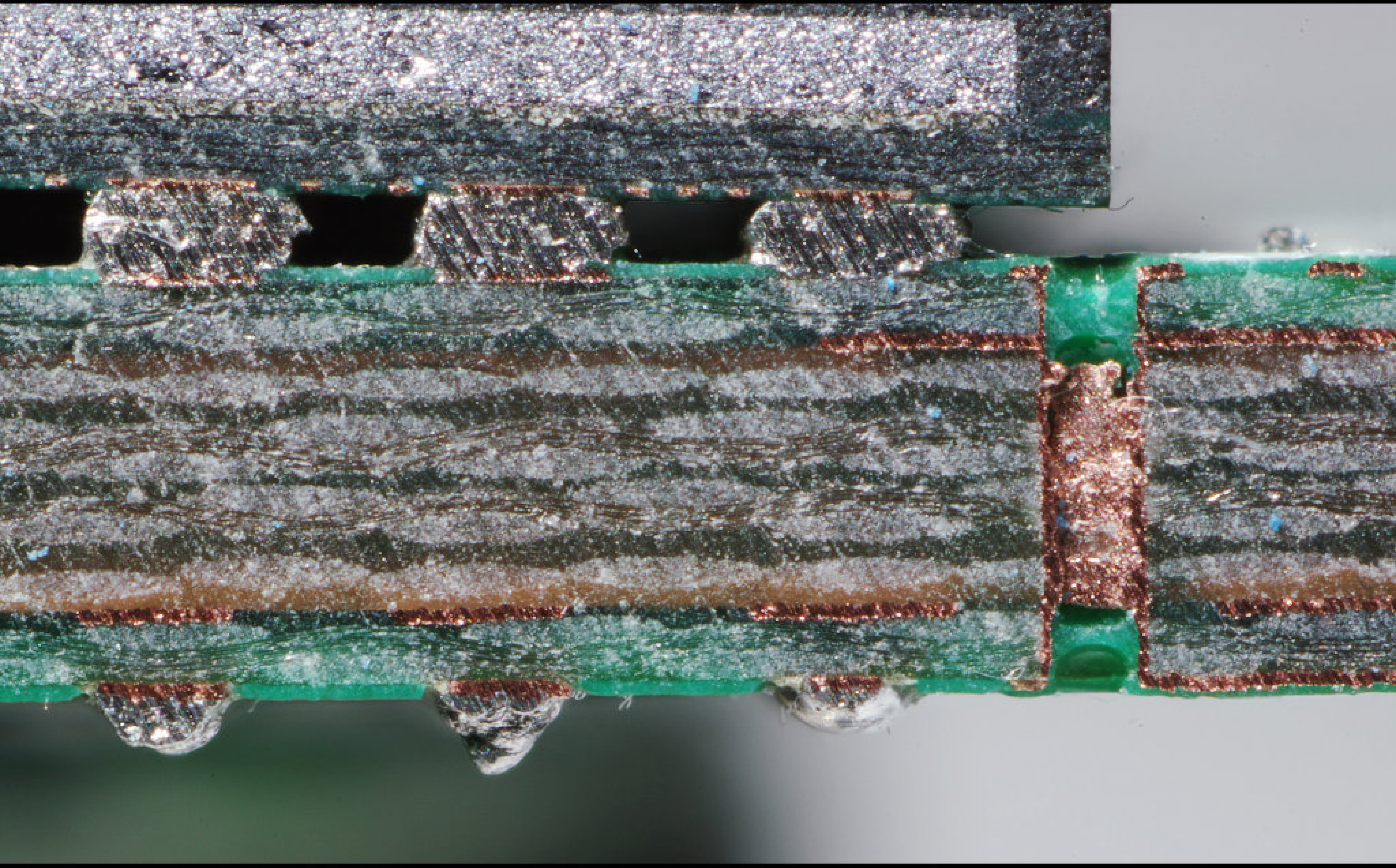
The snake-style traces are fancy, but serve a purely functional purpose. This is a high-speed bus on a PCB of a Rigol power supply from 2012. The snakes ensure that the trace lengths of the bus match up and the signal integrity for high speed signals is maintained.

[(c) 2013 Dave Jones, CC BY-SA 3.0]



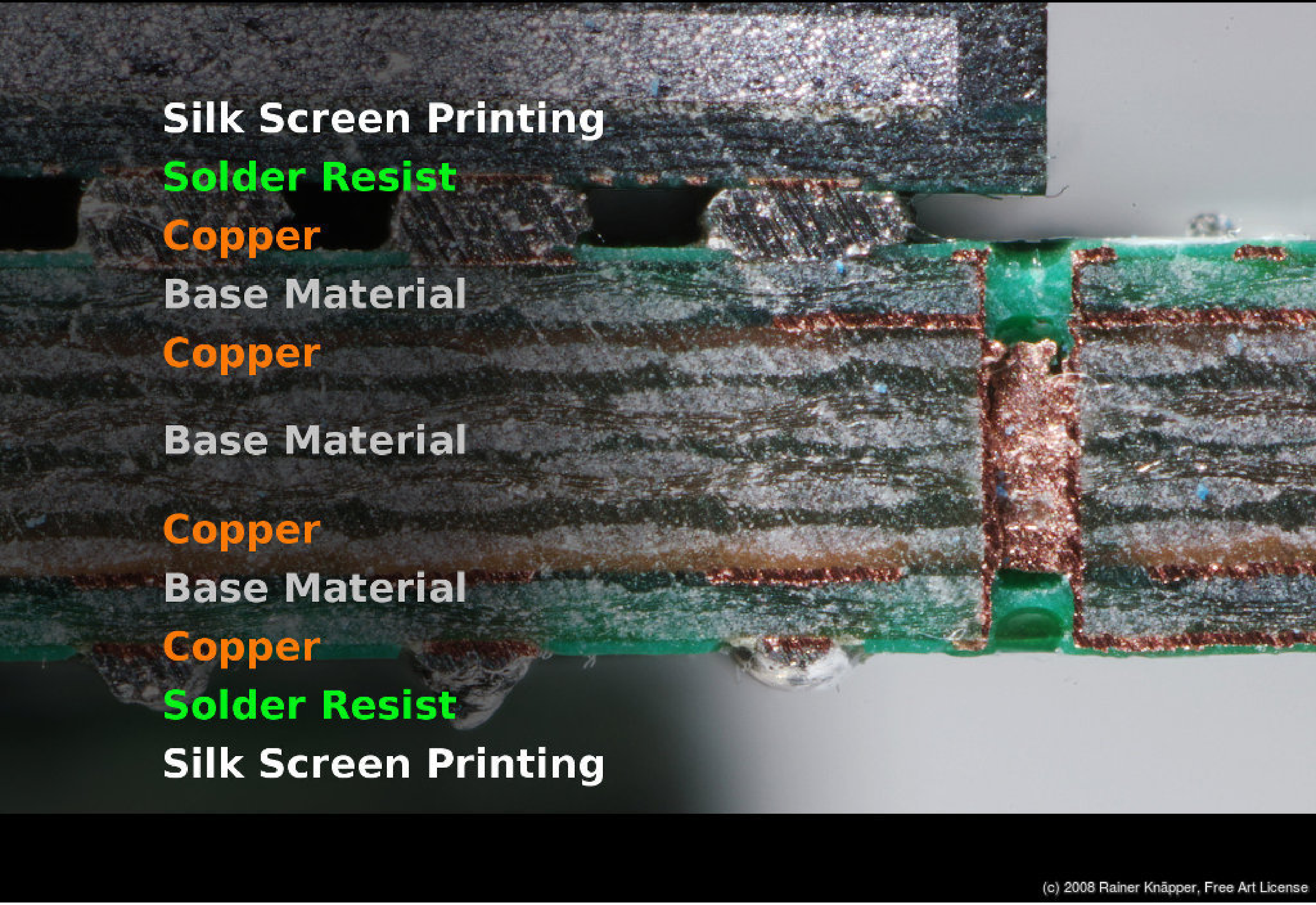
And here we enter high-frequency-voodoo. This is not an attempt at drawing ventilators, it is some sort of filter for very high frequency signals, found in an Agilent Spectrum Analyzer from 2010.

(c) 2013 Dave Jones, CC BY-SA 3.0

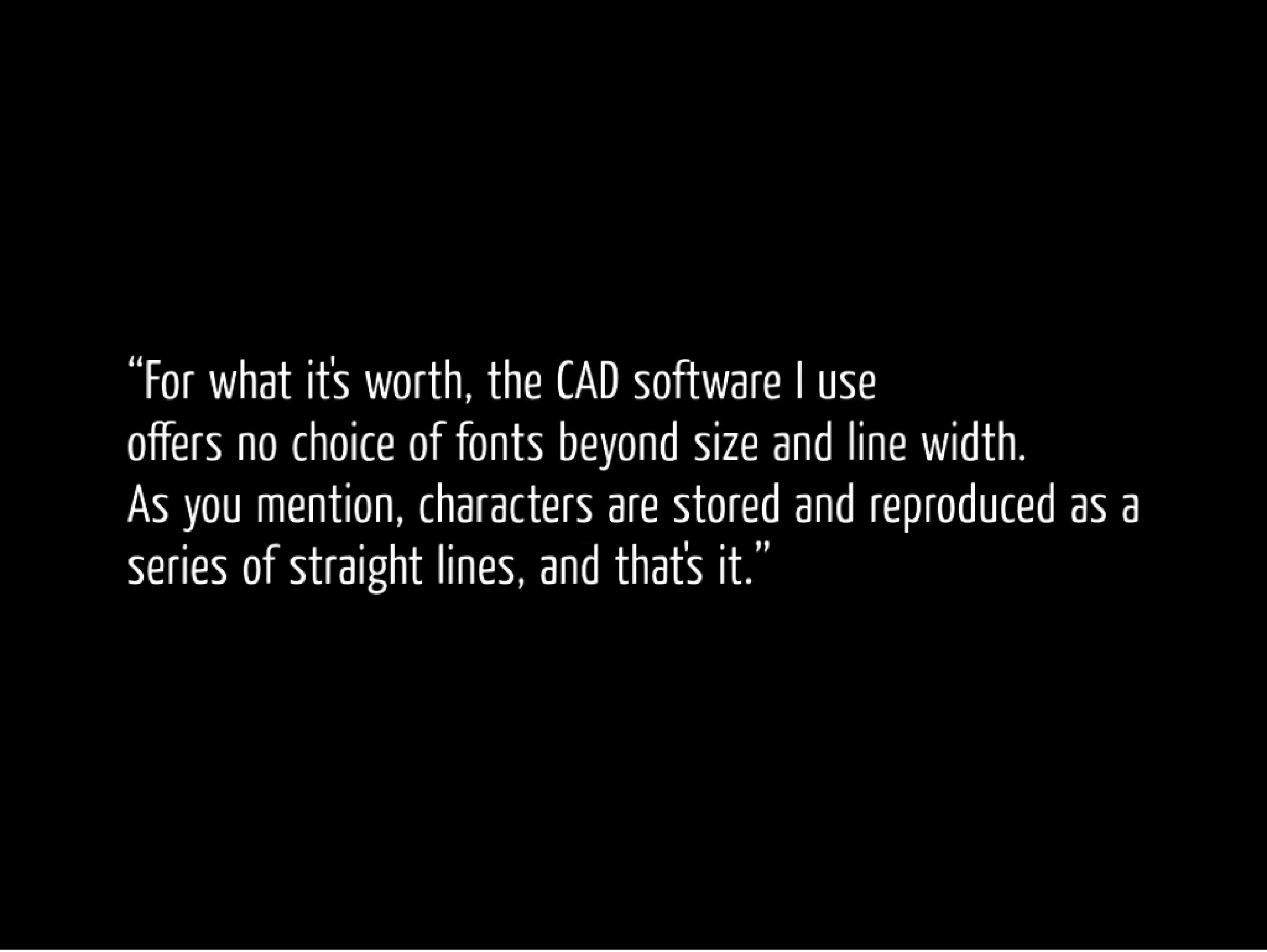


(c) 2008 Rainer Knäpper, Free Art License

This is a cross section of a 4 layer PCB. Photographic film is used with photosensitive material to form a protective layer on top of thin copper sheets bonded to base material. Unwanted copper is then etched away, imposing certain minimum requirements. A typical 4-layer PCB is composed of 11 layers... (c) 2008 Rainer Knäpper, Free Art License

