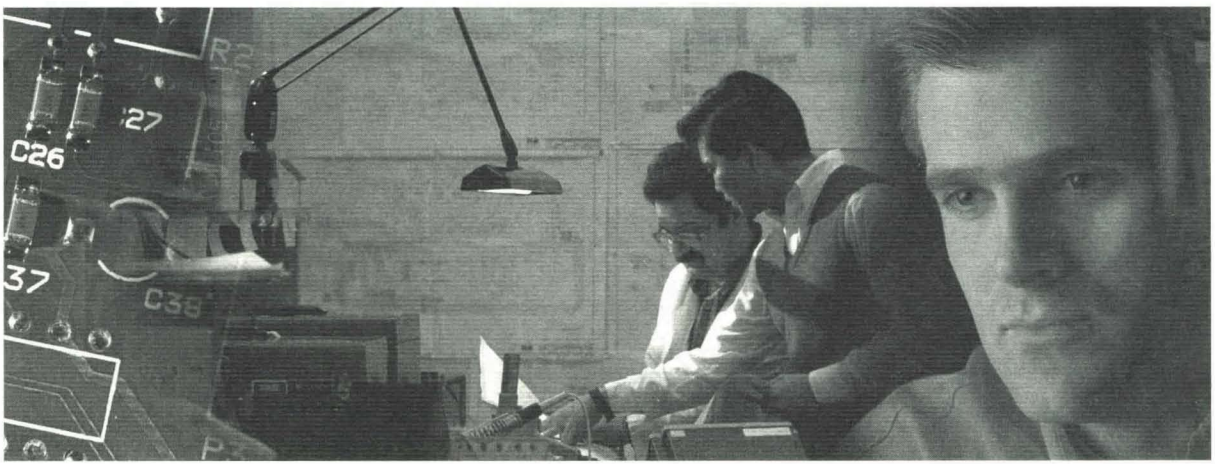
OrCAD®

LAYOUT™

FOR WINDOWS®



GERBTOOL USER'S GUIDE

OrCAD Layout™ for Windows®

GerbTool™ User's Guide

MN-01-5048

First Edition 30 June 96

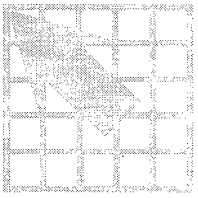
Technical support	(503) 671-9400
Bulletin board system	(503) 671-9401
Administration	(503) 671-9500
Fax	(503) 671-9501
General email	info@orcad.com
Technical support email	techsupport@orcad.com
Web site	http://www.orcad.com

OrCAD®

9300 S.W. Nimbus Ave.
Beaverton, OR 97008 • USA

Contents

Chapter 1



Introduction

Welcome to GerbTool, the easiest, most powerful, and versatile CAM station available.

GerbTool provides a powerful set of Windows-based CAM tools, including a feature rich and robust Gerber/NC editor for ensuring a seamless link between PCB design and manufacturing. GerbTool is designed to provide CAD/CAM professionals with the tools they need for complete control over their CAM databases. From visual verification to high-level CAM tools, GerbTool simplifies and automates your PCB CAD post processing and pre-manufacturing tasks.

GerbTool's consistent and intuitive Graphical User Interface (GUI), and programmable mouse buttons and function keys, allow you to focus on accomplishing tasks, rather than on the technical details of operating the software.

Using this manual

This manual was designed to assist the CAD/CAM professional in utilizing GerbTool's features. *Chapter 3: Quick start* is especially geared toward providing the information you need to become immediately productive. A prior knowledge of CAD/CAM concepts and your computer's operating system is assumed.

Compatibility with OrCAD Layout for Windows

GerbTool is designed to work with both OrCAD Layout Plus for Windows and OrCAD Layout for Windows. GerbTool Ltd. is designed to work with OrCAD Layout Ltd. for Windows.

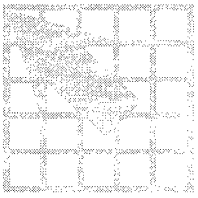
GerbTool Ltd. has all of the features of GerbTool, with the following exceptions:

- Gerber format conversion is not supported
- Gerber files may not be modified
- Macros are not supported
- DRC is not supported

Product features

- Fast and easy to use, causing less user fatigue.
- Unlimited file sizes.
- Accurate to 1/100 mil (.00001 in.).
- Fully automatic panelization and venting.
- Complete undo to beginning of session.
- Full Design Rule Checking (DRC), including Annular Ring Checking and Stub Detection.
- Snoman™ pad/trace filleting.
- NC Drill optimizing, including Step and Repeat.
- Isolated pad removal.
- Automatic removal of silkscreen data from pads.
- Full support for true multilayer netlists, including net highlighting.
- Scalable check plots to HPGL, PostScript®, Laser printers, and all printers/plotters supported by Windows.
- Conversion of drawn pads to flashes.
- Macro language allows the addition of new commands.
- Metric and Imperial formats supported.
- Photo plotter support includes 274-X, FIRE9xxx, EIE, BARCO DPF and IPC-D-350.
- Accurate display of power and ground plane composites.
- Allows aperture scaling to create soldermasks, shrink/expand traces, and so on.
- Ability to scale layers to shrink or expand the database.
- Merge a complete design or a single Gerber file into another.
- Import NC Drill, HPGL, or BARCO files.
- View up to 999 layers simultaneously.
- Handles over 4000 apertures in up to 999 aperture lists.
- Aperture list conversion tools allow the addition of custom aperture list converters.
- Easily created custom apertures and custom fonts.

Chapter 2



Configuration

This chapter describes the configuration of GerbTool. The installation process creates a master configuration file that GerbTool reads every time it starts. This configuration file will most likely be sufficient for your needs. If you find that it is not, or you have special configuration requirements, see the next section.

Configuring GerbTool

GerbTool uses a configuration file and a color list file to control its operating environment. Many of GerbTool's startup defaults such as grid size, film box size, and so on are controlled through the configuration file. Mouse button actions and function key assignments are also controlled through the configuration file.

GerbTool uses the registration database to locate its configuration files.

Configuration file

When starting up, GerbTool looks for a local configuration file named GT.CFG in the GerbTool directory.

A configuration file contains statements called *configuration parameters* that control GerbTool's startup operating environment. While all configuration file parameters can be set from within GerbTool, you may, instead, use a text editor or word processor (in ASCII mode) to create or modify GT.CFG.

Configuration parameter descriptions

See For a complete list of configuration parameters and a sample configuration file, see *Appendix B: Configuration files*.

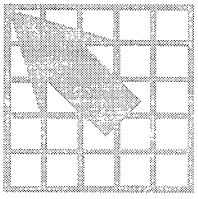
Color list file

When starting up, GerbTool looks for a color list file named COLOR.RGB in the same manner that it looks for a configuration file. Once the color list file is found, GerbTool first reads the available colors from a red-green-blue (RGB) color and name pair list, then reads a list of the currently chosen colors. The currently chosen colors are those presented whenever you select colors from within GerbTool (e.g., flash and draw colors).

```
# maximum 1024 colors available...
[RGB Color/Name pairs]
128   0   0        vga16red
  0 128 128        vga16cyan
  0 128   0        vga16green
245 245 245        WhiteSmoke
           .
           .
           .
255 250 240        FloralWhite
253 245 230        OldLace
250 240 230        linen
250 235 215        AntiqueWhite
# maximum 32 current choice colors...
[Choice Colors]
blue
vga16green
white
black
coral
    .
    .
    .
SteelBlue
SaddleBrown
DarkSalmon
DarkOrange
DeepPink
```

Sample color list.

Chapter 3



Quick start

In order to help you get started quickly, this chapter provides a quick overview of using GerbTool. A more comprehensive description for each GerbTool function is provided in chapters 4 through 8.

Starting GerbTool

To start GerbTool, select it from the Tools menu in the OrCAD Layout for Windows session frame.

Creating a new aperture list

To create a new aperture list, select the *Apertures/Load* command. The file chooser displays. Enter the name of a new aperture list and choose the OK button. GerbTool tells you that the requested aperture list doesn't exist and gives you the opportunity to create it. If you respond Yes, the new aperture list will be created on disk and loaded into GerbTool. You can then edit the aperture list by selecting the *Apertures/Edit* command (see *Chapter 7: Command reference*).

Converting a CAD aperture list

GerbTool provides aperture list conversion for most of the CAD and photo-plotter aperture list formats in use today. The conversion process translates a CAD aperture list directly into GerbTool format, thereby reducing data-entry-related problems.

The following table shows the aperture list formats supported by GerbTool, along with the name of the Aperture Conversion Rule (ACR) file used for the conversion.

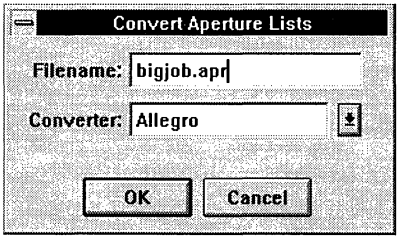
Aperture list format	*GerbTool ACR file*
ALLEGRO	ALLEGRO.ACR
CADSTAR	CADSTAR.ACR
CADSTAR 2	CADSTAR2.ACR
CONSULTEK	CONSULTK.ACR
CSI	CSI.ACR
CSI V4	CSI4.ACR
CSI Report	CSIRPT.ACR
DC-CAD	DC-CAD.ACR
DC-CAD 2	DC-CAD2.ACR
EAGLE	EAGLE.ACR
EDT	EDT.ACR
EDT 2	EDT2.ACR
EE Designer	EED.ACR
GraphiCode Report	GCREP.ACR
GerbTool Report	GTREP.ACR
HIWIRE	HIWIRE.ACR
IVEX	IVEX.ACR
Lavenir Report	LAVINER.ACR
Lavenir View	VIEW.ACR
MASSTECK	MASSTEK.ACR
OrCAD Layout (up to v6.42)	MASSTEK.ACR
McCAD	MCCAD.ACR

Supported aperture list formats (page 1 of 2).

Aperture list format	*GerbTool ACR file*
MENTOR	MENTOR.ACR
OrCAD PCB II	ORCAD.ACR
PADS	PADS.ACR
P-CAD	PCAD.ACR
P-CAD V6	PCAD6.ACR
P-CAD V7/V8	PCAD7_8.ACR
P-CAD Report	PCADRPT.ACR
PRANCE	PRANCE.ACR
PRANCE 2	PRANCE2.ACR
PROTEL 1.0	PROTEL.ACR
PROTEL for Windows	PFW.ACR
SCICARDS 2	SCICARD2.ACR
SCICARDS	SCICARDS.ACR
TANGO	TANGO.ACR
ULTIBOARD	ULTIBRD.ACR
UNICAD	UNICAD.ACR
VALID	VALID.ACR

Supported aperture list formats (page 2 of 2).

To convert a supported aperture list to GerbTool format, select the *Apertures/Convert* command, specify an input filename, then select the appropriate converter in the Convert Aperture Lists dialog box.



Convert Aperture Lists dialog box.

See also For more information about converting aperture lists, see *Chapter 7: Command reference.*

Creating a new design

To have GerbTool create a design file for you automatically, choose *Auto* from the *File/New* sub-menu. This command builds a design file for you automatically by examining the contents of a specified directory and determining which files are Gerber and/or aperture lists. The *Layers/Edit* form (see *Chapter 7: Command reference*) then displays, so that you can make any final adjustments, if necessary.

To create a new design file manually, select *Manual* from the *File/New* sub-menu The *Layers/Edit* form will display. After filling in the *Layers/Edit* form, you continue the loading process by choosing the OK button, at which point the files you specified in the *Layers/Edit* form are loaded.

Whether creating design files automatically or manually, GerbTool creates a design file named UNTITLED.GTD in the current directory. You can use the *File/Save* command to save your design file under a different name.

Loading an existing design

To load an existing design, select the *File/Open* command. You will be prompted for a design filename. You can enter an exact filename or use wildcard specifications. If you use wildcard specifications, a list of matching files displays. To select a filename from the list, click on the filename. After selecting a filename, you can accept your selection by choosing the OK button, or you can cancel the load operation by choosing the Cancel button. After choosing a design file, the *Layers/Edit* form displays. You can make any necessary modifications in the *Layers/Edit* form, or you can accept the previously saved layers data "as is." After choosing the OK button, the files you specified in the *Layers/Edit* form are loaded.

Saving a modified layer

GerbTool will prompt you to save a layer if it detects that the layer has been modified or otherwise changed. If a layer has been modified or changed, you will be given an opportunity to save it when you select the *Files/Save* command.

Note When prompted with a list of files to save, you must click on each file you want to save. Only those files selected or otherwise highlighted will be saved.

Exiting GerbTool

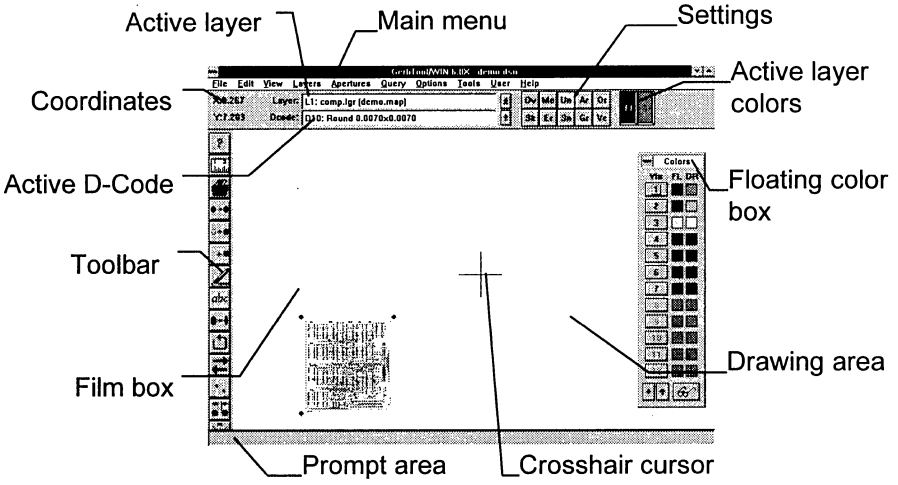
To exit GerbTool, select the *File/Exit* command. If any layers have been modified, GerbTool will request confirmation that you really want to exit.

Chapter 4

GerbTool basics

This chapter provides information on the basics of operating GerbTool.

The GerbTool desktop



The GerbTool desktop consists of the following components:

- Main menu bar where you can access the command menus.
- Toolbar where you can invoke commands with a single click.
- Active layer status box where you can specify the currently active layer for editing commands.
- D-Code status box where you can specify the currently active D-Code for editing commands.
- Coordinates status area where the current X-Y coordinates are displayed according to the current crosshair cursor position.
- Settings control area where you control various program settings such as whether Metric display mode and Grid Snap are enabled.

- Active layer color buttons you can use to make changes to the flash/draw colors of the currently active layer.
- Floating color dialog box where you can change layer colors and visibility.
- Drawing area where all database items are displayed.
- Crosshair cursor that indicates the position of the mouse within the drawing area.
- Film box graphic that indicates the size of the current film box.
- Prompt area where GerbTool commands issue user prompts.

Main menu bar

The main menu bar appears across the top of the desktop display. Each word in the menu bar represents a menu of related commands. When you select a word in the menu bar by moving your mouse over the word and clicking the menu button, the menu will display. Each item on the pull-down menu can be executed by selecting it.

Toolbar

The toolbar appears vertically along the left side of the desktop display. Each icon within the toolbar represents an alternate method of invoking a command. When you click on a icon in the toolbar, the command associated with the icon will be invoked.

Active layer

The active layer status box allows you to control the currently active layer. To change the active layer, select a new one from the drop-down list of layers. You can also click on the field and type a new layer number.

Floating color chooser dialog box

The floating color chooser dialog box is activated by choosing the push pin button in the *Layers/Colors* dialog box. Once activated this floating dialog box remains on the GerbTool desktop until it is manually closed using the dialog box's system button. It is available at all times to change layer colors and visibility.

D-Code

The D-Code status box allows you to control the currently active D-Code. This is the D-Code that will be used when adding new items to the database using the *Edit/Add* commands such as *Edit/Add/Text*. To change the D-Code, click on the down arrow to drop down a list of D-Codes to choose from. You can also click directly on the field and type a new D-Code.

Coordinate display

The coordinate display is for information purposes only. It shows you at a glance the current location of the crosshair cursor. The format of the display is controlled by the *Settings Me* button, described below, and the file format of the active layer.

Settings

The Settings control area allows you to control various program options with a single mouse click. This section describes each check button within the *Settings* control box.

Sk (sketch)

This button toggles Sketch mode on/off. When *Sketch* mode is enabled, pads are shown with an outline only, and traces are displayed as a single thin line. Besides slightly speeding up redraw times, this mode can also help you spot stacked pads.

Ov (overlay)

This button toggles Overlay mode on and off. When *Overlay* mode is enabled, items become transparent when drawn atop each other. When *Overlay* mode is disabled, new items obscure whatever was previously drawn. *Overlay* mode makes it easier to spot stacked pads.

Sn (snap)

This button toggles grid snap mode on and off. When *Snap* mode is enabled, your crosshair cursor will automatically jump to the nearest grid point.

See For information on grids, see *Chapter 7: Command reference.*

Me (metric)

This button toggles Metric mode on/off. When *Metric* mode is enabled, all information and editing fields within GerbTool that represent sizes and distances (i.e., coordinates) will be shown in metric format.

Un (undo)

This button toggles the saving of undo information on/off. If undo is currently disabled, it will become enabled. If undo is currently enabled, any current undo information will be destroyed and undo will then be disabled.

Er (errors)

This button toggles the display of rule violation errors on/off. After executing DRC or Snoman, any rule violation errors are shown. These items are displayed indefinitely until you reload or run DRC or Snoman again. If don't need to see the highlighted rule violation errors, you can use this command to disable their display.

Vc (view composites)

This button toggles the way composite layers (274-X and FIRExxxx only) are displayed. When this button is checked the polarity of each layer, specified by the *Key* field within the *Edit/Layers* form, will be honored. If a layer is specified as Clear, all data in that layer will be displayed with the current background color.

Or (orthogonal snap)

This button allows you to toggle orthogonal snap mode on/off. When enabled, lines drawn interactively will be forced to the specified angle.

Note The current setting can be temporarily overridden by holding down the CTRL key.

Ar (arcs 360°)

This button toggles the method of creating arcs used by the *Edit/Arc* and *Edit/Circle* commands. If enabled all arcs will be created using 360° circular interpolation. If disabled, all arcs will be created using small line segments. This does NOT affect the way Gerber data is read from a disk file. It only pertains to adding new arcs with the *Edit/Arc* and *Edit/Circle* commands.

Gr (grid)

This button toggles the system grid display on or off.

See For information on grids, see *Chapter 7: Command reference.*

Drawing area

The drawing area is the area between the main menu bar and the prompt areas. All database items are displayed here.

Crosshair cursor

While the mouse position is within the drawing area, the cursor will be displayed as a full-screen crosshair cursor. When the cursor is moved out of the drawing area, the cursor will usually be displayed as a small arrow.

Film box

The film box represents the size of the film that you will plot on, and is a graphic display only. It does NOT become part of your Gerber database(s).

Tip You can control the size and color of the film box with the *Options/Film Box* command detailed in *Chapter 7: Command reference.*

Prompt area

GerbTool editing commands issue prompts in this area to avoid cluttering the screen display with dialog boxes.

Design files

GerbTool utilizes the concept of a *design file.* A design file created by GerbTool contains information about the Gerber files and their associated aperture files that constitute a single PCB design. This includes filenames for inner and outer signal layers, silkscreen layers, soldermask layers, and so on.

Note The default file extension for design files is configurable, and can be changed using the *Options/Defaults* command.

GerbTool also stores its operating environment in each design file. This means that when you load an existing design file, the complete GerbTool environment in use at the time the design file was saved is loaded also, thus eliminating the need to continually reconfigure GerbTool each time a different design is loaded.

Aperture list files

Aperture list files are used to define the characteristics of each Gerber D-Code used in a design. For each D-Code specified in an aperture list file, the shape, size, type, and NC Drill tool number are defined (see *Chapter 7: Command reference*). GerbTool stores aperture lists in ASCII format. This makes it easy to create and modify aperture lists outside of GerbTool if you want. It also allows easy conversion from most CAD systems' aperture lists.

See For details of the aperture list format, along with an example aperture list, see *Appendix C: Aperture list file format.*

Note The default file extension for aperture list files is configurable, and can be changed using the *Options/Defaults* command.

Invoking GerbTool commands

This section describes the different ways to invoke GerbTool commands.

Mouse-button and function-key commands

GerbTool comes pre-configured with the following mouse-button and function-key assignments.

Key	*Assignment*
Left mouse button	View/Window
Middle mouse button	View/Zoom In
Right mouse button	View/Zoom Out
F1	View/Redraw
F2	View/Errors
F3	View/Previous
F4	Layers/Colors
F5	Layers/Edit
F6	Apertures/Edit
F7	Apertures/Report
F8	Query/Highlight/Dcode
F9	Query/Item Info
F10	Query/Measure
F11	Edit/Select/Add
F12	Edit/Select/Remove

The assigned mouse and function key commands are available any time GerbTool is idle (i.e., there is no command prompt in the prompt area).

See For complete information on customizing your mouse buttons and function keys, see *Chapter 2: Configuration*.

Selecting from the main menu

At any time, you can position your cursor in the main menu bar and select a command by clicking a mouse button. If you complete a selection, any previous command will be terminated before executing the new selection.

Nested commands

Nested commands are available anytime GerbTool has prompted you to enter a point or is idle. Below is a list of the nested commands. The nested commands are executed immediately without affecting the current command.

Key	*Action*
ENTER	Enter coordinate at cursor location
HOME	Snap cursor to center of item
PGUP	View/Zoom In
PGDN	View/Zoom Out
+ *or* I	View/Zoom In
- *or* O	View/Zoom Out
0-9	Bring a layer to the top (1-10)
CTRL+0-9	Bring a layer to the top (11-20)
A	Turn on all layers
CTRL+A	Turn off all but active layer
B	Pop-up floating color box
C	Enter absolute coordinates
CTRL+C	Enter relative coordinates
D	Increment current D-Code
CTRL+D	Decrement current D-Code
CTRL+F	Edit configuration flags
CTRL+G	Edit system grid
H	Toggle highlights on/off
CTRL+H	Show this list
L	Increment active layer
CTRL+L	Decrement active layer

Nested commands (page 1 of 2).

Key	*Action*
M	Run macro
CTRL+M	Toggle metric mode
P	View/Pan
CTRL+P	Toggle auto pan mode
CTRL+ALT+Q	Quit immediately without confirmation
R	View/Redraw
CTRL+R	View/All
S	Toggle grid snap
CTRL+S	Screen print
U	Undo last edit
CTRL+U	Undo all edits
V	Toggle composite viewing
CTRL+V	Toggle virtual panel mode

Nested commands (page 2 of 2).

Interrupting a drawing process

Anytime GerbTool is redrawing the display or highlighting a window of data, you can halt the drawing process by touching the ESC key or clicking the right mouse button. This will not affect the operation of the command and in many cases will speed up the operation of a command.

Ending a command

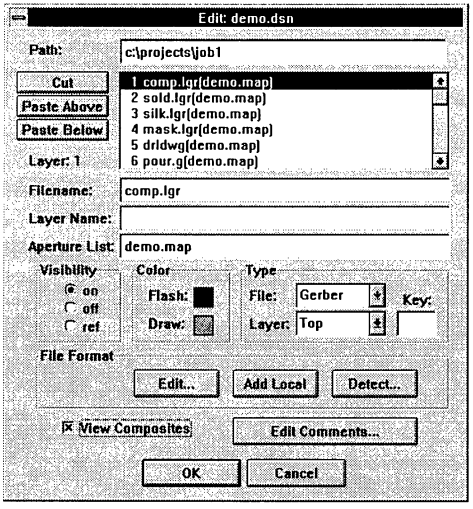
You can end a command, or end at least one level of a multistep command, by touching the ESC key or right mouse button.

Editing forms, dialog boxes, and the file chooser

GerbTool makes use of editing forms, dialog boxes, and the file chooser to obtain information from you. These elements are explained below.

Editing forms

Editing forms are used to enter information into GerbTool. They contain data entry fields, checkable buttons, color buttons, scroll bars, and exit buttons. The *Layers/Edit* form shown below is an example of an editing form.



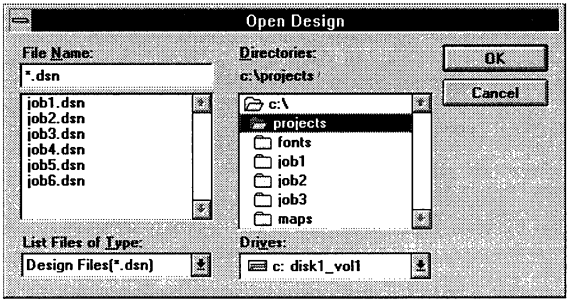
Editing form.

Dialog boxes

Dialog boxes are a method of communicating with the user. A Dialog box can contain one or more data entry fields and/or one or more exit buttons.

File chooser

The file chooser provides a convenient way of selecting filenames so you don't have to remember them all. There are two forms of the file chooser form. The first, allows you to select only one filename. You select the file by clicking directly on a filename. The chosen filename appears in the *Filename* field.



File chooser form.

The second, allows you to select multiple filenames by clicking and dragging your mouse directly over each filename you want. Each selection remains highlighted. The files chosen will be returned in the order in which they were selected. The behavior of the file chooser depends on which form/field you are currently editing.

Chapter 5

Performance tips

This chapter provides tips on obtaining optimal performance from GerbTool.

Speeding up GerbTool operations

Using nested commands

A powerful feature of GerbTool is the availability of nested commands. These commands are available at all times when GerbTool is waiting for you to enter a coordinate (point) or is idle (i.e., no command has been selected). With these commands you can move around, snap to the center of a database item, change which layers are viewed, undo edits, and so on.

See For a complete list of available nested commands, see *Chapter 4: GerbTool basics*.

Interrupting, redrawing, and highlighting

Any command that redraws the database or highlights a group of items can be speeded up by canceling the drawing process. By clicking the right button or touching the ESC key, you can halt the redrawing of the display. This doesn't affect the operation of the command, only the redraw is affected. Once you're comfortable with the operation of GerbTool commands you will find that this ability significantly speeds things up.

Undoing edits

The Undo command provides a high level of freedom when making database edits. You can experiment and try different edits without fear of data loss when undo is enabled. Since undo is available as the nested command U, you can undo edits immediately without having to exit the current command. Undo works for all edits regardless of size, and there is no limit to the number of edits you can undo. Remember to enable undo with the *Options/Undo* command **before** making your edits, then use the *Edit/Undo* or the nested command U to undo as necessary.

Note You can tell at a glance if undo is enabled by checking the *Settings Un* button.

Programming mouse buttons and function keys

GerbTool's easy-to-use GUI (Graphical User Interface) is further enhanced with the versatility of programmable mouse buttons and function keys. Using the *Options/Key Cmds* command, you can program the mouse buttons and function keys F1 through F12 with commands that you frequently use. While the programming can be saved using the *Options/Save* command, you can also make temporary changes to the current programming. This allows you to adapt GerbTool to a particular situation.

See also For more information on how to program your mouse and function keys, see *Chapter 7: Command reference* and *Chapter 2: Configuration*.

Memory considerations

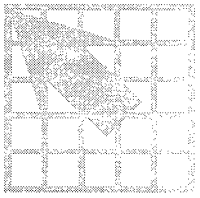
GerbTool was developed to operate in a true 32-bit environment with virtual memory. This allows GerbTool to address the entire memory range of the CPU even if the actual installed amount of RAM memory is less (e.g., 8 Mb).

Note While virtual memory is a powerful feature, there is no substitute for RAM memory for maximum speed. For example, if you load 16 Mb of Gerber files into GerbTool on a 8 Mb system, you will notice a lot of disk activity as the virtual memory manager begins to experience problems due to the disproportionately small amount of real memory.

Memory allocation errors and disk space

If your system has exhausted its allocated swap space, you will receive a memory allocation error message. You can help keep the swap file usage down by occasionally using the *Edit/Purge* command (see *Chapter 7: Command reference*) and by disabling the undo feature. Purging compacts GerbTool's internal database and allows more efficient use of memory.

Chapter 6



Uses for GerbTool

This chapter provides several examples of the kinds of tasks that can be accomplished with GerbTool.

Layer alignment

Layer alignment involves lining them up all layers so that when multiple layers are viewed simultaneously, they are correctly aligned. Proper layer alignment is also crucial to the successful creation of a multilayer netlist.

First determine the layer that all other layers should be aligned with (a master layer) and select an item to use as a reference point. Invoke the *Edit/Align* command and select the item you chose as a reference point. You then select an item, on each layer to be aligned, that corresponds to the reference point. As you select each additional item, the entire layer will be automatically aligned.

Tip You can use the nested zoom in/out and pan keys (see *Chapter 4: GerbTool basics*) to make it easier to locate the reference and corresponding items.

Creating NC Drill files

Using the *Tools/NC Drill* command, GerbTool allows you to create an NC Drill file from any layer. The format of the created drill file is selected by choosing the *NC Format* button within the NC Drill editing form (shown in *Chapter 7: Command reference*). The layer chosen to create a drill file from usually represents the pad master for the entire design. When creating NC Drill files GerbTool translates the Gerber flashes (except targets and thermals) into drill "hits." The *Tool* field, in the corresponding aperture list for the selected layer, is used to determine the tool call-out for each drill hit that is output.

Note Use the *Apertures/Report* command to determine if you have a tool assigned to each flash used. Edit the aperture list if required so all flashes are assigned a tool.

The drill hits are then optimized, according to your specifications, for fastest through-put.

Panelization of the image should be performed prior to executing the *Tools/NC Drill* command. If your drilling equipment has a small memory capacity, you should perform a "virtual" panelization. This will allow GerbTool to insert the needed step and repeat codes into the output drill file. Preferably, if your drilling equipment has enough memory, you should perform a normal non-virtual panelization. This will result in a fully optimized panel for the maximum in efficient drilling.

Importing NC Drill files

Using the *File/Import/NC Drill* command, you can load a NC Drill file into the active layer. If you want, you can create a new empty layer first by selecting the *Layers/Edit* command and entering a filename into a blank filename field. Make sure that the layer you choose is the active layer.

When loading a NC drill file, GerbTool converts the drill hits into Gerber flashes. Each tool called out in the drill file is located in the aperture list for the active layer. If a tool can't be found, an aperture will be added to the list with an "Unknown" shape and the correct tool assignment. You can then edit the aperture to correct the shape, size, and so on.

Note Use the *Apertures/Report* command to determine if any apertures were added. Those added will be highlighted.

Panelizing

GerbTool makes panelizing a simple, one-step process when using the *Auto Panel* feature. After turning on only the layers to be panelized, select the *Tools/Panelize* command, ensure that the *Auto Panel* button is checked (shown in *Chapter 7: Command reference*), and enter the minimum image border-to-border spacing in the *X* and *Y* fields. The spacing you specify should be between adjoining edges of the intended images. GerbTool will automatically calculate the maximum number of images that will fit inside the current film box. After asking for confirmation, GerbTool will complete the panelization process. Depending on the setting of the *Virtual* button, GerbTool will either copy the proper number images into the database or note the number of copies and their location for display purposes.

Note You can right click or touch the ESC key to stop the drawing process anytime during the panelizing process. This usually provides a noticeable improvement in the overall time to complete the panelizing process without affecting the finished panel in any way.

Viewing or printing 274-D composite layers

By allowing the use of black and white for layer colors, GerbTool allows accurate viewing of composite power and ground layers. Setting the negative layer to white on a black background and the positive layers to black will result in a realistic depiction of the final film.

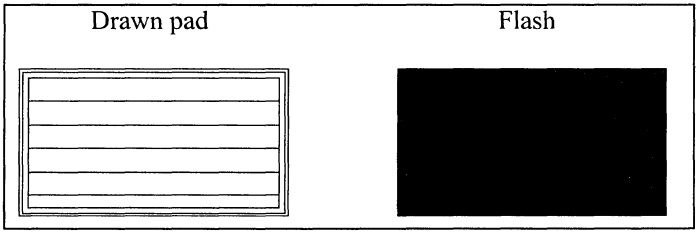
Note Since the negative layer must be displayed first, it is important that the negative layer be before the positive layers (i.e., a lower layer number) and not the active layer.

To print a composite layer, view your composite layers as described above, then use the *File/Print* command. The printed image will appear on the page exactly as it does in the display.

Note Since the image for printing is created in a high resolution off screen bitmap, the film box and display grid may appear on the output page. You can disable this by setting the film box color to the background color using the *Options/Film box* command and disabling the display of the grid using the *Options/Grid*, or nested command G.

Drawn pads

Occasionally, CAD systems may output a irregularly shaped or sized pad using multiple draws to "fill in" the shape, rather than a more efficient single flash. This results in larger than necessary Gerber files and increased processing times. Also, it is virtually impossible for high-level CAM tools such as DRC to recognize the drawn pads as pads rather than as collections of traces. The difference between a typical drawn pad and a comparable flash is shown below.



Drawn pad versus a flash.

The drawn pad shown requires 27 separate Gerber commands to accomplish what one Gerber flash can accomplish. As you can see, if you have 2000 of these drawn flashes, you'll have a Gerber file with at least 54,000 lines when flashes could accomplish the same thing in only 2000.

Using the *Tools/Convert/Pads* command, you can convert all your drawn pads to flashes. You do this by identifying one occurrence of a drawn pad and allowing GerbTool to find all drawn pads that match. And, to increase GerbTool's ability to recognize matching drawn pads, you can specify a tolerance value to compensate for some CAD systems' round-off errors. By specifying a tolerance, you allow GerbTool to relax its criteria for determining matching drawn pads.

