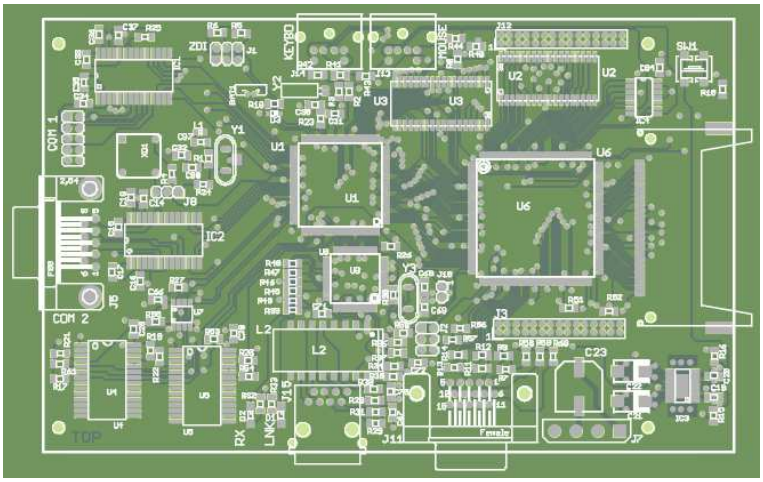
PCB: Command-Line Tools
User's Manual

Rev. 4



Guillaume Rosanis

July 2008

Contents

Chapter 1

Introduction

1.1 Introduction

These tools have been developed, as is often the case, because of the lack of proper tools for prototyping PCBs. I was not satisfied with the usual *Gerber viewers*. Besides, I had developed a 2D graphical rendering library before, and that was, I thought, a good application for it. One particular feature here that most *Gerber viewers* don't offer is the use of anti-aliasing to generate high-quality images. Another is what I see as a robust implementation of the *Gerber RS-274X* format.

With them, you can:

- Generate images (*PNG* format) of one or more PCB layers from *Gerber* files at any resolution (in DPI), with user-defined transparency, colors and/or textures.
- Print one or more PCBs on the same sheet of paper at the exact resolution of the printer, without having to generate intermediate *PNG* images. Each printed PCB can be defined as one or more layers with the same options as mentioned above.

Most common uses include:

- Previewing of *Gerber* files at any resolution.
- Near-realistic rendering of PCBs in 2D, thanks to anti-aliasing, transparency and textures.
- Printing of one or more PCB layers on the same sheet, which allows panelization.

1.2 Development Tools

I have used *GCC 3.4.5* and *GCC 4.1.2* (*MinGW32* flavor on $Windows^{®}$) and *libpng*. My other tools were my brain, a pen, sheets of paper (all the theoretical work and algorithms were written on paper before being implemented) and of course, my computer.

1.3 Requirements

All tools but the printing tool are portable and can be compiled for pretty much any target that support the *standard C library* and *libpng*.

The printing tool currently runs only on $Windows^{®}$. The reason being that I needed to be able to send printers raw images directly from memory, and I don't know how to do that on other platforms. I could have generated *PostScript* instead, but that would have led to very huge *PS* files - something I thought was not very practical. Should the need arise, I could write a front-end for *PostScript* output.

A fairly powerful computer is needed to run the tools comfortably. Processing power and RAM requirements are demanding, because these tools synthesize images than can be up to tens of millions of pixels, depending on the resolution. I'd say 512 MB of RAM are a bare minimum, and at least a P3/800 or better. Actually, these are pretty much the same requirements as a photo/graphical package such as $Photoshop^{®}$ if you work on high-resolution pictures. On $Windows^{®}$, you need the system file *msvcrt.dll*, which should be present on any decently recent version. I suggest using $Windows^{®}NT4$ or better.

As a side note, only *RS-274X*-correct files are supported. RS-274D is not directly supported, but you can always add the aperture definitions manually in the files, and some necessary options, and that should work. More on that later.

1.4 References

- "Gerber RS-274X Format, User's Guide" [Rev D March, 2001], *Barco Graphics N.V.*
- Lots of various Gerber files, either generated by me with several different PCB packages, or found here and there on the Web.
- "Algorithms in C", *Robert Sedgewick*, for some ideas on geometrical algorithms.
- Probably a lot of other reference material I can't think of at the moment.

Chapter 2

GerberRender

2.1 Introduction

`GerberRender` (`GerberRender.exe` on $Windows^{®}$) generates a *PNG* image file from a *Gerber RS-274X* file. It's the most simple tool of the package.