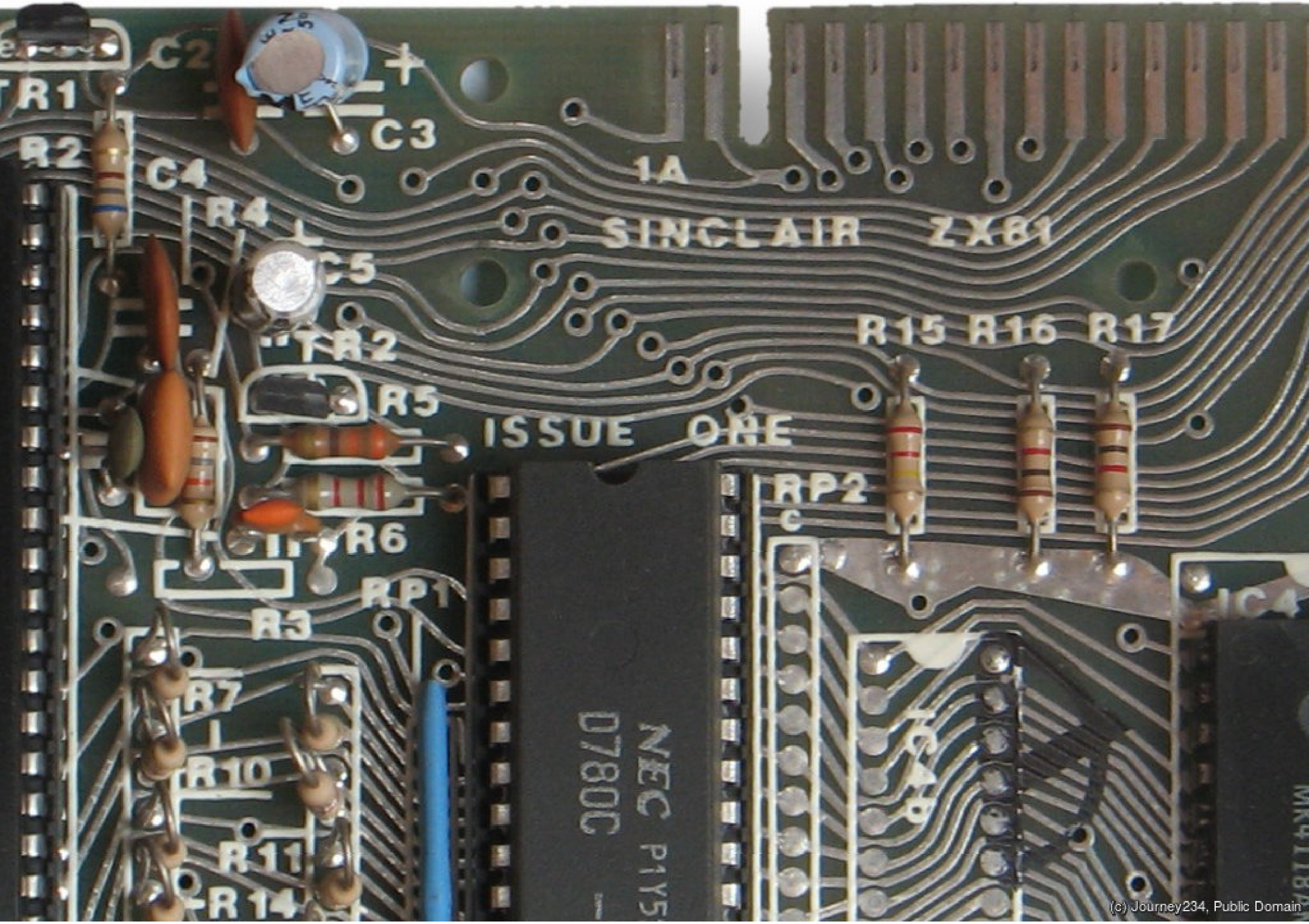


The outer 4 layers of either side – silkscreen, soldermask, copper, base material – are usable for design elements. Printing quality of the silk screen layer varies wildly, copper generally provides a very high resolution and accuracy. The copper layer usually is plated with tin or gold. [(c) 2008 Rainer Knäpper, Free Art License]



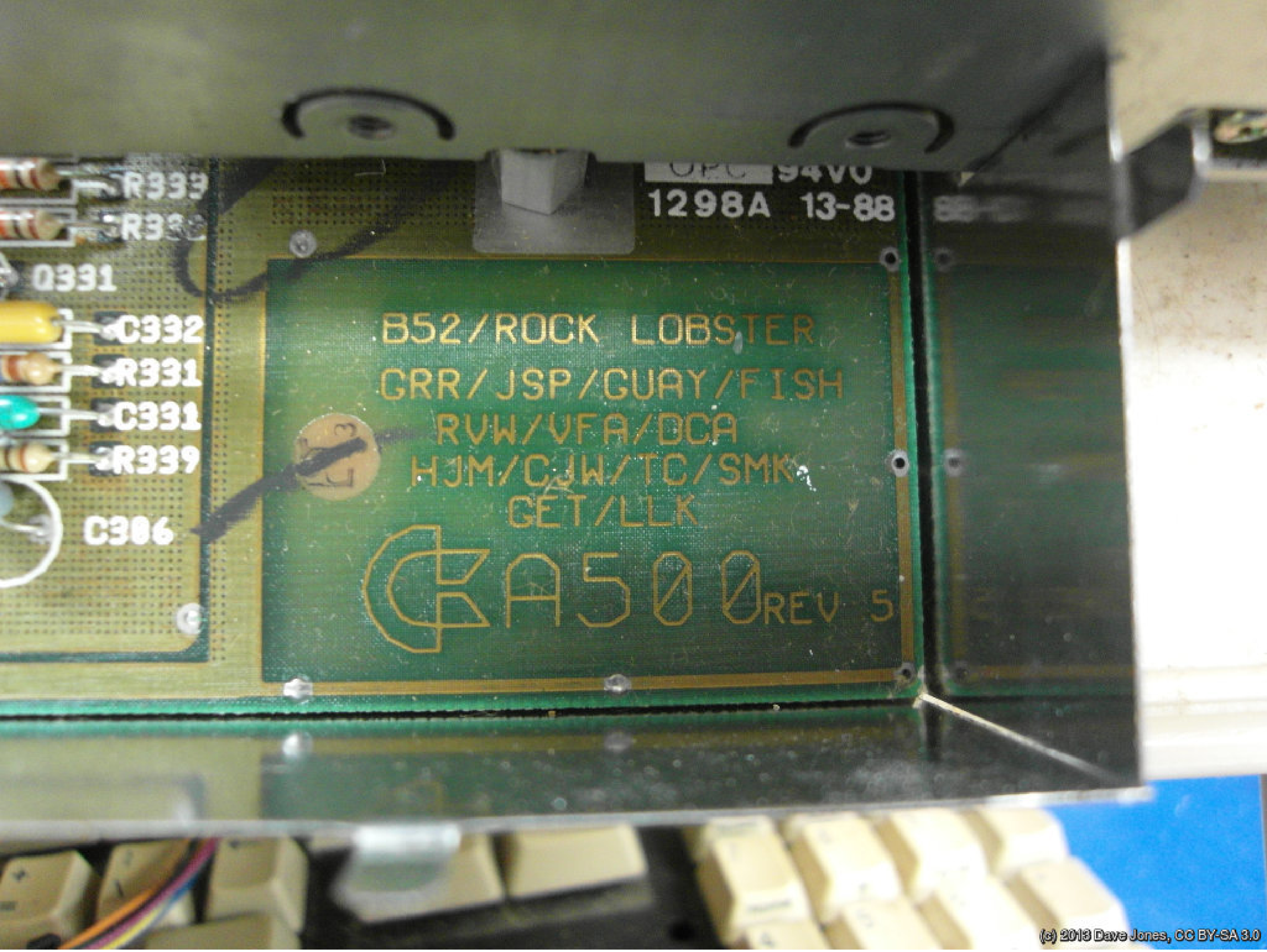
> “For what it's worth, the CAD software I use
> offers no choice of fonts beyond size and line width.
> As you mention, characters are stored and reproduced as a
> series of straight lines, and that's it.”

Lets look at the tools. The current state described by AndyC_772 is kind of a step backwards from earlier days.



The PCB for the ZX81 features a nice font in its silkscreen layer. The silk screen printing is a bit thick, but the glyph shapes are good and clean.

(c) Journey234, Public Domain

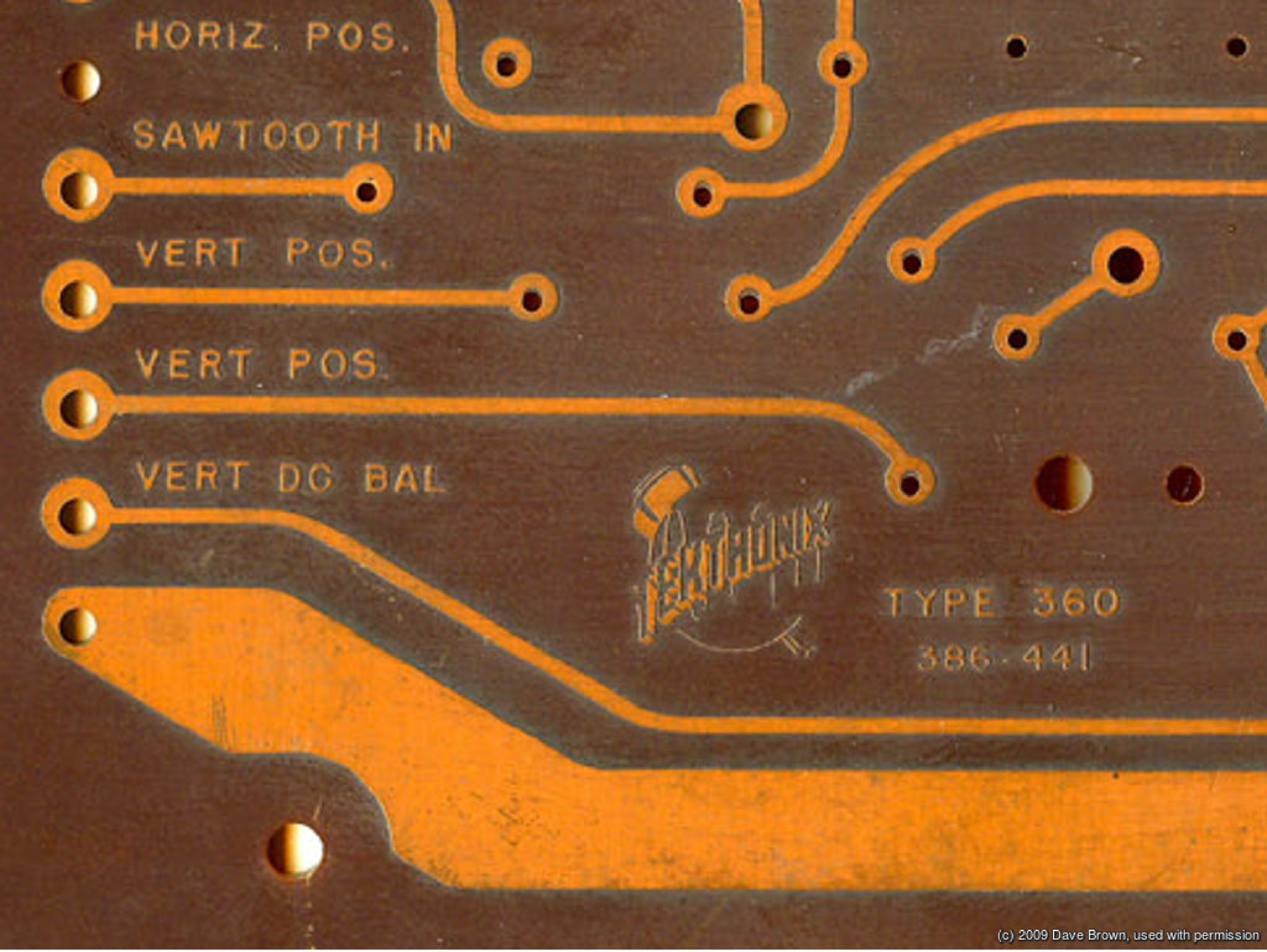


Again the Amiga 500 with some CAD text in copper. The glyphs and logo are constructed from straight lines. In the silkscreen we actually see CAD text as well as properly shaped glyphs.