Tip Converting drawn pads to flashes should be the first thing you do to your designs. This usually ensures trouble-free conversion. Also, you must convert all drawn pads to flashes **before** generating a netlist or running most other CAM tools.

Automatic silkscreen clean-up

GerbTool has the ability to automatically clean up a silkscreen where lines touch or are too close to the pads. Using the *Tools/Fix SS* command, you specify the layer(s) that the silkscreen and pad master are on and the minimum spacing that must be maintained between the silkscreen data and the pads. If you want, you can use window mode to clean up isolated areas rather than the entire silkscreen layer. GerbTool will then clean up all places where silkscreen lines are too close to a pad. Each offending line is moved just enough to eliminate the violation and no more.

See also For more information on silkscreen cleanup, and to see before and after illustrations, see *Chapter 7: Command reference.*

Creating a soldermask layer

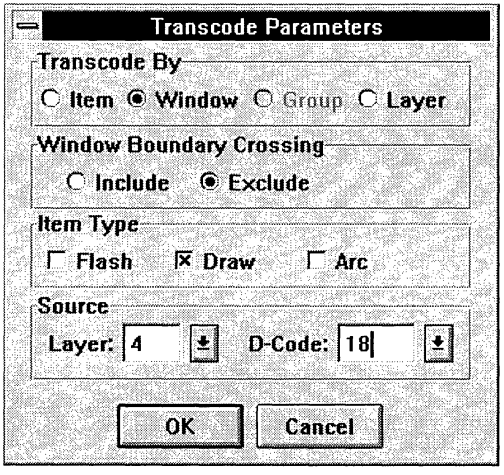
Creating a soldermask is a simple and easy process using the *Edit/D-Code/Scale* command.

First create the soldermask layer by copying the pad master layer onto a new layer. Use *Edit/Copy* command to copy the pad master to the new layer. When copying, select Create Layer from the *Copy to Layer* fields drop-down list. This will create a new layer for the new soldermask data.

Now, select the *Edit/D-Code/Scale* command, enter a scale factor for both X and Y and click on the *Fixed Amount* field. in the D-Code Scale form and click on the *OK* button. GerbTool will add apertures to the corresponding aperture lists as necessary and replace the D-Codes with the new scaled D-Codes. The original D-Codes within the aperture lists are not modified.

Transcoding

Using the *Edit/D-Code/Transcode* command, you can transcode (transform D-Code) either item by item or by selecting a group. Using selection criteria, you can choose exactly which D-Codes are transcoded. For example, to transcode only draws with a D-Code of D18 only on layer 4 and only within a particular window, the following selection criteria would be required:



Restrictive selection criteria.

After selecting and highlighting the D-Codes, GerbTool will prompt you for the new D-Code and then perform the actual transcoding.

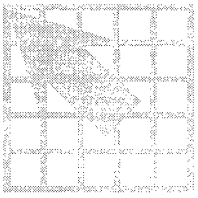
Snoman filleting and teardropping

Snoman is a highly configurable form of the method of eliminating pad/trace separation that is often referred to as *filleting* or *teardropping* (see *Appendix D: Snoman concepts* for a technical description of Snoman). The purpose of Snoman is to increase your manufacturing yield by adding more copper in the area of the pad/trace junction, thereby eliminating any possible pad/trace separation. Snoman is used primarily when dealing with small pads and traces (such as micro vias in the 30 mils or less range) but can be used anywhere to prevent pad/trace separation. Snoman provides additional versatility by allowing you control of the size and location of the generated Snoman pads, along with an integral DRC to eliminate any possible spacing violations.

See For a complete description of how to use the Snoman tool, see *Chapter 7: Command reference.*

Trivia Snoman derives its unusual name from the appearance of a Snoman pad placed on top of a host pad, which resembles a "real" snowman.

Chapter 7



Command reference

This chapter provides details of invoking and using each GerbTool menu command.

File menu

This main menu selection displays a menu of commands for dealing primarily with files and directories. The menu selections are described in the following sections.

New

The New command displays the Auto and Manual commands, which are described in the following sections.

Auto

This command will build a design file for you automatically by examining the contents of a specified directory and determining which files are Gerber and/or aperture lists. The Gerber filenames are sorted first alphabetically and then by layer number if one is found. If an aperture list is found that is not already in GerbTool format, each configured aperture list converter will be tried until a match is found. Finally, each aperture list found will be matched to a suitable Gerber filename. The *Layers/Edit* form will then be displayed where you make any final adjustments if necessary.

Note The speed and usability of this command is directly affected by the `AP_CONV` and `AP_CONV_IGNORE` configuration file parameters detailed in *Appendix B: Configuration files*. In general, the more aperture list converters that are configured and the less filename extensions that are specified to ignore, the slower this command will be. Therefore, if there are aperture list converters configured that you don't use, they should be removed from your configuration file.

Manual

This command will create an empty design file for you, and then display the *Layers/Edit* form for you to enter the Gerber files and aperture lists.

Open

This menu selection will display the file chooser and prompt for a design file to load. You can use a wildcard specification to obtain a list of files from which to choose. After specifying a design file to load, the *Layers/Edit* form will be displayed where you can define or modify the layer structure and, if needed, define or change the Gerber input format specification.

Close

Selecting this menu item allow you to optionally save the current design file and then close and unload the current design.

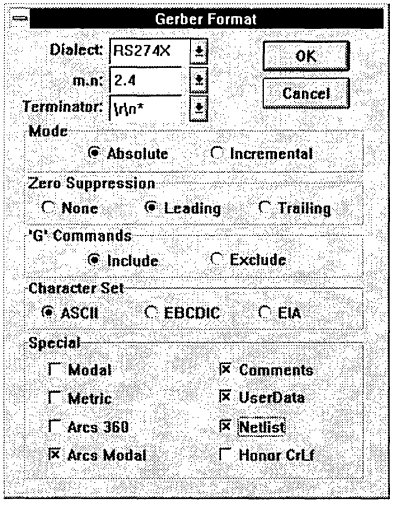
Save

Select this menu item to optionally save the current design file and any modified layers or aperture lists. This command does not clear the current design; you can continue to work on the current design after saving. You must use this command to save modified layer data. Another use of this command is to periodically save your work to guard against an unexpected power outage, or before doing any major edits without undo enabled (see *Chapter 4: GerbTool basics* for more information on undo).

Format

The *Format* forms allow you to specify input/output file formats for the supported file types.

Note GerbTool supports both global and local formats. Global formats apply to all layers that do not have a local format assigned to them. This command allows editing of the global formats only. See the *Layers/Edit* command in *Chapter 7: Command reference* for more information on local formats.



Typical format form.

Editing a Format form allows you to specify the correct format for that type of file (e.g., Gerber). The illustration above shows a Gerber Format editing form, which includes the following fields:

Dialect

Indicates the specific dialect of the Gerber language such as RS274D, RS274X, FIRE9xxx and EIE. If in doubt, choose RS274D.

m.n

Coordinate Format such as 2.3. This specifies 2 decimal digits before an implied decimal point and 3 following. (e.g., 12250 represents 12.250 if the coordinate format is 2.3).

Terminator

Indicate the block terminator (EOB). Use \r to indicate a carriage return (ASCII 13) and \n to indicate a line feed (ASCII 10).

Mode

Choose Absolute or Incremental (see Glossary for descriptions of these terms).

Zero suppression

Indicate whether leading zeros are suppressed, trailing zeros are suppressed, or there is no zero suppression.

"G" commands

Indicate whether "G" commands (e.g., G01) should be included when you output Gerber files.

Special

You can enable *Modal* mode to reduce the size of your files by removing all redundant draft codes and coordinates, enable *Metric* mode indicating that your files are in metric format, specify whether all circular interpolated arcs should be considered 360° and/or modal, enable the saving of G04 *Comments*, enable the output of *UserData* information, enable the output of *Netlist* information embedded within the Gerber file(s), and specify if carriage returns and line feeds should be honored as block terminators.

You can toggle between metric and imperial format, as well as change *m.n* formats as you want after loading a design. If you change formats after loading, all layers will be marked as modified.

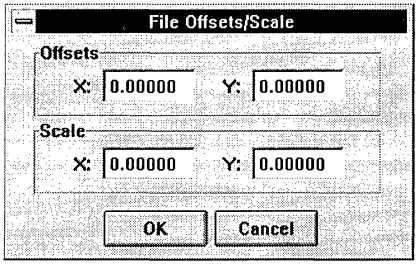
Caution If you change formats after loading and do not save all layers, the next time you load that design, the saved format may not match that of the unsaved Gerber files.

Enabling the *Netlist* button will allow GerbTool to save netlist information within the Gerber file. If you have previously saved a Gerber file with netlist information, you can remove it by disabling the *Netlist* button and saving.

Note It is important that you specify the correct format *before* loading a new design. The critical format items are m.n, mode, and whether trailing zeros are suppressed. If you load a design with an incorrect format, GerbTool will display it with unpredictable results. If you inadvertently load a design this way, reload the design and click on the *Format* button of the *Layers/Edit* form to correct the format.

Offsets

This command allows you to specify the coordinate offsets and scale to be used when loading a Gerber file.



Load offsets editing form.

The offsets and scale are applied during design loading, as well as file merging. By applying a scale factor it is possible to expand or shrink the size of your database. For example, if you design your boards at 2X you can set both the X and Y scale factor to 0.5 to convert your files to 1X.

Merge

The Merge command displays the Design and Gerber commands, which are described in the following sections.

Note All merge commands require that you ensure the critical format items (mode, m.n and zero suppression) of the file or files being merged match those of the currently loaded design.

Design

Selecting this command allows another complete design to be merged layer by layer into the current design. If a layer from the external design doesn't exist in the current design, you will be prompted to create a new layer.

Gerber

This command allows you to merge a Gerber file, on disk, into the currently active layer. You will be prompted for a filename. You can use a wildcard specification to obtain a list of files from which to choose. The specified filename is NOT added to the *Layers* form. Rather, the contents of the file are read in and appended to the active layer.

Import

The Import command displays the BARCO DPF, HPGL, IPC-D-356, and NC Drill commands, which are described in the following sections.

Note All import commands require that you ensure the critical format items (mode, m.n and trailing zero suppression) of the file or files being loaded match those of the currently loaded design.

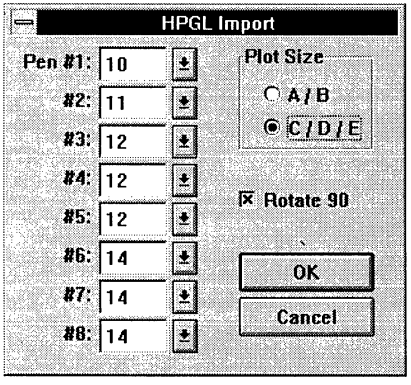
BARCO DPF

This command allows you to import a BARCO DPF file, on disk, into the currently active layer. You will be prompted for a filename. You can use a wildcard specification to obtain a list of files from which to choose.

Note Once a BARCO DPF file is imported into a layer it effectively becomes Gerber data and will indeed be saved as Gerber if the layer is subsequently saved. To output the layer in BARCO DPF format use the *File/Export/BARCO DPF command.*

HPGL

This command allows you to merge an HPGL plot file, on disk, into the currently active layer. You will be prompted for a filename. You can use a wildcard specification to obtain a list of files from which to choose.



HPGL Import form.

Using the HPGL input form, you can specify the expected plot size, whether to rotate the plot data and which D-Codes to use for each HPGL pen.

IPC-D-356

This command allows you to import an IPC-D-356 netlist into your design. Since an IPC-D-356 netlist contains information pertaining to pads and not traces, GerbTool must generate an internal netlist prior to importing an IPC-D-356 netlist, to ensure that your database contains a full and complete netlist after importing. While this may sound somewhat redundant, the added benefit of an "automatic netlist comparison" is well worth it. The netlist comparison feature produces a report file detailing any differences between the internal netlist and the imported netlist, in addition to highlighting any differences. Optionally, the database *UserData* fields can be updated with the component/net data from the IPC-D-356 file. This allows you to use GerbTool commands, including the *Query/Item Info* command, to examine and manipulate the true reference designators, pin numbers, and so on.

NC Drill

This command allows you to import a NC Drill file, on disk, into the currently active layer. You will be prompted for a filename. You can use a wildcard specification to obtain a list of files from which to choose.

Export

The Export command displays the IPC-D-350, IPC-D-356, and BARCO DPF commands, which are described in the following sections.

IPC-D-350

Designs exported to IPC-D-350 format will output into one disk file containing all layer data specified within the currently loaded design. The specified output file will contain all data necessary to reproduce your design on any IPC-D-350 compatible device.

IPC-D-356

Designs exported to IPC-D-356 format will output into one disk file containing all layer data specified within the currently loaded design. The specified output file will contain all netlist data associated with the current design.

BARCO DPF

Designs exported to BARCO DPF format will be output into a separate file for each layer. You select which layers to export and specify the output filenames. If you enable the *Auto Rename* button, GerbTool will output all selected layers, renaming each layer automatically using the filename extension specified in the *File Ext* field.

Plot

This command provides access to the built-in plotting capabilities of GerbTool. The New command displays the HPGL and Postscript commands, which are described in the following sections.

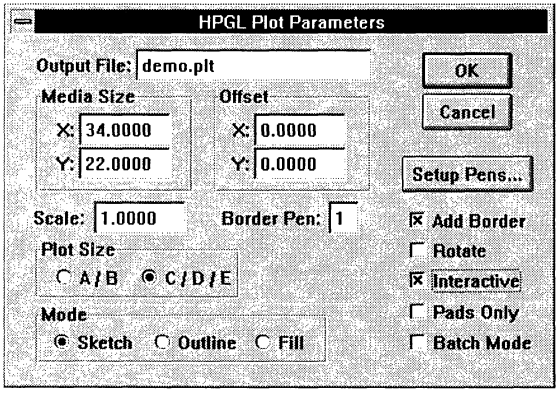
For either command, a form will be displayed that contains the plotter parameters for the plotter you've chosen. Fill in or change the appropriate fields and click on the OK button to begin plotting. Regardless of the plotter chosen, enabling the *Add Border* button adds a border to your plots.

See To determine the text that GerbTool adds to the border, see the description of the `BORDER_TEXT` configuration parameter in *Chapter 2: Configuration*.

Enabling *Batch Mode* instructs GerbTool to output each visible layer to a separate output file. During batch mode operation, if the *Output File* field is empty, the output filenames will be derived from the filename associated with each layer and the currently configured HPGL filename extension (see *Options/Defaults* later in this chapter). If, on the other hand, the output file field contains a filename, GerbTool will append a number representing the number of the input layer (e.g., `demo.001`, `demo.002`).

HPGL

GerbTool provides three modes of output when plotting on a HPGL compatible plotter: Sketch, Outline, and Fill. Sketch mode is the fastest but does not show width on draws and some flashes such as donuts. Outline mode shows true width on all objects but they are outlined only. Fill mode shows true width, and all objects are completely filled in as they would appear on a photoplot. Fill mode is the slowest and is extremely hard on plotter pens.



HPGL Plot Parameters form.

You can also specify output file, media size, plot offset, pen width, pen speed, pen number for flashes and draws, pen number for the optional border, scale, whether to rotate 90°, and whether to plot only pads (flashes). The offset values are applied independent of the scale specified. Plot offsets allow you to plot multiple images on one sheet.

Enabling Interactive mode allows you to interactively position each layer on the output page. To position an image on the page, click your mouse over an image to select it and then drag the image to the proper location and release the mouse button (or click again). During interactive plot positioning, a menu of buttons is provided along with several plot-specific nested commands.

HPGL interactive control form.

The *Plot* button saves the page layout and plot the data. The *OK* button saves the page layout and quits the interactive session without plotting. The *Reset* button allows you to reset the images to their initial positions for the session (if the form has been pinned) or quit the interactive session without saving the page layout or plotting the data.

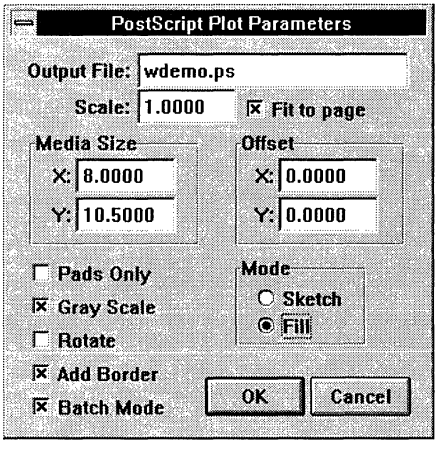
The nested commands available during a interactive plot session are: C for absolute coordinate entry, I for page layout initialization, L to cycle the current layer forward, CTRL+L to cycle the current layer backward, S to snap (align) the current layer on top of another layer, and R to redraw the page layout.

Note There are two files within the GerbTool program directory that affect each HPGL plot. The files HPGL.INI and HPGL.DEI are prefixed and appended, respectively, to the plot output. If you have special requirements, you can edit these files as needed.

PostScript

GerbTool provides PostScript output, allowing you to plot your data on any device that supports PostScript. This includes typesetters capable of producing production quality artwork. Two modes of output are provided when outputting PostScript: Outline and Fill. Outline mode shows true width on all objects, but they are outlined only. This allows you to check for overlapping features. Fill mode shows true width, and all objects are completely filled in as they would appear on a photoplot. Fill mode may produce a larger output file.

Enabling Gray Scale mode allows you to output accurate black and white composites as well as halftone images. When Gray Scale mode is disabled, all colors other than the background color are printed as black. When enabled, all colors (other than black/white) are converted to a different gray scale.

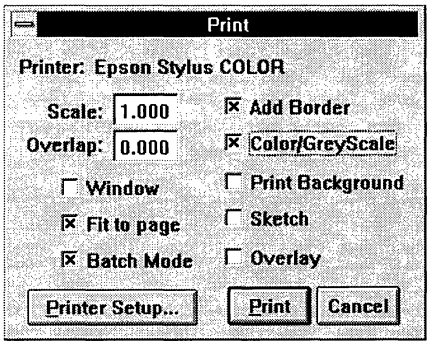


PostScript Plot Parameters form.

You can also specify output file, media size, plot offset, scale including **Fit to page**, whether to rotate 90° and whether to plot only pads (flashes). The offset values are applied independent of the scale specified. Plot offsets allow you to position the image anywhere on the media.

Print

Select this command when you want to print the viewed layers to the current Windows printer. This command allows you to print your design on any printer/plotter supported by Windows.



Print parameters form.

With this dialog box you specify the scale, including Fit to page, window mode, color or greyscale, whether to print the background color, sketch or overlay mode and whether you want batch mode. If Batch Mode is enabled, each visible layer will automatically be sent to the printer as a separate print job.

The Overlap field allows you to indicate how much to overlap the pages of a multipage plot to allow proper alignment when taping the pages together.

Printer setup

The Printer Setup command allows you to select and configure the current Windows default printer prior to using the Print command.

Change directory

Use the Chgdir menu item to change the default directory.

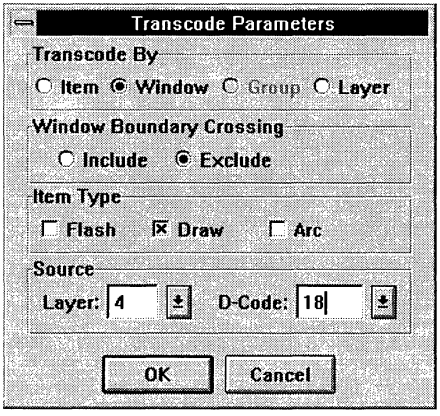
Exit

Select this command when you want to exit GerbTool. The current design file can be saved, and you will be prompted to confirm that you want to quit if any layers have been modified.

Edit menu

The Edit command displays the Add, Copy, Move, Erase, Clip, Join, Rotate, Mirror, Item, D-Code, Align, Origin, Undo, Purge, and Select commands, which are described in the following sections.

All editing commands that require you to modify one or more database items will allow you to edit the selection criteria for determining which database items to select or modify. GerbTool commands are flexible in the selection of data to modify. For example, depending on the command, you can choose from single item, window, group, or complete layer selections, as well as restricting your selections to particular layers, D-Codes, and so on.



Typical selection criteria.

With the form shown above, you can control whether flashes, draws, arcs or any combination of all three are selected. Whether a single item, window, group or complete layer is selected. In the case of *Window* mode, whether to include items that cross the window boundary. And finally, whether to restrict the selection to a particular layer or D-Code.

All editing commands can be terminated by clicking the right button, touching the ESC key, or selecting another menu item.

See For details on using GerbTool nested commands, see *Chapter 4: GerbTool basics*. Nested commands are selected with one keystroke and operate immediately, even during another command.

Add

The Add command displays the Flash, Draw, Rectangle, Vertex, Circle, Arc Ctr, Arc 3 Pt, Polygon, and Text commands, which are described in the following sections.

Note All circles and arcs are created using 360° interpolation or with multiple line segments depending on the style indicated by the *Settings* button *Ar*. Use 360° interpolation with care as not all photo-plotters support circular interpolation. Segmented circles and arcs use the chord angle specified using the *Options/Defaults* command.

Flash

This command allows you to add a flash to the active layer. GerbTool prompts for a point at which to add the flash. As you move the cursor around the screen an outline shape of the current D-Code is displayed. Click left to add a flash at that location.

Draw

This command allows you to draw line segments in the active layer. GerbTool prompts for a starting point and subsequent points to form continuous traces. Click the right mouse button or touch the ESC key to start a new trace.

Rectangle

This command allows you to draw line segments in the shape of a rectangle to the active layer. GerbTool will prompt for a starting corner point and a opposite corner point.

Vertex

This command allows you to add (and move by dragging the mouse) a vertex anywhere on an existing line segment.

Circle

This command allows you to draw a circle by entering a center point and a point on the radius. The circle is drawn on the active layer, using the current D-Code, in a counter-clockwise direction. See the note at the beginning of this section concerning how circles are created.

Arc (center point)

With the Arc Ctr command you define an arc by entering a center point, a point defining the radius and starting angle, followed by a point defining the ending angle. The arc is drawn on the active layer, using the current D-Code, in a counter-clockwise direction. See the note at the beginning of this section concerning how arcs are created.

Arc (3-point)

With the Arc 3 Pt command you define an arc by entering its end points and then a point on its circumference. The arc is drawn on the active layer, using the current D-Code, in a counter-clockwise direction. See the note at the beginning of this section concerning how arcs are created.

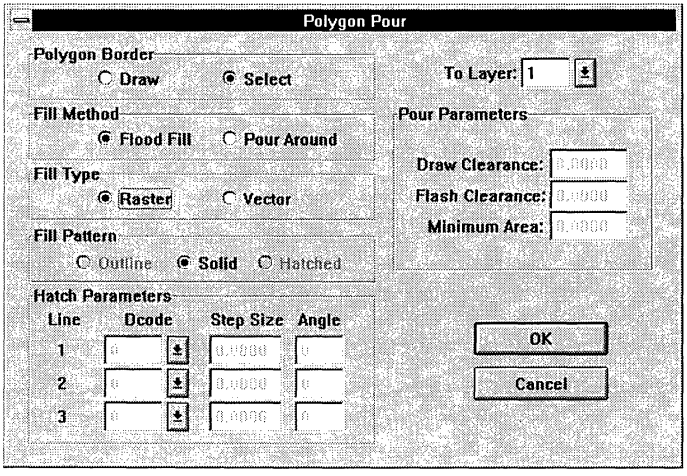
Tip To create 90° arcs, press the 9 key. This automatically creates a 90° arc.

Polygon

This command allows you to select or add a closed polygon and GerbTool will fill the interior of the polygon using either a raster fill or vector fill method. This command is commonly used to create ground plane areas.

Note Raster filling is not supported in RS274D format Gerber files.

When entering a polygon, you can automatically close the polygon by touching the END key at any time. You can also close the polygon manually by entering a point at the point that began the polygon. Regardless of the method of closure, GerbTool will outline the polygon with the current D-Code, as displayed in the status bar, and begin filling the interior of the polygon. In *Flood Fill* mode, GerbTool will fill the interior of the polygon with increasing aperture sizes. As it fills toward the center of the polygon, the aperture sizes will become larger and larger. In *Pour Around* mode, GerbTool will fill the interior of the polygon, as above, while maintaining clearance, as specified by the *Draw Clearance* and *Flash Clearance* fields, around all circuitry.

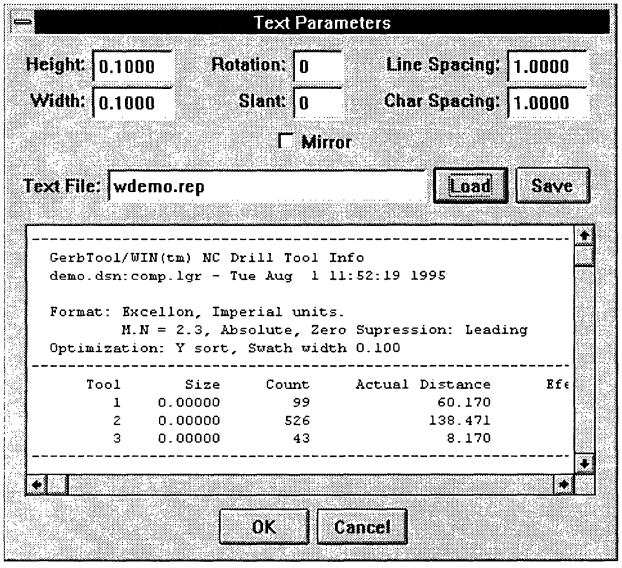


Polygon parameters form.

Because many smaller polygons may be generated to effectively "pour" around the circuitry, the **Min Area** parameter specifies the minimum size area. Any filled areas smaller than the **Min Area** will be eliminated. The **Pour Around** option supports three additional modes: **Outline**, **Solid**, and **Hatch** mode. If **Outline** mode is selected, no filling of the resultant polygons takes place. This type of output can be used to drive PCB prototyping equipment. If **Solid** mode is selected, the resultant polygons are filled completely using the same methods described for the **Fill** command above. If **Hatch** mode is selected, the polygons will be filled with a cross hatched pattern as specified in the *Hatch Parameters* section of the editing form. Up to three lines can be used with different sizes and angles for each line.

Text

The *Add/Text* command provides the ability to insert text into the database as a sequence of line segments. Therefore, you can control the line thickness of the inserted text by changing the current D-Code. Text can be rotated, mirrored or slanted. The height and width of the text is also user specified as is the inter character and line spacing. A text editing window is provided where you can enter as many lines of text as needed. You have full editing and scrolling capabilities. You can even load and save text files. The *Text* command displays the editing form shown below:



Text Parameters form.

Copy

You can use this command to copy single items, windows or groups of items. By specifying a valid destination layer in the *Copy to Layer* field, you can copy all selected items to that layer.

Note If you select data from more than one layer (i.e., more than one layer is visible) and copy to a destination layer, all copied data will be merged into the destination layer. If you do not choose a destination layer then the data will be copied into the respective source layers.

Move

You can use this command to move a single item, a window or groups of items. By specifying a valid destination layer in the *Move to Layer* field, you can move all selected items to that layer. As with *Copy* above, if you select data from more than one layer (i.e., more than one layer is visible) and move to a destination layer, all moved data will be merged into the destination layer. If you do not choose a destination layer then the data will be moved into the respective source layers.

Erase

Select this command when you want to erase items from one or more layers. Either vertices, single items, a window or groups of items can be erased.

Note If Undo is disabled, you will be prompted for confirmation when erasing items.

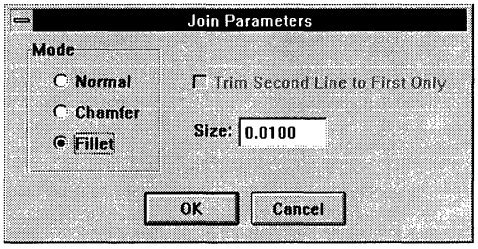
Clip

This command provides the ability to specify a window in which all data will be erased with automatic clipping of draws that pass through the window. If group mode is selected only items within the group will be considered when examining data within the specified window.

Note The *On Boundary* selection controls whether flashes that straddle a window boundary are erased or not.

Join

This command provides the ability to join two line segments together using several different methods.



Join Parameters form.

Using **Normal** mode, the two line segments chosen will be extended or trimmed as needed so that they connect. Naturally, this command will not work for parallel or near parallel lines. An option to normal mode, *Trim Second Line to First Only*, helps when you have a long line in one direction and several lines intersecting the long line. With this option only the second line you select will be modified. The remaining modes, **Chamfer** and **Fillet**, use the *Size* field to determine how far back to trim each of the two selected lines before adding the chamfer or fillet accordingly.

Rotate

Use this command when you need to rotate a window or group of items. You can select *Window* mode or *Group* mode. You can also supply a pivot point (interactive) or allow automatic calculation of the center of the data (calculated) for the required pivot point.

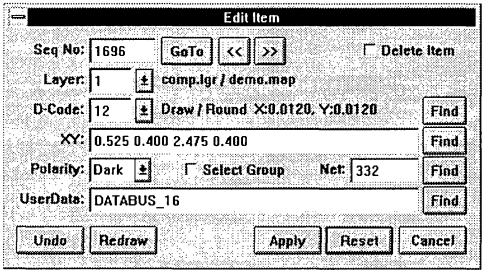
Note If the rotation factor entered is 90° or 270° and the Auto 90° button is pushed, this command will automatically compensate for asymmetrical pads, such as rectangles, by replacing the D-Code with an equivalent D-Code with opposite dimensions. New D-Codes can be added to the appropriate aperture list.

Mirror

Use this command if you need to mirror (flip) a group of items either horizontally or vertically. GerbTool will allow you to specify the direction to mirror and whether to prompt for the pivot point or automatically calculate it. This command can also be used to flip a secondary side layer that was designed as seen from the primary side.

Item

This command displays a dialog box that allows you to edit pertinent information associated with each database item. In addition to allowing you to edit each database item there are extensive controls for navigating from one item to another including the ability locate an item based on its sequential position in the database, D-Code, X-Y coordinate, net and *UserData* value. You can also step forward and backward one item at a time using the supplied directional buttons. You can also use the N key to automatically advance to the next item in the database.



Edit Item form.

The *UserData* field is of special note as this field allows you to attach any textual information you would like to each individual database item. Any text you associate with your database will automatically be saved within your Gerber files the next time you save them. This also allows you to pass on this data to other groups in your organization transparently.

An obvious use is to associate actual reference designators, pin numbers and net names with each pad thereby adding intelligence to your Gerber databases. Besides being able to see *UserData* using the *Query/Item Info* command, macros also have complete read/write access to each *UserData* field. This allows some pretty powerful tools to be built upon GerbTool.

Other than a 256 character size limit, there are no other restrictions on the text that can be associated with a database item.

D-Code

The D-Code command displays the Transcode, Expand, Scale, and Polarity commands, which are described in the following sections.

Transcode

This command allows you to change the D-Code of an individual item, window, group or complete layer. By changing the D-Code of an item, you can alter its size and shape. Another way to change an items size and shape is to edit the aperture list directly.

Expand

Use this command to expand one or all custom apertures in a design. This command is required if you want to plot a design that contains custom apertures and your photo-plotter is unable to create the apertures you need. GerbTool will prompt for a D-Code to search for. You can enter a specific D-Code or you can enter zero to instruct GerbTool to expand all custom apertures found.

Scale

Use this command to shrink or expand the size of one or more D-Codes. One use of this command is to create soldermasks automatically. GerbTool will add new apertures to the corresponding aperture list as needed based on your specified scale factor. If the *Fixed Amount* check button is enabled, the scale values will be added to each D-Code. Otherwise, each D-Code size will be multiplied by the scale values specified.

Polarity

Use this command to control the item level polarity of EIE and BARCO format files as well as FIRE9xxx raster fill polygons. When using item level polarity, the ordering of the data is crucial. You may find that you need to move data "in place," thereby placing the "moved" data at the end of the database.

Note Gerber (RS-274D) does not support polarity at all. Extended Gerber (274X) files only support polarity at the layer level, which is controlled using the *Layers/Edit* command. FIRE9xxx format only supports raster fill polygons at the item level. Otherwise, layer oriented polarity is assumed.

Align

This command allows you to align any misaligned layers. First determine the layer to which all other layers should be aligned with (a master layer) and select an item to use as a reference point. Select the item you chose as a reference point. Then select an item, on each layer to be aligned, that corresponds to the reference point. As you select each additional item, the entire layer will be automatically aligned.

Origin

This command allows you to relocate the origin (0,0 point) of the database. GerbTool will prompt for a point to define the new origin. The film box will be moved to the new origin.

Note This command causes GerbTool to mark all layers as modified.

Undo

This command allows you to fully undo changes you've made to the currently loaded database. Undo information is saved in a "last in, first out" fashion. This means that you undo changes in the reverse order in which the changes were made. This allows you to undo the most recent changes first. You can also use the nested command U to invoke the undo command even during another editing command.

Notes Undo must be enabled with the *Settings/Un* button prior to making any edits if you plan to use this command.

Undo increases the amount of memory GerbTool requires. If you do not require the undo capability, you can disable undo with the *Settings/Un* button. Disabling undo will release any memory currently associated with undo information and prevent further undo memory use.

Purge

Use this command to compact the currently loaded database for more efficient use of memory. Since GerbTool doesn't actually erase data from memory during edits, memory may become fragmented and less efficient. Therefore, occasional purging can help GerbTool perform optimally.

Note Purging destroys any undo information that currently exists. Do not use this command unless you are sure you don't need to undo any previous edits.