A *Gerber* file is rendered as black on white, or with the given *textures*.

2.2 Usage

```
GerberRender [Options] <Gerber file> [<Output PNG file> [<Background PNG> [<Foreground PNG>]]]
```

Everything in brackets (*[...]*) is optional. Note the simplified syntax: if you want to use a background texture, you need to provide the output file name. If not specified, the default output file name is `Output.png`.

Options:

```
-r dpi      Sets resolution in DPI (default: 200)
-a width    Sets anti-aliasing width in mm (default: 0.1)
-m margin   Sets margin in mm (default: 5)
--mirror    Mirrors image (default: not mirrored)
--strict    Strict RS-274X interpretation  (default: relaxed mode)
```

- Note that a space is needed between the options and their argument.
- Also note that the base unit for distances is mm (millimeter) and not inches!
- The dimensions of the Gerber layer are computed automatically. An outter margin can be defined.
- See Appendix for a definition of the strict interpretation mode.
- And finally, a bit of warning: setting too high a DPI could lead to gigantic image files. Keep that in mind when playing around with options.

2.3 Textures

Background and foreground textures are *PNG* files that are used in a tiling manner. Typically, they are small-sized "tilable" images, but you can use any kind of image you want. For realistic rendering of a PCB copper layer, for instance, you could use a copper texture for the foreground and an epoxy/fiberglass-like texture for the background. This notion of texture files will apply to the rest of this documentation.

2.4 Resolution

The resolution is specified in DPI (Dots Per Inch). A common resolution for modern screens is 96 DPI. For printers, you will most often deal with 300 DPI, 600 DPI and 1200 DPI.

To give you an idea of the target image sizes, a PCB in "Europe" format (160x$100\,mm$) at 600 DPI will give an image of 3780x$2363\,pixels$. That's almost 9 MPixels (Megapixels).

Be aware that the resolution directly affects the minimum size of objects that can be rendered correctly. For instance, a PCB track width of $0.1\,mm$ will be rendered bigger than it actually is if the resolution is less than 300 DPI. For printing PCB layers, I suggest a resolution of at least 600 DPI to get the best result. More on that in the chapter documenting `PCB-Print`.

2.5 Anti-aliasing

I won't explain in great details what anti-aliasing is for images. You can probably find that kind of information in various books and articles.

Basically, it consists in blurring the edges of lines to make up for the jagged effect of lines drawn on a discrete canvas (*ie.* a surface made of discrete points). When we talk of lines, that means straight lines as well as curves and the edges of solid objects.

The 2D library used in this tool implements *perfect* anti-aliasing. That means that lines' edges are blurred in an even way along their whole length. The width of the blurring is what is called *anti-aliasing width* in the options.

Note that due to the rendering algorithm used in this 2D library, the minimum anti-aliasing width should be equivalent to $1\,pixel$. You don't need to bother, though: if you want minimum anti-aliasing, just specify a width of 0. `GerberRender` will compute the minimum width needed and will display it. I suggest using minimum anti-aliasing for printing images at high resolution, and approximately 2-3 times that width (or more) for images destined to be viewed on screen.

This notion of anti-aliasing width will apply to the rest of this documentation.

2.6 Examples

2.6.1 Example 1

```
GerberRender.exe -r 1000 -m 2.2 ON_BOARD_PREAMP_V3.TOP
```

`GerberRender` displays:

```
Gerber file [ON_BOARD_PREAMP_V3.TOP] read successfully.
Layer dimensions: 60.30 x 20.30 mm
Resolution: 1000.00 DPI, AA: 0.1000 mm, Margin: 2.20 mm
Rendering layer...
Image: 2547x973  [2.48 MPixels]
Writing image...
PNG file created successfully.
```

2.6.2 Example 2

```
GerberRender.exe -r 300 layer1.gdo Test7.png green-epoxy.png copper-brushed.png
```

`GerberRender` displays:

```
Gerber file [layer1.gdo] read successfully.
Layer dimensions: 180.19 x 100.20 mm
Resolution: 300.00 DPI, AA: 0.1000 mm, Margin: 5.00 mm
Rendering layer...
Image: 2246x1302  [2.92 MPixels]
Writing image...
PNG file created successfully.
```

Chapter 3

PCB-Render

3.1 Introduction

`PCB-Render` (`PCB-Render.exe` on $Windows^{®}$) generates a *PNG* image file from a *PCB Job* file. This tool allows more sophisticated rendering than the previous one: it can render more than one layer at once and each layer can be rendered with user-defined color or texture and transparency.