(c) 2013 Dave Jones, CC BY-SA 3.0

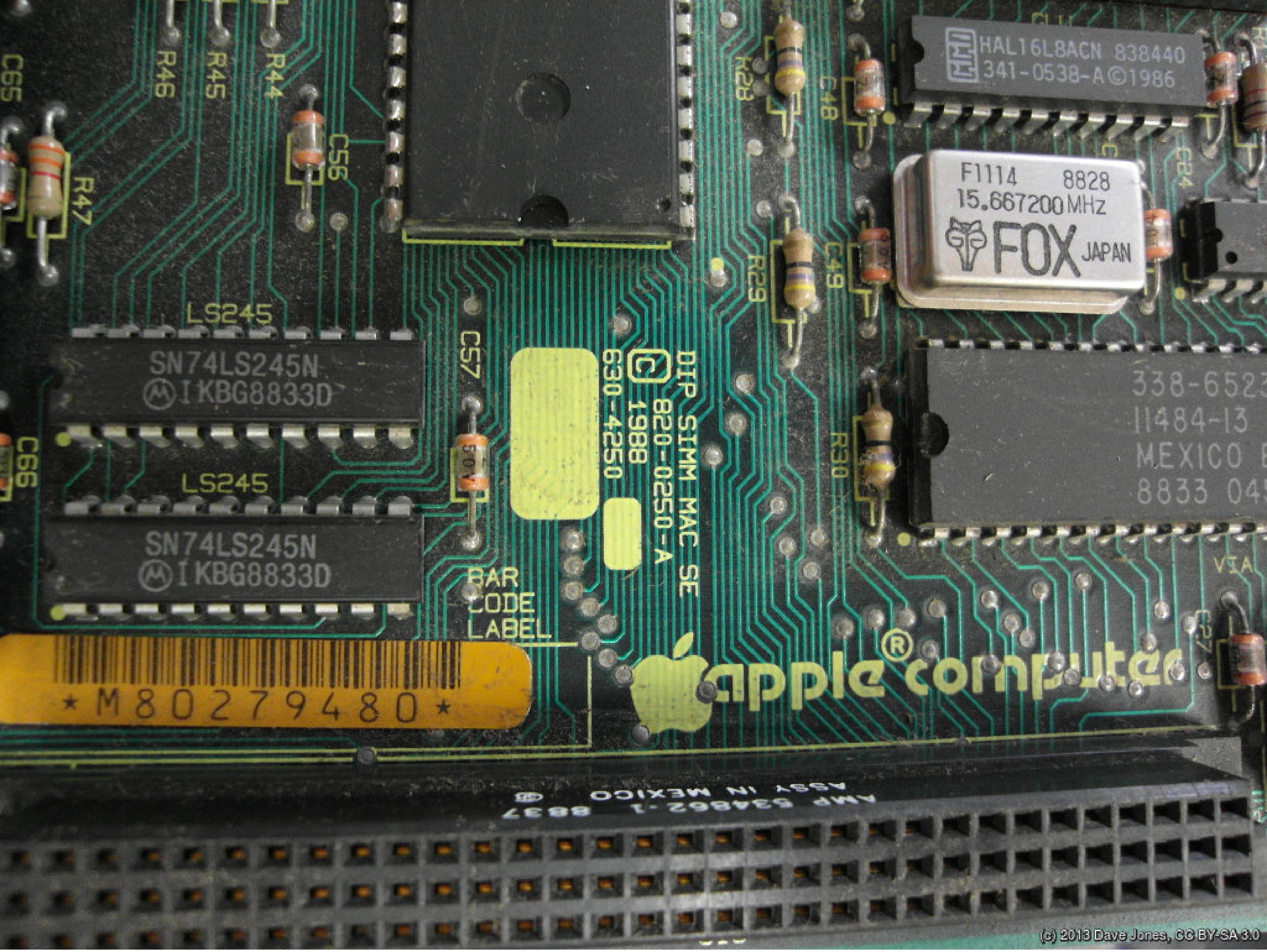
“I can import graphics from outside (such as a company logo),
but even that is a time consuming process with multiple steps,
and not something I would ever do
just to change the look of my board even if I wanted to.”

I can confirm that working with design elements is cumbersome, yet this happens despite these problems.



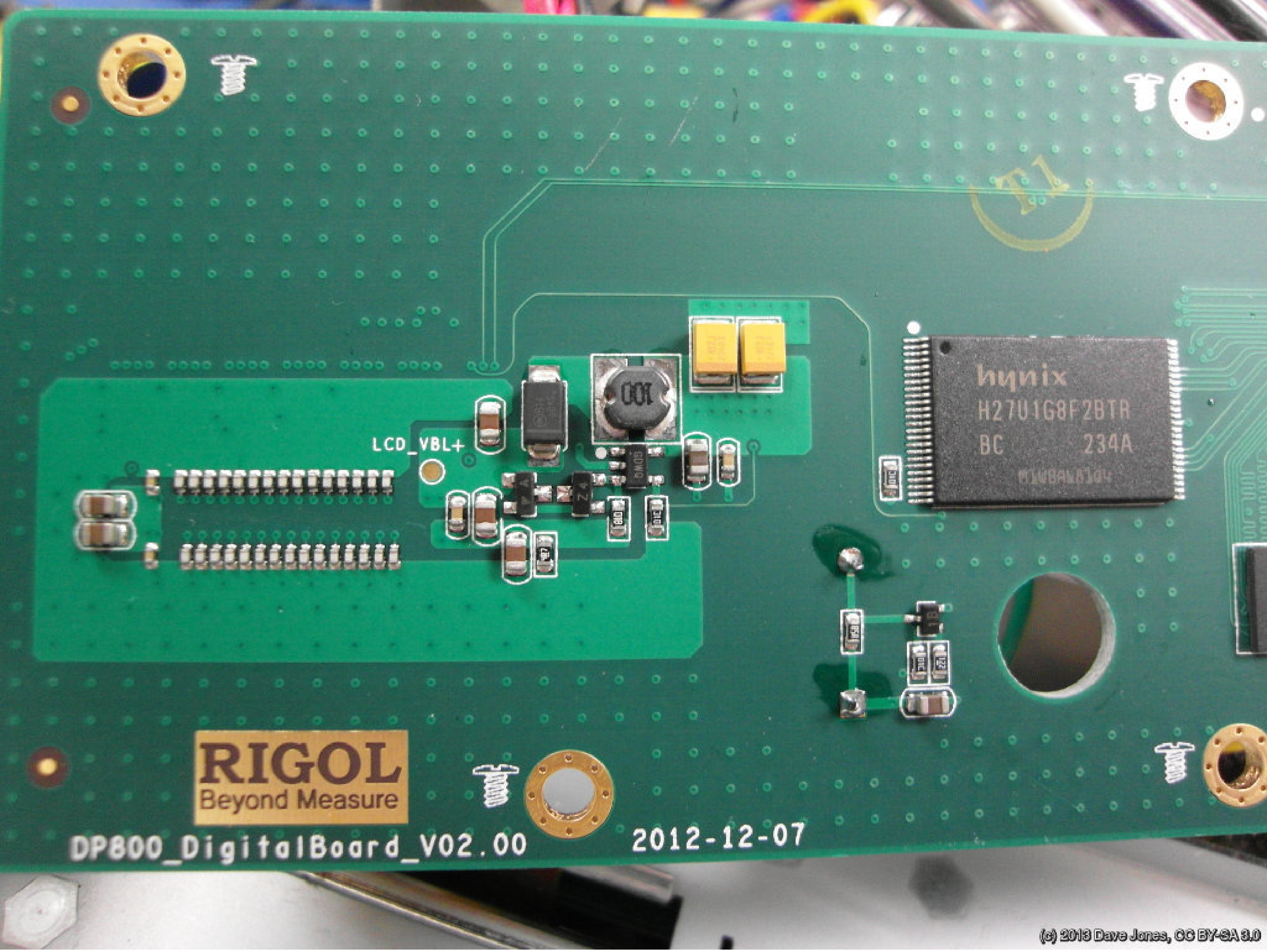
Again the experimental Tektronix PCB from 1966. They made a point of incorporating the logo. It turned out to be over-etched (too much copper removed) but it is fairly detailed.

(c) 2009 Dave Brown, used with permission



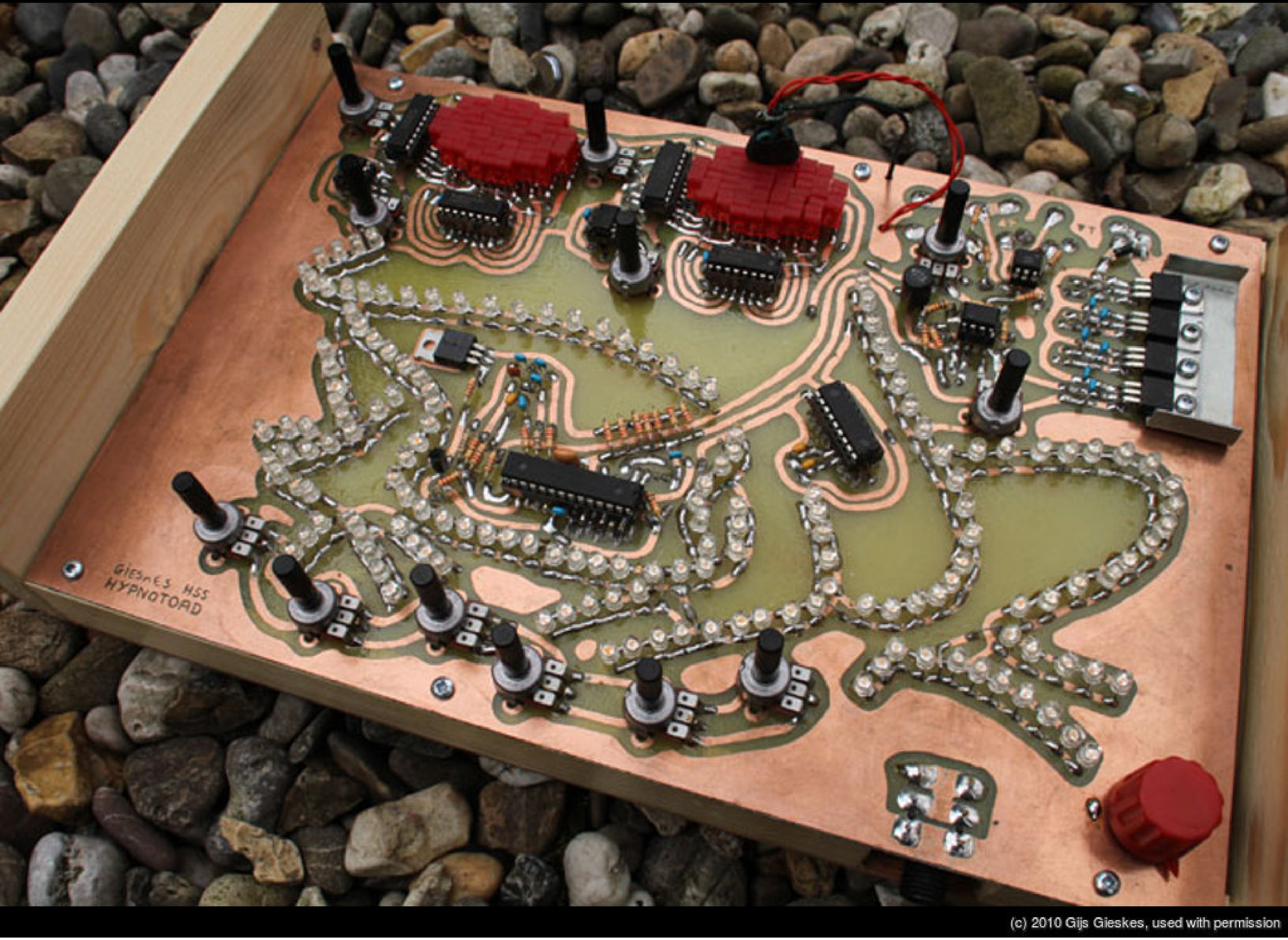
Apple (again the Apple Macintosh SE, 1988) has CAD text as well as really smooth logo artwork in the silkscreen layer.

(c) 2013 Dave Jones, CC BY-SA 3.0



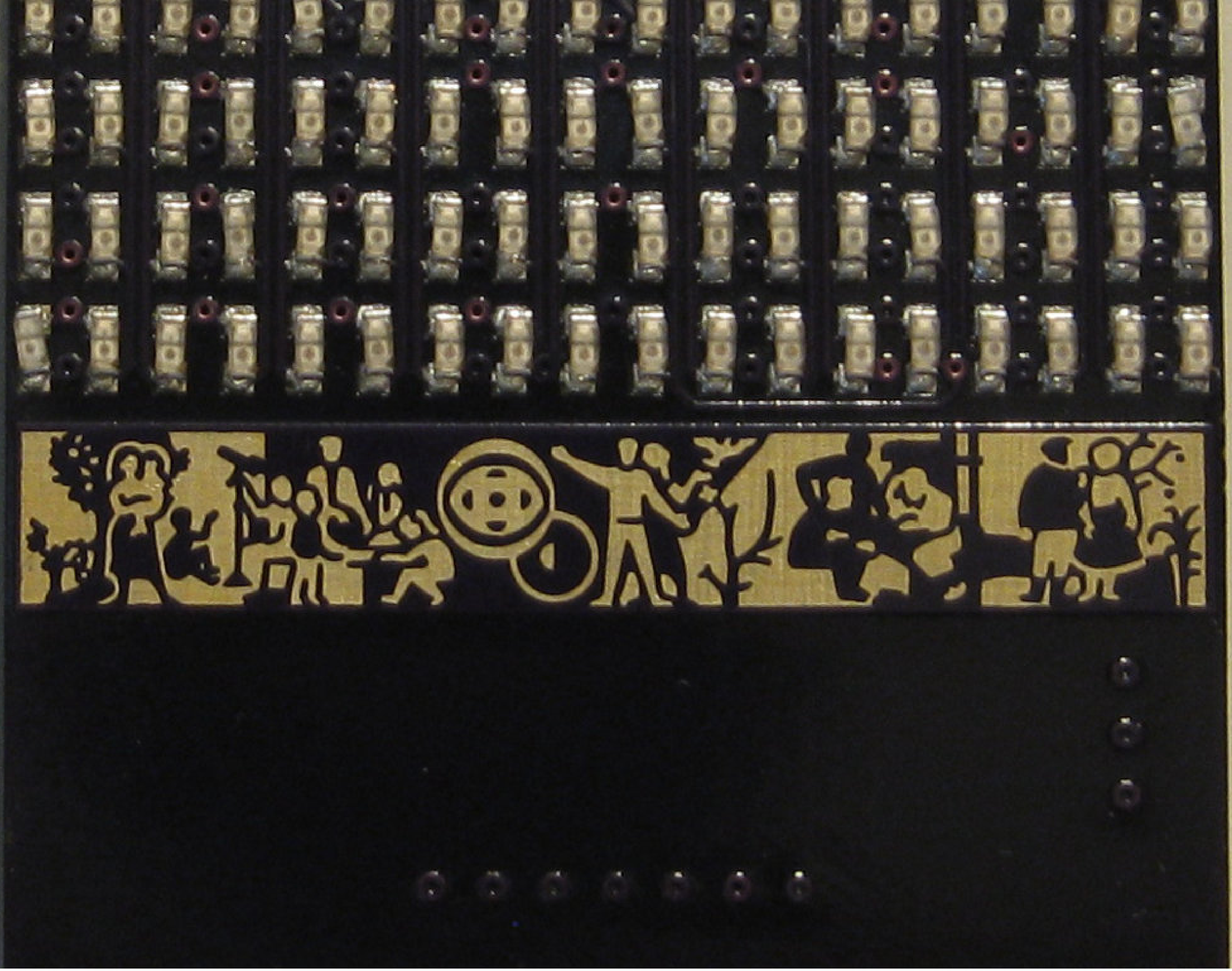
This Rigol power supply from 2012 is lab equipment, so there is a real chance that some curious engineer takes a peek inside. Maybe this is the reason for the nicely done company logo: gold-plated copper on base material without soldermask on top...