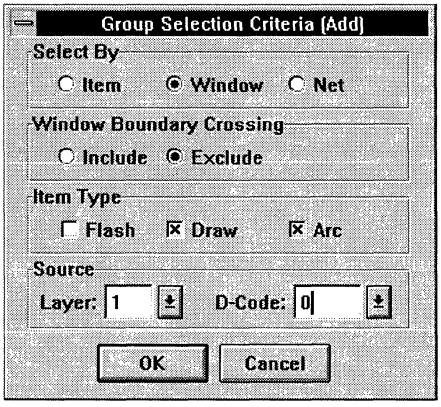
Select

The Select command displays the New Group, Add To, Remove From, Invert, and Off commands, which are described in the following sections. Most editing commands (such as *Copy* or *Move*) allow you to work with single items, windows of items, or groups of items. The commands available in the *Select* sub-menu allow you to manage the grouping of items for use by these editing commands. When a command allows group selection mode, it will use the currently selected group created and maintained by the different *Select* commands. Select groups are also persistent from one command to another. For example, if you rotate the current select group, the rotated data will remain selected ready for another command.

New group

This command allows you to start a new group of selected items. You will be prompted for confirmation to clear the current select group if any. This does NOT destroy any data. It deselects the current select group.

If you respond affirmatively, the Group Selection Criteria form will be presented and you will be automatically placed in the *Select/Add To* command.



Group Selection Criteria form.

Add to

Use this command to select more items and place them in the current select group. The Group Selection Criteria form will be presented where you specify the types of items you would like to select.

Remove from

Use this command to selectively remove items from the current select group. The Group Selection Criteria form will be presented where you specify the types of items you would like to deselect.

Invert

Use this command to invert the current select group. That is, all currently selected items are deselected and all deselected items become selected. One use of this command is to allow you to select all but a few items by first selecting the items you don't want and then inverting the select group.

Off

Use this command to clear the current select group and deselect any highlighted items. This does NOT destroy any data. It deselects the current select group.

View menu

The View command displays the Window, Zoom In, Zoom Out, Pan, All, Film Box, Redraw, Errors, Save, Recall, and Previous commands, which are described in the following sections.

Window

This command allows you to select a new viewing window. Two points are required to define a window. The two points define a rectangle that encompass the area that is to become the new viewing window. Use this command when you want precise control over the viewing window.

Zoom in

This menu item halves the size of the current viewing window using a center point that you supply. This command provides a closer look at the displayed data.

Zoom out

Doubles the size of the current viewing window using a center point you supply. Use this command to increase the size of the viewing window.

Pan

Moves the current viewing window to a new location. The new location is centered about a point you supply. This command does not change the size of the viewing window.

All

This menu item adjusts the size of the viewing window to encompass the extremes of the currently displayed layer(s). No user interaction is required. If data has been deleted from any displayed layers you may need to use the *Query/Extents* command to calculate the current extremes of the database.

Film box

Select this command to adjust the size of the viewing window to display the contents of the currently specified *Film Box*. This command does not check to see that all data lies within the film box borders. Therefore, depending on the film box size, not all data may be displayed. No user interaction is required.

Redraw

This command redraws the current viewing window.

Errors

This command is used to view rule violation errors after performing a DRC or running Snoman. Each time this command is executed, the viewing window is moved to the location of the next highlighted rule violation error, if any. After reaching the last error, GerbTool will cycle back to the first error.

Note An error will only be highlighted when its corresponding layer is viewed.

Save

This command is used to save the current viewing window for later recall. There are eight positions available, 1-8, for saving. The current viewing window will be saved in the position that you click on. Use the *View/Recall* command to recall any of the saved viewing windows.

Recall

This command is used to recall a previously saved viewing window (see *Save* above). If any of the eight possible positions does not have a viewing window associated with it, the corresponding position in the sub-menu will be disabled.

Previous

This command is used to recall the last viewing window. This allows you to toggle between two viewing locations.

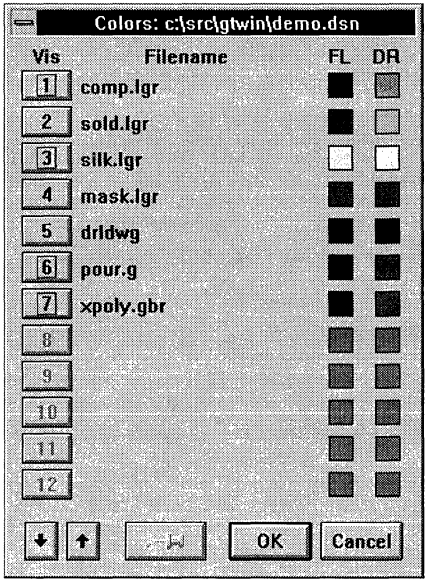
Layers menu

The Layers command displays the Colors and Edit commands, which are described in the following sections.

Colors

The *Colors* command allows you to edit the *Layers* color and visibility form. Within the *Colors* form you specify:

- Visibility: ON, OFF or REF.
- Draw and flash color.

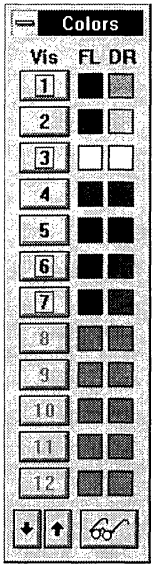


Layers color form.

When a layer is *on*, indicated by a red box around the layer number, it is both visible and editable. When a layer is *off* it is neither visible nor editable. When a layer is *ref*, indicated by a black box around the layer number, it is visible but not editable.

The push pin button activates the floating *Colors* form. This form stays pinned to your desktop while you work. It is available to control layer visibility and colors at any time regardless if any other command is active. As you make changes within this form, the changes take place immediately but the display is not automatically updated. To update the display (redraw) click on the eyeglass viewer button. You can also move this floating form to any convenient location.

Note If you find that you don't use the *ref* visibility setting, you can disable the availability of the *ref* status using the LAYERVIS_REF configuration parameter. See *Chapter 2: Configuration* for more information.



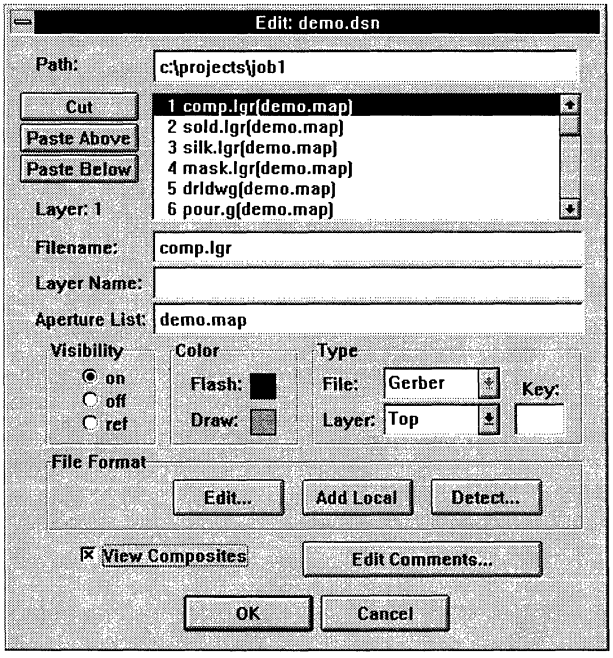
Layers floating color form.

Edit

The *Edit* command allows you to edit the *Layers* form. The effects of editing certain fields within the *Layers* form differ, depending on whether you are loading a design or editing after loading.

Within the *Layers* form you specify:

- The path used to locate the Gerber and aperture list files
- Gerber files
- Aperture list files
- Layer names
- Layer visibility
- Flash/draw colors
- Layer type
- File format
- Extended Gerber compositing instructions



Layers form.

While loading, edit the *Path* field to tell GerbTool where to find the specified files if they do not contain a path as part of the filename. Entering a wildcard specification (e.g., *.GBR) in the *Filename* field will display the file chooser. You can click on more than one filename and all selected filenames will be entered in one step. You can also enter a wildcard in an *Aperture List* field to obtain the file chooser. You can select a filename and it will be entered in the current field.

Note You do not need to fill in the *Aperture list* field for each Gerber file specified. If a *Aperture List* field is left blank, it will assume the contents of the previous *Aperture List* field. If the *Aperture List* field for the first specified layer is blank then the currently configured default aperture list file will be assumed.

After loading, changing the *Path* field will cause GerbTool to mark all Gerber and aperture list files as modified. This allows them to be saved in a different location then they were loaded from. Changing the contents of a *Filename* field after a design is loaded will cause that layer to be marked as modified. This allows you to save a layer under a new filename. If you enter a filename into a previously empty *Filename* field, GerbTool will attempt to load the newly specified Gerber file. If it does not exist, you will be given an opportunity to create it. Changing the contents of an *Aperture List* field will cause GerbTool to load the specified aperture list, if it is not already loaded, and link it to the corresponding Gerber file.

Regardless of whether you are loading or not, the *Visibility* button controls the visibility of the specified layer, the *Flash* and *Draw* color buttons control the color of flashes and draws respectively and the *Layer Type* button displays a menu of layer types that you can choose from.

Note It is important to specify which *Layer type* each layer is, as several GerbTool commands check this field for the proper type before processing each layer. For example, the *Tools/Pad Removal* command will only operate on layers with a type of *Inner*.

Following is a description of each field within the *Layers/Edit* form.

Path

Path to the directory where Gerber and aperture list files will be found.

Cut, paste above, and paste below

The ***Cut***, ***Paste Above*** and ***Paste Below*** buttons allow you to re-order the layer structure both before and after a design is loaded. In addition, if you cut a layer from a loaded design without pasting the layer, you will be prompted if you would like to unload that layer from memory. This allows you to free up memory if your resources become low.

Layer

Current layer. To make a layer current, click on the layer within the scrollable layer list.

Filename

Filename of a Gerber file to be loaded into current layer. If an explicit path is not included in the provided filename, the contents of the *Path* field will be prefixed to this filename before attempting to open the file.

Layer name

Used by 274-X format files to specify a composite layer name. This is **NOT** a filename.

Aperture list

Filename of aperture list to be associated with the current layer.

Visibility

Controls the visibility of the specified layer. Options are On, Off or Ref.

Color

Color buttons control the color of flashes and draws respectively.

Type

Allows you to specify a layer type of *Top*, *Inner*, *Bottom*, *Plane*, *Composite* or *Other*.

Note It is important to specify which Layer Type each layer is, as several GerbTool commands check this field for the proper type before processing each layer. For example, the *Tools/Pad Removal* command will only operate on layers with a type of *Inner*.

Key

Used to define polarity and link layers together to form composites. Enter D# or C#. D indicates Dark (positive), C indicates Clear (negative), and # is a numeric value. Layers with similar key numbers will be linked together to form a composite.

File format

The *File Format* buttons give you the opportunity to specify the correct data format BEFORE loading begins. With these buttons, you can edit the selected layers format, whether global or local, add/remove local formats, and automatically detect the format of one or more layers.

The ***Edit*** button allows you to edit the file format of the selected layer. If the selected layer has had a local format added (the *File Format* ***Edit*** button will have Local to its left), the format displayed for editing will be specific to the selected layer. Otherwise, the global format will be displayed for editing (See the *Files/Format* command in *Chapter 7: Command reference* for more information on editing file formats.)

The ***Add Local*** button adds a local format to the currently selected layer, which allows you to specify that the layer has a different format than other layers of the same file type. By default, each layer references a global format common to all layers of a particular type (e.g., Gerber). You can use local formats to load different file types into the same design. This allows you to simultaneously view and edit any files in the same design regardless of their file type.

The ***Del Local*** button allows you to remove a local format.

The ***Detect*** button will automatically detect the file format of the selected layer and update the format associated with the selected layer.

View composites

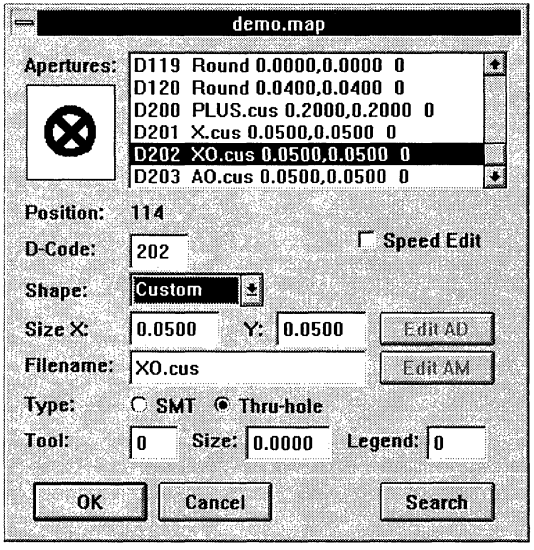
Enables the correct viewing of composite layers. When this button is checked the polarity of each layer, specified by the *Key* field, will be honored. If a layer is specified as Clear, all data in that layer will be displayed with the current background color.

Apertures menu

The Apertures command displays the Edit, Load, Unload, Report, Merge, Compact, Convert, and Save commands, which are described in the following sections.

Edit

This command allows you to edit a previously loaded aperture list. A list of currently loaded aperture lists, if any, will be displayed for you to choose from. The selected aperture list will be presented on the screen for you to edit.



Aperture editing form.

If, after editing an aperture list, you decide you want to discard the changes you've made, you can click on the *Cancel* button or dismiss the editing window using the window manager. If, on the other hand, you like your changes and would like to keep them, at least temporarily, then click on *OK* and GerbTool will update the in memory copy of the aperture list. To save an aperture list to disk, you must use the *Apertures/Save* command.

The Aperture editing form contains two sections; a list for selecting apertures to edit and the actual editable fields. The Position field is for reference only and is not editable. The nine remaining fields are:

Field	*Possible values*
D-Code	10 - 4095
Shape	Round, Square, Rectangle, Oblong, Donut, Diamond, Octagon, Thermal, Therm45, Target, Complex, Custom
Size X	0.0 - 9.9999
Size Y	0.0 - 9.9999
Filename	Custom aperture filename or aperture macro
Type	Surface-mount or through-hole
Tool	0 - 999
Tool Size	0.0 - 9.9999
Legend	0 - 4095

D-Code

Normally, you select a D-Code from the apertures list, but you can change this field to add new apertures.

Shape

Click on the shape you want. If you click on Custom, the *Filename* field will become available for you to specify the filename of the custom aperture (see *Chapter 11: Using custom apertures*). GerbTool will automatically add the required .CUS extension, if needed, when loading the custom aperture. If you click on Complex, the *Filename* field will become available for you to specify a valid aperture macro name.

Size X/Size Y

When editing the *Size X* field, if the *Size Y* field contains 0.0, then it will be set to the value of the *X* field.

Filename

If the current aperture shape is Custom, enter a filename of a custom aperture file. You can use a wildcard to invoke the file chooser. If the current aperture shape is Complex, enter a valid aperture macro name.

Type

This field specifies whether the D-Code represents a surface mount or through-hole pad. This information is needed when building multilayer netlists (see *Chapter 7: Command reference*).

Tool

You will need to edit this field if you intend to extract NC Drill information from a layer, or merge a NC Drill file into a layer, using this aperture list.

Size

This field specifies the size of the tool indicated in the *Tool* field.

Legend

You can enter a D-Code that will be used to represent this tool in a drill legend. This field is used when creating a drill drawing using the *Tools/Drill/Drawing* command.

Speed edit

If the **Speed Edit** check box is selected, GerbTool changes the operation of this dialog box to make it easier to rapidly enter aperture lists manually. Normally, when editing an aperture list, touching the ENTER key updates the current aperture and advances to the next aperture. When the end of the aperture list is reached, new apertures are added to the list automatically. Moving from field to field is accomplished using the TAB key or mouse. When **Speed Edit** is selected, only the Shape and X/Y size fields are active. Furthermore, touching the ENTER key moves from field to field except for the Y size field. Touching the ENTER key while editing the Y size field advances to the next record, as usual, before moving to the Shape field. This change in operation allows fast aperture list creation using only the ENTER key to move from field to field and to advance to the next record.

Edit AD and Edit AM

The *Edit AD* and *Edit AM* buttons are only active if the shape is Complex. These buttons allow you to edit the 274-X aperture definition (AD) and the aperture macro (AM) respectively. For FIRE9xxx aperture lists, the *Edit AD* button allows you to edit an aperture definition in native FIRE9xxx format.

Search

The **Search** button allows you to search for an aperture that contains the text string you specify. Any text appearing in the scrollable aperture list can be searched for. For example, you could enter `D200` to find that particular D-Code or you could enter `rect` to find the next occurrence of a Rectangular aperture. You could also enter `.05` to find the next occurrence of a 50-mils aperture. The search text is remembered between uses and the search cycles through the aperture list so you can continue to search forward repeatedly.

Load

Select this menu item when you need to load or create an aperture list. You will be prompted with the file chooser. You can use a wildcard specification to obtain a list of files from which to choose. If the specified aperture list doesn't exist, you will be given an opportunity to create a new one. If creating a new aperture list, you will then be asked for the highest D-Code expected. GerbTool will create an aperture list on disk, using default values, then load it. If you are loading an existing aperture list, GerbTool loads the specified aperture list.

Note You can load and edit aperture lists independently from a design.

Unload

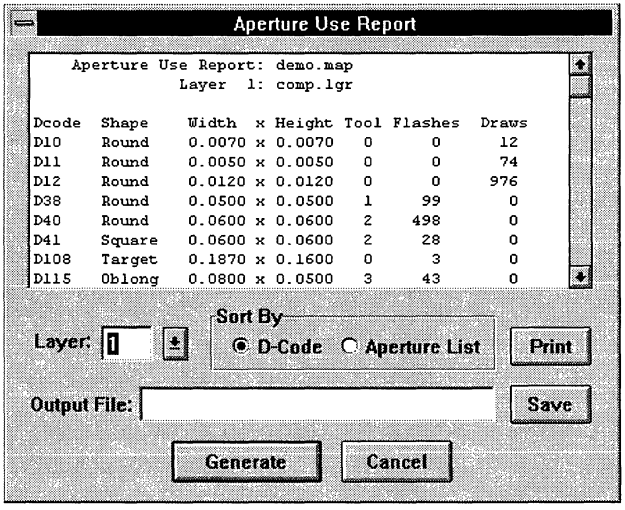
You can use this command to remove a previously loaded aperture list. A list of loaded aperture lists will be presented for you to choose from. The selected aperture list, if not required by the currently loaded design, will be removed from within GerbTool. If the aperture list has been modified and not saved to disk, you will be prompted to do so.

Report

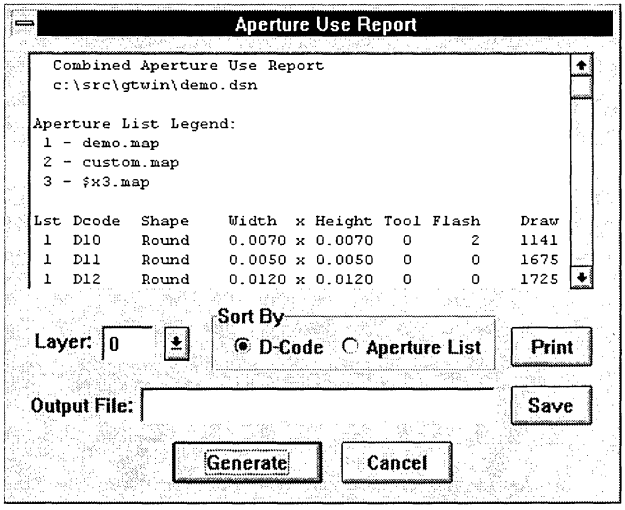
Select this command when you want an *Aperture Use Report.* An aperture use report details which D-Codes, along with their definitions, are being used on a per-layer basis. Included in the report are usage counts for both flashes and draws.

Note If an aperture has an unknown shape, or is sized at zero, it will be highlighted for easy recognition.

Each time the *Generate* button is pressed a report will be displayed for the layer specified in the *Layer* field. Entering a zero in the *Layer* field will instruct GerbTool to generate a *Combined Aperture Use Report* for all loaded layers. You can use the scroll bar to view all of the report if it does not fit entirely within the window. You can also edit the report to add or delete title information. You can print the report using the *Print* button or save the report to a file for later printing by entering a filename in the *Output File* field and choosing the *Save* button.



Single-layer Aperture Use Report form.



Combined-layer Aperture Use Report form.

Merge

This command allows you to merge two or more loaded aperture lists into one. All aperture lists associated with the currently viewed layers will be merged into a new aperture list. Each layer will then be associated with the new aperture list and the D-Codes of each layer will be remapped accordingly.

Note It is important that the new aperture list be saved if any of the remapped layers are saved.

Compact

This command allows you to remove unused and redundant apertures within an aperture list. Select an aperture list to compact. Each layer associated with the selected aperture list is then re-associated with the new aperture list and the D-Codes remapped accordingly.

Note It is important that the new aperture list be saved if any of the remapped layers are saved.

Convert

GerbTool has the ability to convert most CAD and photo plotter aperture list formats directly into GerbTool format (See *Chapter 3: Quick start* for a complete list.)

You can specify an input filename and select the appropriate converter using the pull-down list.

Note There are two ways to use wildcards in the *Filename* field. If you enter a wildcard followed by the ENTER key, the file chooser will be presented and you can locate the file you want. If a wildcard is entered but you **do not** touch the ENTER key, the wildcard will be passed to the converter. This allows you to convert many files at one time, if needed.

The GerbTool aperture list(s) created by the *Convert* command will be named FILENAME.MAP.

See For information about adding additional aperture list converters to GerbTool, see *Chapter 2: Configuration*. For information on creating your own aperture list converters, see *Chapter 9: Aperture Conversion Rule files*.

Save

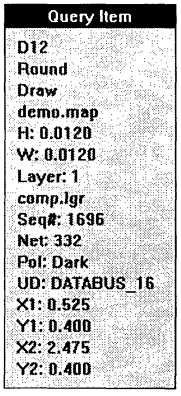
Use this command to optionally save any modified aperture lists.

Query menu

The Query command displays the Item Info, Measure, Highlight, Copper, and Extents commands, which are described in the following sections.

Item information

The Item Info command allows you to obtain information on individual items within the database. As you cycle through the database selecting items, each selected item is highlighted and its D-Code definition, along with its X-Y location and other information, are displayed in a form as shown below:



Item Info form.

You can select items either by clicking directly on a database item or you can use the N key to automatically advance to the next sequential item in the database.

Measure

The Measure command displays the Point to Point and Edge to Edge commands, which are described in the following sections.

Point to point

Use this command to obtain accurate measurements of your data. GerbTool first prompts for a base point to measure from. As you move the cursor away form the base point the distance in X and Y as well as true length will be displayed in the prompt area. A left click will change the base point to the current cursor position.

Edge to edge

This command measures the actual minimum distance between two Gerber data items. GerbTool first prompts for you to select a base item. As you select additional items, the actual minimum distance between items in X and Y as well as true length will be displayed in the prompt area.

Highlight

The Highlight command displays the D-Code, Net, and Off commands, which are described in the following sections.

D-Code

Use this command to highlight all occurrences of a specified D-Code. You can restrict your selection to flashes, draws or both and a particular layer. The selected D-Codes remain highlighted until you turn off the highlight with the nested command H or you select another group of items with this or any other command.

Net

This command allows you to highlight true multilayer nets by pointing and clicking anywhere on a line segment or flash. All viewed items in the chosen net are highlighted and remain so until canceled with a right mouse click or escape key. At any time, you can change the color used for highlighting subsequent nets by typing ALT+C.

You can also inform GerbTool that you would like to exit this command and leave the currently selected nets highlighted by typing ALT+X. You can then toggle those nets highlighted on and off with the nested command H, or cancel the highlights with the *Query/Highlight/Off* command.

Note This command relies on the netlist information supplied by a previous invocation of the *Tools/Netlist* command. If netlist information does not yet exist you will be prompted whether to create one.

Off

This command will turn off any current highlights.

Copper

This command will accurately calculate the amount of copper used on a layer using a high resolution bitmap method. All visible layers will be scanned.

Extents

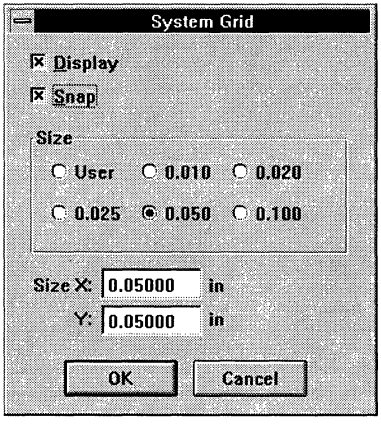
Use this command to determine the data extents of all layers loaded. In addition to displaying the extents information, GerbTool also updates its internal data extent information. This will allow the *View/All* command to correctly center the data after you've made edits to the database.

Options menu

The Options command displays the Grid, Ortho, Sketch, Overlay, Key Cmds, Defaults, Film Box, Bg Color, Show Errs, Undo, Arcs 360, Status, Metric, and Save commands, which are described in the following sections.

Grid

This command displays the system grid form, as shown below.



System Grid form.

You can turn the display of the grid on/off, toggle grid snapping on/off, as well as specifying the grid size. You can select a built-in grid size or, by entering a value in the *Size X/Y* field, you can specify a non-standard grid size.

Note This command is also available as the nested command CTRL+G. See *Chapter 4: GerbTool basics* for more information about nested commands.

Ortho

This command allows you to toggle orthogonal snap mode on/off and specify a snap angle. When enabled, all lines drawn interactively will be forced to the specified angle.

Note The current setting can be temporarily overridden by holding down the CTRL key.

Sketch

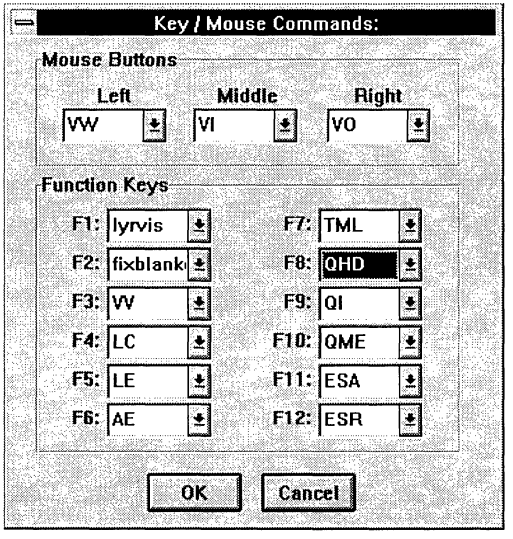
This command toggles *Sketch* mode on/off. When sketch mode is enabled, pads are shown with an outline only, and traces are displayed as a single thin line. Besides speeding up redraw times considerably, this mode can also help you spot stacked pads. A check mark is placed to the left of the menu entry and the *Settings Sk* button appears depressed when this mode is enabled.

Overlay

This command toggles *Overlay* mode on/off. When overlay mode is enabled, items become transparent when drawn atop each other. When disabled, items obscure whatever was previously drawn. Overlay mode makes it easier to spot stacked pads. A check mark is placed to the left of the menu entry, and the *Settings Ov* button appears depressed when this mode is enabled.

Key commands

The Key Cmds menu selection displays a form showing the current mouse button and function key assignments.



Mouse/function key assignment form.

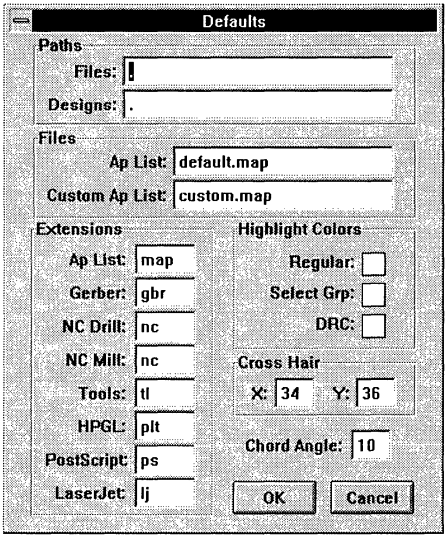
You can change any of the commands assigned to the mouse and function keys by editing the corresponding fields within this form. Any changes you make will become effective immediately after choosing the *OK* button. To make your changes permanent, use the *Options/Save* command. This will save the current key assignments in a GerbTool configuration file.

See For a list of command IDs available for mouse button and function key assignments, see *Appendix A: Command ID values*.

Note In addition to command IDs, function keys can also be programmed with GerbTool macros, allowing virtually all of GerbTool's power to be within one keystroke.

Defaults

This command displays a form containing fields for program default values.



Defaults editing form.

Paths

The **Files** field specifies where GerbTool should look for data files, other than design files, if no other directory is specified.

The **Designs** field specifies where GerbTool should look for design files if no other directory is specified.

Files

The **Ap List** field specifies the default aperture list file that GerbTool will load if no other aperture list is specified.

The **Custom Ap List** field specifies the aperture list used by all custom aperture files (see *Chapter 11: Using custom apertures*).

Extensions

The **Ap List** extension field indicates the default filename extension to be used when dealing with aperture list (map) files.

The **Gerber**, **NC Drill**, **Tools**, **HPGL**, **PostScript** and **LaserJet** extension fields indicate the default filename extensions to be used when dealing with Gerber, NC Drill, Tool, HPGL, PostScript and LaserJet files respectively.

Highlight colors

The **Highlight Colors** buttons allow you to control the colors used when highlighting database items.

Crosshair

The **X** and **Y** fields provide control over the size of the drawing area crosshair cursor. Enter 0,0 for a full screen cursor.

Chord angle

The **Chord Angle** field allows you to specify the chord angle used when creating segmented arcs using editing commands. For example, a chord angle of 5° would result in a 18 separate line segments for a 90° arc.

Film box

This menu selection displays a form containing the *Film Box* size and color. You can change the current film box size, by editing the *X-Size*, *Y-Size* fields, and/or the film box color by choosing the *Color* button.

Background color

The Bg Color command provides the ability to change the *Drawing Area* background color. As with all color buttons within GerbTool, click on the color button for a list of available colors. (See *Chapter 2: Configuration* for a description of how to change the available colors.)

Show errors

The Show Errs command toggles the display of rule violation errors on/off. After executing DRC or Snoman, any rule violation errors are shown highlighted. These highlighted items are displayed indefinitely until you reload or run DRC or Snoman again. If you no longer need to see the highlighted rule violation errors, you can use this command to disable their display. A check mark is placed to the left of the menu entry and the *Settings Er* button appears depressed when enabled.

Undo

This command toggles the saving of undo information on/off. If undo is currently disabled, it will become enabled and a check mark is placed to the left of the menu entry and the *Settings Un* button will appear depressed. If undo is currently enabled, any current undo information will be destroyed and undo will then be disabled.

Arcs 360

This command toggles the method of creating arcs used by the *Edit/Add/Arc Ctr*, *Edit/Add/Arc 3 Pt* and *Edit/Add/Circle* commands. If enabled all arcs will be created using 360° circular interpolation. If disabled, all arcs will be created using small line segments. This does NOT affect the way Gerber data is read from a disk file. It only pertains to adding new arcs with the above mentioned commands.

Note Not all photoplotters support circular interpolation.

Status

This menu selection toggles the status bar display on/off. To increase the size of the drawing area you can turn off the status bar display. Selecting this command again will re-display the status bar.

Metric

This menu item toggles *metric* mode on/off. When metric mode is enabled, all information and editing fields within GerbTool that represent sizes and distances (i.e., coordinates) will be shown in metric format.

Save

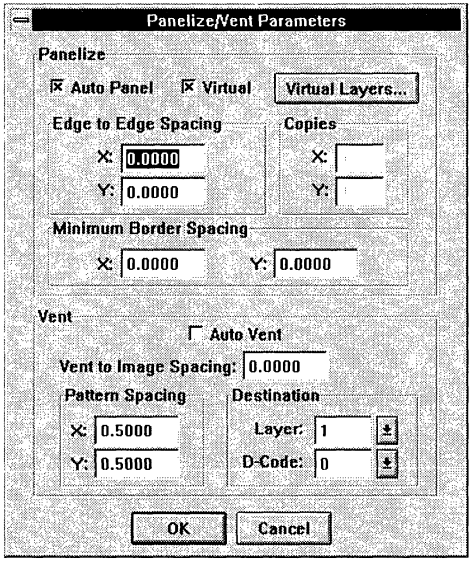
This command allows the current program environment, including defaults, to be saved in the GerbTool configuration file (see *Chapter 2: Configuration*). This file is loaded upon program startup.

Tools menu

The Tools command displays the Panelize, DRC, Snoman, Netlist, Pad Removal, NC Drill, Vent, Convert, Lyr Spread, Fix SS, and Macros commands, which are described in the following sections.

Panelize

The *Panelize* command is used to create multiple (array) copies of a design. This allows multiple copies of the design to be manufactured as one panel.



Panelize editing form.

Automatic panelization

To panelize an image, locate the data you want within the Film Box (see *Edit/Origin*), view the layers that are to be panelized, and enter the image border to border spacing you want in the X and Y fields of the **edge to edge** spacing group box within the *Panelize* editing form.

Note Although only visible layers will be copied, all layers of the original image will remain aligned after panelization.

Manual panelization

To panelize manually, remove the check mark from the *Auto* check button if needed. You must also enter the number of rows and columns in the appropriate fields as well as the **point to point** distance between copies.

Automatic venting

Automatic venting occurs during panelization, whether automatic or not. To vent a panel automatically, check the *Auto Vent* button within the *Panelize* editing form. You can also define the spacing between the image data extents and the venting area with the *Vent/Image Spacing* field, specify the D-Code and spacing between the flashes in the vent pattern and the layer to add the vent pattern to.

Tip In both automatic and manual venting, the style of vent pattern can use custom apertures. For example, you could create a hatch or cross-hatch pattern using a diagonal or cross-shaped custom aperture. Be sure to set the height and width of the overall size of the custom aperture in the aperture list.