For instance, you can render the top copper layer of a PCB, add the solder mask layer on top of it and then the silk screen: you get a realistic view of what the top side of your PCB will look like. You can also render several copper layers on top of one another; choosing the right colors and transparencies allows to have a better view of a PCB routing.

3.2 Usage

```
PCB-Render [Options] <Job file> -o <PNG file>
```

Everything in brackets (*[...]*) is optional. You need to specify the output file name, which is a PNG image file.

Options (override options specified in the Job File):

```
-r dpi      Sets resolution in DPI
-a width    Sets anti-aliasing width in mm
-m margin   Sets margin in mm
--mirror    Mirrors image
--strict    Strict RS-274X interpretation  (default: relaxed mode)
```

- Note that a space is needed between the options and their argument.
- Options given on the command line override the global options set in the Job File.
- Also note that the base unit for distances is mm (millimeter) and not inches!
- The dimensions of the Gerber layers are computed automatically. An outter margin can be defined.
- See Appendix for a definition of the strict interpretation mode.
- And finally, a bit of warning: setting too high a DPI could lead to gigantic image files. Keep that in mind when playing around with options.

3.3 PCB Job Files

3.3.1 Definition

PCB Job Files are text files that describe a rendering job in terms of a background and a list of one or more layers. The convention is to give these files an extension of '`.pcbjob`'.

The layers appear in the order that they will be "piled up", thus in a bottom-to-top fashion. The top-most layer is always the last one listed in the *PCB Job* file. To be able to "see through" some layer, you will need to make it transparent by using the transparency attribute (alpha coefficient).

3.3.2 Syntax

General syntax

```
<PCB Job File> := List of <Sections>
<Section> := <Section Name> { List of <Parameters> }
<Parameter> := <Parameter Name> = <Parameter Value>;
```

The character '#' can be used to write comments. Everything following this character up to the end of the current line is ignored.

Sections

Section	Parameter	Description	Default Value
GLOBAL	DPI	Image resolution in DPI	300
	AA	Anti-aliasing width in mm	0.1
	MARGIN	Margin in mm	5
	MIRROR	Mirror image (TRUE, FALSE)	FALSE
BACKGROUND	COLOR	Background solid color	None
	IMG	Background texture file (PNG)	None
LAYER	COLOR	Layer solid color	Black: 0,0,0
	IMG	Layer texture file (PNG)	None
	ALPHA	Alpha coefficient for the layer (transparency)	1.0
	MIRROR	Mirror layer (TRUE, FALSE)	FALSE
	INVERT	Invert layer (TRUE, FALSE)	FALSE
	GERBER	Gerber RS-274X file	None

Colors

Solid colors (parameter 'COLOR') are expressed as 3 comma separated numbers:

```
<Red>,<Green>,<Blue>
```

which are the R,G,B component values, each between 0 and 1.

Alpha coefficients

Alpha coefficients set the level of transparency of layers. They are numbers between 0 and 1, 1 being fully opaque, and 0 being fully transparent.

File names

All file names can be either relative or absolute. If relative, they are always relative to the directory where the *PCB Job* file resides.
For instance:

```
GERBER = ..\SomeGerberfile
```

refers to a file named 'SomeGerberfile' that is located in the *PCB Job* file's parent directory.
This rule applies to all files, be it Gerber files or texture files.

Layer inversion

Inversion (parameter 'INVERT') allows to render the negative of a layer. This is useful, for instance, to render solder mask layers of PCBs as a layer of "coating", because solder masks actually represent the areas that are "masked" when the PCB is being coated.
You would specify an Alpha coefficient of less than 1 to make the layer transparent, just like regular PCB coating would be.

3.4 Examples: PCB Job files

3.4.1 Example 1

```
GLOBAL
{
    DPI = 600;
    MARGIN = 2.0;
    AA = 0.1;
}

BACKGROUND
{
    COLOR = 0.81,0.90,0.60;
}

LAYER
{
    GERBER = ..\top_copper.grb;
    COLOR = 0.8,0.8,0.4;
}

LAYER
{
    GERBER = ..\top_mask.grb;
    COLOR = 0,0.2,0.1;
    INVERT = TRUE;
    ALPHA = 0.45;
}

LAYER
{
    GERBER = ..\top_silk.grb;
    COLOR = 1,1,1;
}
```

Renders the top side of a PCB with greenish coating and white silk screen. In this example, the Gerber files are supposed to be in the parent directory of the directory containing the above PCB Job file.