(c) 2013 Dave Jones, CC BY-SA 3.0

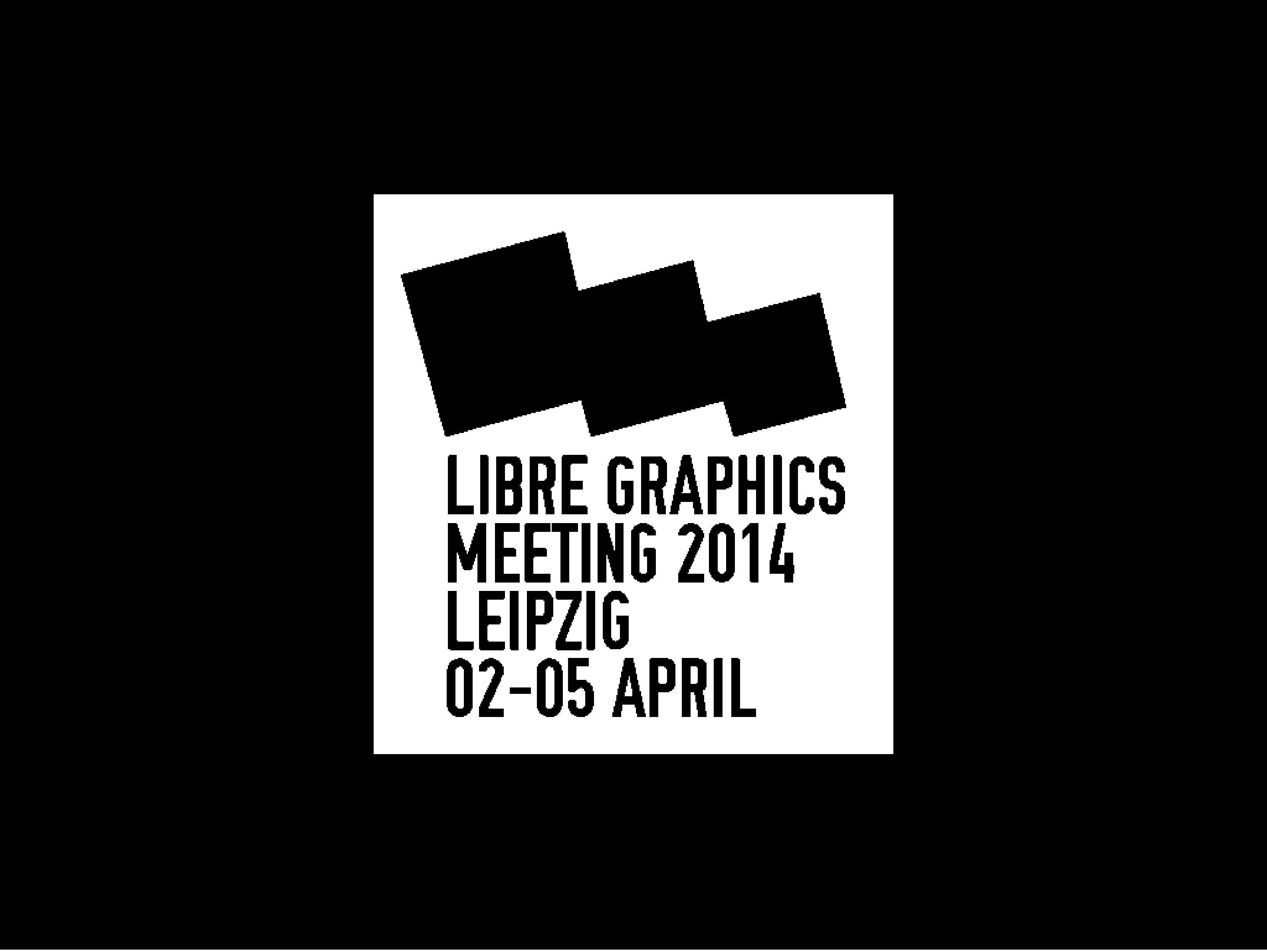


(c) 2010 Gijs Gieskes, used with permission

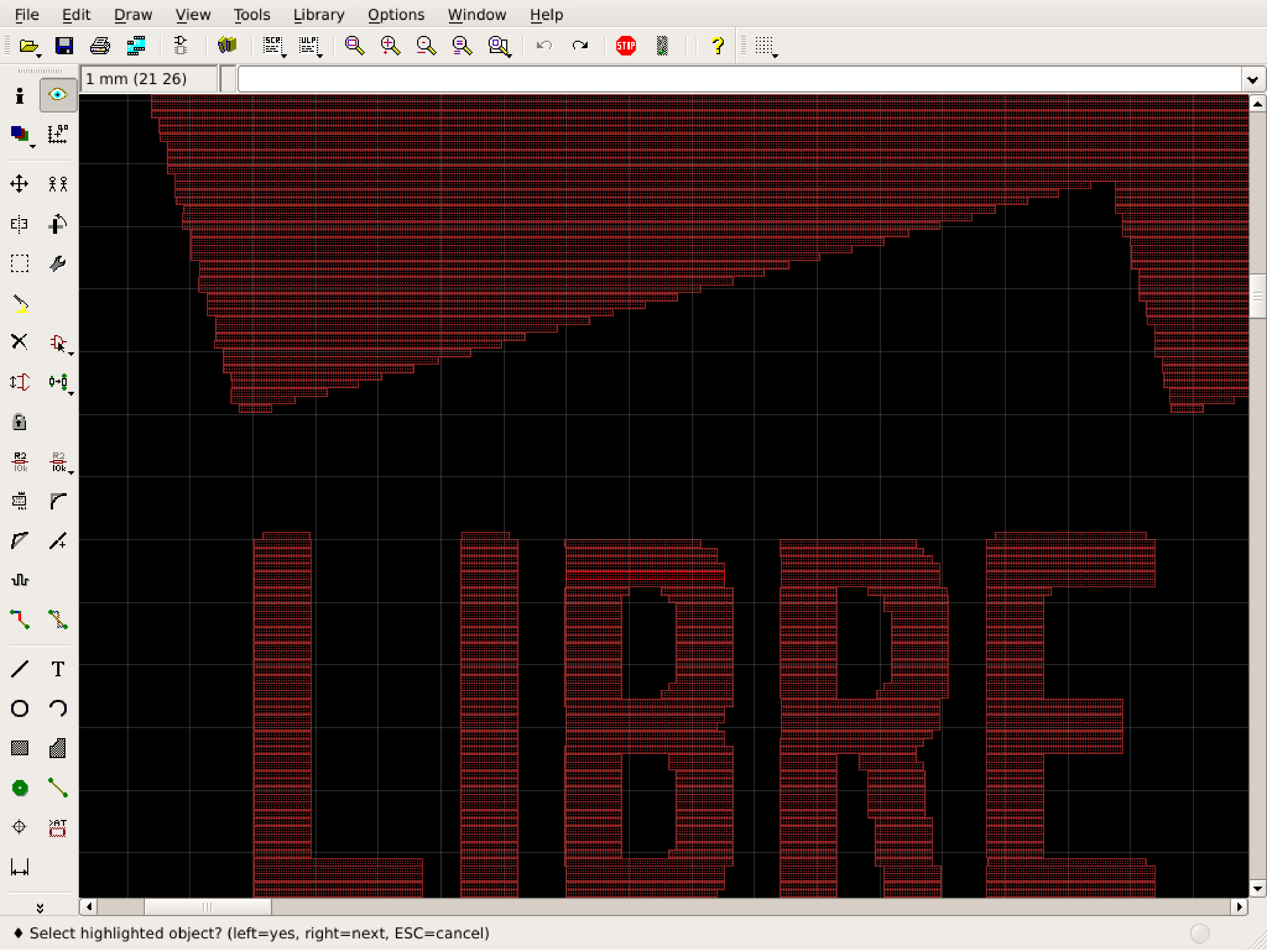
Enter the artists: This is a music synthesizer creating the sounds of the Hypnotoad (Futurama). The manually drawn PCB (unplated copper, no silk screen, no soldermask) replicates the appearance of the toad. We need to make more use of the PCB manufacturing process to create art! (c) 2010 Gijs Gieskes, used with permission



Here is my attempt. It is a small model of the original Blinkenlights installation. The decor is a black/white interpretation of the frieze on the Haus des Lehrers at Alexanderplatz in Berlin. It is dark violet soldermask printed on solid copper (gold plated), the PCB is 32mm wide.

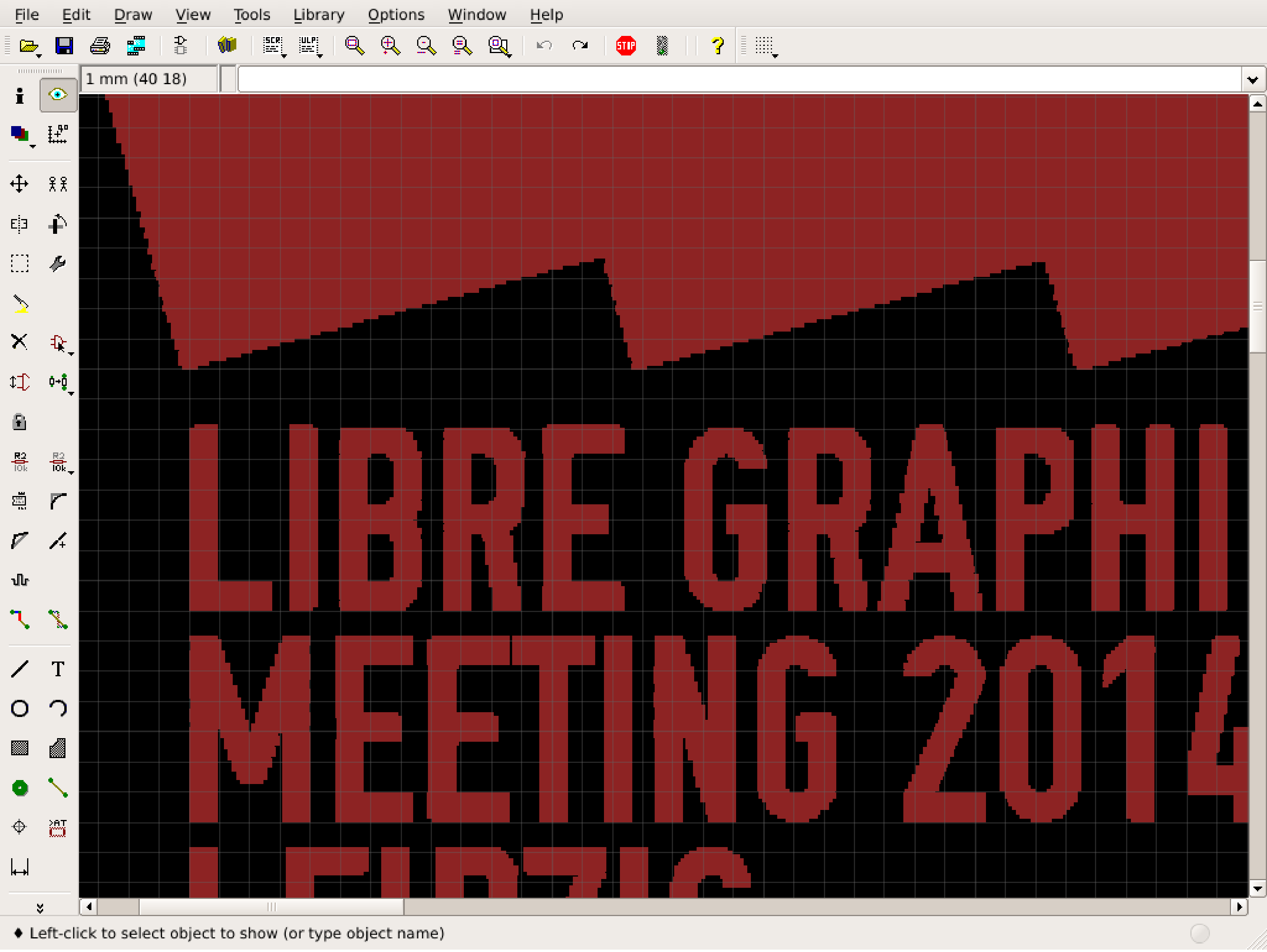


Lets see how we can deal with artwork. This is a black/white BMP, how can we incorporate it into the PCB design?