Virtual panelization

Enabling the *Virtual* button within the *Panelize* editing form allows GerbTool to panelize your design without actually duplicating layer data.

Note Although no data is duplicated during virtual panelization, the data origin is modified to center the images within the panel. Therefore, it is still necessary to save your design after panelization.

Virtual panelization provides many benefits including automatic updating of all images during edits and drastically reduced file sizes. Furthermore, if your designs are to be plotted on a 274-X, FIRE9xxx or EIE compatible plotter, GerbTool will automatically insert the proper step and repeat codes into your Gerber data.

Note If your designs are plotted on a plotter that does not support step and repeat codes, you must execute the *Tools/Panelize* command without the *Virtual* button enabled and save your panelized Gerber files before you send them to the plotter.

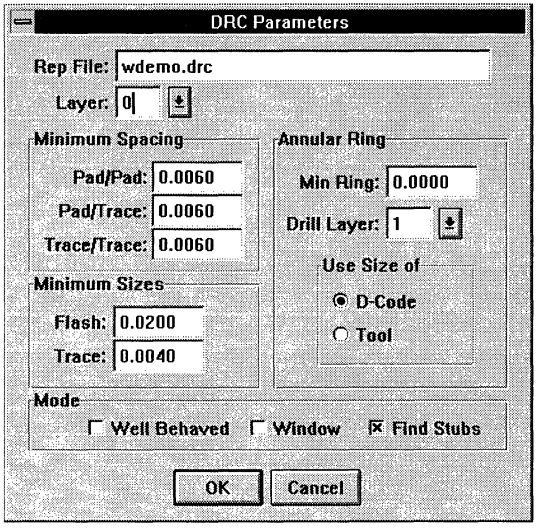
GerbTool will also insert step and repeat codes into NC Drill output data if the *Virtual* button is enabled. This may be necessary to drill large panels if your NC equipment is memory limited.

Virtual panel mode and hence the display of virtual panels can be toggled on/off using the nested command CTRL+V. See *Chapter 4: GerbTool basics* for more information about nested commands.

Choosing the ***Virtual Layers*** button will display a list of loaded layers so that you can choose which layers are to be included during virtual panelization.

DRC

Selecting this menu item invokes the DRC tool. The DRC tool is available to verify that your design meets minimum item-to-item spacing requirements.



DRC editing form.

The DRC form is presented with the current active layer in the *Layer* field. You can override this by entering another layer.

Note If you enter a zero in the *Layer* field, all viewed layers will be processed.

Enter a valid filename in the *Rep File* field and all errors will be logged to this file. Edit the spacing parameters according to your needs. Fields for pad to pad, pad to trace, trace to trace, and minimum flash/trace sizes are provided. Optionally, you can specify a minimum Annular ring required and corresponding drill layer. If either the annular ring size or drill layer are zero, no annular ring check will be performed. The annular ring check compares the DRC layer to the drill layer, with the assumption that the drill layer will normally contain a flash at each pad location using a smaller size than the DRC layer. You can also specify whether the size of the drill layer flashes are taken from the D-Code size or the Tool size.

Tip You can use the annular ring check to verify a soldermask layer also.

A through-hole pad that does not have a corresponding drill flash will be reported as a "missing" drill.

The *Find Stubs* check button will allow GerbTool to locate and highlight all trace stubs. A trace stub is defined as any trace that touches a pad or trace on one end, but does not on the opposite end.

You can optionally select window mode to run DRC on a window of data versus the complete layer.

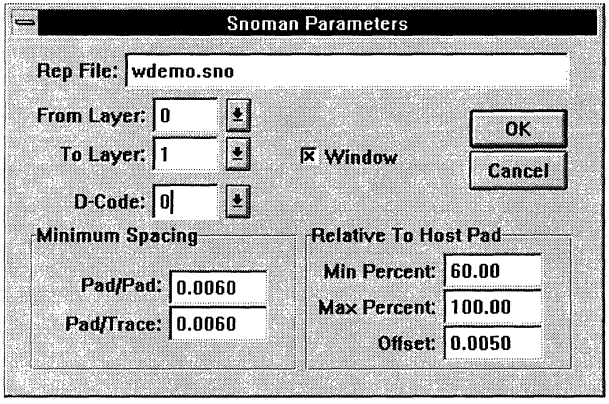
The DRC command supports two separate modes: well-behaved and normal. In the well-behaved mode, GerbTool assumes that legal pad/trace or trace/trace connections will have common X-Y locations (see *Chapter 7: Command reference* for a description of well-behaved Gerber files). This means that ANY actual contact between items that don't share a common X-Y location, and are in different nets, will be considered a violation. Conversely, in normal mode, any actual contact between items will not be considered a violation. Only items that are not in contact but are within the minimum spacing rules will be considered in violation. The well-behaved mode is preferred.

Note If a valid netlist does not already exist, you will be prompted whether to generate one now. While a netlist is not a prerequisite to DRC, a netlist increases the usefulness and correctness of DRC.

Use the *View/Errors* command (see *Chapter 7: Command reference*) to view rule violation errors, if any, after executing this command.

Snoman

This menu selection will invoke the Snoman tool. The Snoman tool will create a *maximum material condition* at the point of trace entry into a pad. See *Appendix D: Snoman concepts* for a more technical description.



Snoman editing form.

Enter a valid filename in the *Rep File* field as any errors will be logged to this file. You must specify a layer to operate on (*From Layer*) as well as an output layer (*To Layer*) for the generated Snoman pads.

Note If you enter a zero in the *From Layer* field, all viewed layers will be processed, with the resultant Snoman pads being added to their respective layers.

You can restrict the generation of Snoman pads to a particular D-Code by entering a D-Code in the *D-Code* field. A D-Code of zero matches all. Edit the spacing parameters to specify the design rules that Snoman must adhere to. The *Host Offset* field contains the offset maintained between the host pad centroid and the edge of the generated Snoman pad. This value may be negative. If Snoman detects a spacing rule violation while placing a Snoman pad, it will reduce the size of the Snoman pad to avoid such errors. You can control to what percentage of the host pad size that Snoman can reduce the size of the Snoman pad. Use the *Min Percent* field to specify this value. The *Max Percent* field allows you to control the maximum size of the generated Snoman pad as a percentage of the host pad size. You can also indicate whether Snoman should operate on a window of data versus a complete layer.

Note If a valid netlist does not already exist you will be prompted whether to generate one now. A netlist is required for the Snoman tool to work properly.

Use the *View/Errors* command to view potential rule violation errors, if any, after executing this command.

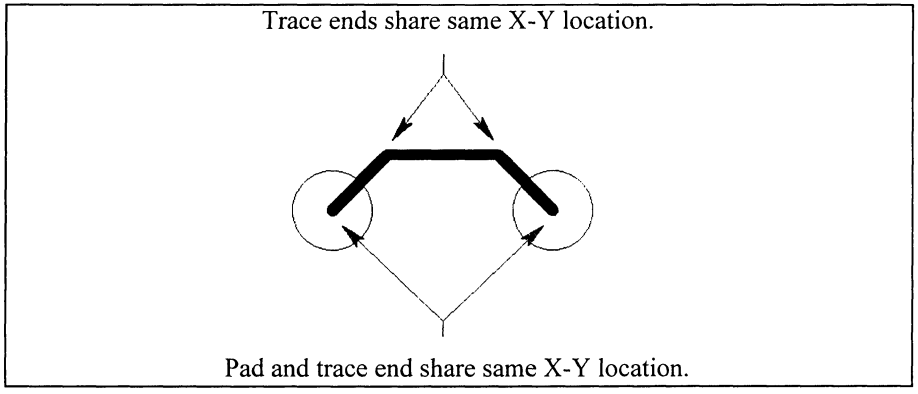
Netlist

The Netlist command displays the Generate and Write commands, which are described in the following sections.

Generate

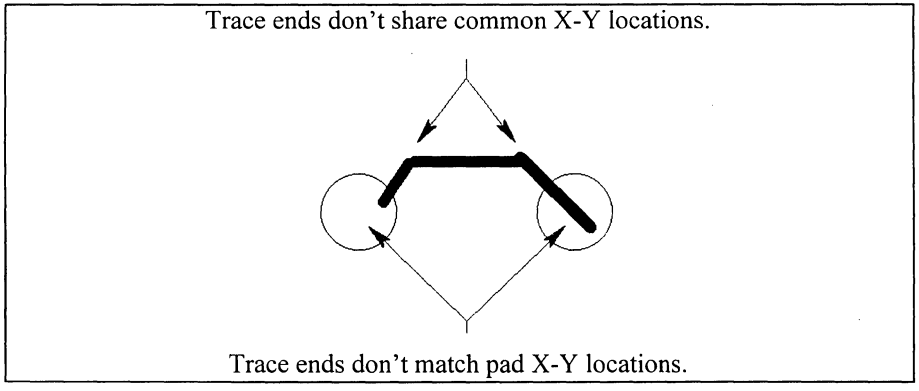
The *Generate* command will process all viewed layers and create a single multilayer netlist that becomes part of the internal database. The netlist can then be used by other GerbTool commands that require a netlist.

GerbTool allows you to indicate whether your database is well-behaved or not. A well-behaved Gerber file is defined as one where all items that are to be considered connected share a common X-Y location, as shown below:



Example of a well-behaved Gerber file.

The following is an example of a Gerber file that is NOT well-behaved:



Example of a Gerber file that is NOT well-behaved.

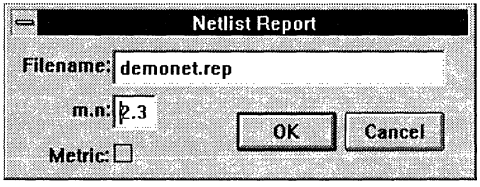
If you determine that your Gerber files are indeed well-behaved, it is recommended that you choose this mode when generating a netlist, as there is a dramatic increase in processing speed due to the well-behaved nature of the Gerber files.

Since so many of GerbTool's features require a netlist to perform properly, you can save the generated netlist within your Gerber files for later use. If netlist saving is enabled (see *Chapter 7: Command reference*), and a netlist is present, it will be saved when the layer is saved to disk. To remove a netlist from a Gerber file, load the layer (or layers), disable netlist saving using the *Files/Format* command and then save the necessary layers.

Note GerbTool uses the G04 command to embed a netlist within a Gerber file. This will cause the Gerber file to increase slightly in size. It is recommended that netlists be removed as described above before submitting your files to be photoplotted, due to their increased size and the possibility of the photoplot equipment not properly recognizing the G04 command.

Write

This command will generate an ASCII netlist file consisting of pad X-Y coordinates.



Netlist editing form.

This command creates one netlist for all viewed layers. You can specify the output file m.n values and whether you want metric output. You specify an output filename to which GerbTool will write the netlist.

Note If a valid netlist does not already exist you will be prompted whether to generate one now. A netlist is required for this command to work properly.

Pad removal

The Pad Removal command displays the Isolated and Stacked commands, which are described in the following sections.

Isolated

Selecting this command will remove any unused pads (isolated/floating pads) from your inner layers.

Note Only layers with a layer type of *Inner* will be considered. Use the *Layers/Edit* command to change this if necessary.

GerbTool does not remove targets and/or thermal pads. You specify the layer to remove the pads from and whether you want window mode versus processing the entire layer.

Stacked

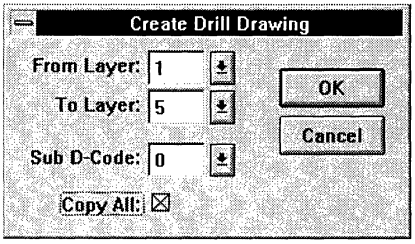
Selecting this command will remove any unnecessary pads that are identical and stacked exactly one on top of another on the same layer. You specify the layer to remove the pads from and whether you want window mode versus processing the entire layer.

NC Drill

The NC Drill command displays the Drawing and Write commands, which are described in the following sections.

Drawing

This command creates a drill drawing using the *Legend* field associated with each D-Code in a aperture list.

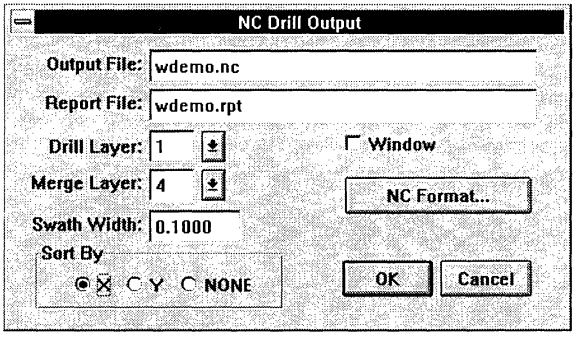


Create Drill Drawing form.

For each D-Code in the *From Layer*, the D-Code specified by the corresponding *Legend* field will be added to the *To Layer*. The *Copy All* option allows you to indicate whether D-Codes with an invalid *Legend* D-Code should be copied. If the *Copy All* option is enabled, the *Sub D-Code* field allows you to specify a particular D-Code to use as a replacement for invalid *Legend* D-Codes. If *Sub D-Code* is zero, all D-Codes with an invalid *Legend* D-Code will use the original D-Code value in the *To Layer*.

Write

The *NC Drill* command creates an ASCII output file containing X-Y pad locations in the selected NC format. The output is optimized and duplicate hits within a single tool are removed.



NC Drill editing form.

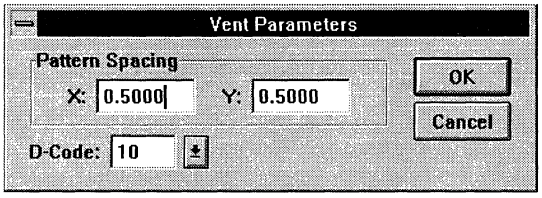
You specify the layer to operate on as well as the output filename and format. You can also specify window mode as well. This command relies on the *Tool* assignments within the aperture list assigned to the selected input layer. Optimization is controlled by the *Swath Width* value and secondarily by whether an X or Y first sort is performed. The report file contains an approximate distance that the drill head will travel. Therefore, by adjusting the swath width and examining the report file you can achieve the fastest drilling through put.

Note Panelization of the image should be performed prior to executing this command. If you perform a virtual panelization the output of this command will contain step and repeat codes. Step and repeat codes should only be used if your drilling equipment has limited memory capacity. Otherwise, a fully optimized non-virtual panel will result in more efficient drilling.

Occasionally there may be items that you don't want optimized, but do want included in the same drill file, such as test coupons and mounting holes. These items should be placed on a layer in the order that they should be drilled. This layer would then be entered into the *Merge Layer* field. If a valid layer number is entered in this field, its drilling information will be inserted into the drill file without optimization after inserting the optimized information from the layer specified in the *Layer* field. This is done on a tool by tool basis so that information for tool #1 on the drill layer will be sorted and then output, followed by the tool #1 information from the merge layer. The same will occur for tool #2, and so on. This also works when using Virtual panels and you want to include some drill data that is not panelized.

Vent

This command allows you to manually add Venting/Thieving patterns to your database. GerbTool will display the *Vent Parameters* editing form where you can edit the venting parameters such as pattern spacing and aperture selection.



Vent Parameters editing form.

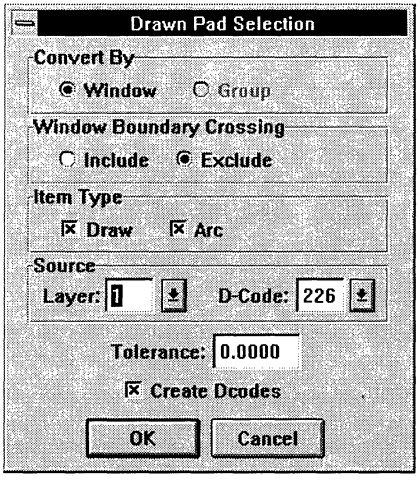
You can then define a rectangular area by entering two coordinate points. After confirmation, GerbTool will fill the specified area with a pattern of flashes as specified.

Convert

The Convert command displays the Drawn Pads and Circles commands, which are described in the following sections.

Drawn pads

Use this command to convert pads that are created with Gerber draws into flashes. This command should be used prior to attempting any other editing or data extraction such as NC Drill. This command may significantly decrease the size of your database if it contains drawn pads.

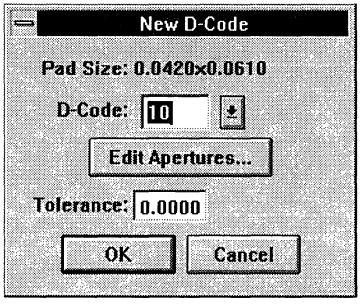


Drawn Pad conversion form.

The Drawn Pads command will prompt you to enter a window around a drawn pad that is to be converted.

If the *Create D-Codes* check button is *enabled*, GerbTool will create new D-Codes as necessary to match the dimensions of the drawn pads selected for conversion.

If the *Create D-Codes* check button is *disabled*, you will be informed of the calculated size of the pad as shown below:



Drawn pad replacement D-Code form.

Find or create a corresponding flash in the aperture list for this layer. Enter the appropriate D-Code in the *New D-Code* field and a tolerance value, if needed, in the *Tolerance* field. GerbTool will then locate and highlight all occurrences of any matching drawn pads and prompt you whether to continue.

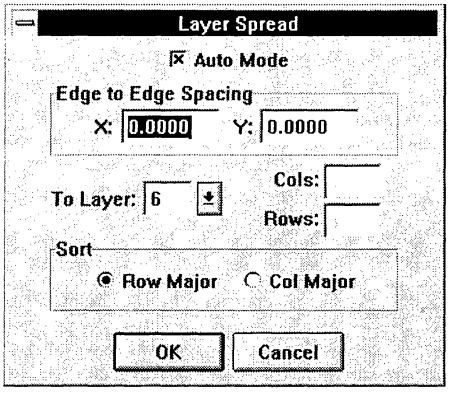
Note The tolerance value allows GerbTool to increase its match frequency when the CAD system that generated the drawn pads exhibits round off errors. Usually a value of 0.002 (inches) will suffice.

Circles

This command will convert circular interpolated circles into segmented circles individually or by window. Use this command if your photoplotter can't handle circular interpolated draws.

Layer spread

Use the Lyr Spread command to reduce your film costs by automatically copying and spreading all viewed layers onto one layer and thus one sheet of film.



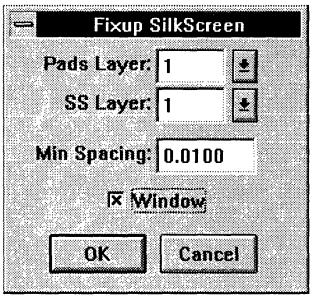
Layer Spread editing form.

You can select automatic or manual mode using the *Lyr Spread* editing form as shown above. If you select *Auto Mode* then GerbTool will automatically calculate how many images will fit in the film box as well as the position of each image. In auto mode the X and Y spacing fields specify the opposing border-to-border minimum spacing requirements. In manual mode, you must specify the number of rows and columns and the center to center spacing in the X and Y spacing fields. In either case, you can select either row major or column major placement. While the *To Layer* field may specify one of the layers to be spread, it usually is an empty layer created to accept the properly spread out images.

After choosing the *Lyr Spread* editing form *OK* button, you will be prompted to select the order in which the layers are spread. You must click on each layer to define the proper order. After doing so, the placement of all layers will be shown for your approval. If you respond affirmatively, the layers will be copied and spread as shown.

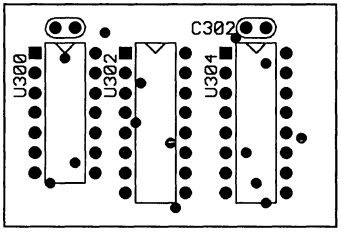
Fix silkscreen

The Fix SS command will automatically move silkscreen data away from pads.

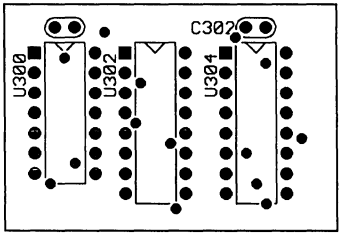


Fixup Silkscreen editing form.

You specify the layer that contains the pads (*Pads Layer*), the layer that contains the silkscreen data (*SS Layer*), a minimum spacing to be maintained, and finally whether you want window mode. GerbTool will then automatically move lines segments that violate the minimum spacing requirement as shown in the following before and after sequence:



Before the Tools/Fix SS command.



After the Tools/Fix SS command.

Macros

The Macros command displays the Load and Run commands, which are described in the following sections.

Load

Use this command to load additional macro files into GerbTool. This allows any macros present in the specified file to be included in GerbTool's list of available macros.

Run

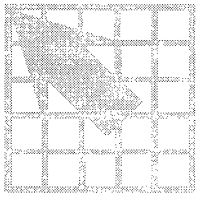
This command will prompt you to select a macro to run. All macros loaded at program startup and through the *Tools/Macros/Load* command will be available for execution.

User menu

The *User* main menu selection will display a menu of commands that are configured through a GerbTool configuration file. The purpose of the *User* menu is to allow you to make your favorite macros and commands as accessible and easy to use as any other GerbTool command. Between the *User* menu and programmable mouse/function keys (see *Chapter 7: Command reference*) the commands you use the most can be truly a keystroke or mouse click away.

See For details on configuring the *User* menu, see *Chapter 2: Configuration.*

Chapter 8



Macros

GerbTool provides a powerful macro command language that allows you to create new commands to accomplish everything from simplifying repetitive tasks to implementing entirely new functions. The macro language provides the ability to make decisions, repetitively execute a group of commands, scan the database, prompt the user for data and more. This chapter details the command language used in creating a GerbTool macro.

Creating a macro

Macros are created using any text editor that supports plain ASCII text.

Note If you use a word processor you may have to specifically save your macro file in ASCII mode.

A macro file can contain multiple macros, each with a name up to sixteen characters long. If a macro is defined more than once, only the last occurrence of the macro is visible to GerbTool. While any number of macro files can be loaded, GerbTool allows a maximum of 1024 macros to be defined.

Each macro consists of a MACRO statement followed by one or more macro language commands or statements and terminated by and END statement as shown below:

```
MACRO testmacro
.
.
.
ENDMACRO
```

Using variables

A variable is a way of saving a piece of information, such as a database coordinate, for later use. Macros allow you to define variables with names up to 32 characters long. You can use a variable anywhere a number or text string is expected within a macro. To use a variable, prefix a symbolic name with a $, as shown in the following example:

```
GETPOINT   "Enter New Coord", $XCOORD,$YCOORD
CALC       $XCOORD = $YCOORD * 2.0
ADDFLASH   $XCOORD,$YCOORD + 0.5
```

GerbTool automatically defines many system level variables that provide basic program information, such as the number of layers configured, and variables that contain the specific results of certain macro commands after they have been executed. All system level variables are defined with an additional leading dollar sign ($) character, such as $$STATUS.

Note The system variable $$STATUS is set by most commands, indicating the success or failure of the command. Positive values indicate success; negative failure. In some cases the value may indicate a count, such as in the *Edit/Copy* command.

A list of system level variables is shown below.

$$ACTIVELAYER	$$GROUPMODE	$$PLATFORM
$$CALLDEPTH	$$ITEMMODE	$$RECTANGLE
$$COMPLEX	$$LAYERMODE	$$ROUND
$$CURRENTDCODE	$$MAXLAYERS	$$SELGRPCNT
$$CUSTOM	$$MODEL	$$SQUARE
$$DIAMOND	$$NO	$$STATUS
$$DONUT	$$OBLONG	$$TARGET
$$DRAW	$$OCTAGON	$$THERMAL
$$DRILLCOUNT	$$PANELCOLS	$$THERM45
$$DRILLEFTRAVEL	$$PANELROWS	$$TRUE
$$DRILLTRAVEL	$$PANELXOFF	$$VERSION
$$DSNNAME	$$PANELXSPACING	$$WINDOWMODE
$$FALSE	$$PANELYOFF	$$YES
$$FLASH	$$PANELYSPACING	

You can also create arrays of variables using the ARRAY and STRARRAY functions. An array is a list of variables referenced through a single variable name and an index. Array indexes can be any expression including another array variable. For example:

```
STRARRY $names(3)
STRCPY $names(1), "Bob"
STRCPY $names(2), "Betty"
STRCPY $names(3), "Jessie"
```

Coordinate lists

For added ease of use, you can specify a range of array indexes when using numeric arrays in coordinate lists submitted to a macro command. For example, ADDFLASH $xy(1,4) would be equivalent to ADDFLASH $xy(1),$xy(2),$xy(3),$xy(4). This is especially useful when using variables as indexes.

Coordinate lists also support both *absolute* and *relative* coordinate modes. The default coordinate mode is absolute. To change to relative mode, prefix a coordinate with an R. Once turned on this way, all coordinates following will be relative to the previous coordinate. You can turn relative mode off by prefixing a coordinate with an A. The specified mode only applies to the command in which it was used.

For example:

```
ADDDRAW $xs,$ys,R0.1,0,0,0.1,-0.1,0,A$xs,$ys
```

In the above example the ADDDRAW command is provided an initial absolute coordinate followed by three relative coordinates and finally a absolute coordinate.

Repeating blocks of commands

Macros allow you to repeat a series of commands until an event occurs that terminates the loop. An example of the REPEAT statement is shown below.

```
REPEAT  $cnt > 0
        ...will be executed as long as $cnt > 0...
END
```

The above example will repetitively execute the commands between the REPEAT and END statements until the variable $cnt is less than or equal to zero. Nested REPEAT blocks are also allowed, providing powerful looping capabilities.

Making decisions

Macros provide the ability to make decisions as to which commands are executed within a macro. The IF statement allows such control, and is exemplified below:

```
IF  $shape == $$RECTANGLE
      ...will be executed if above test is TRUE...
END
IF  $shape == $$RECTANGLE
      ...will be executed if above test is TRUE...
ELSE
      ...will be executed if above test is FALSE...
END
```

Combining IF statements within REPEAT blocks provides virtually unlimited macro programming possibilities.

Loading macros

There are two methods of loading macros:

- You can load a macro file using the *Tools/Macro/Load* command. This command allows you to load a macro file after GerbTool is up and running.
- To load one or more macro files every time GerbTool starts, you can add one or more MACRO_FILE=filename parameters to your GerbTool configuration file.

See For information on configuration files, see *Chapter 2: Configuration.*

Running macros

There are four methods of running a macro:

- The first method is to use the normal menu and select the *Tools/Macro/Run* command. This command allows you to choose a previously loaded macro for execution.
- The second, easier method is to type the macro nested command M.
- The third, and still easier method, is to assign a macro to the *User* menu using the USERMENU configuration parameter detailed in *Appendix B: Configuration files*.
- The fourth, and easiest method, is to assign a macro to a function key using the *Options/KeyCmds* command. This allows you to run a macro with a single keystroke.

Macro language reference

This section describes each macro command and statement, and the parameters expected for each.

Conventions used

...	Three dots (an ellipsis) indicate that additional similar parameters are allowed.
[x, y]	Items inside square brackets indicate optional parameters.
<a | b>	Angle brackets and vertical bars indicate a choice among two or more items.
literal	A literal numeric or string value, such as `14.125` or `Yes`.
variable	A numeric or string value stored in a variable, such as `$XCOORD` or `$REFDES`.
numvar	A numeric variable.
strvar	A string variable.

operator	One of the following mathematical or comparison operators:
	`+ addition`
	`- subtraction`
	`/ division`
	`* multiplication`
	`== equal`
	`!= not equal`
	`< less than`
	`> greater than`
	`<= less than or equal`
	`>= greater than or equal`

exp	Numeric expression of the form: <literal | numvar> [operator <literal | numvar>]
yesno	`<"Yes" | "No" | $$YES \| $$NO>`
\	The backward slash can be used as the last character on a line to indicate that a long command is to continue on the next line.
#	This character denotes that the remainder of the line is a comment and will be ignored.

Add functions

ADD3PTARC

Purpose	Allows the user to enter arcs by specifying two end points and a point on its circumference.
Menu command	*Edit/Add/Arc 3 Pt*
Syntax	`ADD3PTARC [x1, y1, x2, y2, x3, y3]...`
Parameters	
`x1`	The x coordinate of the first end point of the arc.
`y1`	The y coordinate of the first end point of the arc.
`x2`	The x coordinate of the second end point of the arc.
`y2`	The y coordinate of the second end point of the arc.
`x3`	The x coordinate of a point on the circumference of the arc.
`y3`	The y coordinate of a point on the circumference of the arc.
Description	This command is used to add three-point arcs into your Gerber layer. This arc is drawn on the active layer using the current D-Code in a counter-clockwise direction. The arc is created either as a 360° interpolated circle or with multiple line segments, depending on the style that is currently active. Note that this command can be used to draw multiple arcs with a single call by passing all of the coordinates for all of the arcs to the command.
Example	The following example adds an arc to layer 1 using D-Code 10, whose end points are at (0, 0) and (1, 0) and passes through (0.5, 0.5).

```
ACTIVELAYER     1
CURRENTDCODE    10
ADD3PTARC       0,0, 1,0, 0.5, 0.5
```

See also	ADDARC, ADDCIRCLE

ADDARC

Purpose	Allows the user to enter arcs by specifying its center, and two points defining its starting and ending angles.
Menu command	*Edit/Add/Arc Ctr*
Syntax	`ADDARC [x1, y1, x2, y2, x3, y3]...`
Parameters	
`x1`	The x coordinate of the center of the arc.
`y1`	The y coordinate of the center of the arc.
`x2`	The x coordinate of the starting point of the arc.
`y2`	The y coordinate of the starting point of the arc.
`x3`	The x coordinate of the ending point of the arc.
`y3`	The y coordinate of the ending point of the arc.
Description	This command is used to add an arc into your Gerber layer. This arc is drawn on the active layer using the current D-Code in a counter-clockwise direction. The arc is created either as a 360° interpolated circle or with multiple line segments, depending on the style that is currently active. Note that this command can be used to draw multiple arcs with a single call by passing all of the coordinates for all of the arcs to the command.
Example	The following example adds an arc to layer 1 using D-Code 10, whose center is at 0,0 and its end points are at -1,0 and 1,0.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDARC          0,0, -1,0, 1,0
```

See also	ADD3PTARC, ADDCIRCLE

ADDCIRCLE

Purpose	Allows the user to draw a circle by specifying its center, and a point defining its radius.
Menu command	*Edit/Add/Circle*
Syntax	`ADDCIRCLE [x1, y1, x2, y2]...`
Parameters	
`x1`	The x coordinate of the center of the circle.
`y1`	The y coordinate of the center of the circle.
`x2`	The x coordinate of a point on the radius of the circle.
`y2`	The y coordinate of a point on the radius of the circle.
Description	This command is used to add a circle into your Gerber layer. This circle is drawn on the active layer using the current D-Code in a counter-clockwise direction. The arc is created either as a 360° interpolated circle or with multiple line segments, depending on the style that is currently active. Note that this command can be used to draw multiple circles with a single call by passing all of the coordinates for all of the circles to the command.
Example	The following example adds a circle to layer 1 using D-Code 10, whose center is at 0,0 and has a radius of 3 inches.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDCIRCLE       0,0, 3,0
```

See also	ADD3PTARC, ADDARC

ADDDRAW

Purpose	Allows the user to draw a line by specifying its end points.
Menu command	*Edit/Add/Draw*
Syntax	`ADDDRAW [x1, y1, x2, y2]...`
Parameters	
`x1`	The x coordinate of the starting point of the line.
`y1`	The y coordinate of the starting point of the line
`x2`	The x coordinate of the ending point of the line.
`y2`	The y coordinate of the ending point of the line.
Description	This command is used to add a line or trace into your Gerber layer. This line is drawn on the active layer using the current D-Code. Note that this command can be used to draw multiple lines with a single call by passing all of the coordinates for all of the lines to the command. When drawing more than one line, only the ending points of subsequent lines need to be passed to the command. GerbTool will automatically start each line at the endpoint of the previous line, which is the same way that lines are added with the menu command.
Example	The following example creates a 1 inch box out of 4 lines, whose lower left corner is at the 0,0 coordinate. Note how only the changed coordinates need to be entered.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDDRAW         0,0, 0,1, 1,1, 1,0, 0,0
```

ADDFILL

Purpose	Allows the user to perform a polyfill.
Menu command	*Edit/Add/Fill*
Syntax	`ADDFILL [x, y]...`
Parameters	
`x`	The x coordinate of a point on the fill border.
`y`	The y coordinate of a point on the fill border.
Description	This command allows you to enter a closed polygon and GerbTool will fill the interior of the polygon. This command is commonly used to create ground plane areas. To use this command you should enter at least 3 x, y coordinate pairs to specify the outline you want filled. If you enter fewer, GerbTool will stop and prompt the user. GerbTool will then outline the polygon with the current D-Code, and begin filling the interior of the polygon. GerbTool will fill the interior of the polygon with increasing aperture sizes as long as it can continue to find an aperture that is twice the size of the current aperture being used. As it fills toward the center of the polygon the aperture sizes will become larger and larger. This allows the polygon to be filled with the least amount of data thereby keeping the database size as small as possible. If an aperture twice the size of the last aperture used cannot be located then the remainder of the polygon will be filled with the last aperture found.
Example	The following example creates a 1-inch filled box, whose lower left corner is at the 0,0 coordinate.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDFILL         0,0, 0,1, 1,1, 1,0, 0,0
```

ADDFLASH

Purpose	Allows the user to add a flash at a specified location.
Menu command	*Edit/Add/Flash*
Syntax	`ADDFLASH [x, y]...`
Parameters	
`x`	The x coordinate of the location to add a flash.
`y`	The y coordinate of the location to add a flash.
Description	This command allows you to add a flash to your Gerber data. The flash is created on the active layer using the current D-Code. If you want, multiple coordinates can be used with this command, and the system will place a flash at each.
Example	The following example adds four flashes forming the corners of a 1 inch box, whose lower left corner is at the 0,0 coordinate.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDFLASH        0,0, 0,1, 1,1, 1,0, 0,0
```

ADDPOUR

Purpose	Allows the user to perform a polypour.
Menu command	*Edit/Add/Pour*
Syntax	`ADDPOUR [x, y]...`
Parameters	
`x`	The x coordinate of a point on the pour border.
`y`	The y coordinate of a point on the pour border.
Description	This command allows you to enter a closed polygon and GerbTool will fill the interior of the polygon using an intelligent polypour. This command is commonly used to create ground plane areas. To use this command you should enter at least 3 x, y coordinate pairs to specify the outline you want filled. If you enter fewer, GerbTool will stop and prompt the user. This command uses the pour settings that were either set up in the menu command or from the POUR command. This command is used mainly as an easier to use version of the POUR command.
Example	The following example creates a 1-inch filled box, whose lower left corner is at the 0,0 coordinate.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDPOUR         0,0, 0,1, 1,1, 1,0, 0,0
```

See also	POUR

ADDTEXT

Purpose	Allows the user to add a string of text to the database.
Menu command	*Edit/Add/Text*
Syntax	`ADDTEXT string [x, y]...`
Parameters	
`string`	A string containing the text to be added.
`x`	The x coordinate of the location to add the text.
`y`	The y coordinate of the location to add the text.
Description	This command allows you to add user specified text to a Gerber layer. The text is drawn using the current D-Code on the active layer This command uses the settings that were either set up in the menu command or from the `TEXT` command. This command is used mainly as an easier to use version of the `TEXT` command.
Example	The following example adds the text string "Hello World" starting at coordinate 0,0.

```
ACTIVELAYER     1
CURRENTDCODE    10
ADDTEXT         "Hello World" 0,0
```

See also	TEXT

ADDVERTEX

Purpose	Allows the user to add a vertex to a line.
Menu command	*Edit/Add/Vertex*
Syntax	`ADDVERTEX [x, y]...`
Parameters	
`x`	The x coordinate of the vertex to add.
`y`	The y coordinate of the vertex to add.
Description	This command allows you to add a vertex to an existing trace, creating two traces from the original. The coordinate specified is the location of the new vertex and must be located on an existing trace.
Example	The following example adds a vertex to the center of an existing line, whose end points are located in the variables $x1, $y1 and $ux, $uy.

```
CALC   $ptx = $ux - $lx
CALC   $ptx = $ptx / 2.0
CALC   $ptx = $ptx + $lx
CALC   $pty = $uy - $ly
CALC   $pty = $pty / 2.0
CALC   $pty = $pty + $ly

ADDVERTEX $ptx, $pty
```

Aperture functions

APREPORT

Purpose — Produces an aperture usage report.