3.4.2 Example 2

```
GLOBAL
{
    DPI = 1200;
    MARGIN = 2.0;
    AA = 0;
}

BACKGROUND
{
    COLOR = 1,1,1;
}

LAYER
{
    GERBER = top_copper.grb;
    COLOR = 0,0,0;
}
```

Renders the top copper layer of a PCB in black on a white background, in 1200 DPI. In this example, the Gerber file is supposed to be in the same directory as the one containing the above PCB Job file.

3.4.3 Example 3

```
GLOBAL
{
    DPI = 600;
    MARGIN = 2.0;
    AA = 0.1;
}

BACKGROUND
{
    COLOR = 0.81,0.90,0.60;
}

LAYER
{
    GERBER = ..\top_copper.grb;
    IMG = Metal_Texture.png;
}

LAYER
{
    GERBER = ..\top_mask.grb;
    COLOR = 0,0.2,0.1;
    INVERT = TRUE;
    ALPHA = 0.45;
}

LAYER
{
    GERBER = ..\top_silk.grb;
    COLOR = 1,1,1;
}
```

Renders the top side of a PCB with greenish coating and white silk screen. The copper layer is rendered using a metal-like texture. In this example, the Gerber files are supposed to be in the parent directory of the directory containing the above PCB Job file, and the metal texture PNG file in the same directory as the one containing the PCB Job file.

3.5 Example: Command line

```
PCB-Render.exe Top_PCB_1200dpi.pcbjob -o Top_PCB_1200dpi.png
```

PCB-Render displays:

```
PCB dimensions: 59.82 x 29.34 mm
Resolution: 1200.00 DPI, AA: 0.0212 mm, Margin: 2.00 mm
Rendering: 100 %   [3 of 3]
Writing PNG file...
```

Chapter 4

PCB-Print

4.1 Introduction

`PCB-Print.exe` allows to print PCB layers from a *Print Job* file. This tool currently runs only on $Windows^{®}$.

It prints at the exact resolution of the selected printer with the highest possible quality. Since it allows to print several "PCBs" on the same sheet, "panelization" is possible. There is no "automatic panelization" function in this version, but this is planned for a future version.

4.2 Usage

```
PCB-Print [Options] <Print Job file>
```

Everything in brackets (*[...]*) is optional.

Options:

```
--verbose | -v    Verbose mode: display additional info
--pretend | -p    Pretend: don't actually print
--strict          Strict RS-274X interpretation  (default: relaxed mode)
```

- The "pretend" mode does everything that the normal mode does, except that nothing is actually printed. Useful for testing purposes.
- See Appendix for a definition of the strict interpretation mode.

4.3 Print Job Files

4.3.1 Definition

Print Job Files are text files that describe a printing job in terms of a list of one or more "PCBs". The convention is to give these files an extension of '`.printjob`'.

Here, "PCBs" are in fact *PCB Jobs*, each associated with a list of *placements*. A placement is the coordinates of the top left corner of a PCB image within the printing area. A PCB image is always considered rectangular and is the result of the rendering of a *PCB Job*, as defined in the previous chapter.

4.3.2 Syntax

General syntax

```
<Print Job File> := List of <Sections>
<Section> := <Section Name> { List of <Parameters> }
<Parameter> := <Parameter Name> = <Parameter Value>;
```

The character '#' can be used to write comments. Everything following this character up to the end of the current line is ignored. The general syntax is identical to that of PCB Job files.

Sections

Section	Parameter	Description	Default Value
GLOBAL	NAME	Print Job Name	None
PCB	NAME	PCB Name	None
	MARGIN	Margin in mm (overrides the margin defined in the PCB Job file)	None
	PCBJOB	PCB Job file	None
	PLACE	List of placements	None

Placements

```
<List of placements> := { <Placement> [<Placement> ... ] }
<Placement> := (<x>,<y> [,R])
```

As mentioned before, a placement is the coordinates of the top left corner of a PCB image on the printing area. The printing area is determined at run-time, when the user selects a printer and a paper format. The coordinates are expressed in mm (millimeters) in a top-down manner with the origin $(0,0)$ at the top left corner of the printing area.

In a placement statement, the optional 'R' parameter indicates that the PCB image will be rotated clockwise by 90 degrees. Rotating PCB images allows in some cases to fit more images on the same sheet.

All placements of PCB images are then checked to see if there is no overlapping and if they are all contained within the printing area. If not, the printing will not occur and a message will be displayed.