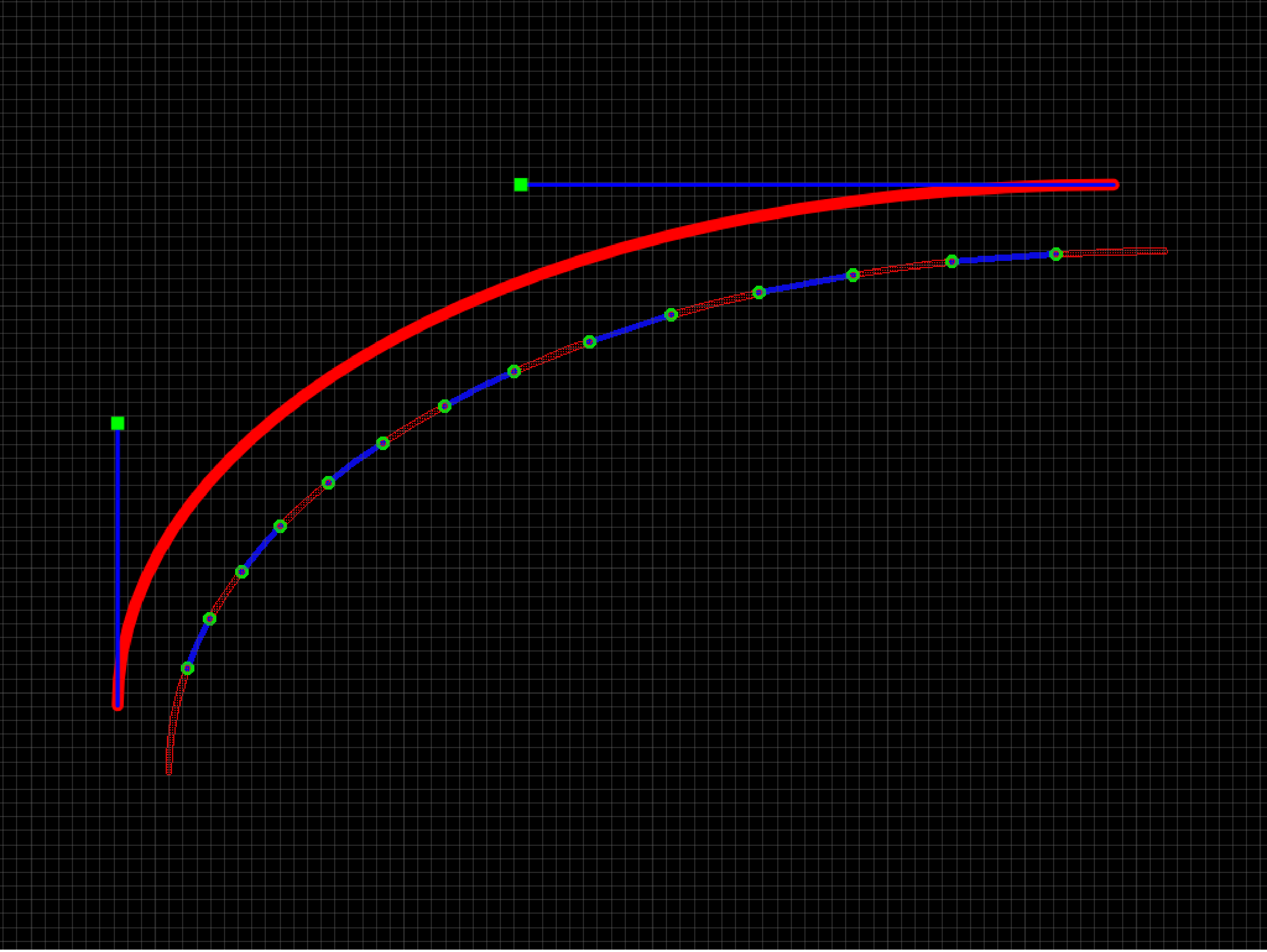


♦ Select highlighted object? (left=yes, right=next, ESC=cancel)

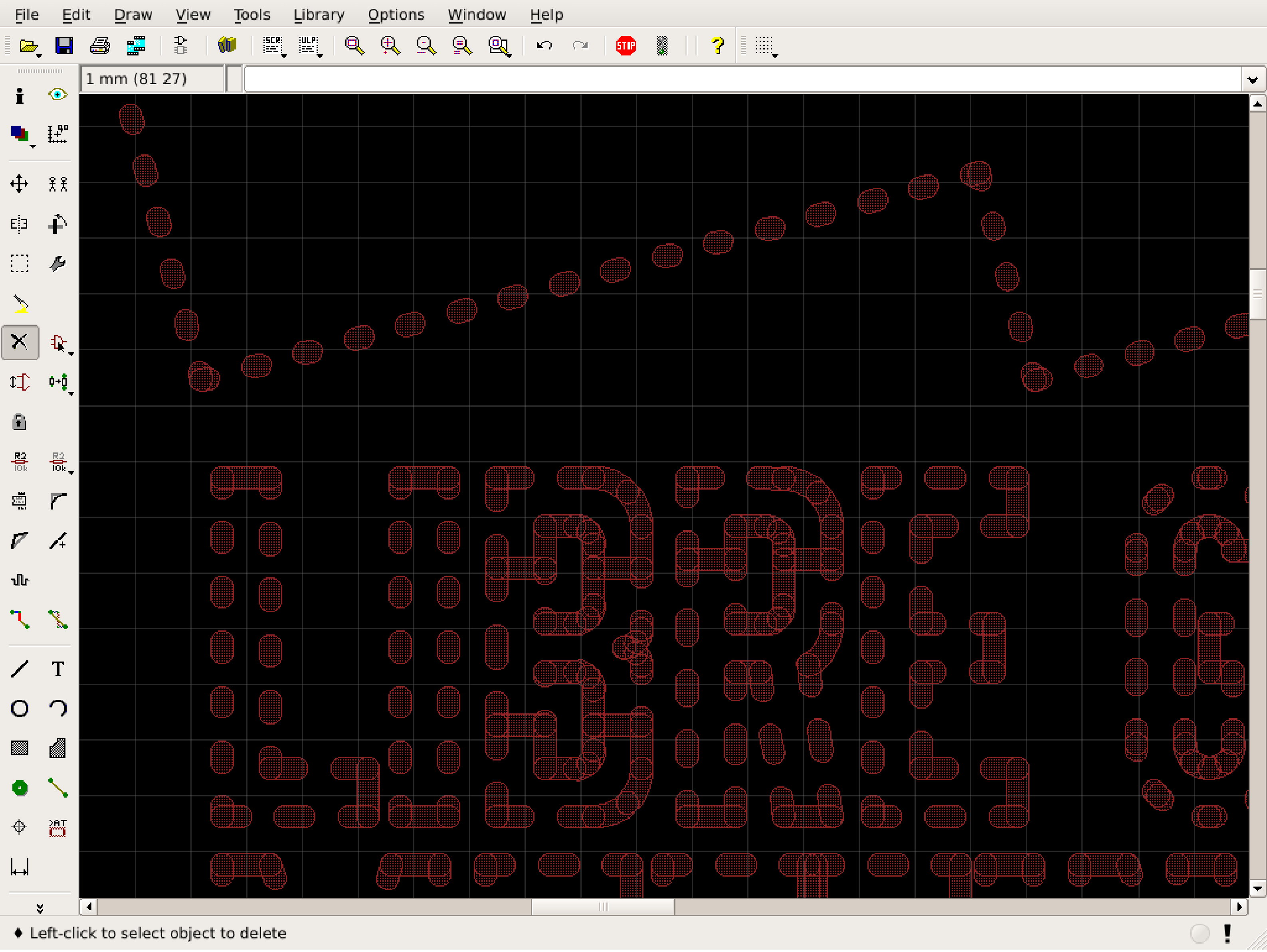
CadSoft Eagle provides a script to import BMPs. It straightforwardly converts pixel lines into PCB traces. It creates huge amounts of data, yet the results are still aliased. If we'd use thinner lines we'd run into problems with the design rule check, since that one demands a minimum trace width (typically 0.15mm), which is applied to each line individually.



This is the result, the aliasing is obvious. Can we do better?

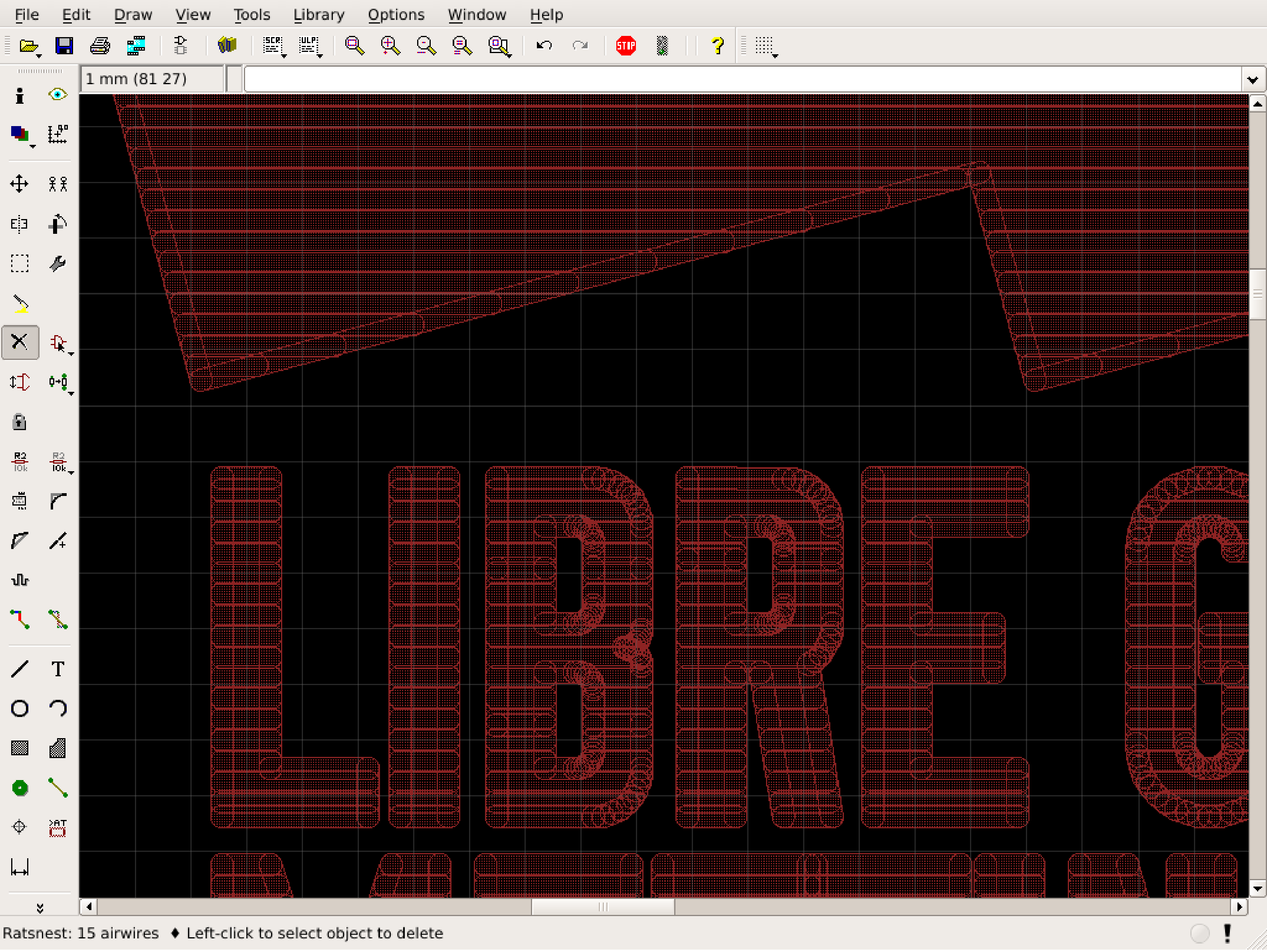


Bezier segments are much more expressive than circular segments. This simple bezier shape needs 15 circular segments for a reasonable approximation. Conversely, a circle typically gets approximated with 4 bezier segments... However, circular segments are easy to handle, with one caveat: scaling non-uniformely destroys their circularity.