Menu command — *Apertures/Report*

Syntax

```
APREPORT
    REPFILEstring
    LAYERexp
    SORTBYstring
    GO
END
```

Parameters

`REPFILE` — This is the name of the report file the command will produce.

`LAYER` — Specifies the layer to produce the report on. If zero is given, all layers will be included in the report.

`SORTBY` — Must be either D-Code or List. This is used to specify the ordering of the report when more than one layer is being specified.

Description — This command is used to produce an aperture usage report. The format of this report is the same as that produced by the Apertures/Report command. This report details which D-Codes along with their definitions, are being used on a per layer basis. Included in the report are use counts for both flashes and draws.

Example — The following example produces an aperture report for all visible layers sorted by aperture list and places the result in the file OUTPUT.RPT.

```
APREPORT
    REPFILE "output.rpt"
    LAYER 0 #output rep based on all lyrs
    SORTBY "List"
    GO
END
```

GETAPINFO

Purpose — To obtain information about an aperture in an aperture list.

Menu command — *Apertures/Edit*

Syntax

```
GETAPINFO [layer, dcode, shape, xsize,
ysize, type, tool, toolsize, legend]
```

Parameters

`layer` — An expression specifying the layer whose aperture list is to be used.

`dcode` — An expression specifying the D-Code to get information on.

`shape` — A numeric variable that returns the shape of the aperture.

`xsize` — A numeric variable that returns the width of the aperture.

`ysize` — A numeric variable that returns the height of the aperture.

`type` — A numeric variable that returns the type of the aperture.

`tool` — A numeric variable that returns the tool number of the aperture.

`toolsize` — A numeric variable that returns the toolsize for the aperture.

`legend` — A numeric variable that returns the legend value for the aperture.

Description — The user specifies the number of a layer that uses the aperture list and the aperture D-Code. The system then places the appropriate information into the other variables passed into the command. All of the variables listed with this command must be included at least once within a macro, even if they are not going to be used by the macro. Subsequent calls can specify the first two parameters (layer, dcode) or can omit all parameters. Those variables used in the last complete call are remembered and used again.

Example — The example gets the aperture information used by layer 1 of a design, and writes its toolsize and D-Code out to a file.

```
IF $dcode != 0
    GETAPINFO 1, $dcode, $shp, $xs, $ys, \
       $type, $tool, $toolsize, $legend
    FILEWRITE $fd,"%n %n",$toolsize,$dcode
END
```

See also — PUTAPINFO

PUTAPINFO

Purpose

To update the information about a specific aperture in an aperture list.

Menu command

Apertures/Edit

Syntax

```
PUTAPINFO [layer, dcode, shape, xsize,
ysize, type, tool, toolsize, legend]
```

Parameters

See the parameter list for the GETAPINFO command. The PUTAPINFO and GETAPINFO parameter lists are identical and are shared between the two commands. For this reason, you need only specify the parameter list for one command, and the other command will automatically use the same variables.

Description

This command is used to update information about an aperture. The user specifies the number of a layer that uses the aperture list and the aperture D-Code. The system then places the appropriate information from the other variables passed into the command into the actual aperture list. All of the variables listed with this command must be included at least once within a macro, even if they are not going to be used by the macro. Subsequent calls can specify the first two parameters (layer, dcode) or can omit all parameters. Those variables used in the last complete call are remembered and will be used again.

Example

The following example gets the aperture information for an aperture used by layer 1 of a design, changes the size and updates the aperture list.

```
IF $dcode != 0
    GETAPINFO $lyr,$dcode,$shp,$xs,$ys, \
       $type, $tool, $toolsize, $legend
    CALC $xs = $xs * 1.2
    CALC $ys = $ys * 1.2
    PUTAPINFO
END
```

See also

GETAPINFO

Control statements

CALLMACRO

Purpose	To execute a macro within another macro.
Menu command	None
Syntax	`CALLMACRO macroname, parameter0-9`
Parameters	
`macroname`	A string indicating which macro to execute.
`parameter0-9`	From 0 to 9 variables, literals, or expressions.
Description	Allows a macro to "call" another macro so that general purpose macros can be written and shared by other macros. A system variable `$$CALLDEPTH` detects if a macro is being called: the parameters passed are visible to the called macro as parameters `$1-$9`. Any changes to these variables are reflected in the "caller" macro. The system variable `$$STATUS` is passed back to the calling macro. Note: If you pass in a literal value such as `four`, it is not updated when the called macro returns.
Example	Within the called macro "maxes," `$top_layer` and `$top_dcode` are automatically defined as `$1` and `$2`. When maxes ends, `$top_layer` and `$top_dcode` are updated with values of `$1` and `$2`.

```
MACRO test1
    CALLMACRO "maxes", $top_layer,
    $top_dcode
    STRWRITE $msg,"Max:lyr=%n\ndcode=%n",\
       $top_layer, $top_dcode
    MESSAGEBOX "MAX Info", $msg, 0
END
MACRO maxes
    # $1 is synonym for $top_layer
    SET $1 = $$MAXLAYERS + 1
    REPEAT $$STATUS == $$FALSE
       CALC $1 = $1 - 1
       ACTIVELAYER $1
    END
    # $2 is synonym for $top_dcode
    CALLMACRO "maxuseddcode", $2
END
```

DEBUG

Purpose	To turn macro debug mode on/off.
Menu command	None
Syntax	`DEBUG exp | strlit`
Parameters	
`exp`	An expression where 0=OFF and 1=ON, or a string literal of `"Y"` or `"N"`.
Description	This function is used to toggle debug mode on/off. When debug mode is on, debug information is output to a file named _MACRO_.DEB.
Example	The following example turns debug mode on and off.

```
DEBUG 1        # on
CALLMACRO      "NewMac"
DEBUG 0        # off
```

IF

Purpose — To conditionally execute blocks of macro commands.

Menu command — None

Syntax — `IF [numvar = ] exp`

Parameters

`numvar` — An optional numeric variable that will receive the result of the expression on the right side of the equal sign.

`exp` — An expression indicating whether the macro statements between the `IF` and the corresponding `ELSE` or `END` statement will be executed.

Description — This function is used when you need to execute a block of macro statements only when a certain condition exists.

Example — The following example shows two `IF` statements, one `IF/ELSE/END` trio and a nested `IF/END` pair.

```
GETSTRING "Enter your name", $name
STRLEN $name
IF $$STATUS == 0
   #this block only executed if the
   #variable $$STATUS is equal to 0
   STOP "Bad Username"
ELSE
   #this block only executed if the
   #variable $$STATUS is NOT equal to 0
   GETSTRING "Enter your passwd", $passwd
   STRCMP $passwd, "dilbert"
   IF $$STATUS != 0
      #this block only executed if the
      #variable $$STATUS NOT equal to 0
      STOP "Bad Passwd"
   END
END
```

REPEAT

Purpose	Repeats a series of macro commands until the test condition becomes false.
Menu command	None
Syntax	`REPEAT [numvar = ] exp`
Parameters	
`numvar`	An optional numeric variable that will receive the result of the expression on the right hand side of the equal sign.
`exp`	An expression indicating whether the macro statements between the `REPEAT` and the corresponding `END` statement will be executed.
Description	This function executes a block of macro statements repeatedly while a certain condition exists. As long as the test condition is not zero, the commands between the `REPEAT` and `END` will be executed repeatedly.
Example	The following macro will continue to prompt the user for a text string as long as a blank string isn't entered.

```
SET $cnt = 1
GETSTRING "Enter a component", $comp
STRLEN $name
REPEAT $$STATUS != 0
    FILEWRITE $fid, \
              "COMP%.0n: %s", \
              $cnt, $comp
    CALC $cnt = $cnt + 1
    GETSTRING "Enter a component", $comp
    STRLEN $name
END
```

STOP

Purpose	Causes the macro to stop.
Menu command	None
Syntax	`STOP [message]`
Parameters	
`message`	An optional string variable or string literal giving a message that will be presented to the user when the macro is stopped.
Description	This command will cause the macro to stop, and an optional message will be presented to the user.
Example	The following example stops a macro if the user enters no.

```
GETYESNO "Quit? ", $value
IF $value == $$YES
    STOP
END
```

Database functions

COPYITEM

Purpose	To copy an item in a Gerber database.
Menu command	*Edit/Copy*
Syntax	`COPYITEM layer, seqno, dx, dy`
Parameters	
`layer`	An expression representing the layer containing the item to copy.
`seqno`	An expression representing the sequence number of the item to copy.
`dx`	An expression representing the amount of offset to apply to the x coordinate of the object.
`dy`	An expression representing the amount of offset to apply to the y coordinate of the object.
Description	This function is used to copy Gerber items in a layer. It takes the layer and sequence number of the object you want to copy. The sequence number used is the same as that returned by the menu command *Query/Item Info* or the macro command `GETFIRSTITEM`. This function returns $$TRUE in the $$STATUS variable it the command was successful and $$FALSE if it was not.
Example	The following example scans a layer and copies all flashes 1 inch to the left.

```
GETFIRSTITEM $layer, $seqno, $net, \
             $dcode,$type, $x, $y, \
             $x2, $y2,$dia,$cw
REPEAT $$STATUS != $$FALSE
    IF $type == $$FLASH
       COPYITEM $layer, $seqno, -1.0, 0
       GETNEXTITEM
    END

END
```

See also	GETFIRSTITEM, GETNEXTITEM

DELETEITEM

Purpose — To delete an item from a Gerber database.

Menu command — *Edit/Erase*

Syntax — `DELETEITEM layer, seqno`

Parameters

`layer` — An expression representing the layer containing the item to delete.

`seqno` — An expression representing the sequence number of the item to delete.

Description — This function is used to delete Gerber items from a layer. It takes the layer and sequence number of the object you want to delete. The sequence number used is the same as that returned by the menu command *Query/Item Info* or the macro command `GETFIRSTITEM`. This function returns $$TRUE in the $$STATUS variable it the deletion was successful and $$FALSE if it was not.

Example — The following example scans a layer and removes all flashes.

```
GETFIRSTITEM $layer,$seqno,$net,$dcode, \
             $type, $x, $y, $x2, $y2, \
             $dia, $cw
REPEAT $$STATUS != $$FALSE
    IF $type == $$FLASH
       DELETEITEM  $layer, $seqno
    END
       GETNEXTITEM
END
```

See also — GETFIRSTITEM, GETNEXTITEM

GETEXTENTS

Purpose

To calculate and return the extents of the Gerber data on any one or all loaded layers.

Menu command

Query/Extents

Syntax

```
GETEXTENTS layer, lx, ly, ux, uy
```

Parameters

`layer` An expression indicating the layer or layers you want the extents of.

`lx` A numeric variable that returns the x coordinate of the lower left extent of the data on the requested layer(s).

`ly` A numeric variable that returns the y coordinate of the lower left extent of the data on the requested layer(s).

`ux` A numeric variable that returns the x coordinate of the upper right extent of the data on the requested layer(s).

`uy` A numeric variable that returns the y coordinate of the upper right extent of the data on the requested layer(s).

Description

This calculates the extents of the requested layer or layers. If `layer` contains a value greater than zero, the command determines the extent for that layer. If the value is zero, the extent for all visible layers is calculated, and if -1 is used, the extents of all layers regardless of their visibility is returned. The coordinates calculated by this command are returned in the other four variables.

Example

The following example gets the extents of all the layers in a design and draws a box around it.

```
GETEXTENTS   -1, $lx, $ly, $ux, $uy
ADDDRAW      $lx,$ly,$lx,$uy,$ux,$uy, \
             $ux,$ly, $lx, $ly
```

GETFILMBOX

Purpose	To return the size of the film box.
Menu command	*Options/Filmbox*
Syntax	`GETFILMBOX xsize, ysize`
Parameters	
`xsize`	A numeric variable that returns the horizontal size of the film box.
`ysize`	A numeric variable that returns the vertical size of the film box.
Description	This function returns the size of the current film box. Since the lower left corner of the film box is always at coordinate 0,0, These values can be used to determine the proper positioning of any Gerber information.
Example	The following example gets the size of the film box and draws a box on the active layer in the same location.

```
GETFILMBOX    $xs, $ys
ADDDRAW       0,0,0,$ys,xs,$ys,$xs,0,0,0
```

GETFIRSTITEM

Purpose

To retrieve information about an object in a Gerber layer. This function also sets up the variables needed for the `GETNEXTITEM` function.

Menu command

Query/Item Info

Syntax

```
GETFIRSTITEM layer, seqno, net, dcode,
type, x, y, x2, y2, dia, cw, flags
```

Parameters

`layer` A variable containing the layer to scan for information. If the layer specified by this variable is not a valid layer, all layers will be scanned, and the original contents of this variable will be replaced with the layer of the item being returned.

`seqno` A variable that returns the sequence number of the item being scanned. An item's sequence number is its relative location in the Gerber file. This is the same number as that displayed in the command *Query/Item Info*.

`net` A variable that returns the net id number associated with this item. If the item does not have a net associated with it, -1 will be returned.

`dcode` A variable that returns the D-Code of the item scanned.

`type` A variable that returns the type of the item being scanned. Valid return values are `$$FLASH` and `$$DRAW`.

`x` A variable that returns an x coordinate for the item being scanned. If the object in question is of type $$FLASH, this value represents the coordinate of the center of the flash. If the variable is of type $$DRAW, it represents the x coordinate of one of its end points.

`y` A variable that returns an y coordinate for the item being scanned. If the object in question is of type $$FLASH, this value represents the coordinate of the center of the flash. If the variable is of type $$DRAW, it represents the y coordinate of one of its end points.

`x2` A variable that returns the x coordinate of the second end point for lines. These are Gerber items that return a type of `$$DRAW`. If the type of an item is `$$FLASH`, this variable is not used.

`y2`	A variable that returns the y coordinate of the second end point for lines. These are Gerber items that return a type of `$$DRAW`. If the type of an item is `$$FLASH`, this variable is not used.
`dia`	A variable that returns the radius of a Gerber arc. If the object being scanned is not an arc, the value 0.0 will be returned.
`cw`	A variable that returns `$$TRUE` if the item being scanned is drawn in a clockwise direction, and `$$FALSE` if it is drawn in a counter-clockwise direction. If the item being scanned is not an arc, this variable is not used.
`flags`	A variable that returns the flags value for the current item. This parameter is currently used to indicate TOP (2048) or BOTTOM (4096) of the test point layer items. This parameter may have more uses in the future.

Description

This function is used to scan loaded Gerber files for information. When called, it examines the layer and seqno parameters for valid data. If they are valid, the specified item will be loaded into the variables described above. If not, the first item in the first loaded layer will be returned. To move to the next item, use the `GETNEXTITEM` command, which uses the variables initialized by this command. If this command is unable to find a Gerber item on this layer, a status of `$$FALSE` is returned, otherwise `$$TRUE` is returned.

Example

The following example uses the `GETFIRSTITEM` and `GETNEXTITEM` commands to loop through all of the items on the layer represented by $layer. When there are no more items left on the layer, `$$STATUS` will return `$$FALSE` and the repeat loop will stop.

```
GETFIRSTITEM $layer,$seqno,$net,$dcode, \
 $type, $x, $y, $x2, $y2, $dia, $cw
REPEAT $$STATUS != $$FALSE
 ...process data here...
    GETNEXTITEM
END
```

See also

GETNEXTITEM

GETLAYER

Purpose — Allows the user to obtain information about a GerbTool layer (Gerber file).

Menu command — *Layers/Edit*

Syntax

```
GETLAYER layer, fn, ln, an, vis, fc,
dc, type, polarity, key, ft, lx, ly,
ux, uy, netid
```

Parameters

`layer` — A numeric variable containing the layer to obtain the information for. Note: A variable must be used.

`fn` — A string variable that returns the Gerber filename associated with this layer.

`ln` — A string variable that returns the layer name associated with this layer. This variable is only used on 274X files.

`an` — A string variable that returns the name of the aperture list associated with this layer.

`vis` — A numeric variable that returns the visibility of the layer. The possible values are: 0=OFF, 1=ON, 2=REF.

`fc` — A string variable that returns the color of the flashes on this layer.

`dc` — A string variable that returns the color of the draws on this layer.

`type` — A numeric variable that returns the layer type of this layer. The possible values are: 0=TOP, 1=INNER, 2=BOTTOM, 3=PLANE, 4=COMPOSITE, 5=OTHER.

`polarity` — A numeric variable returning the polarity of this layer. The possible values are 0=clear, 1=dark. This variable is only used on 274X files.

`key` — A numeric variable that returns the key value associated with this layer. This variable is only used on 274X files.

`ft` — A numeric variable that returns the type of file loaded on this layer. This value is currently not used.

`lx`	A numeric variable that returns the x coordinate of the lower left most extent of the data on this layer.
`ly`	A numeric variable that returns the y coordinate of the lower left most extent of the data on this layer.
`ux`	A numeric variable that returns the x coordinate of the upper right most extent of the data on this layer.
`uy`	A numeric variable that returns the y coordinate of the upper right most extent of the data on this layer.
`netid`	A numeric variable that returns the netid value that is associated with this layer.

Description

This command is used to retrieve information about a given Gerber layer. This information is returned in the variables described above. Note that all of the variables must be included with this command even if they are not used.

Example

The following example retrieves information regarding the layer whose number is stored in $layer.

```
GETLAYER $layer,$fn,$ln,$an,$vis, \
 $fc,$dc,$type,$polarity, \
 $key,$ft,$lx,$ly,$ux, $uy,$netid
```

GETNEXTITEM

Purpose	To retrieve information about an object in a Gerber layer. This function is used in conjunction with the GETFIRSTITEM function.
Menu command	*Query/Item Info*
Syntax	GETNEXTITEM
Parameters	*None*
Description	This function is used along with GETFIRSTITEM to scan loaded Gerber files for information. The command GETFIRSTITEM must be called before this function in order to setup the variables used and retrieve the first item. When GETNEXTITEM is called, the information for the next Gerber object on the layer is placed into the same variables that were created and used by the GETFIRSTITEM command, and the $$STATUS variable is set to $$TRUE. When there are no more objects to process, a status of $$FALSE will be returned.
Example	The following example uses the GETFIRSTITEM and GETNEXTITEM commands to loop through all of the items on the layer represented by $layer. When there are no more items left on the layer, $$STATUS will return $$FALSE and the repeat loop will stop.

```
GETFIRSTITEM $layer,$seqno,$net,$dcode, \
      $type, $x, $y, $x2, $y2, $dia, $cw
REPEAT $$STATUS != $$FALSE
      ...process data here...
   GETNEXTITEM
END
```

See also GETFIRSTITEM

GETUSERDATA

Purpose	To obtain the UserData field for a specific database item.
Menu command	*Query/Item Info* or *Edit/Item*
Syntax	`GETUSERDATA [layer, seqno userdata]`
Parameters	
`layer`	A numeric variable indicating the layer.
`seqno`	A numeric variable indicating the sequence number of the item.
`userdata`	A string variable that will receive the UserData from the specified database item.
Description	This function locates the specified item and copies its UserData field to the specified string variable. The above parameters should be specified on the first call to `GETUSERDATA` but can thereafter be omitted. The variables used for the `layer` and `seqno` parameters are usually those used in a `GETFIRSTITEM/GETNEXTITEM` loop.
Example	The following example scans the active layer allowing the user to edit the UserData field of each item.

```
MACRO getuserdata
    SET $lyr = $$ACTIVELAYER
    SET $seqno = 0
    STRSET $user, ""

    GETUSERDATA  $lyr, $seqno, $user
    GETFIRSTITEM $lyr,$seqno,$net,$dcode,\
             $type, $x, $y, $x2, $y2, $d
    REPEAT $$STATUS
       GETUSERDATA
       GETSTRING "Edit UserData:", $user
       GETNEXTITEM
    END
ENDMACRO
```

See also	PUTUSERDATA, GETFIRSTITEM, GETNEXTITEM

GETVIEWEXTENTS

Purpose	To obtain the extents of the current viewing window.
Menu command	None
Syntax	`GETVIEWEXTENTS lowerx, lowery, upperx, uppery`
Parameters	
`lowerx`	A numeric variable that will receive the lower left X limit.
`lowery`	A numeric variable that will receive the lower left Y limit.
`upperx`	A numeric variable that will receive the upper right Y limit.
`uppery`	A numeric variable that will receive the upper right Y limit.
Description	This function returns the extents of the current viewing window in the four numeric variables specified.
Example	The following example gets the size extents of the current view window and then zooms in at its center.

```
GETVIEWEXTENTS $lx, $ly, $ux, $uy
CALC   $tx = $ux - $lx
CALC   $ty = $uy - $ly
CALC   $tx = $tx / 2
CALC   $ty = $ty / 2
CALC   $tx = $lx + $tx
CALC   $ty = $ly + $ty
ZOOMIN $tx, $ty
```

MOVEITEM

Purpose	To move a selected item in a Gerber database.
Menu command	*Edit/Move*
Syntax	`MOVEITEM layer, seqno, dx, dy`
Parameters	
`layer`	An expression representing the layer containing the item to move.
`seqno`	An expression representing the sequence number of the item to move.
`dx`	An expression representing the amount of offset to apply to the x coordinate of the object.
`dy`	An expression representing the amount of offset to apply to the y coordinate of the object.
Description	This function is used to move Gerber items in a layer. It takes the layer and sequence number of the object you want to move. The sequence number used it the same as that returned by the menu command *Query/Item Info* or the macro command `GETFIRSTITEM`. This function returns `$$TRUE` in the `$$STATUS` variable it the move was successful and `$$FALSE` if it was not.
Example	The following example scans a layer and moves all flashes 1 inch to the right.

```
GETFIRSTITEM $layer,$seqno,$net,$dcode, \
 $type, $x, $y, $x2, $y2, $dia, $cw
REPEAT $$STATUS != $$FALSE
    IF $type == $$FLASH
       MOVEITEM $layer, $seqno, 1.0, 0
    END
    GETNEXTITEM
END
```

See also	GETFIRSTITEM

PUTUSERDATA

Purpose — To update the UserData field for a specific database item.

Menu command — *Edit/Item*

Syntax — `PUTUSERDATA [layer, seqno, userdata]`

Parameters

`layer` — A numeric variable indicating the layer.

`seqno` — A numeric variable indicating the sequence number of the item.

`userdata` — A string variable that will be used to update the UserData in the specified database item.

Description — This function locates the specified item and copies the string from the userdata parameter into its UserData field. The above parameters should be specified on the first call to `GETUSERDATA` but can thereafter be omitted. The variables used for the `layer` and `seqno` parameters are usually those used in a `GETFIRSTITEM/GETNEXTITEM` loop.

Example — The following example scans the active layer initializing the UserData field of each item.

```
MACRO putuserdata

    SET $lyr = $$ACTIVELAYER
    SET $seqno = 0
    SET $cnt = 0
    STRSET $user, ""

    GETUSERDATA $lyr, $seqno, $user
    GETFIRSTITEM $lyr,$seqno,$net,$dcode,\
             $type, $x, $y, $x2, $y2, $d

    REPEAT $$STATUS

       CALC $cnt = $cnt + 1
       STRWRITE $user, "U%.0n", $cnt
       PUTUSERDATA
       GETNEXTITEM

    END

ENDMACRO
```

See also — GETUSERDATA, GETFIRSTITEM, GETNEXTITEM

Editing functions

ALIGNLAYERS

Purpose	To align Gerber layers based on common items.
Menu command	*Edit/Align*
Syntax	`ALIGNLAYERS [x, y...]`
Parameters	
`x, y`	A variable number of parameters specifying the coordinates of common objects to align.
Description	This function is used to align Gerber layers who for some reason no longer share a common origin. The first coordinate represents the location on a reference layer of an item that you want to align all of the other layers to. The remainder of the coordinates represent the locations of objects on other layers that you want to align to the first specified object.
Example	The following example performs a layer alignment of two layers. The location of the reference object is at 0,0. The location of the object on the second layer has been calculated, and is located in the variables $x, $y.

```
ALIGNLAYERS 0, 0, $x, $y
```

CLIP

Purpose — Allows a macro to erase items, allowing clipping of lines.

Menu command — *Edit/Clip*

Syntax

```
CLIP
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
    GO        [x1, y1, x2, y2 ...]
END
```

Parameters

BY	An expression indicating how to perform the clipping. Valid values are: 1=window, 2=group.
BOUNDARY	Controls erasure of flashes straddling the window boundary.
DCODE	Indicates a D-Code filter to use for the command. If zero is used, all D-Codes may be affected by the command.
LAYER	Indicates a layer filter to use for the command. If zero is used, all visible layers may be affected by the command.
GO	A variable number of coordinates used by the system.

Description — This is used to erase a selection of items with automatic clipping of lines that cross the window boundary.

Example — The following example clips all arcs using window mode.

```
CLIP
    By        $$WINDOWMODE
    Boundary $$NO    # no flashes
    Flashes   $$NO
    Draws     $$NO
    Arcs      $$YES
    Layer 0   #Erase from all visible lyrs
    Dcode 0   #Erase all D-Codes
    GetWindow"Enter Clip Window", \
              $lx,$ly,$ux,$uy
    GO $lx,$ly,$ux,$uy
END
```

See also — ERASE

COPY

Purpose — Allows a macro to perform a copy.

Menu command — *Edit/Copy*

Syntax

```
COPY
    BY          exp
    BOUNDARY    yesno

    FLASHES     yesno
    DRAWS       yesno

    ARCS        yesno
    DCODE       exp

    LAYER       exp
    TOLAYER     exp
    GO          [x1, y1, x2, y2 ...]
END
```

Parameters

`BY` — An expression indicating how to perform the copy. Valid values are: 0=item, 1=window, 2=group.

`DCODE` — An expression indicating a D-Code filter to use for the copy. If zero is used, all D-Codes may be affected by the command.

`LAYER` — An expression indicating a layer filter to use for the copy. If zero is used, all visible layers may be affected by the command.

`TOLAYER` — An expression indicating the layer to copy all of the selected objects to. If zero is used, the objects will be kept on their original layers.

`[x1,y1,x2, y2...]` — A variable number of coordinates used by the system to complete the command.

Description — This function is used to perform a copy of a number of items. If window mode is selected the first 2 x, y coordinate pairs given are used to specify the window, if item mode is used, the first pair is used to select the item to copy, and in group mode the select group is used and none of the coordinate pairs are used to select the items. The next coordinate is used to specify the starting location for the copy, and all subsequent coordinate pairs are used to specify to locations where the copied data is to be placed.

Example

The following example obtains a window, obtains from and to locations from the user, and performs a copy.

```
COPY
    By        $$WINDOWMODE
    Boundary  $$YES
    Flashes   $$YES
    Draws     $$YES
    Arcs      $$YES
    Layer     0 # Copy from all visible
    Dcode     0 # Copy all D-Codes
    GetWindow"Enter Copy Window", \
              $lx,$ly,$ux,$uy
    GetPoint "Enter from location", \
              $fx,$fy
    GetPoint "Enter to location", \
              $tx,$ty
    GO        $lx,$ly,$ux,$uy, \
              $fx,$fy, $tx, $ty
END
```

DCEXPAND

Purpose

Allows a macro to expand a custom aperture into normal Gerber entities.

Menu command

Edit/DCode/Expand

Syntax

```
DCEXPAND
    DCODE     exp
    LAYER     exp
    GO
END
```

Parameters

DCODE

An expression indicating a D-Code filter to use when transcoding. If zero is used, all D-Codes may be affected by the command.

LAYER

An expression indicating a layer filter to use for transcoding. If zero is used, all visible layers may be affected by the command.

Description

This function is used to expand custom apertures into their basic Gerber constructs.

Example

The following example expands all of the custom apertures on all visible layers.

```
DCEXPAND
    LAYER 0   # Change all visible layers
    DCODE 0   # Change all D-Codes
END
```

DCODESCALE

Purpose — Allows a macro to scale D-Codes in a design.

Menu command — *Edit/DCode/Scale*

Syntax

```
DCODESCALE
    FIXED     yesno
    SCALE     exp, exp
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
    GO        [x1, y1, x2, y2 ...]
END
```

Parameters

`SCALE` — A pair of expressions indicating the X-Y scaling factors.

`BY` — An expression indicating how to select the items to change. Valid values are: 0=item, 1=window, 2=group.

`DCODE` — An expression indicating a D-Code filter to use for the command. If zero is used, all D-Codes may be affected by the command.

`LAYER` — An expression indicating a layer filter to use for the command. If zero is used, all visible layers may be affected by the command. `[x1,y1, x2,y2...]`. A variable number of coordinates used by the system to complete the command.

Description — This function is used to scale the D-Codes of a number of selected items. If window mode is selected the first 2 x, y coordinate pairs given are used to specify the window, if item mode is used, the first pair is used to select the item to copy, and in group mode the select group is used and no coordinates need to be specified. If fixed is chosen, the values specified for the scale are added to the sizes of the apertures in question. If fixed is given the value of `$$NO`, the sizes of the apertures are multiplied by the scale values.

Example

The following example obtains a window's from and to locations from the user, and scales all of the flashes in this area up by 5 percent.

```
DCODESCALE
    FIXED     $$NO
    SCALE     1.05, 1.05
    BY        $$WINDOWMODE
    BOUNDARY  $$YES
    FLASHES   $$YES
    DRAWS     $$NO
    ARCS      $$NO
    LAYER 0   #Copy from all visible layers
    DCODE 0   #Copy all D-Codes
    GETWINDOW"Enter Window", \
              $lx,$ly,$ux,$uy
    GO $lx,$ly,$ux,$uy
END
```

ERASE

Purpose — Allows a macro to erase items.

Menu command — *Edit/Erase*

Syntax

```
ERASE
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
    GO        [x1, y1, x2, y2 ...]
END
```

Parameters

`BY` — An expression indicating how to perform the erasure. Valid values are: 0=item, 1=window, 2=group.

`DCODE` — An expression indicating a D-Code filter to use for the command. If zero is used, all D-Codes may be affected by the command.

`LAYER` — An expression indicating a layer filter to use for the command. If zero is used, all visible layers may be affected by the command.

`GO` — A variable number of coordinates used by the system to complete the command.

Description — This function is used to erase a selection of items. If window mode is selected the first 2 x, y coordinate pairs given are used to specify the window, if item mode is used, the first pair is used to select the item to delete, and in group mode the select group is used and no coordinates need to be specified.

Example

The following example obtains a window and erases all arcs and draws in it.

```
ERASE
    By        $$WINDOWMODE
    Boundary  $$YES
    Flashes   $$NO
    Draws     $$YES
    Arcs      $$YES
    Layer 0   #Erase from all visible lyrs
    Dcode 0   #Erase all D-Codes
    GetWindow"Enter Erase Window", \
              $lx,$ly,$ux,$uy
    GO $lx,$ly,$ux,$uy
END
```

See also

CLIP

MIRROR

Purpose — Allows a macro to mirror objects.