File names

All file names can be either relative or absolute. If relative, they are always relative to the directory where the *Print Job* file resides.

For instance:

```
PCBJOB = ..\SomePCBJobFile.pcbjob
```

refers to a file named 'SomePCBJobFile.pcbjob' that is located in the *Print Job* file's parent directory.

4.4 Examples: Print Job files

4.4.1 Example 1

```
GLOBAL
{
    NAME = B-1 On-board Electronics;
}

PCB
{
    NAME = Audio I/O;
    MARGIN = 3;
    PCBJOB = ..\FreePCB\B-1_AudioIO\CAM\Images\Top_PCB_1200dpi.pcbjob;
    PLACE = { (0,0) (80,0) (100,200) };
}

PCB
{
    NAME = Display CPU;
    MARGIN = 3;
    PCBJOB = ..\FreePCB\B-1_DisplayCPU\CAM\Images\Top_PCB_1200dpi.pcbjob;
    PLACE = { (0,40) (97,40) (0,200) };
}

PCB
{
    NAME = Main CPU;
    MARGIN = 3;
```

```
    PCBJOB = ..\FreePCB\B-1_MainCPU\CAM\Images\Top_PCB_1200dpi.pcbjob;
    PLACE = { (0,100) (90,100) };
}
```

Prints the top copper layer of 3 different PCBs defined by their respective PCB Job file. In this example, there are 8 PCBs printed in total.

4.5 Example: Command line

```
PCB-Print.exe -v -p B-1_OBE_1.printjob
```

PCB-Print displays:

```
Print Job: B-1_OBE_1.printjob
PCB dimensions: 59.82 x 29.34 mm
Resolution: 600.52 DPI, AA: 0.0423 mm, Margin: 3.00 mm
Rendering: 100 %   [3 of 3]
Printing: <B-1 On-board Electronics>    33 %   [1 of 3]
PCB dimensions: 88.39 x 37.59 mm
Resolution: 600.52 DPI, AA: 0.0423 mm, Margin: 3.00 mm
Rendering: 100 %   [3 of 3]
Printing: <B-1 On-board Electronics>    66 %   [2 of 3]
PCB dimensions: 73.79 x 79.50 mm
Resolution: 600.52 DPI, AA: 0.0423 mm, Margin: 3.00 mm
Rendering: 100 %   [3 of 3]
Printing: <B-1 On-board Electronics>    100 %   [3 of 3]
List of rectangles:  (Total = 8)
->    (0.00,0.00) - (65.82,35.34)
->    (80.00,0.00) - (145.82,35.34)
->    (100.00,200.00) - (165.82,235.34)
->    (0.00,40.00) - (94.39,83.59)
->    (97.00,40.00) - (191.39,83.59)
->    (0.00,200.00) - (94.39,243.59)
->    (0.00,100.00) - (79.79,185.50)
->    (90.00,100.00) - (169.79,185.50)
Surface coverage: 59.8 %
```

Note that in 'verbose' mode, PCB-Print displays the list of the placed rectangles and the surface coverage, which is computed as the ratio of the sum of the areas of the placed rectangles to the total printing area.

Appendix A

Gerber

A.1 Gerber RS-274X support

The RS-274X standard is fully implemented, with the following exceptions.

A.1.1 Not implemented: will give an error

X Code	Description
KO	Knockout
SR	Step and Repeat

A.1.2 Not implemented: will be ignored

X Code	Description
OF	Offset
IJ	Image Justify
IC	*Not known, but found in some files*
IO	Image Offset
IR	Image Rotation
PF	Plotter Film
RO	Rotate

A.2 Default parameters

If no "Format Statement" ('FS' X Code) is specified, the default format used is:

- Unit: Inches
- Omit leading zeros
- 2.3

I strongly suggest always to put a Format Statement in Gerber files.

Other default parameters are:

- Image Polarity: positive
- Scale on axis A and B: 1

A.3 Strict RS-274X conformance mode

The strict RS-274X conformance mode can be used for interpreting Gerber files with the general option '`--strict`' in the command-line tools. In this strict interpretation mode:

- Local variables in aperture macros are not allowed
- Internal layer changes reinitialize current tool selection
- `G04` commands require a '*' at the end of the line, as any other command

A.4 Supporting Gerber RS-274D

To support RS-274D files, you will need to edit the files and add at the top of the file, at least:

- A Format Statement.
- All apertures definitions.

according to the RS-274X specification.