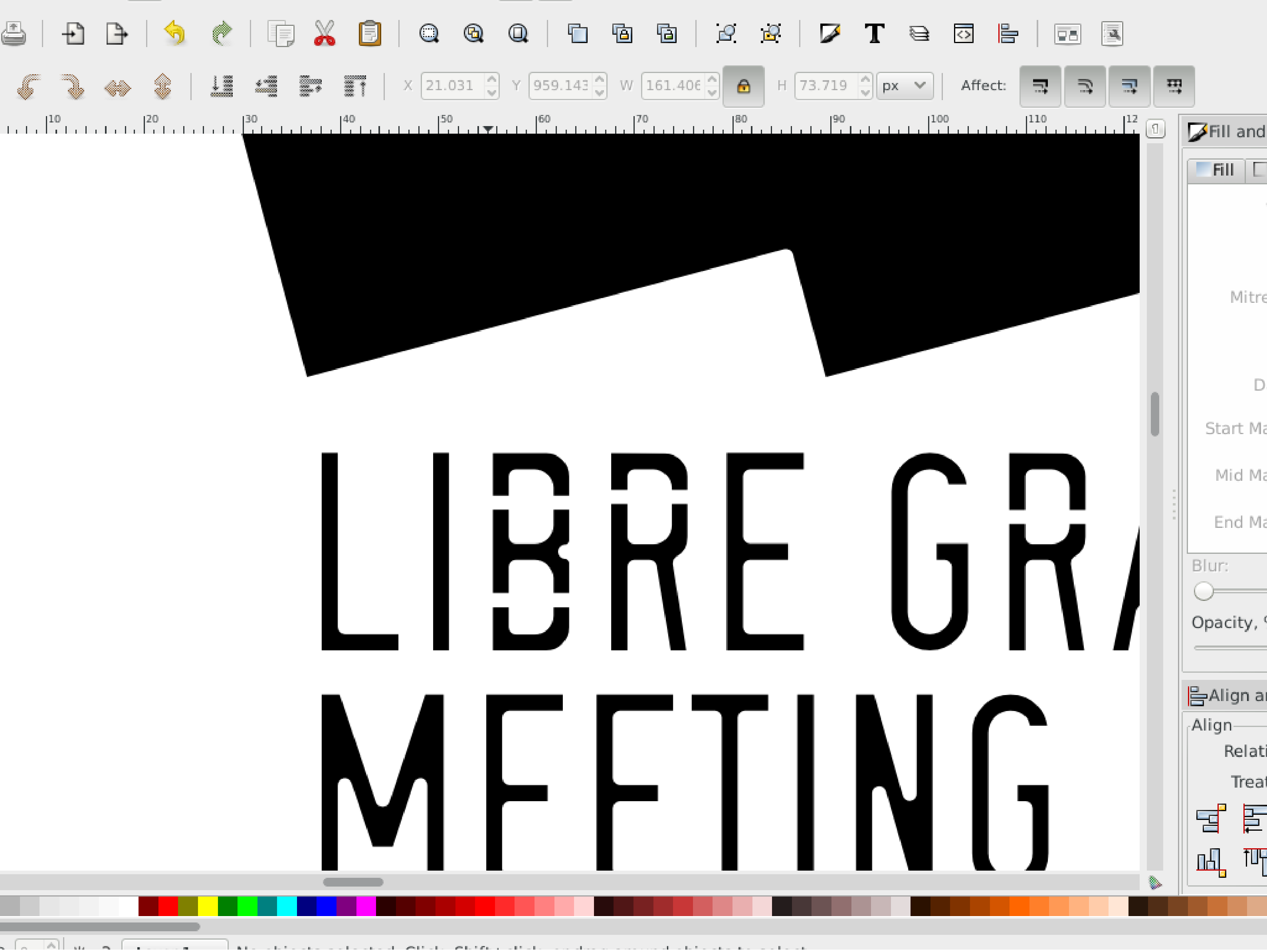
Left-click to select object to delete

This is an attempt at vector artwork being converted to straight lines and circular segments. Holes in polygons need breaking up, since Eagle can't handle holes in polygons. I wrote a python script to convert the beziers into circular segments, creating an Eagle script to construct them. A side effect: one can see that OSP DIN uses circular corners... :)

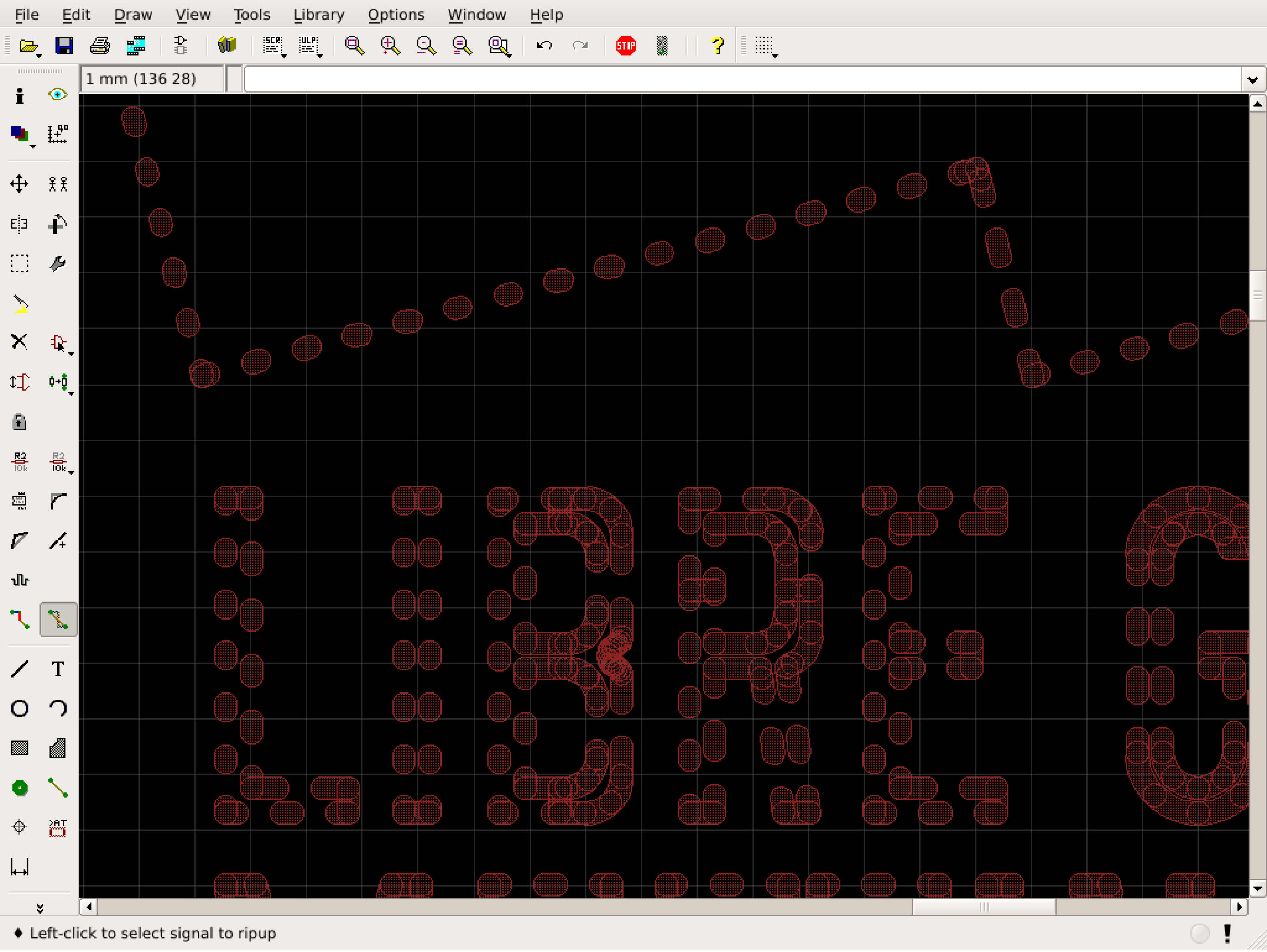


File
Edit
Draw
View
Tools
Library
Options
Window
Help
1 mm (81 27)
Ratsnest: 15 airwires ♦ Left-click to select object to delete

This is the result after Eagle has computed the filling strokes. Note that the outline still is nicely smooth and rounded. The strokes filling the shape are not incorporated into the gerber file, a data explosion doesn't really happen here. However, the result is too fat, since the outline is stroked with a stroke of a significant width.

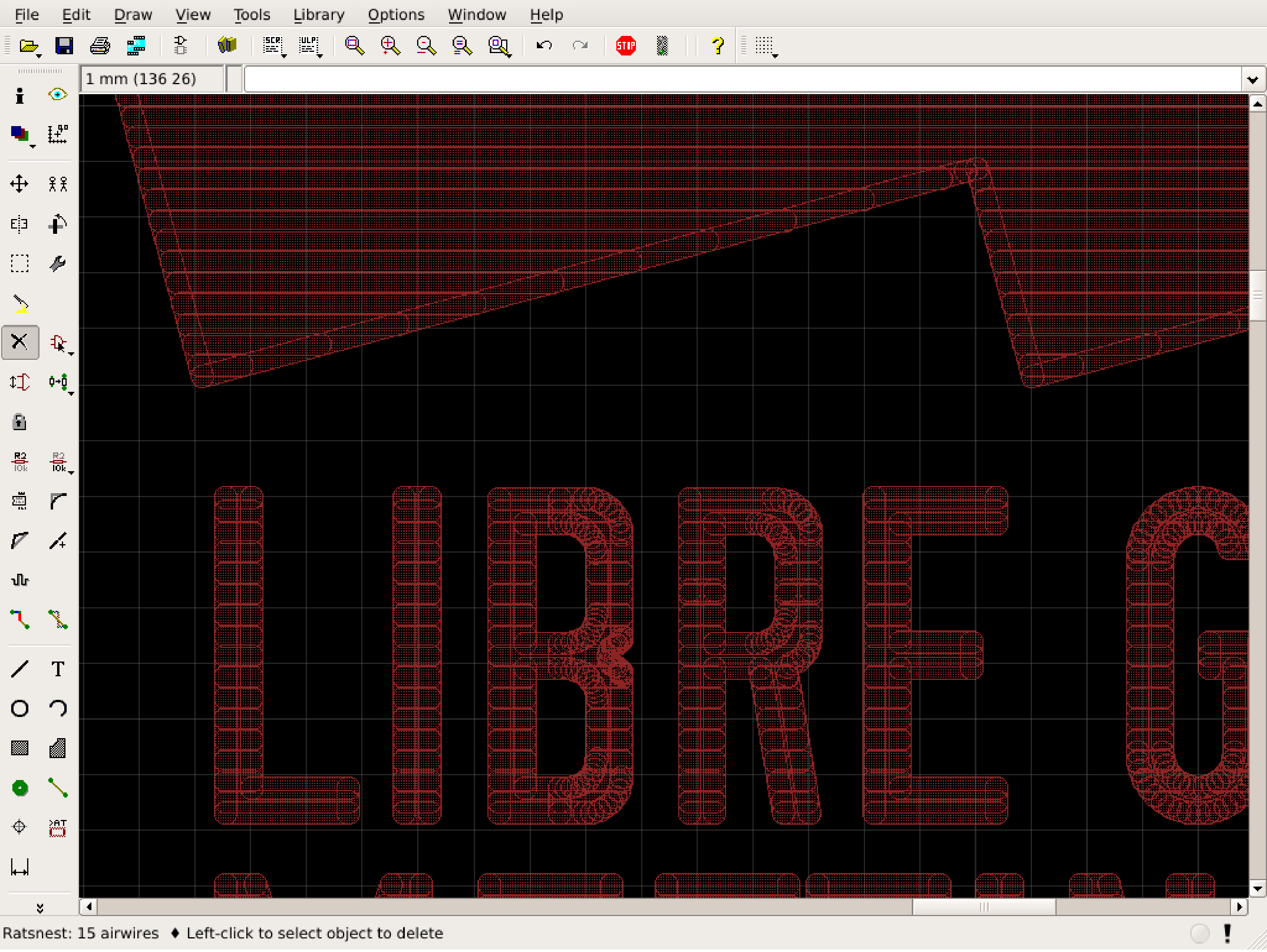


A workaround is, to shrink the path in Inkscape beforehand. Note that the shrinking amount depends on the stroke width later used in Eagle.
In this screenshot I accidentally worked in the wrong order. Usually you'd shrink first and then cut open the holes.



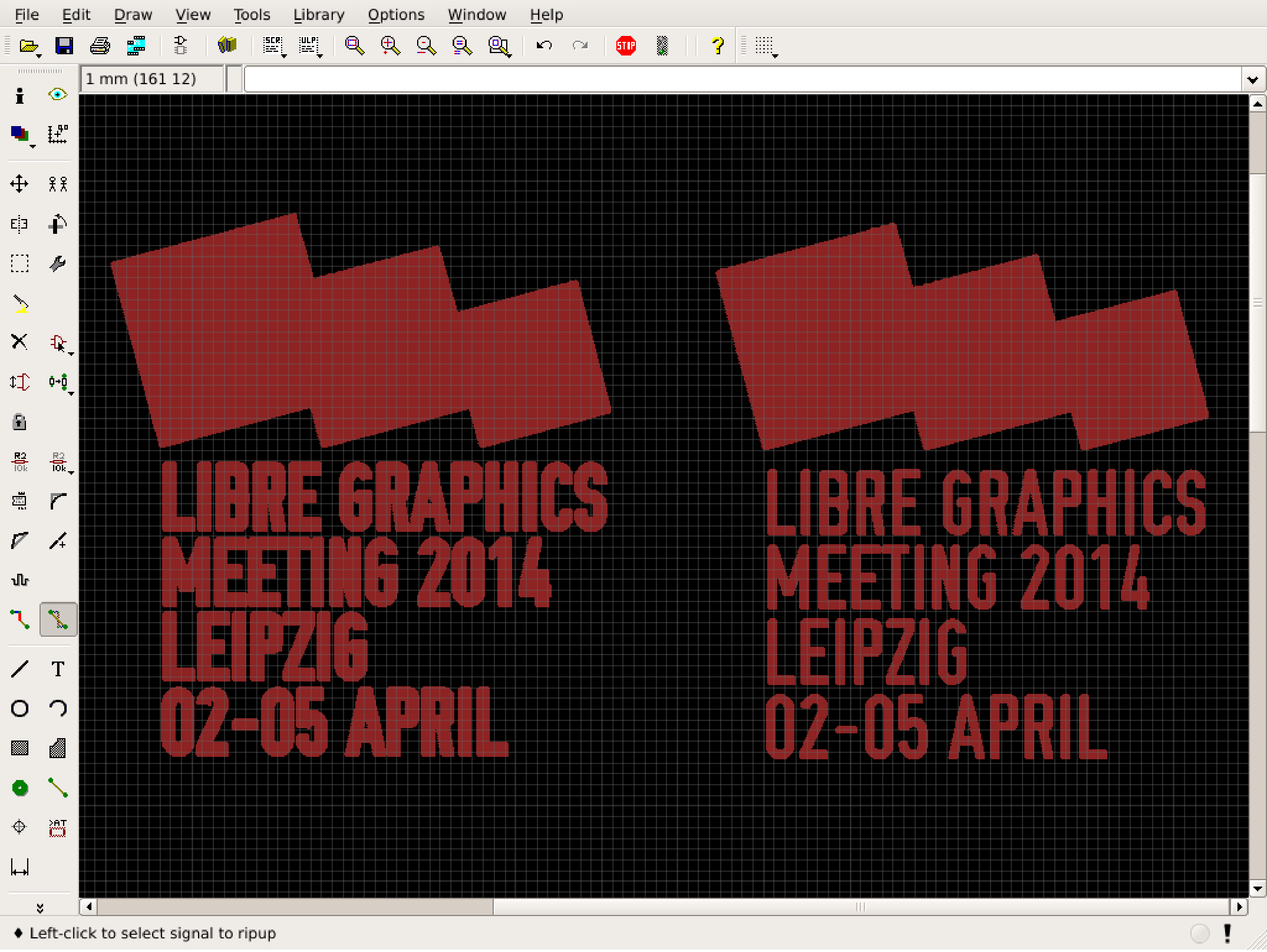
Left-click to select signal to ripup

Now we have something that resembles the logo more faithfully.