Menu command — *Edit/Mirror*

Syntax

```
MIRROR
    DIRECTION       <exp | "H" | "V">
    CENTERED        yesno
    BY              byexp
    BOUNDARY        yesno
    FLASHES         yesno
    DRAWS           yesno
    ARCS            yesno
    DCODE           dcrexp
    LAYER           lyrexp
    GO              [x1, y1, x2, y2 ...]
END
```

Parameters

`DIRECTION` — An expression indicating the direction of the mirroring. Valid values are: 0=horizontal and 1=vertical. Also acceptable are the text characters `H` and `V`.

`BY` — An expression indicating how to perform the mirror. Valid values are 0=item, 1=window, 2=group.

`DCODE` — An expression indicating a D-Code filter to use mirroring. If zero is used, all D-Codes may be affected by the command.

`LAYER` — An expression indicating a layer filter to use for mirroring. If zero is used, all visible layers may be affected by the command.

`GO` — A variable number of coordinates used by the system to complete the command.

Description — This function is used to mirror a selection of items. If window mode is selected the first 2 x, y coordinate pairs given are used to specify the window, if item mode is used, the first pair is used to select the item to copy, and in group mode the select group is used and none of the coordinate pairs are used to select the items. If centered is set to `$$NO`, the next coordinate will be used to specify the center of rotation, otherwise the center of the selected items will be used.

Example

The following example obtains a window, from the user and mirrors all of the items in it.

```
MIRROR
    DIRECTION    0   # horz
    CENTERED     $$YES
    By           $$WINDOWMODE
    Boundary     $$YES
    Flashes      $$YES
    Draws        $$YES
    Arcs         $$YES
    Layer 0      # Mirror all visible layers
    Dcode 0      # Mirror all D-Codes
    GetWindow    "Enter Mirror Window", \
                 $lx,$ly,$ux,$uy
    GO           $lx,$ly,$ux,$uy
END
```

MOVE

Purpose — Allows a macro to perform a move.

Menu command — *Edit/Move*

Syntax

```
MOVE
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
    TOLAYER   exp
    GO        [x1, y1, x2, y2 ...]
END
```

Parameters

`BY` — An expression indicating how to perform the move. Valid values are: 0=item, 1=window, 2=group.

`DCODE` — An expression indicating a D-Code filter to use for the move. If zero is used, all D-Codes may be affected by the command.

`LAYER` — An expression indicating a layer filter to use for the move. If zero is used, all visible layers may be affected by the command.

`TOLAYER` — An expression indicating the layer to move all of the selected objects to. If zero is used, the objects will be kept on their original layers.

`GO` — A variable number of coordinates used by the system to complete the command.

Description — This function is used to perform a move on a selection of items. If window mode is selected the first 2 x, y coordinate pairs given are used to specify the window, if item mode is used, the first pair is used to select the item to copy, and in group mode the select group is used and none of the coordinate pairs are used to select the items. The next coordinate is used to specify the starting location for the move, and all subsequent coordinate pairs are used to specify to locations where the moved data is to be placed.

Example

The following example obtains a window, from and to locations from the user and performs a move.

```
MOVE
    By          $$WINDOWMODE
    Boundary    $$YES
    Flashes     $$YES
    Draws       $$YES
    Arcs        $$YES
    Layer 0     #Move from all visible layers
    Dcode 0     #Move all D-Codes
    GetWindow"Enter Move Window", \
                $lx,$ly,$ux,$uy
    GetPoint "Enter from location",$fx,$fy
    GetPoint "Enter to location", $tx, $ty
    GO          $lx,$ly,$ux,$uy, $fx,$fy,$tx,$ty
END
```

ORIGIN

Purpose	To enable a macro to change the origin used by GerbTool.
Menu command	*Edit/Origin*
Syntax	`ORIGIN [x, y...]`
Parameters	
`x, y`	A variable number of coordinates representing new origins.
Description	This function is used to change the origin used by GerbTool. The origin is the 0,0 coordinate of the Gerber files and is always located at the lower left corner of the film box.
Example	The following example calculates the center of the extents of all visible Gerber files and moves the origin to that point.

```
GETEXTENTS   -1, $lx, $ly, $ux, $uy
CALC $x =    $ux - $lx
CALC $x =    $x / 2
CALC $x =    $x + $lx
CALC $y =    $uy - $ly
CALC $y =    $y / 2
CALC $y =    $y + $ly
ORIGIN       $x, $y
```

POUR

Purpose — Allows a macro to perform a polypour.

Menu command — *Edit/Add/Pour*

Syntax

```
POUR
    DRAWCLR     exp
    FLASHCLR    exp
    MINAREA     exp
    TYPE        exp
    LAYER       exp
    HATCHLINE   line, dcode, step, angle
    DCODE       exp
    GO          [x1, y1, x2, y2 ...]
END
```

Parameters

DRAWCLR	An expression indicating the clearance the pour is to maintain from draws.
FLASHCLR	An expression indicating the clearance the pour is to maintain from flashes.
MINAREA	An expression that indicates the minimum area to be filled by a pour.
TYPE	An expression giving the type of pour to perform. Valid values are: 1=outline, 2=solid, 3=hatch.
LAYER	An expression indicating the layer to place the resulting polypour onto.
DCODE	An expression that represents D-Code to be used for this particular hatch line.
HATCHLINE	A quad of expressions indicating describing one of the three possible hatch lines. The first expression is a line number of 1-3. The remaining parameters describe the dcode, step size and angle of the selected hatch line. This parameter can be used multiple times in a single pour command for more complicated pour patterns.
GO	A variable number of coordinates the outline of the area to perform the polypour. The starting coordinate should also be specified at the end of the coordinate list in order to close the polygon and complete the pour.

Description — This function is used to perform a polypour.

Example

The following example draws a doubly nested rectangle in such a way that it can be filled with the cross hatch pattern set up for this pour.

```
POUR
    DRAWCLR   0.02
    FLASHCLR  0.02
    TYPE      2       # 0 == OUTLINE,
                      # 1 == SOLID,
                      # 2 == HATCHED
    HATCHLINE 1, $$CURRENTDCODE, 0.35, 45
    HATCHLINE 2, $$CURRENTDCODE, 0.35, 135
    HATCHLINE 3,0,0.0,0 #only use 2 lines
    GO $olx,$oly, $olx,$ouy, $oux,$ouy, \
       $oux,$oly, $iux,$ily, $iux,$iuy, \
       $ilx,$iuy, $ilx,$ily, $iux,$ily, \
       $oux,$oly, $olx,$oly
END
```

PURGE

Purpose	To compact the Gerber databases and purge the undo queue.
Menu command	*Edit/Purge*
Syntax	`PURGE`
Parameters	None
Description	This function compacts the Gerber files that are loaded into GerbTool. GerbTool doesn't actually erase data from memory during edits, and therefore memory may become fragmented and less efficient. The purge command deleted these items from the Gerber database and removes the contents of the undo queue.
Example	The following is an example of using the undo command.

```
GETYESNO "Purge?", $value
IF $value == $$YES
    PURGE
END
```

ROTATE

Purpose — Allows a macro to rotate items.

Menu command — *Edit/Rotate*

Syntax

```
ROTATE
    DEGREES   exp
    CENTERED  yesno
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
    GO        [x1, y1, x2, y2 ...]
END
```

Parameters

DEGREES	An expression indicating the number of degrees to rotate the selected objects in a counter clockwise direction.
BY	An expression indicating how to perform the rotation. Valid values are: 0=item, 2=group.
DCODE	An expression indicating a D-Code filter to use for the command. If zero is used, all D-Codes may be affected by the command.
LAYER	An expression indicating a layer filter to use for the command. If zero is used, all visible layers may be affected by the command.
GO	A variable number of coordinates used by the system to complete the command.

Description — This function is used to rotate a selection of items. If window mode is selected the first 2 x, y coordinate pairs given are used to specify the window, and in group mode the select group is used. If CENTERED is set to $$NO, the next coordinate is used to specify a pivot point.

Example

The following example obtains a window and rotates all arcs and draws but not the flashes.

```
ROTATE
    DEGREES   90.0
    CENTERED  $$YES
    BY        $$WINDOWMODE
    BOUNDARY  $$YES
    FLASHES   $$NO
    DRAWS     $$YES
    ARCS      $$YES
    LAYER     0
    DCODE     0
    GETWINDOW"Enter Window to rotate", \
              $lx,$ly,$ux,$uy
    GO $lx,$ly,$ux,$uy
END
```

SELECTCRITERIA

Purpose

Allows a macro to modify the selection criteria that is used by many of the editing commands.

Menu command

All editing commands.

Syntax

```
SELECTCRITERA
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
END
```

Parameters

BY

An expression that describes which items are to be selected. Valid values are: 0=item, 1=window, 2=group, 3=layer, 4=net.

DCODE

An expression that describes the D-Code you are interested in operating on. Use a value of zero to enable selection of all D-Codes.

LAYER

An expression that describes the layer you select objects from. Use a value of zero to select from all visible layers.

Description

This function allows you to set the selection criteria that is shared by most GerbTool editing commands. Note that since this is a block command, only the variables that you want to change need to be given when using the command.

Example

The following example sets the selection criteria to window mode, includes items that cross the window boundary, includes flashes, excludes draws and arcs, sets the layer to the value of the $layer variable, and sets the D-Code to zero.

```
SELECTCRITERA
    BY        $$WINDOWMODE
    BOUNDARY  $$YES
    FLASHES   $$TRUE
    DRAWS     $$FALSE
    ARCS      $$FALSE
    LAYER     $layer
    DCODE     0
END
```

SELECTGROUP

Purpose

Allows a macro to manipulate the select groups used in GerbTool.

Menu command

Edit/Select

Syntax

```
SELECTGROUP
    BY        exp
    BOUNDARY  yesno
    FLASHES   yesno
    DRAWS     yesno
    ARCS      yesno
    DCODE     exp
    LAYER     exp
    MODE      exp
    GO        [lx, ly, ux, uy...]
END
```

Parameters

`BY` An expression that describes how to perform the selection. Valid values are: 0=item, 1=window, 4=net.

`DCODE` An expression that describes the D-Code you are interested in operating on. Use a value of zero to enable selection of all D-Codes.

`LAYER` An expression that describes the layer you select objects from. Use a value of zero to select from all visible layers.

`MODE` An expression that describes which action to perform on the select set. Valid values are: 0=reset, 1=add, 2=remove, 3=invert.

`GO` A variable number of variables that are used to pass coordinates to this command. If you are selecting by item or net, each pair of x, y coordinates will be used to select an item. If you are selecting by window, each two pair of coordinates will be used to determine the window the command will use. If you are resetting or inverting the select group, no coordinates need to be passed into this command.

Description

This function is the macro interface into the Select Group command of GerbTool. Note that since this is a block command, only the variables that you want to change need to be given when using the command. The system variable `$$SELGRPCNT` is used to return the current number of items in the select group.

Example

The following example performs a selection by window of all the flashes on a given layer. Note how this example also shows both, how only the variables to be changed need to be mentioned, but also how they can be called multiple times in the same block.

```
SELECTGROUP
    By          $$WINDOWMODE
    Flashes     $$TRUE
    Draws       $$FALSE
    Arcs        $$FALSE
    Layer       $layer
END

REPEAT          $$TRUE
    GetWindow "Enter Component Window", \
              $lx,$ly, $ux,$uy
    SELECTGROUP
       MODE  0      #reset the select set
       GO
       MODE  1      #select by window
       GO    $lx,$ly,$ux,$uy
    END
END
```

TEXT

Purpose — Allows a macro insert text.

Menu command — *Edit/Add/Text*

Syntax

```
TEXT
    HEIGHT          exp
    WIDTH           exp
    ROTATE          yesno
    SLANT           exp
    MIRROR          yesno
    FILE            string
    LINESPACE       exp
    CHARSPACE       exp
    GO              [x, y, ...]
END
```

Parameters

`HEIGHT` — An expression that describes the height of the characters in the text to be added.

`WIDTH` — An expression that describes the width of the characters in the text to be added.

`SLANT` — An expression that describes the amount of slant, in degrees, you want to apply to each individual character.

`FILE` — A string that gives the name of the file that contains the text to be added.

`LINESPACE` — An expression that gives the spacing between lines of text, where 1.0 represents single-spaced text, 2.0 represents double-spaced, and so on.

`CHARSPACE` — An expression that represents the spacing between characters in the added text, where 1.0 gives normal character spacing, 2.0 gives double spaces between characters, and so on.

`GO` — A variable number of coordinates specifying where to add the text. If more than one set of coordinates are given, the same text will be added to each location.

Description — This function is uses to add text to a Gerber file. The text to be added is in read in from a file whose name is specified in the `FILE` variable.

Example

The following example places a logo one-half inch in from the lower left corner of the board.

```
GETEXTENTS -1, $lx, $ly, $ux, $uy
CALC $lx = $lx + 0.5
CALC $ly= $ly+ 0.5
TEXT
    HEIGHT 0.25
    WIDTH  0.25
    FILE   "logo.txt"
    GO $lx, $ly
END
```

TRANSCODE

Purpose — Allows a macro to change the D-Codes of selected items.

Menu command — *Edit/DCODE/Transcode*

Syntax

```
TRANSCODE
    NEWDCODE exp
    BY       exp
    BOUNDARY yesno
    FLASHES  yesno
    DRAWS    yesno
    ARCS     yesno
    DCODE    exp
    LAYER    exp
    GO       [x1, y1, x2, y2 ...]
END
```

Parameters

`NEWDCODE` — An expression indicating the new D-Code that you want all selected items changed to.

`BY` — An expression indicating how to perform the transcode. Valid values are: 0=item, 1=window, 2=group.

`DCODE` — An expression indicating a D-Code filter to use when transcoding. If zero is used, all D-Codes may be affected by the command.

`LAYER` — An expression indicating a layer filter to use for transcoding. If zero is used, all visible layers may be affected by the command.

`GO` — A variable number of coordinates used by the system to complete the command.

Description — This function is used to change the apertures or transcode a selection of items. If window mode is selected, the first two x, y coordinate pairs given are used to specify the window; if item mode is used, the first pair is used to select the item to copy; and in group mode, the select group is used and none of the coordinate pairs are used to select the items.

Example

The following example obtains a window from the user and changes all of the D10 flashes in the window to D15.

```
TRANSCODE
    NEWDCODE 15
    By       $$WINDOWMODE
    BOUNDARY $$YES
    FLASHES  $$YES
    DRAWS    $$NO
    ARCS     $$NO
    LAYER 0  # Change all visible layers
    DCODE 0  # Change all D-Codes

    GETWINDOW"Enter Window to change", \
             $lx,$ly,$ux,$uy
    GO $lx,$ly,$ux,$uy
END
```

Environment functions

ACTIVELAYER

Purpose	Allows the user to set the active layer.
Menu command	*Layers/Active*
Syntax	`ACTIVELAYER layer`
Parameters	
`layer`	An expression indicating the layer to make active.
Description	This function allows the user to change the active layer inside a macro. The active layer is the layer where any newly created objects are placed. If the macro attempts to set the active layer to a layer that doesn't exist, the function returns `$$FALSE`, otherwise it returns `$$TRUE`.
Example	The following example loops through a number of layers, and adds a flash to each layer by changing the active layer to the value of the loop.

```
SET $layer = 0
REPEAT $layer <= $$MAXLAYERS
    ACTIVELAYER $layer
    IF $$STATUS
       ADDFLASH $lx,$ly
    END
    CALC $layer = $layer + 1
END
```

BKCOLOR

Purpose	Allows the user to set the background color on the screen.
Menu command	*Options/Bg Color*
Syntax	`BKCOLOR color`
Parameters	
`color`	A text string representing the color to set the background to.
Description	This function allows the user to change the color of the background screen. The color string can be any of the colors given in the color file COLOR.RGB, even if these colors are not available in GerbTool's color chooser.
Example	The following example sets the background color in GerbTool to a rather unusable color.

```
MACRO TESTCOLOR
        BKCOLOR "PapayaWhip"
ENDMACRO
```

CURRENTDCODE

Purpose	Allows the user to set the current D-Code.
Menu command	*Apertures/Change*
Syntax	`CURRENTDCODE dcode`
Parameters	
`dcode`	An expression indicating the D-Code to make current.
Description	This function allows the user to change the current D-Code inside a macro. The current D-Code is the aperture that any newly created objects are created with. If the macro attempts to set the current D-Code to an aperture that doesn't exist, the function returns `$$FALSE`, otherwise it returns `$$TRUE`.
Example	The following example loops through a number of layers, and adds a different flash to each layer by changing the active layer to the value of the loop, and incrementing the current D-Code.

```
REPEAT      $layer <= $$MAXLAYERS
    ACTIVELAYER    $layer
    CURRENTDCODE$dcode
    IF         $$STATUS
       ADDFLASH    $lx,$ly
    END
    CALC       $layer = $layer + 1
    CALC       $dcode = $dcode + 1
END
```

EXTENSIONS

Purpose

To change the default file extensions associated with the different file types supported by GerbTool.

Menu command

Options/Defaults

Syntax

```
EXTENSIONS
    GERBER          string
    APLISTS         string
    DESIGNS         string
    DRILL           string
    MILL            string
    HPGL            string
    POSTSCRIPT      string
    TOOL            string
    LASERJET        string
END
```

Description

This command allows you to control the default file extensions that GerbTool will use when searching for files of a particular type.

Example

The following example modifies the default extensions.

```
EXTENSIONS
    GERBER    "lgr"
    APLISTS   "apr"
    DESIGNS   "job"
    HPGL      "plt"
END
```

FILESPATH

Purpose	To change the location where GerbTool looks for files.
Menu command	*Options/Defaults*
Syntax	`FILESPATH string`
Parameters	
`string`	A text string defining the new path.
Description	This function accepts a character string and uses it to change the path GerbTool uses to find the Gerber files it is working on. This can be used to change the location where files are to be saved or loaded from.
Example	The following example changes the current directory to a temporary one so the files can be saved there and not overwrite the original ones.

```
FILESPATH "C:\temp\gerbers"
```

FILMBOX

Purpose	To adjust the size and color of the film box.
Menu command	*Options/Film Box*
Syntax	`FILMBOX xsize, ysize, color`
Parameters	
`xsize`	An expression giving the width of the film box.
`ysize`	An expression giving the height of the film box.
`color`	A string giving the color that the film box is to be drawn with.
Description	This command is used to change the size and color of the film box that GerbTool displays. Since the lower left corner of the film box is always at the 0,0 coordinate, only the width and height need to be given. The color string can be any of the colors given in the color file COLOR.RGB, even if these colors are not available in GerbTool's color chooser.
Example	The following example sets the film box to 8.5 by 11 and the color of the film box to a particular color.

```
FILMBOX 8.5, 11.0, "SeaGreen"
```

FLAGS

Purpose	To allow the user to modify the flags variable associated with a design.
Menu command	There are no menu commands to affect this setting: the nested command CTRL+F can be used, however.
Syntax	`FLAGS exp`
Parameters	
`exp`	An expression describing the value to assign to the flags field.
Description	This function is used to modify the system's environment flags. The flags field controls GerbTool settings that are either too infrequently used or too recently added to the system to have a more conventional menu access. This command is intended for use by software developers. The values used by the flags may change at any time, and improper usage may have unpredictable results.
Example	No example is provided for this function.

FORMAT

Purpose

To change the file format parameters for a specific file type.

Menu command

File/Format or *Layers/Edit*

Syntax

```
FORMAT        string
    DIALECT                 string
    M.N                     exp, exp
    MODE                    string
    ZEROSUPPRESSION         string
    TERMINATOR              string
    CHARSET                 string
    METRIC                  yesno
    MODEL                   yesno
    NETS                    yesno
    USERDATA                yesno
    GCMDS                   yesno
    ARCS360                 yesno
    ARCSMODAL               yesno
    HONORCRLF               yesno
    COMMENTS                yesno
END
```

Parameters

FORMAT	The selected file type: Gerber, Drill, or Mill.
DIALECT	A supported dialect such as Excellon or RS247X.
M.N	A pair of expressions specifying the m.n of the specified format.
MODE	A (Absolute) or I (Incremental).
ZEROSUPPRESSION	L (Leading), T (Trailing), or N (None).
TERMINATOR	A string indicating which characters should be output at the end of every line of output when writing a file of this type.
CHARSET	ASCII, EBCDIC, or EIA.

Description

This command allows you to control the format of the specified file type. Normally, this command operates on a global format. To operate on a local format you can use this command from within a LAYERN block.

Example

The following example modifies the format associated with the layer specified in the `$lyrno` variable regardless if it is a local or global format.

```
LAYERN $lyrno
    FORMAT "Gerber"
       NETS $$YES
       METRIC $$NO
       MODAL  $$YES
       MODE "A"
       TERMINATOR "*\r\n"
    END
END
```

GRIDSIZE

Purpose	To allow the user to change the size of the display grid.
Menu command	*Options/Grid*
Syntax	`GRIDSIZE xsize, ysize`
Parameters	
`xsize`	An expression specifying the horizontal spacing of the grid.
`ysize`	An expression specifying the vertical spacing of the grid.
Description	This command allows the user to change the grid that is displayed in GerbTool. Note that the macros themselves do not make use of the grid.
Example	The following macro changes the grid 0.1 inches horizontally and 0.15 inches vertically.

```
GRIDSIZE 0.1, 0.15
```

GRIDSNAP

Purpose	To allow the user to change the grid snap settings.
Menu command	*Options/Grid*
Syntax	`GRIDSNAP yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This command allows the user to change the grid snap settings that are used in GerbTool. Note that the macros themselves do not make use of the grid.
Example	The following macro turns on grid snap. `GRIDSNAP $$YES`

GRIDVISIBLE

Purpose	To allow the user to change the visibility of the GerbTool grid.
Menu command	*Options/Grid*
Syntax	`GRIDVISIBLE yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This command allows the user to change the grid visibility settings that are used in GerbTool. Note that the macros themselves do not make use of the grid.
Example	The following macro turns on grid visibility.

```
GRIDVISIBLE $$YES
```

HILICOLORS

Purpose — To change the highlight colors used by the system.

Menu command — *Options/Defaults*

Syntax

```
HILICOLORS
    QUERY     color
    SELECT    color
    DRC       color
END
```

Parameters

`QUERY` — A text string indicating the color to use when highlighting items while using the *Query/Item Info* command.

`SELECT` — A text string indicating the color to use when highlighting items while using the *Edit/Select* command.

`DRC` — A text string indicating the color to use when performing a DRC.

Description — This function is used to change the highlight colors that are used by the system. The color string can be any of the colors given in the color file COLOR.RGB, even if these colors are not available in GerbTool's color chooser. Since this is a block command, only the variables that you want to change need to be used.

Example — The following macro sets the color used to display DRC errors and select groups.

```
HILICOLORS
    SELECT    $selectColor
    DRC       "vga16magenta"
END
```

LAYERN

Purpose: To change the settings of a particular Gerber layer.

Menu command: *Layers/Edit*

Syntax:

```
LAYERN        layer
    FILE              string
    LYRNAME           string
    APLIST            string
    VISIBILITY        exp
    FLASHCOLOR        string
    DRAWCOLOR         string
    TYPE              string
    POLARITY          string
    KEY               exp
    FTYPE             string
    EXTENTS           lx, ly, ux, uy
    NETID             exp
    VIRTUAL           exp
END
```

Parameters

`LAYERN`	An expression indicating the layer to process.
`FILE`	A text string indicating the filename to associate with this layer.
`LYRNAME`	A text string indicating the layer name to associate with this layer.
`APLIST`	A text string indicating the aperture list name to associate with this layer.
`VISIBILITY`	An expression used to control the visibility of this layer. Valid values are: 0 (off), 1 (on), and 2(ref).
`FLASHCOLOR`	A text string indicating the color to use when displaying flashes from this layer.
`DRAWCOLOR`	A text string indicating the color to use when displaying draws from this layer.
`TYPE`	A text string indicating the type for this layer. Valid values are Top, Bottom, Inner, Plane, Composite.
`POLARITY`	A text string indicating the polarity of this layer in composite formats such as 274X. Valid settings are `dark` and `clear`.

KEY	An expression used to indicate the key field used for this layer in composite formats such as 274X.
FTYPE	A text string used to indicate the file type for this layer. This variable is for future expansion and as such the only valid value currently is Gerber.
EXTENTS	A quad of expressions used to set the extents of this layer. Note that these settings are only temporary, and can be changed when another function is called.
NETID	An expression used to indicate the net id for this layer. This value is used to determine layer to layer netlist accuracy, and should not generally be modified.
VIRTUAL	An expression used to indicate whether the layer should be included in a virtual panelization and step and repeat patterns.

Description

This function is used to set many of the parameters concerning individual layers in GerbTool. Since this is a block command, only the variables that you want to change need to be used.

Example

The following example obtains the visibility of a layer, stores it in the layers net id and then turns the visibility of the layer off. The stored information can then be used later to restore the layers visibility to its original condition. Note how only the variables for the information we want to change are included in the command.

```
GetLayer $Layer,$fn,$ln,$an,$vis,$fc, \
       $dc,$type,$pol,$key,$ft, \
       $lx,$ly,$ux,$uy, $netid
LAYERN $Layer     #Set this layers info
    Netid $vis    #save true visibility
    Visibility 0 #turn the layers vis off
END
```

MAPPATH

Purpose	To change the location where GerbTool looks for its aperture lists.
Menu command	None
Syntax	`MAPPATH path`
Parameters	
`path`	A string defining the new path.
Description	This function accepts a character string and uses it to change the path GerbTool uses to find its aperture lists. This can be used to change the location where files are to be saved or loaded from.
Example	The following example changes directory where GerbTool looks for aperture lists to a temporary one.

```
MAPPATH "C:\temp\aperturs"
```

NETID

Purpose — To allow the user to change the net id number for a layer.

Menu command — None

Syntax — `NETID exp`

Parameters

`exp` — An expression representing the new value to set the netid to.

Description — This function allows the user to set the net id value for the currently active layer. Since these values are only used internally to keep the layer to layer net lists in sync, there are few instances where the user should attempt to modify this value. One thing this function can be used for in a macro is for storing information with a layer, that can be retrieved and used later. Doing this destroys the layer to layer netlist information however, and the macro writer should be sure to warn the user of this fact.

Example — The following example obtains the visibility of a layer, stores it in the layers net id and then turns the visibility of the layer off. The stored information can then be used later to restore the layer's visibility to its original condition.

```
GetLayer $Layer,$fn,$ln,$an,$vis,$fc,$dc, \
        $type,$pol,$key,$ft, $lx,$ly,$ux, \
        $uy $netid
LAYERN $Layer          # Chg settings for this lyr
    Netid $vis         # save true visibility
    Visibility 0       # turn lyr visibility off
END
```

OFFSETS

Purpose

To change the offsets applied to files loaded and merged into the system.

Menu command

File/Offsets

Syntax

```
OFFSETS xoff, yoff
```

Parameters

`xoff`

An expression describing the offset to be applied to the x coordinates of objects.

`yoff`

An expression describing the offset to be applied to the y coordinates of objects.

Description

This function is used to apply an offset to any files loaded or merged into the system.

Example

The following example merges in a Gerber file 2 inches to the left and 3 inches up from where it would normally be located.

```
OFFSETS -2.0, 3.0
MERGEGERBER  "infile.gbr"
       # reset the offsets so they will
       # not affect future merging.
OFFSETS 0,0
```

OVERLAYMODE

Purpose	To change the state of the overlay mode setting.
Menu command	*Options/Overlay*
Syntax	`OVERLAYMODE yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This function is used to change the state of the overlay mode that GerbTool is currently using.
Example	The following example turns overlay mode off.
	`OVERLAYMODE $$YES`

PREVIOUSVIEW

Purpose	Allows a macro to redefine the previous view.
Menu command	*View/Previous*
Syntax	`PREVIOUSVIEW lx, ly, ux, uy`
Parameters	
`lx`	An expression defining the x coordinate of the lower left corner of the view you are defining.
`ly`	An expression defining the y coordinate of the lower left corner of the view you are defining.
`ux`	An expression defining the x coordinate of the upper right corner of the view you are defining.
`uy`	An expression defining the y coordinate of the upper right corner of the view you are defining.
Description	This function allows a macro to redefine the previous view that is held internally by GerbTool. Normally this value is updated automatically whenever the user performs any of the View commands. One use of this command is to restore a view that was saved when first starting a macro. This way, once the macro has finished, the users viewing ability will not be changed.
Example	The following example sets the previous view to pre-calculated values.

```
PREVIOUSVIEW $lx, $ly, $ux, $uy
```

SCALE

Purpose — To allow a macro to change the scale of a file that is loaded or merged into GerbTool.

Menu command — *File/Offsets*

Syntax — `SCALE xscale, yscale`

Parameters

`xscale` — An expression giving the scale factor to be applied in the x dimension.

`yscale` — An expression giving the scale factor to be applied in the y dimension.

Description — This command allows the user to change the scale of files that are loaded or merged into GerbTool. Note that this only effects the coordinates of the Gerber file. If you want to modify the sizes of apertures, you should use the `DCODESCALE` command.

Example — The following example merges in a Gerber file and increases the scale of the file by 0.5 percent in the x dimension. Small changes like this are often used to make allowances for shrinkage.

```
SCALE        1.005, 0.0
MERGEGERBER  "infile.gbr"
OFFSETS 0,0  #reset the offsets so
             # they will not affect
             # future merging.
```

SHOWERRORS

Purpose	Controls the state of the show errors option in GerbTool.
Menu command	*Options/Show Errs*
Syntax	`SHOWERRORS yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This function controls the state of the Show errors setting in GerbTool.
Example	The following example turns on the show errors setting.

```
SHOWERRORS $$YES
```

SKETCHMODE

Purpose	To change the state of the sketch mode setting.
Menu command	*Options/Sketch*
Syntax	`SKETCHMODE yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This function is used to change the state of the sketch mode that GerbTool is currently using.
Example	The following example turns sketch mode off.

```
SKETCHMODE $$NO
```

UNDO

Purpose	Controls the state of the system undo.
Menu command	*Options/Undo*
Syntax	`UNDO yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This function allows the user to toggle the state of the undo queue inside a GerbTool macro. Note, that turning off undo destroys all of the current undo information.
Example	The following example turns disables the undo queue and destroys any existing undo information.

```
UNDO $$NO
```

VIEWCOMPOSITES

Purpose	To change the setting that determines how Composite 274X files are viewed.
Menu command	*Layers/Edit* (View composites button)
Syntax	`VIEWCOMPOSITES yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This function is used to change the state of the View Composites setting that GerbTool is currently using. This setting only affects the viewing of composited 274X Gerber files.
Example	The following example turns composite viewing on.

```
VIEWCOMPOSITES $$YES
```

VIEWMETRIC

Purpose	To control whether files are view*ed* in Metric or Imperial units.
Menu command	*Options/Metric*
Syntax	`VIEWMETRIC yesno`
Parameters	
`yesno`	Your choice of the values `$$YES` and `$$NO`.
Description	This command controls whether the coordinates used to display the Gerber files are presented in Metric or Imperial units.
Example	The following macro turns off metric mode so the files will be displayed in inches.

```
VIEWMETRIC $$NO
```

File handling functions

FILECLOSE

Purpose	To close a previously opened disk file.
Menu command	None
Syntax	`FILECLOSE fid`
Parameters	
`fid`	A file id returned by a previous call to `FILEOPEN`.
Description	This function closes a file that was previously opened by a call to `FILEOPEN`.
Example	See `FILEOPEN` for a example of closing a file.
See also	FILEOPEN, FILEREAD, FILEWRITE

FILEOPEN

Purpose	To open a disk file for reading or writing.
Menu command	None
Syntax	`FILEOPEN fid, filename, mode`
Parameters	
`fid`	A numeric variable that will receive a file identification number.
`filename`	A string containing the filename of a disk file to open.
`mode`	A string containing a mode string of `r` for reading, `w` for writing, and `a` for append.
Description	This function attempts to open the specified disk file in the mode indicated by the mode parameter. The value returned in the `fid` parameter can be used in subsequent calls to `FILEREAD`, `FILEWRITE` and `FILECLOSE`. WARNING: Opening a existing file in write mode will destroy any data that previously existed in the file.

Example	The following example opens a file for writing, processes the file, and then closes it.
	
	`FILEOPEN $fid, "MYCMD.LOG", "w"`
	`...file processing...`
	`FILECLOSE $fid`

See also	FILECLOSE, FILEREAD, FILEWRITE

FILEREAD

Purpose	To read a line of data from a disk file.
Menu command	None
Syntax	`FILEREAD fid, format, output_variables`
Parameters	
`fid`	A file id returned by a previous call to `FILEOPEN`.
`format`	A string describing the fields in the input line.
`output_ variables`	A list of variables that will receive the data from the input line.
Description	This function reads a line from the input file and converts the data into individual string and numeric variables. The first parameter is the file id created by a previous call to `FILEOPEN`. The remaining parameters are exactly like those in the `STRREAD` command.
Example	The following example reads a line from a file and parses it into a string variable and two numeric variables. `FILEREAD $fid, "%s %n %n", $ref, $x, $y`
See also	FILEOPEN, FILECLOSE, FILEWRITE

FILEWRITE

Purpose	To output a formatted string to a disk file.
Menu command	None
Syntax	`FILEWRITE fid, format, input_variables`
Parameters	
`fid`	A file id returned by a previous call to `FILEOPEN`.
`format`	A string describing the variables that will used to construct the output line.
`input_ variables`	A list of variables that will provide the data for the output line.
Description	This function writes to the output file converting the input data. The first parameter is a file id created by a previous call to the `FILEOPEN` command. The remaining parameters are exactly like those of the `STRWRITE` command.
Example	The following example outputs a line to a file consisting of a string and two numbers.

```
FILEWRITE    $fid, "%s %n %n", $ref, \
             $x + 10.5, $y
```

See also FILEOPEN, FILECLOSE, FILEREAD

File merging functions

MERGEDRILL

Purpose	To merge (import) a drill file into the active layer.
Menu command	*File/Import/NC Drill*
Syntax	`MERGEDRILL filename`
Parameters	
`filename`	A string defining the disk file to import.
Description	This function imports a NC Drill file into the currently active layer using the current Drill format setup. All drill hits become Gerber flashes appended to the active layer.

Example	The following example changes the active layer and then imports a NC Drill file.
	`ACTIVELAYER $drilllayer`
	`MERGEDRILL $drillfile`

See also	FORMAT, MERGEGERBER, MERGEHPGL

MERGEGERBER

Purpose	To merge a Gerber file into the active layer.
Menu command	*File/Merge/Gerber*
Syntax	`MERGEGERBER filename`
Parameters	
`filename`	A string defining the disk file to merge.
Description	This function merges a Gerber file into the currently active layer using the current Gerber format associated with the active layer.
Example	The following example changes the active layer and then merges in a NC Gerber file.

```
ACTIVELAYER $mergelayer
MERGEGERBER $newfile
```

See also FORMAT, MERGEDRILL, MERGEHPGL

MERGEHPGL

Purpose

To merge (import) a HPGL plot file into the active layer.

Menu command

File/Import/HPGL

Syntax

```
MERGEHPGL    filename
    PLOTSIZE        string
    ROTATE          yesno
    PEN             exp, exp
    GO              string
END
```

Parameters

`filename` A string defining the disk file to import.

`PLOTSIZE` A string defining the expected size of the plotter that the input file was generated for. Use S (small) for A/B plotters and L (large) for C/D/E plotters.

`PEN` A pair of expressions matching a pen number to a D-Code.

`GO` A string indicating the filename of the file to import.

Description

This function imports an HPGL file into the currently active layer using the current HPGL format setup. All data becomes Gerber flashes/draws on the active layer.

Example

The following example changes the active layer and then imports a HPGL file.

```
ACTIVELAYER $hpgllayer
MERGEHPGL $hpglfile
```

See also

FORMAT, MERGEDRILL, MERGEGERBER

Mathematical functions

ABS

Purpose	Calculates the absolute value of a given value.
Menu command	None
Syntax	`ABS numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the absolute value of the expression on the right hand side of the equal sign.
`exp`	The expression to take the absolute value of.
Description	This function allows the user to calculate the absolute value of a number. This is a number with the negative sign removed if one exists. The resulting value will be assigned to `numvar`. While the object on the right hand side of the equal sign can be any numeric expression, the value on the left of the equal sign must be a numeric variable.
Example	The following example takes the absolute value of -7.0 and assigns its value (7.0) to the variable `$answer`.

```
ABS $answer = -7.0 * $zaxis
```

ARRAY

Purpose	To create an array of numeric variables.
Menu command	None
Syntax	`ARRAY $name(size)`
Parameters	
`name`	The name of the variable.
`size`	An expression indicating the size of the array.
Description	This function creates an array of numeric variables that can be accessed with one name and a index expression.
Example	The following example shows how to fill an array of items.

```
ARRAY $v(50)
CALC $index = 1
REPEAT $index <= 50
    GETVALUE "Enter Next Value: ", $val
    CALC $v($index) = $val
END
```

See also STRARRAY

ASIN

Purpose	Calculates the principal value of the arcsin function of a given value.
Menu command	None
Syntax	`ASIN numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the arcsin of exp.
`exp`	An expression to take the arcsin of.
Description	This function allows the user to calculate the arcsin of a number. The resulting value will be assigned to numvar While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable. Note that an error occurs if the argument is less than -1.0 or greater than 1.0.
Example	The following example takes the arcsin of 0.5984271 and assigns its value (2.5 radians) to the variable $answer.

```
ASIN $answer = 0.5984271
```

ATAN

Purpose	Calculates the principal value of the arctangent function of a given value.
Menu command	None
Syntax	`ATAN numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the arctangent of exp.
`exp`	The expression to take the arctangent of.
Description	This function allows the user to calculate the arctangent of a number. The resulting value will be assigned to numvar. While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable.
Example	The following example takes the arctangent of -0.7470223 and assigns its value (2.5 radians) to the variable `$answer`.

```
ATAN $answer = -0.7470223
```

CALC

Purpose	To perform mathematical calculation.
Menu command	None
Syntax	`CALC numvar = exp1 operator exp1`
Parameters	
`numvar`	A numeric variable that is assigned the results of the mathematical operation.
`exp1`	The first variable or literal used in the calculation.
`operator`	The mathematical operation to perform. Valid operations are + - / *.
`exp2`	The second variable or literal used in the calculation.
Description	The calc function allows the user to perform mathematical calculations inside a macro. While the objects on the right side of the equal sign can be either numeric variables or numeric literals, the value on the left of the equal sign must be a numeric variable. Like any other programming language, it is possible for a variable to be on both sides of the equal sign at the same time. The result of doing this is that the value of the variable is first used in the calculation, and the answer is placed back into the variable.
Example	The following example takes the contents of the variable $size, multiplies it by 2.5 and assigns the result to the variable `$result`.

```
CALC $result = $size * 2.5
```

CEIL

Purpose	Calculates the smallest integer not less than the input value.
Menu command	None
Syntax	`CEIL numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the ceiling value of the expression on the right hand side of the equal sign.
`exp`	The expression to take the absolute value of.
Description	This function allows the user to find the largest integer value of a number.
Example	The following example takes the ceiling value of -7.658 and assigns its value (-7.0) to the variable `$answer`.

```
CEIL $answer = -7.658
```

CSIN

Purpose	Calculates the trigonometric cosin value of a given value.
Menu command	None
Syntax	`CSIN numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the cosin of exp.
`exp`	The expression to take the cosin of.
Description	This function allows the user to calculate the cosin of a number. The resulting value will be assigned to numvar While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable. Note that this functions assumes the value given it is represented in radians and not degrees.
Example	The following example takes the cosin of 2.5 radians and assigns its value (-0.8011436) to the variable `$answer`.

```
CSIN $answer = 2.5
```

FLOOR

Purpose	Calculates the largest integer not greater than the input value.
Menu command	None
Syntax	`FLOOR numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the floor value of the expression on the right hand side of the equal sign.
`exp`	The expression to take the floor value of.
Description	This function allows the user to find the smallest integer value of a floating point number.
Example	The following example takes the value of -7.658 and assigns the integer value (-8.0) to the variable `$answer`.

```
FLOOR $answer = -7.658
```

ROUND

Purpose	Calculates the nearest integer of the input value.
Menu command	None
Syntax	`ROUND numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the nearest integer value of the expression on the right hand side of the equal sign.
`exp`	The expression to take the absolute value of.
Description	This function allows the user to round a floating point number to its nearest integer value.
Example	The following example rounds the input of 7.658 and assigns the result of 8.0 to the variable `$answer`.

```
ABS $answer = 7.658
```

SET

Purpose	Creates and assigns a value to a numeric variable.
Menu command	None
Syntax	`SET numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the value of exp.
`exp`	An expression whose value is assigned to numvar.
Description	The `SET` function allows the user to assign a value to a variable. While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable.
Example	The following example assigns the contents of the variable `$value` to the variable `$var1` and the number 4.12 to the variable `$var2`.

```
SET $var1 = $value
SET $var2 = 4.12
```

SETGLOBAL

Purpose	Creates and assigns a value to a global numeric variable.
Menu command	None
Syntax	`SETGLOBAL numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the value of exp.
`exp`	An expression whose value is assigned to numvar.
Description	The `SETGLOBAL` function allows the user to assign a value to a global variable. Global variables must be defined before any other variables are defined in a macro. Once set, global variables last the duration of your GerbTool session and are available to all macros. Global variables are persistent from one macro invocation to another.
Example	The following example shows some typical global variable assignments.

```
SETGLOBAL $gvar1 = 1.75
SETGLOBAL $gvar2 = $gvar1 * 4.12
```

SIN

Purpose	Calculates the trigonometric sin of a value.
Menu command	None
Syntax	`SIN` numvar = exp
Parameters	
`numvar`	A numeric variable that is assigned the sin of exp.
`exp`	The variable or literal to take the sin of. This value is assumed to be in radians.
Description	This function allows the user to take the sin of a number. The resulting value will be assigned to numvar. While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable. Note that this functions assumes the value given it is represented in radians and not degrees.
Example	The following example takes the sin of 2.5 radians and assigns its value (0.5984721) to the variable `$answer`.

```
SIN $answer = 2.5
```

SQRT

Purpose	Calculates the square root of a value.
Menu command	None
Syntax	`SQRT numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the square root of exp.
`exp`	The variable or literal to take the square root of.
Description	This function allows the user to take the square root of a number. The resulting value will be assigned to numvar While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable. Attempting to take the square root of a negative number will result in an error.
Example	The following example takes the square root of 36 and assigns its value (6) to the variable `$answer`.

```
SQRT $answer = 36
```

TAN

Purpose	Calculates the trigonometric tangent of a given value.
Menu command	None
Syntax	`TAN numvar = exp`
Parameters	
`numvar`	A numeric variable that is assigned the tangent of exp.
`exp`	The variable or literal to take the tangent of.
Description	This function allows the user to calculate the tangent of a number. The resulting value will be assigned to `numvar` While the object on the right side of the equal sign can be either a numeric variable or numeric literal, the value on the left of the equal sign must be a numeric variable. Note that this functions assumes the value given it is represented in radians and not degrees. Also, attempting to produce the tangent of odd multiples of $\pi/2$ will produce an error.
Example	The following example takes the tangent of 2.5 radians and assigns its value (-0.7470223) to the variable `$answer`.

```
TAN $answer = 2.5
```

Plotting functions

PLOTHPGL

Purpose: To create an HPGL plot.

Menu command: *Files/Plot/HPGL*

Syntax:

```
PLOTHPGL
    OUTFILE         string
    MEDIASIZE       exp, exp
    SCALE           exp
    MODE            string
    ROTATE          yesno
    BORDER          yesno
    BORDERTEXT      string
    BORDERPEN       exp
    PENWIDTH        exp
    PENSPEED        exp
    PADSONLY        yesno
    GO
END
```

Parameters

`OUTFILE` A text string specifying the name of the file to place the plot.

`MEDIASIZE` A pair of expressions specifying the horizontal and vertical size of the printable area of the output media.

`OFFSETS` A pair of expressions specifying the horizontal and vertical offsets. This can be used to position the plot at a specific location on the media.

`SCALE` An expression specifying the scale factor to apply to the plot.

`MODE` A string indicating the plotting mode. Choose from "S" (Sketch), "O" (Outline), or "F" (Fill).

`BORDERTEXT` This parameter allows you to specify the text that will appear in the border, if borders are enabled. GerbTool looks for the key words $DATE, $TIME, $DESIGN and $PROG. NOTE: These are not variable names. If GerbTool finds any of these keywords they will be replaced with the appropriate text. All other text specified will be included in the border verbatim.

BORDERPEN	An expression specifying the plotter pen to use when drawing the border if enabled.
PENWIDTH	An expression specifying the size of the pens used.
PENSPEED	An expression specifying the speed of the plotter pens.
GO	Executes the command using the current parameters.

Description

This function plots the currently visible layer to a HPGL plotter.

Example

The following example plots all loaded layers, one layer per output file, using a predefined plotting scale and media size.

```
CALLMACRO "MaxLoadedLayer", $maxlyr
SET $lyr = 1
REPEAT $lyr <= $maxlyr
    ACTIVELAYER $lyr
    IF $$STATUS
        CALLMACRO "GetLayerFileName", $lfn
        SPLITPATH $lfn, $dir, $fn, $ext
        STRWRITE $fn, "%s.%.0n", $fn, $lyr
        PLOTHPGL
              OUTFILE      $fn
              MEDIASIZE $mediax, $mediay
              SCALE $fscale
              GO
        END
    END
    CALC $lyr = $lyr + 1
END
```

PLOTPS

Purpose — To create an PostScript plot.

Menu command — *Files/Plot/PostScript*

Syntax

```
PLOTPS
    OUTFILE         string
    MEDIASIZE       exp, exp
    OFFSET          exp, exp
    SCALE           exp
    MODE            string
    ROTATE          yesno
    WINDOWMODE      yesno
    PADSONLY        yesno
    BORDER          yesno
    BORDERTEXT      string
    GREYSCALE       yesno
    FITTOPAGE       yesno
    GO
END
```

Parameters

OUTFILE	A text string specifying the name of the output file.
MEDIASIZE	A pair of expressions specifying the horizontal and vertical size of the printable area of the output media.
OFFSETS	A pair of expressions specifying the horizontal and vertical offsets. This can be used to position the plot at a specific location on the media.
SCALE	An expression specifying the scale factor for the plot.
MODE	A string indicating the plotting mode. Choose from S (sketch) or F (fill).
BORDERTEXT	This parameter allows you to specify the text to appear in the border, if borders are enabled. GerbTool looks for the key words $DATE, $TIME, $DESIGN and $PROG. NOTE: These are not variable names. If GerbTool finds any of these keywords they will be replaced with the appropriate text. All other text specified will be included in the border verbatim.
GO	Executes the command using the current parameters.

Description — This function plots the currently visible layer to a PostScript plotter.

Example

The following example plots all of the visible layers using the `FITTOPAGE` option to automatically scale the image to the media size.

```
PLOTLJ
    OUTFILE         "output.ps"
    MEDIASIZE       $mediax, $mediay
    OFFSETS         0.0, 0.0
    FITTOPAGE       $$YES
    GO
END
```

Query functions

HILIDCODE

Purpose

Allows the user to highlight items in their database based on specified criteria.

Menu command

Query/Highlight/DCode

Syntax

```
HILIDCODE
    FLASHES         yesno
    DRAWS           yesno
    ARCS            yesno
    DCODE           exp
    LAYER           exp
END
```

Parameters

DCODE

An expression that describes the D-Code you are interested in highlighting. Use a value of zero to select all D-Codes.

LAYER

An expression that describes the layer you are interested in highlighting objects from. Use a value of zero to select from all visible layers.

Description

This function is used to highlight objects on the screen that match the given criteria. Note that since this is a block command, only the variables that you want to change need to be given when using the command.

Example

The following example highlights all of the flashes on layer 10 that use D-Code 45.

```
HILIDCODE
        FLASHES     $$YES
        DRAWS       $$NO
        ARCS        $$NO
        DCODE       45
        LAYER       10
END
```

HILIITEM

Purpose	To highlight an item on the display for easy recognition by a user.
Menu command	None
Syntax	`HILIITEM layer, seqno`
Parameters	
`layer`	A numeric variable containing the layer of the item.
`seqno`	A numeric variable containing the sequence number of the item.
Description	This function highlights an item using the current regular highlight color. The `layer` and `seqno` parameters would normally be returned by `GETFIRSTITEM/ GETNEXTITEM` calls.
Example	The following example scans the database highlighting all flashes.

```
REPEAT $$STATUS
    IF $type == $$FLASH
       HILIITEM $layer, $seqno
    END
    GETNEXTITEM
END
```

MEASUREE2E

Purpose	To measure the minimum distance between two database items.
Menu command	*Query/Measure/Edge 2 Edge*
Syntax	`MEASUREE2E layer, seqno1, seqno2`
Parameters	
`layer`	A numeric variable containing the layer of the item.
`seqno1`	A numeric variable containing the sequence number of a item.
`seqno2`	A numeric variable containing the sequence number of a item.
Description	This function measures the minimum distance between two database items. The calculated distance is returned in the `$$STATUS` variable. The `layer` and `seqno1/seqno2` parameters would normally be returned by `GETFIRSTITEM/GETNEXTITEM` calls.
Example	The following example scans the database measuring the distance between flashes.

```
REPEAT $$STATUS
    IF $type == $$FLASH
       MEASUREE2E $layer,$seqno,$lastSeqno
       IF $$STATUS < $minDist
             STOP "Too Close!"
       END
       Calc $lastSeqno = $seqno
    END
    GETNEXTITEM
END
```

String handling functions

STRARRAY

Purpose	To create an array of string variables.
Menu command	None
Syntax	`STRARRAY $name(size)`
Parameters	
`name`	The name of the variable.
`size`	An expression indicating the size of the array.
Description	This function creates an array of string variables that can be accessed with one name and a index expression. Each element of the array can hold up to 256 characters.
Example	The following example shows how to fill an array of items.

```
STRARRAY $s(50)
CALC $index = 1
REPEAT $index <= 50
    GETSTRING "Enter Next Line:", $str
    STRCPY $s($index), $str
END
```

See also	ARRAY

STRCAT

Purpose	To concatenate two strings.
Menu command	None
Syntax	`STRCAT destination, source`
Parameters	
`destination`	A string variable to which the source string will be appended.
`source`	A string that will be appended to the destination string variable.
Description	This function copies the source string to the end of the destination string variable.
Example	The following example appends the string literal `"400"` to the `$refdesg` string variable. The value of `$refdesg` following the `STRCAT` function would be `"U400"`.

```
STRSET $refdesg, "U"
STRCAT $refdesg, "400"
```

STRCMP

Purpose	To determine if two strings are equal.
Menu command	None
Syntax	`STRCMP string1, string2`
Parameters	
`string1`	A string.
`string2`	A string.
Description	This function compares two strings, without regard to case, and determines whether `string1` is less than, equal or greater than `string2`. A string is less than another when it would come first in the ASCII collating sequence. A value of zero indicates that the two strings are equal.
Example	In the following example $$STATUS would contain a value less than zero, indicating that ONE is less than TWO. `STRCMP "ONE", "TWO"`

STRCPY

Purpose	To make a copy of a text string.
Menu command	None
Syntax	`STRCPY destination, source`
Parameters	
`destination`	A string variable that will receive a copy of the source string.
`source`	A string that will be copied into the destination string variable.
Description	This function copies a string variable or literal into another string variable.
Example	The following example sets the value of `$refdes` to `"U400"`.

```
STRCPY $refdes, "U400"
```

STRLEN

Purpose	To calculate the number of characters in a string.
Menu command	None
Syntax	`STRLEN string`
Parameters	
`string`	A string.
Description	This function counts the length of a string in characters. The calculated length is returned in the `$$STATUS` variable.
Example	In the following example `$$STATUS` would contain the value of 12 after executing the `STRLEN` function.

```
STRLEN "Short string"
```

STRLOC

Purpose	To find an occurrence of a string within another string.
Menu command	None
Syntax	`STRLOC source, search`
Parameters	
`source`	The string to search in.
`search`	The string to search for.
Description	This function attempts to locate the search string anywhere within the source string. If found, the `$$STATUS` variable will contain the index of the first matching character within the source string.
Example	In the following example `$$STATUS` will contain the value 3 after the `STRLOC` function is executed.

```
STRSET $line, "This is a test"
STRLOC $line, "is"
```

STRREAD

Purpose	To parse a line of text into a series of variables.
Menu command	None
Syntax	`STRREAD source, format, output_variables`
Parameters	
`source`	A text string.
`format`	A string describing the format of the data fields in the input line.
`output_ variables`	A list of variables that will receive the data from the input line.
Description	This function reads the `source` string, converts the data according to the `format` string and places the converted data into individual string and numeric variables. The format string describes the position and type of each data field within the `source` string. White space characters in the source string are not converted and serve only to delimited the data fields. Within the format string `%s` matches a text string and `%n` matches a numeric value. Note: For those familiar with the C programming language, the format string is similar to the `scanf` format string, with `%n` being mapped to `%f`.

Example	The following example results in `$ref` containing `"U1"` `$x` containing 5.0, and `$y` containing 4.25.
	
	`STRSET $line = "U1 5.000 4.250"`
	`STRSET $ref = ""`
	`SET $x = 0`
	`SET $y = 0`
	`STRREAD $line, "%s %n %n", $ref, $x, $y`

See also	STRWRITE

STRSET

Purpose	Creates and assigns a value to a string variable.
Menu command	None
Syntax	`STRSET strvar, string`
Parameters	
`strvar`	A string variable that is assigned the value of string.
`string`	A string whose value is assigned to strvar.
Description	The `STRSET` function allows the user to assign a value to a string variable. While the object on the right side of the comma can be either a string variable or literal, the value on the left of the comma must be a string variable.
Example	The following example assigns the contents of the variable `$value` to the variable `$var1` and the number 4.12 to the variable `$var2`.

```
SETSTR $var1, "This is a string"
SETSTR $var2, $var1
```

STRSETGLOBAL

Purpose	Creates and assigns a value to a global numeric variable.
Menu command	None
Syntax	`STRSETGLOBAL strvar, string`
Parameters	
`strvar`	A string variable that is assigned the value of string.
`string`	A string whose value is assigned to strvar.
Description	The `STRSETGLOBAL` function allows the user to assign a value to a global variable. Global variables must be defined before any other variables are defined in a macro. Once set, global variables last the duration of your GerbTool session and are available to all macros. Global variables are persistent from one macro invocation to another.
Example	The following example shows some typical global variable assignments.

```
STRSETGLOBAL $gvar1, "A Global Str Var"
STRSETGLOBAL $gvar2, $gvar1
```

STRSUB

Purpose	To copy a portion of one string to another.
Menu command	None
Syntax	`STRSUB destination, start, count, source`
Parameters	
`destination`	A string variable that will receive the sub-string.
`start`	An expression indicating the index of the first character in the sub-string.
`count`	An expression indicating the number of characters in the sub-string.
`source`	A string that contains the sub-string.
Description	This function copies a specified portion of a string into another.
Example	In the following example `$subline` would contain the string `"sub-string"` after the `STRSUB` function executes.

```
STRSET $line, "A small substring example"
STRLOC $line, "sub"
STRSUB $subline, $$STATUS, 10, $line
```

STRTOK

Purpose	To parse a text string into individual tokens.
Menu command	None
Syntax	`STRTOK destination, delimiters, source`
Parameters	
`destination`	A string variable that will receive the string token.
`delimiters`	A string of characters that are used to separate tokens in the source string.
`source`	A string that contains the list of tokens or the numeric literal 0.
Description	This function parses the source string for sub-strings that are separated by any character present in the `delimiters` parameter. This function is meant to be initialized by a call with a valid `source` parameter. Subsequent calls are then made with the `source` parameter set to the numeric literal 0. As long as there are more tokens in the `source` parameter, this function will continue to return the next string token. The `$$STATUS` variable is set to the length of the returned token. A `$$STATUS` value of zero, therefore, indicates that there are no more tokens.
Example	In the following example, `$loops` will contain the value 5 when the `REPEAT` block completes.

```
STRSET $line, "This is a token test"
SET $loops = 0
STRTOK $token, " ", $line
REPEAT $$STATUS > 0
    Calc $loops = $loops + 1
    STRTOK $token, " ", 0
END
```

STRWRITE

Purpose	To output a formatted string to a string variable.
Menu command	None
Syntax	`STRWRITE destination, format, input_variables`
Parameters	
`destination`	A string variable that will receive the formatted output string.
`format`	A string describing the variables that will used to construct the output line.
`input_ variables`	A list of variables that will provide the data for the output line.
Description	This function writes to the `destination` string converting the input data according to the `format` string. The `format` string contains a combination of conversion specifiers and characters that will be output verbatim. The `%` character is used to indicate a conversion specifier. Currently only two types are supported: `%s` and `%n`. When a `%s` is found, the next string input variable will be output in its place. Similarly, if a `%n` is found, it will be replaced with the value of the next numeric input variable. Each conversion specifier can be further enhanced with a precision specifier such as `%6.3n`. This indicates that the output data should be 6 characters wide with 3 places after the decimal point. For string conversions, the precision specifier allows you to specify a output field width such as `%20s`. The output is padded with spaces to achieve the proper width if the input variable is not that wide. Note: For those familiar with the C programming language, the format string is similar to the `printf` format string, with `%n` being mapped to `%f`.
Example	In the following example the variable `$line` would contain the value `U10 10.500 5.500`.

```
STRWRITE $line, "%10s %6.3n %6.3n", \
       "U10", 10.5, 5.5
```

See also	STRREAD

Tool functions

DRAWNPADS

Purpose — To setup and execute the Convert Pads command.

Menu command — *Tools/Convert/Pads*

Syntax

```
DRAWNPADS
    NEWDCODE        exp
    TOLERANCE       exp
    BY              exp
    BOUNDARY        yesno
    DRAWS           exp
    ARCS            exp
    DCODE           exp
    LAYER           exp
    GO              [lx,ly,ux,uy]...
END
```

Parameters

NEWDCODE	Specifies the D-Code to replace found drawn pads.
TOLERANCE	An expression specifying the allowable tolerance to use when finding matching drawn pads.
BY	An expression indicating how to select items to convert. Valid values are: 1=window, 2=group.
DCODE	An expression specifying the D-Code filter. Use zero to operate on all D-Codes.
LAYER	An expression specifying the layer filter. Use zero to operate on all visible layers.
GO	An optional list of expressions describing windows that enclose drawn pads that are to be converted.

Description — This function converts drawn pads into flashes.

Example — This example converts all drawn pads that match the example drawn pad contained in the current select group.

```
DRAWNPAD
    NEWDCODE        $newDcode
    TOLERANCE 0.004
    BY              $$GROUPMODE
    GO
END
```

DRC

Purpose

To setup and execute the DRC command.

Menu command

Tools/DRC

Syntax

```
DRC
    REPFILE         string
    PAD2PAD         exp
    PAD2TRACE       exp
    TRACE2TRACE     exp
    MINFLASH        exp
    MINTRACE        exp
    MINRING         exp
    LAYER           exp
    DRILLLAYER      exp
    USETOOLSIZE     yesno
    WELLBEHAVED     yesno
    WINDOWMODE      yesno
    GO              [lx,ly,ux,uy]
END
```

Parameters

`REPFILE` A string containing the filename to which the report file will be written.

`PAD2PAD` An expression indicating the minimum pad-to-pad spacing.

`PAD2TRACE` An expression indicating the minimum pad-to-trace spacing.

`TRACE2TRACE` An expression indicating the minimum trace-to-trace spacing.

`MINFLASH` An expression indicating the minimum flash size allowed.

`MINTRACE` An expression indicating the minimum trace size allowed.

`MINRING` An expression that indicates the minimum annular ring allowed between drilled hole and pad size.

`LAYER` An expression that indicates the layer on which to perform the DRC. A value of zero indicates that all signal layers are to be processed.

`DRILLLAYER` An expression that indicates the drill layer to be used when performing the annular ring check.

GO	Executes the command using the current parameters, using an optional quad of expressions indicating a window to operate on.
Description	This function allows the setup and execution of the DRC command.
Example	The following example executes the DRC command.

```
DRC
    REPFILE   "drc.rep"
    PAD2PAD          0.006
    PAD2TRACE        0.006
    TRACE2TRACE      0.005
    MINFLASH         0.020
    MINTRACE         0.004
    MINRING          0.010
    LAYER            0
    DRILLLAYER       $drillLayer
    USETOOLSIZE      $$NO
    WELLBEHAVED      $$YES
    WINDOWMODE       $$NO
    GO
END
```

DRILL

Purpose — To setup and execute the Drill command.

Menu command — *Tools/Drill*

Syntax

```
DRILL
    OUTFILE         string
    REPFILE         string
    SWATH           exp
    SORT            exp
    LAYER           exp
    WINDOWMODE      yesno
    GO              [lx,ly,ux,uy]
END
```

Parameters

`OUTFILE`	A string containing the filename to which the drill file will be written.
`REPFILE`	A string containing the filename to which the report file will be written.
`SWATH`	An expression indicating the swath size.
`SORT`	A string indicating the sort method: None, X, or Y.
`LAYER`	Indicates the layer to be used in generating the drill data.
`MERGELAYER`	Indicates the layer to be merged into the normal drill data.
`GO`	Uses the current parameters, using optional quad of expressions, to indicate a window to operate on.

Description — Sets up and executes the Drill command. This command sets the `$$DRILLHITS` and `$$DRILLTRAVEL` system variables, as well as returning the hit count in `$$STATUS`.

Example — The following example executes the Drill command.

```
DRILL
    OUTFILE "final.drl"
    REPFILE "final.rep"
    SWATH 0.100
    SORT "Y"
    LAYER $drillLayer
    WINDOWMODE $$NO
    GO
END
```

FIXSS

Purpose — To setup and execute the Fix SS command.

Menu command — *Tools/Fix SS*

Syntax

```
FIXSS
    PADLAYER        exp
    SILKLAYER       exp
    SPACING         exp
    WINDOWMODE      yesno
    GO              [lx,ly,ux,uy],...
END
```

Parameters

PADLAYER	An expression indicating the pad master layer.
SILKLAYER	An expression indicating the silkscreen layer.
SPACING	An expression indicating the minimum spacing allowed between a pad and any silkscreen data.
GO	An optional quad of expressions describing a window to operate on.

Description — This function allows the setup and execution of the Fix SS command.

Example — The following example executes the Fix SS command.

```
FIXSS
    PADLAYER        $padMaster
    SILKLAYER       $silkLayer
    SPACING         0.020
    WINDOWMODE      $$NO
    GO
END
```

NETLIST

Purpose

To generate a internal netlist and/or output such a netlist to a disk file.

Menu command

Tools/Netlist

Syntax

```
NETLIST
    OUTFILE          string
    WELLBEHAVED      yesno
    METRIC           yesno
    M.N              exp, exp
    GO               exp
END
```

Parameters

`OUTFILE` A string indicating the output filename. This parameter is only required when writing a netlist to disk.

`M.N` A pair of expressions specifying the `m.n` of the output file.

`GO` An expression indicating whether to generate a netlist (0) or write an existing netlist to the output file (1) specified with the `OUTFILE` parameter.

Description

This function generates an internal netlist and optionally writes the netlist data to a disk file.

Example

The following example executes the Netlist command.

```
NETLIST
    WELLBEHAVED      $$YES
    GO 0             #generate a netlist
    OUTFILE          "final.net"
    M.N              2, 4
    METRIC           $$NO
    GO 1             #write netlist to file
END
```

PADREMOVAL

Purpose — To setup and execute the Pad Removal command.

Menu command — *Tools/Pad Removal*

Syntax

```
PADREMOVAL
    TYPE            string
    LAYER           exp
    DCODE           exp
    WINDOWMODE      yesno
    GO              [lx,ly,ux,uy],...
END
```

Parameters

`TYPE` — A string indicating the type of pad removal to be performed: I (isolated) or S (stacked).

`LAYER` — An expression that indicates the layer to perform the pad removal on. A value of zero indicates that all visible layers are to be processed.

`DCODE` — An expression indicating a D-Code filter to be used when considering a pad for removal.

`GO` — An optional quad of expressions describing a window to operate on.

Description — This function allows the setup and execution of the Pad Removal command.

Example — The following example executes the Pad Removal command.

```
PADREMOVAL
    TYPE            "I"   # isolated only
    LAYER           $$ACTIVELAYER
    DCODE           0
    WINDOWMODE      $$NO
    GO
END
```

PANELIZE

Purpose — To setup and execute the Panelize command.

Menu command — *Tools/Panelize*

Syntax

```
PANELIZE
    ROWS            exp
    COLS            exp
    TABSIZE         exp, exp
    VENTBORDER      exp
    VENTSPACING     exp, exp
    VENTDCODE       exp
    VENTLAYER       exp
    DOCUMENT        string
    AUTOPANEL       yesno
    AUTOVENT        yesno
    VIRTUAL         yesno
    GO              [lx,ly, ux,uy]
END
```

Parameters

ROWS	An expression indicating the number of rows you want.
COLS	An expression indicating the number of columns you want.
TABSIZE	A pair of expressions indicating the X and Y spacing between images on the panel.
VENTSPACING	A pair of expressions specifying the X and Y spacing between flashes in the vent pattern.
VENTBORDER	An expression indicating the spacing to maintain between the vent pattern and each image.
VENTDCODE	An expression that indicates the D-Code to be used in the vent pattern.
VENTLAYER	An expression that indicates the layer to output the vent pattern into.
DOCUMENT	A string containing the filename to which the report file will output.
GO	Executes the command using the current parameters using optional quad of expressions indicating a window to operate on.

Description

This function allows the setup and execution of the GerbTool Panelize command. This command also sets the following system variables: `$PANELXOFF`, `$PANELYOFF`, `$PANELXSPACING`, `$PANELYSPACING`, `$PANELROWS`, `$PANELCOLS`.

Example

The following example execures the Panelize command using auto panel and auto vent modes.

```
PANELIZE
    VENTBORDER      0.5
    VENTSPACING     0.25, 0.25
    VENTDCODE       250
    VENTLAYER       $$ACTIVELAYER
    DOCUMENT        "panel.rep"
    AUTOPANEL       $$YES
    AUTOVENT        $$YES
    VIRTUAL         $$YES
    GO
END
```

SEGMENTARCS

Purpose — To setup and execute the Convert Circles command.

Menu command — *Tools/Convert/Circles*

Syntax

```
SEGMENTARCS
    CHORDANGLE      exp
    BY              exp
    BOUNDARY        yesno
    DCODE           exp
    LAYER           exp
    GO              [x1,y1,x2,y2...]
END
```

Parameters

`CHORDANGLE` — An expression indicating the chord angle, in degrees, to be used when segmenting each arc.

`BY` — An expression indicating how to select items to convert. Valid values are: 0=item, 1=window, 2=group.

`DCODE` — An expression specifying the D-Code filter. Use zero to operate on all D-Codes.

`LAYER` — An expression specifying the layer filter. Use zero to operate on all visible layers.

`GO` — An optional list of expressions describing either locations of arcs or windows that enclose arcs depending on the setting of the `BY` parameter.

Description — This function allows you to convert interpolated arcs in the database into segmented arcs. The converted arcs consist of a series of short line segments. The smoothness of the converted arcs depend on the setting of the CHORDANGLE parameter. The larger the chord angle the coarser the arc wil be.