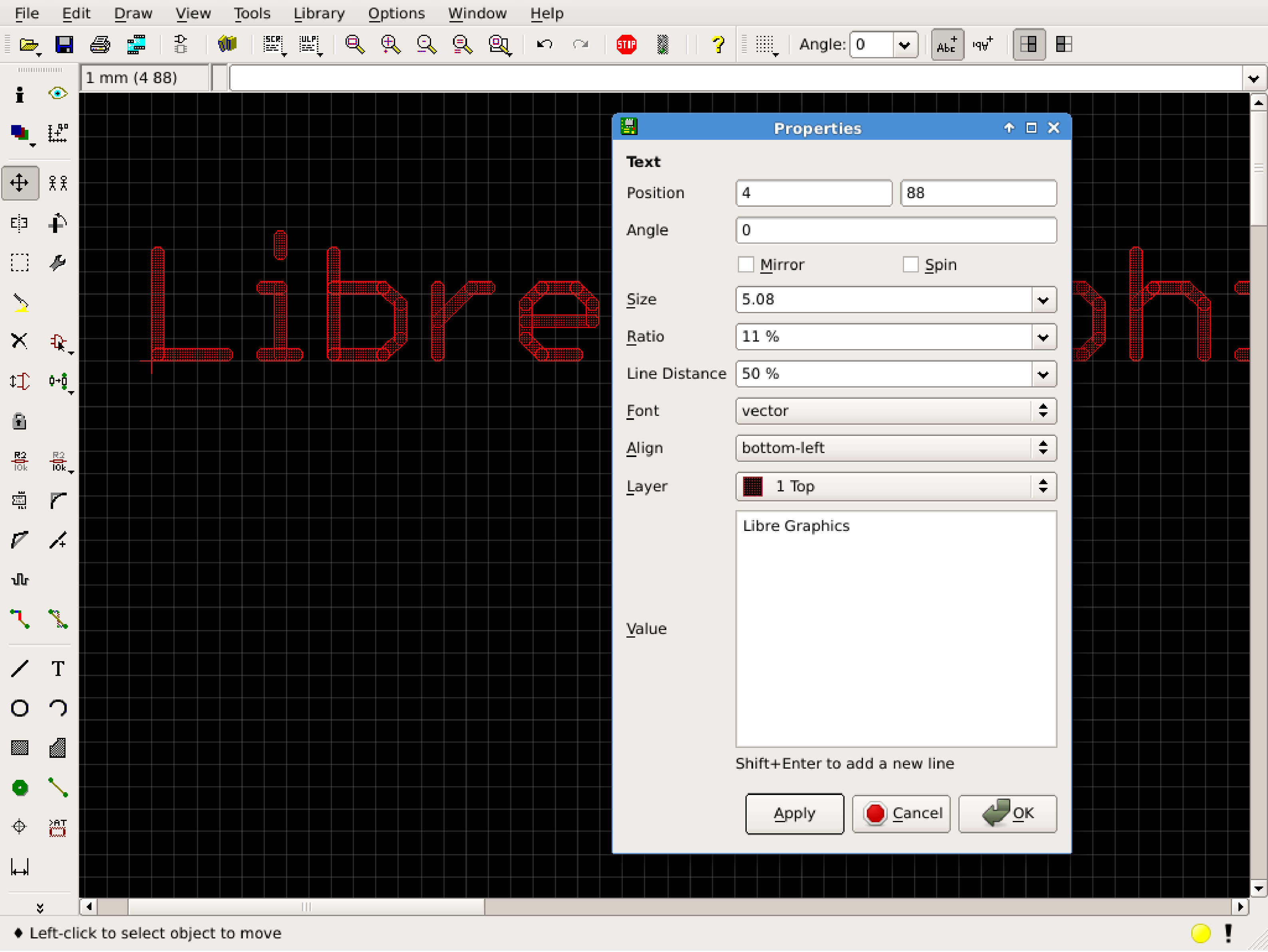
Ratsnest: 15 airwires ♦ Left-click to select object to delete

Lets fill the shape and admire the result :)



Left-click to select signal to ripup

In direct comparison we can immediately see the improvement.

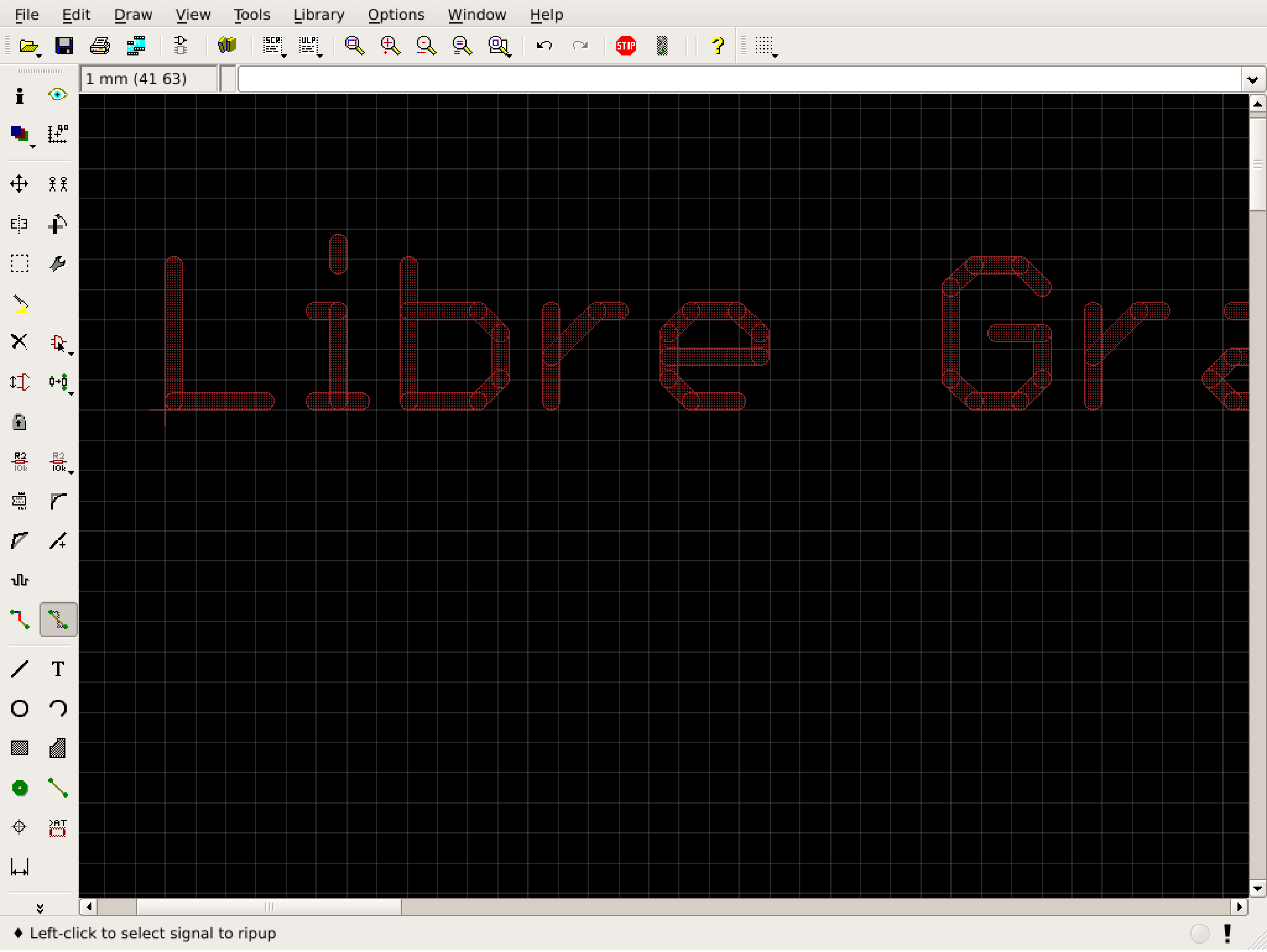


Text

Position	4	88
Angle	0	
	□ Mirror	□ Spin
Size	5.08	
Ratio	11 %	
Line Distance	50 %	
Font	vector	
Align	bottom-left	
Layer	1 Top	
Value	Libre Graphics	

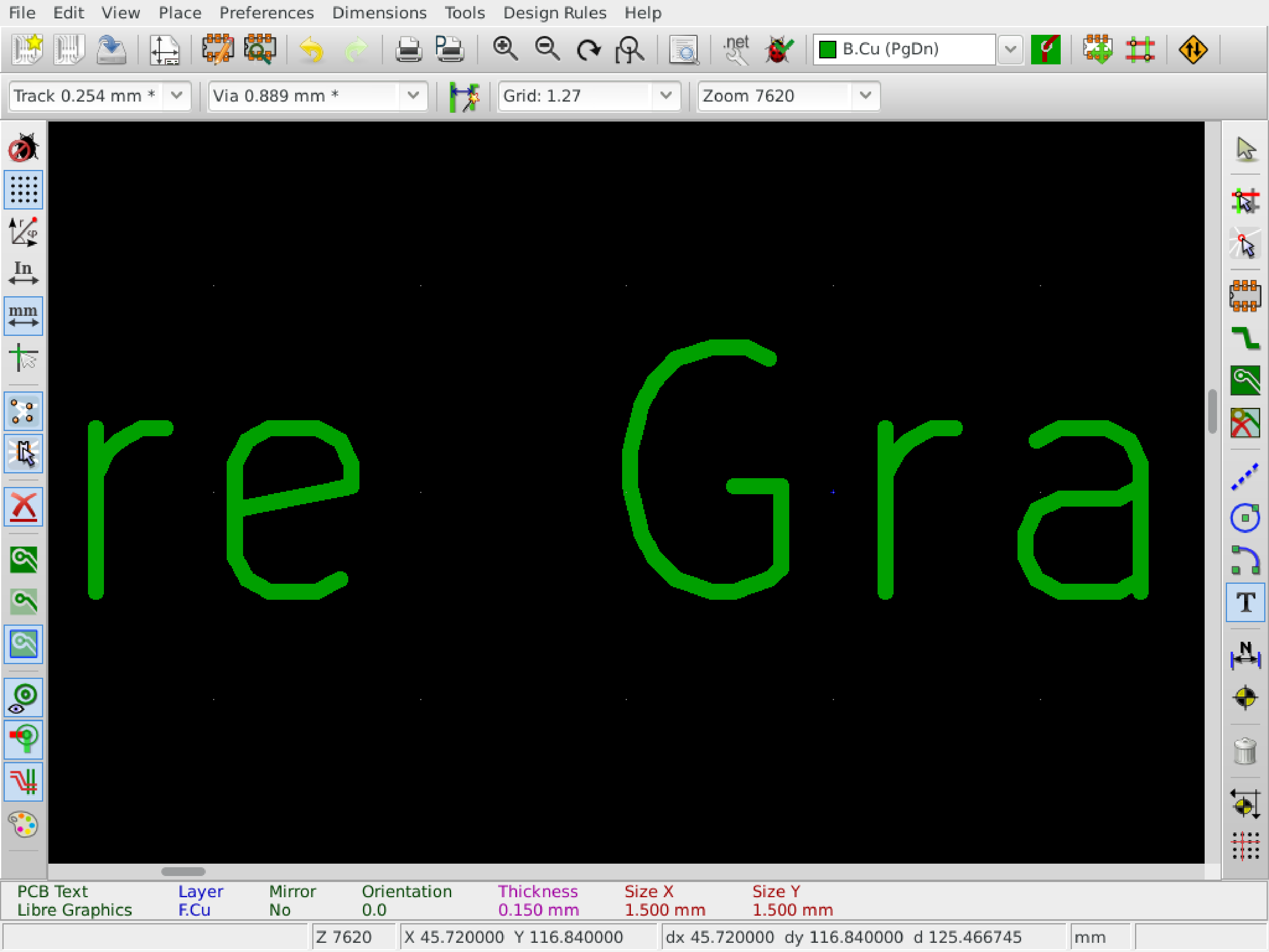
Shift+Enter to add a new line

This is how fonts are handled in Eagle. There is just one font working reliably and it is ugly. The text dialog allows to change size and the size/line-width ratio. The rest is not that interesting...



Left-click to select signal to ripup

Our font needs to be scalable and should adapt to different stroke widths, which is hard when using polygons. We need something stroke based, so that the design rule check later won't complain. Since nobody cares about design within plastic enclosures anyway, lets just throw a font together – as long as it resembles letters everything is fine, right?



B.Cu (PgDn)

Track 0.254 mm *

Via 0.889 mm *

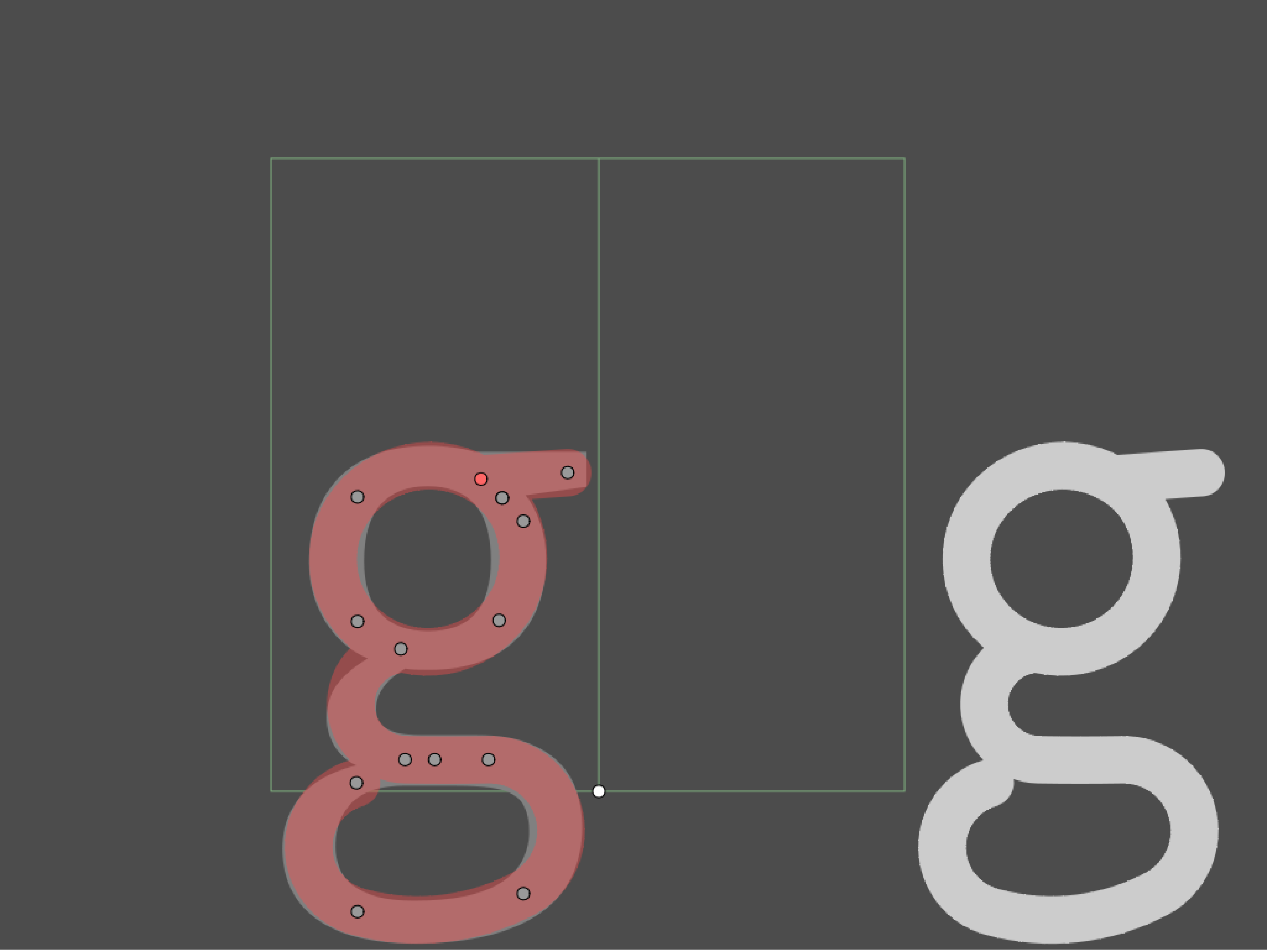
Grid: 1.27

Zoom 7620

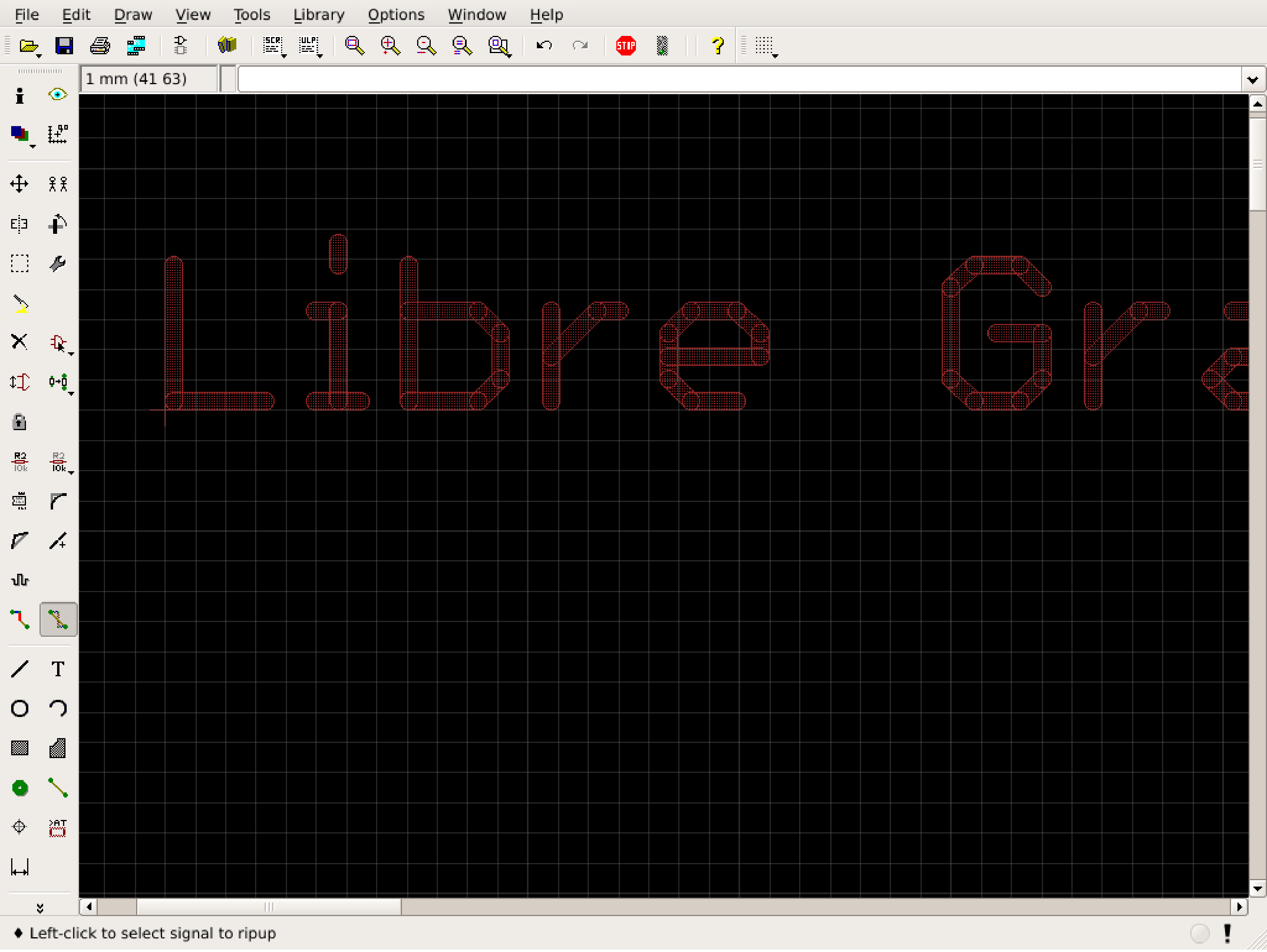
PCB Text	Layer	Mirror	Orientation	Thickness	Size X	Size Y
Libre Graphics	F.Cu	No	0.0	0.150 mm	1.500 mm	1.500 mm

Z 7620 X 45.720000 Y 116.840000 dx 45.720000 dy 116.840000 d 125.466745 mm

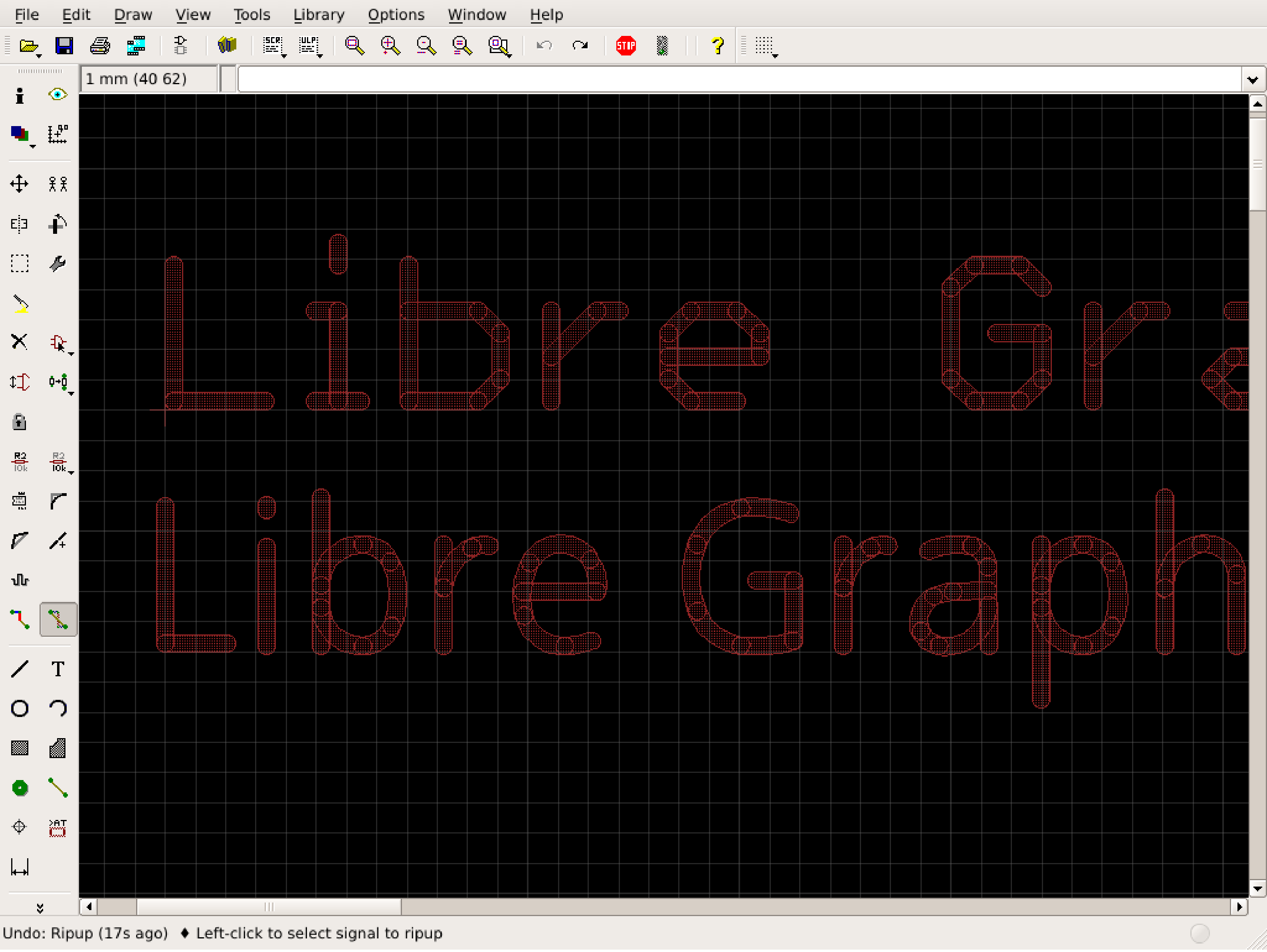
Actually KiCad, the free software alternative to Eagle, disagrees. They have put some effort into their font (which however still is the only one available in the program), using more line segments to create better letter shapes. But I believe that there still is room for improvement.



So I wrote a 500 line python hack, to allow editing shapes with straight lines and circular segments on top of a glyph from an opentype font (Droid Sans). The circular segments are built from start point, start direction and end point, providing C1 continuity. Still very much a hack it can create an Eagle script to generate the shapes looking like letters.



So after manually creating the ASCII set of glyphs and adapting the base metrics from the source fonts we're coming from this...



Undo: Ripup (17s ago) ♦ Left-click to select signal to ripup

... to this. There are some obvious drawbacks: The result consists of multiple objects, Eagle has no clue about the object being text (therefore it is not editable). But the result has a much nicer appearance, even without a greatly increased number of segments. We'll see if at some point this might get incorporated into Eagle and/or KiCad.