Example — The following example converts all interpolated arcs that are in the current select group.

```
SEGMENTARCS
    CHORDANGLE      5
    BY              $$GROUPMODE
    GO
END
```

SNOMAN

Purpose — To setup and execute the Snoman command.

Menu command — *Tools/Snoman*

Syntax

```
SNOMAN
    REPFILE         string
    PAD2PAD         exp
    PAD2TRACE       exp
    OFFSET          exp
    MINPERCENT      exp
    MAXPERCENT      exp
    FROMLAYER       exp
    TOLAYER         exp
    DCODE           yesno
    GO              [lx,ly,ux,uy]
END
```

Parameters

REPFILE	A string containing the filename to which the report file will be written to.
PAD2PAD	An expression indicating the minimum pad-to-pad spacing.
PAD2TRACE	Indicates the minimum pad-to-trace spacing.
OFFSET	An expression indicating the host pad offset.
MINPERCENT	An expression indicating the minimum percent of host pad size that a Snoman pad is allowed to be.
MAXPERCENT	An expression indicating the maximum percent of host pad size that a Snoman pad is allowed to be.
FROMLAYER	An expression that indicates the layer to perform the Snoman on. A value of zero indicates that all signal layers are to be processed.
TOLAYER	An expression that indicates the output layer for the generated Snoman pads. A value of zero indicates the output layer is the same as the input layer.
DCODE	An expression that indicates the D-Code filter used when determining which host pads to consider for generating Snoman pads. A value of zero indicates all D-Codes should be considered.

GO

Executes the command using the current parameters using optional quad of expressions indicating a window to operate on.

Description

This function allows the setup and execution of the Snoman command.

Example

The following example executes the Snoman command.

```
SNOMAN
    REPFILE   "drc.rep"
    PAD2PAD          0.006
    PAD2TRACE        0.006
    OFFSET           -0.005 # close hug
    MINPERCENT       40
    MAXPERCENT       80
    FROMLAYER        0
    TOLAYER          0
    DCODE            0
    WIDOWMODE        $$NO
    GO
END
```

SPREAD

Purpose — To setup and execute the Lyr Spread command.

Menu command — *Tools/Lyr Spread*

Syntax

```
SPREAD
    ROWS            exp
    COLS            exp
    TABSIZE         exp, exp
    LAYER           exp
    AUTOSPREAD      yesno
    SORTTYPE        exp
    GO
END
```

Parameters

ROWS	An expression indicating the number of rows you want.
COLS	An expression indicating the number of columns you want.
TABSIZE	A pair of expressions indicating the X and Y spacing between images in the spread.
LAYER	An expression that indicates the layer to output the spread pattern into.
SORTYPE	An expression where 0=ROW_MAJOR and 1=COL_MAJOR sorting.
GO	Executes the command using the current parameters.

Description — This function allows the setup and execution of the Lyr Spread command.

Example — The following example executes the Lyr Spread command using autospread modes.

```
SPREAD
    SORTTYPE 0      #row major
    LAYER           $$ACTIVELAYER
    AUTOSPREAD      $$YES
    GO
END
```

VENT

Purpose — To setup and execute the Vent command.

Menu command — *Tools/Vent*

Syntax

```
VENT
    SPACING  exp
    DCODE    exp
    GO       [lx,ly,ux,uy]...
END
```

Parameters

`SPACING` — An expression indicating the spacing between the flashes in the generated vent pattern.

`DCODE` — An expression specifying the D-Code to be used in the generated vent pattern.

`GO` — An optional quad of expressions describing a window to operate on.

Description — This function allows the setup and execution of the Vent command.

Example — The following example executes the Vent command.

```
VENT
    SPACING  0.100
    DCODE    $$CURRENTDCODE
    GO       # go interactive
END
```

User data entry functions

GETPOINT

Purpose	Queries the user for a point.
Menu command	None
Syntax	`GETPOINT prompt, x, y`
Parameters	
`prompt`	A string variable or string literal representing the prompt that will be shown the user when this command is run.
`x`	A numeric variable that returns the x coordinate of the point entered by the user.
`y`	A numeric variable that returns the y coordinate of the point entered by the user.
Description	This command allows you to display a message to the user and allow them the chance to enter a location. The user can specify the coordinates in the same fashion as they would if one of the built-in GerbTool commands were querying them. The resulting coordinate is returned in the two provided variables.
Example	The following example queries the user for a location and then adds a flash at that point.

```
GETPOINT "where do you want a flash?", \
                $x,$y
ADDFLASH        $x, $y
```

GETSTRING

Purpose	To prompt a user to enter a text string.
Menu command	None
Syntax	`GETSTRING prompt, destination`
Parameters	
`prompt`	A string containing the prompt text.
`destination`	A string variable that will receive the user's response.
Description	This function allows a macro to display a simple dialog box that contains a prompt message and a text field for the user to enter a response.
Example	The following example prompts you to enter your name.
	`GETSTRING "Enter your name:", $name`

GETWINDOW

Purpose	Queries the user for a window.
Menu command	None
Syntax	`GETWINDOW prompt, x1, y1, x2, y2`
Parameters	
`prompt`	A string variable or string literal representing the prompt that will be shown the user when this command is run.
`x1`	A numeric variable that returns the x coordinate of one corner of the user-specified window.
`y1`	A numeric variable that returns the y coordinate of one corner of the user-specified window.
`x2`	A numeric variable that returns the x coordinate of the second corner of the user-specified window.
`y2`	A numeric variable that returns the y coordinate of the second corner of the user-specified window.
Description	This command allows you to display a message to the user and allow them the chance to enter a window. The user can specify the coordinates in the same fashion as they would if one of the built-in GerbTool commands were querying them. The resulting coordinates are returned in the four provided variables.
Example	The following example queries the user for a window, and views that window.

```
GETWINDOW    "Enter window",$x1,$y1,$x2,$y2
VIEWWINDOW   $x1,$y1, $x2, $y2
```

GETVALUE

Purpose	Queries the user for a numerical value.
Menu command	None
Syntax	`GETVALUE prompt, value`
Parameters	
`prompt`	A string variable or string literal representing the prompt that will be shown the user when this command is run.
`x`	The numeric variable that returns the user-specified value.
Description	This command allows you to display a message box to the user, and allow them to enter a numerical value The resulting value is returned in the provided variable.
Example	The following example queries the user for a layer number.

```
GETVALUE "which layer", $layer
```

GETYESNO

Purpose	Queries the user for a Yes/No value.
Menu command	None
Syntax	`GETYESNO prompt, yesno`
Parameters	
`prompt`	A string variable or string literal representing the prompt that will be shown the user when this command is run.
`yesno`	A numeric variable that returns the user's choice. The possible values are $$YES and $$NO.
Description	This command allows you to display a message box to the user, and allow them to make a Yes/No choice. The resulting value is returned in the provided variable.
Example	The following example queries the user to see if it should terminate the macro.

```
GETYESNO "Quit? ", $value
IF $value == $$YES
    STOP
END
```

MESSAGEBOX

Purpose	To display a dialog box with a title, message, and a choice of button labels.
Menu command	None
Syntax	`MESSAGEBOX title, message, button_style`
Parameters	
`title`	A string containing the dialog box title.
`message`	A string containing the dialog box message.
`button_ style`	An expression indicating your choice of button labels.
Description	This function allows you to display a dialog box with both a title and a message. The body of the message can contain multiple lines of text by separating the lines with the two characters \n. The `button_style` parameter controls which buttons are displayed; 0=Okay, 1=Okay/Cancel, 2=Yes/No. The `$$STATUS` variable is set to 1 (`$$YES`) for Okay/Yes and 0 (`$$NO`) for Cancel/No.
Example	The following example prompts the user for confirmation to continue.

```
MESSAGEBOX  "MyMacro", \
       "Found some errors\nContinue?", 2
IF $$STATUS == $$NO
    STOP
END
```

SETPROMPT

Purpose	Allows the user to control the prompts shown the user while running a macro.
Menu command	None
Syntax	`SETPROMPT cmd_name, [cmd_prompt]`
Parameters	
`cmd_name`	Any type of variable, literal or expression that will be displayed as the command name in the prompt area.
`cmd_prompt`	Any type of variable, literal or expression that will be displayed is the command prompt.
Description	This command updates the prompt area.
Example	The following example sets the prompt bar to reflect the status of a variable used in a macro.

```
REPEAT $counter < $maxnets
    SETPROMPT "Processing net", $counter
    CALC $counter = $counter + 1
    .
    .
    .
END
```

SHOWPROMPT

Purpose	To enable/disable the display of normal GerbTool prompts and messages in the prompt bar.
Menu command	None
Syntax	`SHOWPROMPT yesno`
Parameters	
`yesno`	An expression evaluating to zero for disable, otherwise enable.
Description	This function allows the system prompts to be disabled. This allows some commands to run much faster, as there is less screen writing.
Example	The following example disables prompts, performs some time critical processing and finally turns prompts back on.

```
SHOWPROMPT $$NO
...time critical processing...
SHOWPROMPT $$YES
```

PAUSE

Purpose	Causes the macro to pause.
Menu command	None
Syntax	`PAUSE [time] [cmd_name] [cmd_prompt]`
Parameters	
`time`	An optional expression specifying the amount of time to wait in tenths of a second.
`cmd_name`	An optional string specifying the command name portion of the prompt.
`cmd_prompt`	An optional string specifying the command prompt.
Description	This command will cause the macro to pause, and a message will be presented to the user to press any key to continue. Upon pressing any keyboard key, the macro will continue. If the optional time value is included, the system will wait for that period of time and then resume, even if no key was pressed. A good use for this macro is when preparing demonstration macros showing how to perform some function.
Example	The following example pauses for 10 seconds or until the user presses a key.

```
PAUSE 100, "My Cmd", "hit a key to cont"
```

Utilities and other functions

ABORTCHECKRATE

Purpose	To allow a macro developer to control how often a macro checks for user aborts.
Menu command	None
Syntax	`ABORTCHECKRATE exp`
Parameters	
`exp`	An expression indicating the check rate.
Description	This function allows you to control how often a macro is interrupted to check for a user abort. The lower the checkrate number the more responsive a macro is to a user's request to abort and the slower the macro will run. Raising the checkrate has the opposite effect. The default checkrate is 200.
Example	The following example sets the default abort check rate to a high value inside a database scan loop.

```
# high speed, slow response
ABORTCHECKRATE 5000
REPEAT $$STATUS
...database processing...
    GETNEXTITEM
END

# slow speed, fast response
ABORTCHECKRATE 10
```

CREATELAYER

Purpose	To create and/or load a new layer.
Menu command	None
Syntax	`CREATELAYER layer, filename, ap_filename`
Parameters	
`layer`	An expression evaluating to a layer number.
`filename`	A string containing the filename of a Gerber file.
`ap_filename`	A string containing the filename an aperture list.
Description	This function creates a new layer in the currently loaded design. If the Gerber filename specified is found on disk, it will be loaded. Otherwise, an empty layer will be created. If the `layer` parameter evaluates to zero, this function will search the layer table for the first unused layer. The actual layer created will be returned in the `$$STATUS` variable. A return value less than or equal to zero indicates failure. This function operates similarly to normal design loading in that you do not need to specify an `ap_filename` if you intend to use a previously loaded aperture list.
Example	The following example creates a new unnamed layer in the first available empty layer and returns the new layer number in `$$STATUS`.

```
CREATELAYER 0, "", ""
Calc $tempLayer = $$STATUS
```

SPLITPATH

Purpose	To split a complete file path specification into its separate directory, filename, and extension components.
Menu command	None
Syntax	`SPLITPATH fullpath, dir, filename, ext`
Parameters	
`fullpath`	A string containing a complete file path specification.
`dir`	A string variable that will receive the directory portion of the `fullpath` string.
`filename`	A string variable that will receive the filename portion of the `fullpath` string.
`ext`	A string variable that will receive the extension portion of the `fullpath` string.
Description	This function is used to separate the individual components of a complete file path specification.
Example	In the following example, if `$$DSNNAME` contained the value C:\PROJECTS\JOBS\GROMMIT.GTD, then `$filename` would end up containing the value GROMMIT.ZIP.

```
SPLITPATH $$DSNNAME, $dir, $file, $ext
STRWRITE $filename, "%s.zip", $file
```

SYSCMD

Purpose	To execute another program.
Menu command	None
Syntax	`SYSCMD command_line`
Parameters	
`command_line`	A string that will be passed to the host operating system for execution.
Description	This function allows you to execute external programs while GerbTool waits. More than a simple convenience feature, this function allows you to have an external program perform a specialized task on a file created within a GerbTool macro. This modifed file would then be read back into GerbTool, thereby extending GerbTool beyond its own limits. Note: This function CANNOT be used to execute a Windows executable even when issued from GerbTool.

Example	The following example shows how you can obtain a sorted copy data extracted from GerbTool.
	
	`STRWRITE $cmd, "sort %s %s", $infile, $outfile`
	`SYSCMD $cmd`

Viewing functions

REDRAW

Purpose	To cause the screen to redraw.
Menu command	*View/Redraw*
Syntax	`REDRAW`
Parameters	None
Description	This command causes the screen to redraw. The scale or viewing positions are not changed.

Example	The following example changes the prompt to inform the user that some calculations have been finished, and then performs a redraw so the user can see the results.
	`SETPROMPT "Finished", $count`
	`REDRAW`

VIEWALL

Purpose	To view the extents of all visible layers.
Menu command	*View/All*
Syntax	`VIEWALL`
Parameters	None
Description	This command changes the view scale so that all of the Gerber layers whose visibility is turned on are displayed on the screen.
Example	The following example changes the prompt to inform the user that some calculations have been finished, and then performs a `VIEWALL` so the user can see the results.

```
SETPROMPT "Finished", $count
VIEWALL
```

VIEWFILMBOX

Purpose	To view the film box and its contents.
Menu command	*View/Filmbox*
Syntax	`VIEWFILMBOX`
Parameters	None
Description	This command changes the view scale so that the entire film box and its contents are displayed on the screen.
Example	The following example changes the prompt to inform the user that some calculations have been finished, and then performs a `VIEWFILMBOX` so the user can see the results.

```
SETPROMPT "Finished", $count
VIEWFILMBOX
```

VIEWPAN

Purpose — To set the view window in GerbTool to a particular location.

Menu command — *View/Pan*

Syntax — `VIEWPAN [x, y]...`

Parameters

`x` — An expression describing the x coordinate of the point to pan to.

`y` — An expression describing the y coordinate of the point to pan to.

Description — This function accepts a set of coordinates and changes the current view such that this coordinate is placed in the center of the screen. This function assumes that the point you want to pan to is currently on the screen. If this is not the case, you can use the `VIEWWINDOW` command to change the viewing location of the system.

Example — This example takes a coordinate specified by $x, $y and pans to that location, changing the view window so that the entire screen is taken up by the 2 square inches surrounding the point.

```
CALC   $winLx = $x - 1.0
CALC   $winLy = $y - 1.0
CALC   $winUx = $x + 1.0
CALC   $winUy = $y + 1.0

VIEWWINDOW $winLx, $winLy, $winUx, $winUy
VIEWPAN $x, $y
```

VIEWPREVIOUS

Purpose	To cause the current view to be set to the state before it was last changed.
Menu command	*View/Previous*
Syntax	`VIEWPREVIOUS`
Parameters	None
Description	This command is used to recall the last viewing window. This last viewing window is automatically saved after performing a view command such as View/Window.
Example	The following example views a window whose size has been calculated previously, pauses 10 seconds to allow the user to view the area, and then restores the view to its previous state.

```
VIEWWINDOW $lx,$lx, $ux, $uy
PAUSE 100
VIEWPREVIOUS
```

VIEWRECALL

Purpose	To cause the current view to be set to that stored in the specified save locations.
Menu command	*View/Recall*
Syntax	`VIEWSAVE location`
Parameters	
`location`	An expression giving the save location that contains the view. Valid locations are from 1 to 8.
Description	This command causes the current view to be changed to the value stored in one of the available save locations shown under the *View/Save* command, or set with the `VIEWSAVE` macro function.
Example	The following example saves the current view so that it can restore it after performing some calculations.

```
VIEWSAVE 1
...perform calculations here...
VIEWRECALL 1
```

VIEWSAVE

Purpose	To cause the current view to be saved.
Menu command	*View/Save*
Syntax	`VIEWSAVE location`
Parameters	
`location`	An expression giving the save location to place this view. Valid locations are from 1 to 8.
Description	This command causes the current view to be saved in one of the available save locations shown under the *View/Save* command.
Example	The following example saves the current view so that it can restore it after performing some calculations.

```
VIEWSAVE 1
...perform calculations here...
VIEWRECALL 1
```

VIEWWINDOW

Purpose	To set the view window in GerbTool to a particular location.
Menu command	*View/Window*
Syntax	`VIEWWINDOW lx, ly, ux, uy`
Parameters	
`lx`	An expression describing the lower x coordinate of the view window.
`ly`	An expression describing the lower y coordinate of the view window.
`ux`	An expression describing the upper x coordinate of the view window.
`uy`	An expression describing the upper y coordinate of the view window.
Description	This function accepts four values representing the lower-left and upper-right coordinates of a rectangle. GerbTool then performs the equivalent of a *View/Window* on this area. Since there is no guarantee that the coordinates provided will coincide exactly with the view window, some of the design outside of the specified rectangle may be visible.
Example	The following example gets the size of the film box and then views that area.

```
GETFILMBOX   $ux, $uy
VIEWWINDOW    0, 0, $ux, $uy
```

ZOOMIN

Purpose	To set the view window in GerbTool to a particular location.
Menu command	*View/Zoom In*
Syntax	`ZOOMIN [x, y]...`
Parameters	
`x`	An expression describing the x coordinate of the point to zoom in around.
`y`	An expression describing the y coordinate of the point to zoom in around.
Description	This function accepts a set of coordinates and changes the current view such that this coordinate is placed in the center of the screen and the magnification factor of the view is doubled.
Example	The following example gets the size of the film box and then zooms into its center.

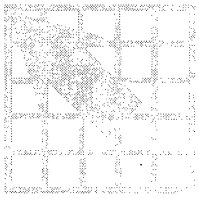
```
GETFILMBOX   $ux, $uy
CALC         $ux = $ux / 2
CALC         $uy = $uy / 2
ZOOMIN       $ux, $uy
```

ZOOMOUT

Purpose	To set the view window in GerbTool to a particular location.
Menu command	*View/Zoom Out*
Syntax	`ZOOMOUT [x, y]...`
Parameters	
`x`	An expression describing the x coordinate of the point to zoom out around.
`y`	An expression describing the y coordinate of the point to zoom out around.
Description	This function accepts a set of coordinates and changes the current view such that this coordinate is placed in the center of the screen and the magnification factor of the view is decreased by half.
Example	The following example gets the size of the film box and then zooms out from its center.

```
GETFILMBOX    $ux, $uy
CALC          $ux = $ux / 2
CALC          $uy = $uy / 2
ZOOMOUT       $ux, $uy
```

Chapter 9



Aperture Conversion Rule files

In addition to providing the ability to convert most popular CAD and photoplotter aperture lists directly into the popular GerbTool format, GerbTool also allows you to create your own Aperture Conversion Rule (ACR) files for specialty, proprietary or otherwise unsupported aperture list formats.

Definition of an ACR file

An Aperture Conversion Rule (ACR) file is an ASCII file used to describe a particular aperture list format using conversion language statements. Using a text editor, you can create your own ACR file that describes the expected format of your aperture list. Once it is read in, GerbTool is able to convert your new aperture list format automatically, just as it converts the supported aperture list formats (see *Converting a CAD aperture list* in *Chapter 3: Quick start*).

Creating an ACR file

An ACR file contains two types of statements. The first type describes the environment, such as the expected file extension, metric mode, number of header lines to skip, and so on. The second type is the actual rule statement, which is used to match incoming aperture list entries to corresponding GerbTool aperture shapes.

The following are descriptions of the environment-type of ACR statements and their expected parameters, if any.

NAME

Syntax	`NAME converter_name`
Parameters	
`converter_name`	The name of the ACR file. Should be a single word.
Description	This statement will place the parameter in the header of the resulting aperture list.
Example	The following example sets the name of the converter to ALLEGRO.ACR.
	`NAME allegro.acr`

VERSION

Syntax	`VERSION version_number`
Parameters	
`version_number`	The version number of the ACR file. The version number should be a single decimal number.
Description	This statement will place the parameter in the header of the resulting aperture list.
Example	The following example sets the version number of the converter to 6.
	`VERSION 6`

HEADER

Syntax

```
HEADER lines_to_skip
```

Parameters

`lines_to_skip` — The number of lines to skip in the header of the aperture list.

Description

If this line is present, the number of lines specified will be skipped from the header of the aperture list file you are attempting to convert. This can be used to bypass information at the top of a file that you know does not contain any apertures.

Example

The following example instructs GerbTool to skip the first twenty lines of the aperture list.

```
HEADER 20
```

SKIP

Syntax

```
SKIP skip_string
```

Parameters

`skip_string` — A text string to mark text to be skipped.

Description

If this line is present, all lines in the aperture list that start with the given character string will be ignored.

Example

The following example will allow GerbTool to skip over lines that begin with `MOIRE`.

```
SKIP MOIRE
```

DEFAULT_UNITS

Syntax	`DEFAULT_UNITS mode`
Parameters	
`mode`	One of $$INCH, $$MIL, or $$MM.
Description	If given, will cause the values read in to be interpreted as Inches, Mils, or Millimeters, depending on the value used.
Example	The following example sets the units mode to metric. `DEFAULT_UNITS $$MM`

CUSTOM

Syntax	`CUSTOM yesno`
Parameters	
`yesno`	Either $$YES or $$NO.
Description	If set to $$YES, GerbTool will attempt to create custom aperture names whenever possible. Otherwise a Diamond shape will be substituted. Note: GerbTool will not create the custom apertures themselves, only their names in the aperture list.
Example	The following example sets the creation of custom apertures to off. `CUSTOM $$NO`

EXTENSION

Syntax

```
EXTENSION extension
```

Parameters

`extension` The default aperture list extension.

Description

The default extension of the aperture lists you will be converting with this rule file. If the value is entered here, you will not need to enter it when specifying the aperture list for conversion.

Example

The following example sets default aperture list extension of MYA.

```
EXTENSION mya
```

DEBUG

Syntax

```
DEBUG mode
```

Parameters

`mode` A value of 0, 1, or 2.

Description

Enables debugging information to be output into the aperture converter's log file. If zero is used, no debug information will be output. If 1 is used, GerbTool will output debug information while parsing the ACR file, and if the value is set to 2, debug information will be output while converting the aperture file itself. This function is for advanced users and should either not be included or be set to zero for normal converter operation.

Example

The following example sets the current debug mode to 2.

```
DEBUG 2
```

XTENSION

Syntax	`XTENSION dll_filename`
Parameters	
`dll_filename`	The name of a .DLL file that you supply.
Description	Causes the converter to look for the specified .DLL file to help in converting the aperture lists.
Example	The example specifies a user-supplied .DLL. `XTENSION myapfmt.dll`

DCODE

Syntax	`DCODE mode`
Parameters	
`mode`	One of $$ONLINE, $$SEQUENTIAL, or $$GERBER_ORDER.
Description	Controls how D-Code values will be derived. If set to $$ONLINE (the default) the codes read on each line will be used. If $$SEQUENTIAL is used, lines that match the rules given will be assigned sequential numbers. Some aperture lists have their D-Codes arranged in a special non-sequential order used in certain Gerber photoplotters. Walcer will use this order if $$GERBER_ORDER is set.
Example	The example sets the D-Code mode to sequential. `DCODE $$SEQUENTIAL`

#

Syntax	`# any_text`
Parameters	
`any_text`	The body of a comment.
Description	This symbol leads comments in an ACR file.
Example	The example shows a typical comment. `# Created By A. Designer`

The following is a description of each rule type of ACR statement and the expected parameters, if any:

FORMAT_shape

Syntax	`FORMAT_shape rule`
Parameters	
`shape`	The possible shapes are: ROUND, SQUARE, RECT, OBLONG, DONUT, DIAMOND, OCTAGON, THERMAL, THERM45, TARGET, and CUSTOM. Note that this parameter should be combined with the `FORMAT_` statement to form a single word such as `FORMAT_ROUND`.
`rule`	A rule for matching apertures that are to be mapped to a GerbTool shape aperture.
Description	If the rule matches a line in the aperture list being converted, that line will be converted into a GerbTool `shape` aperture.
Example	The following example will match the line: JUNK D10 0.060 0.060 ROUND.

```
FORMAT_ROUND $skip +D$dcode
$xsize $ysize ROUND
```

FORMAT_UNITS

Syntax	`FORMAT_UNITS rule`
Parameters	
`rule`	A rule for matching a line in the aperture list that specifies the format of the file.
Description	A line matching this is used to determine the format of the aperture list. This statement allows the aperture list itself to override a previous `UNITS` statement.
Example	The following example will match the line: FORMAT MM.

```
FORMAT_UNITS $skip $units
```

FORMAT_SPECIAL

Syntax	`FORMAT_SPECIAL rule`
Parameters	
`rule`	A rule for matching lines for use by an `XTENSION` DLL.
Description	Does not produce a GerbTool D-Code line. It is used for special processing by an `XTENSION`-specified DLL.
Example	The following example will match the line: SQR D10 0.060 0.060.

```
FORMAT_SPECIAL SQR +D$dcode
$xsize $ysize
```

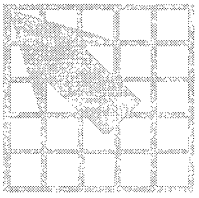
When constructing rules to match apertures, there are special keywords that you place in the rule that will cause GerbTool to assign the values contained in the fields to the corresponding GerbTool aperture list fields. These keywords are as follows:

Keyword	*Meaning*
`$dcode`	Assigned to D-Code
`$xsize`	Assigned to xsize
`$od`	Assigned to xsize
`$ysize`	Assigned to ysize
`$id`	Assigned to ysize
`$rot`	Assigned to rotation
`$tool`	Assigned to tool num
`$skip`	Skip this field
`$custom`	Use this field to make a custom aperture
`$units`	Used to determine the format of the aperture list

The following is a sample ACR file.

```
# Aperture converter for Mentor

NAME Mentor
VERSION 1.0
EXTENSION rpt

# handle swapped X/Y columns
XTENSION mentor.dll

DEBUG 0

CUSTOM $$NO

DEFAULT_UNITS $$INCH

HEADER 1

FORMAT_ROUND $skip +circle +$skip +$xsize +$ysize +$rot +false +false
+$dcode

FORMAT_THERMAL $skip +circle +$skip +$xsize +$ysize +$rot +false +true
+$dcode

FORMAT_RECT $skip +rectangle +$skip +$xsize +$ysize +$rot +false +false
+$dcode

FORMAT_SPECIAL Position +Shape

# Mentor now has multiple formats
FORMAT_ROUND  +$skip +$dcode +circle +$skip +$xsize +$ysize
FORMAT_THERMAL  +$skip +$dcode +circle +$skip +power +$xsize +$ysize
FORMAT_RECT  +$skip +$dcode +rectangle +$skip +$xsize +$ysize
FORMAT_SPECIAL Aperture Position
```

Chapter 10



274-X

GerbTool supports the extended Gerber data format, 274-X, developed by Gerber Systems, Inc. (GSI). This format provides for the inclusion of aperture data directly in the Gerber data files (embedded apertures), flexible aperture definitions and easy single file compositing.

Embedded apertures

Note While it not necessary to understand the syntax of 274-X to manipulate 274-X files within GerbTool, several examples of 274-X syntax are provided below. These examples are provided to acquaint you the 274-X only. See the instruction manuals provided with your photoplotter, or contact GSI directly, for more information on the 274-X syntax.

A 274-X format Gerber file contains all aperture definitions necessary to plot the data thereby eliminating the need for an external aperture list. An aperture is defined within a 274-X file with an AD command as follows:

```
%ADD<code><macro_name>,<parameter_list>*%
```

For example:

```
%ADD10C,0.06X0.020%
```

This example defines D10 as a simple 60-mils round flash using the GSI intrinsic aperture macro "C."

GerbTool allows you to edit aperture definitions using the *Edit AD* button within the *Apertures/Edit* form. See *Chapter 7: Command reference* for more information.

Aperture macros

Aperture macros are used to describe the size and shape of special apertures. Using aperture macro primitives, it is possible to design complex aperture shapes. Each primitive describes a basic shape such as a circle or a line. Each primitive also specifies its polarity (on/off) allowing data to be removed for such features as donuts or spokes in a thermal. Shown below are the different primitives available.

Number	*Type*	*Parameters*
1	Circle	on/off diameter xcenter ycenter
20	Line-Vector	on/off width xbeg ybeg xend xend rot
21	Line-Center	on/off width height xcenter ycenter rot
22	Line-Lower left	on/off width height xloc yloc rot
4	Outline	on/off count x y... rotation
5	Polygon	on/off sides xcenter ycenter diameter rot

274-X aperture macro primitives.

Aperture macros are also programmable by using *replaceable parameters*, which allow a macro to produce different results, depending on the aperture definition specified by the AD aperture definition command (explained in the preceding section). Replaceable parameters are indicated by a dollar sign ($) followed by a numeric value. The numeric value indicates the parameter's position within the AD aperture definition. A typical donut macro and corresponding definitions are shown below.

```
%AMDONUT*
1,1,$1,0.0,0.0*
1,0,$2,0.0,0.0*
%

%ADD10DONUT,0.60X0.40%
%ADD20DONUT,0.08X0.70%
```

In the above example, D10 is defined as a 60-mils donut with a 40-mils hole, and D20 is defined as a 80-mils donut with a 70-mils hole. Note that both D10 and D20 refer to the same macro but have different sizes.

GerbTool allows you to edit aperture macros using the *Edit AM* button within the *Apertures/Edit* form. See *Chapter 7: Command reference* for more information.

Layer compositing

274-X allows a single Gerber file to define a composite image of arbitrary complexity. Each "layer" of data within the Gerber file is prefixed with an appropriate polarity command. Ordering of the layers is critical as the data is processed sequentially. For assistance, check the example files provided and notice how each layer either adds or removes from the initial image.

GerbTool automatically creates separate layers for composite layers when reading a 274-X file and conversely creates a single file for all layers that form a composite when writing out data.

Viewing composites

Composite layers can be displayed by typing the nested command v. This nested command toggles composite viewing on/off. When **enabled**, composite layers will be displayed as they will plot. When **disabled**, composite layers will be displayed as if all layers were dark (positive). Composite viewing can also be controlled using the *Layers/Edit* form.

Converting from 274-D to 274-X

In order to convert a set of standard Gerber 274-D files into a single Gerber 274-X composite file, load the 274-D files as you normally do and then perform the following steps using the *Layers/Edit* command:

- Set the *Layer Name* field of each layer to a meaningful name.

Tip Setting the *Layer Name* field to the original filename of the same layer will label the 274-X "layers" in a fashion that will be familiar to the user.

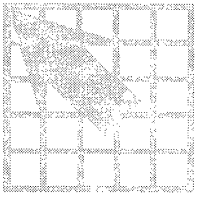
- Decide on the filename you want to use for the new Gerber 274-X file and rename all of the Gerber filenames to this new name. It is important that each 274-X "layer" have the same filename.
- Set the *Layer Type* for each of these 274-X layers to *Composite*.
- Assign a polarity and a common number to the *Key* field for each of the 274-X "layers." For example, D1 for "Dark composite number 1" or C1 for "Clear composite number 1." A polarity of Dark means that the layer is to be displayed in the style a normal Gerber file is displayed. Clear tells GerbTool to display the layer using the current background color. This has the effect of erasing, or "clearing," areas from an image that were previously drawn by a "dark" layer. Negative layers should be set to clear.

Note The common number portion of the *Key* field allows GerbTool to load multiple 274-X composite files at the same time. Each set of layers within a 274-X file should have a common number assign to the *Key* field.

- Click on the *Edit* button within the *File Format* group box. Change the *Dialect* field to 274-X.
- Save the composite file using the *File/Save* command. All the layers will all be written into a single Gerber 274-X file with the name that you specified, along with a Gerber 274-X embedded aperture list.

To load this new 274-X composite file into another design, enter its filename into the *Filename* field as you would with any other Gerber file, making sure the file format has been set to 274-X. There is no need to load in an aperture list as it is included in the 274-X file.

Chapter 11



Using custom apertures

GerbTool allows you to create custom apertures. A custom aperture is nothing more than a Gerber file, and can therefore be of virtually any size or shape. This chapter details the steps for creating a custom aperture.

Create a custom aperture

- Using the *Files/Format* command, set the Gerber format to Imperial, absolute, 2.3 and no zero suppression.
- Select the command *Files/Load.*
- Enter the appropriate design filename.

Tip Use one design file for all of your custom apertures.

- In the *Layers* form, enter a descriptive name in the *Filename* field, such as FIDUCIAL.CUS (the .CUS extension is mandatory).
- Enter the filename of the aperture list that you will be using for this custom aperture in the *Aperture List* field.
- Click on the *OK* button. GerbTool will inform you that the specified Gerber file doesn't exist. Respond affirmatively to create the new layer.
- At this point you can create your custom aperture using any of the apertures defined in the aperture list assigned to the new layer.

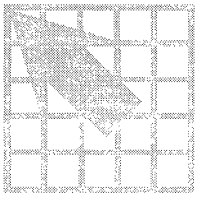
Note Before you save your custom aperture, ensure that the origin is where you want it. You can use the *Edit/Origin* command to relocate the origin.

To use the new custom aperture, enter its filename (less extension) in the *Shape* field of an aperture list using the *Apertures/Edit* command.

Caution The aperture list used while designing your custom aperture must be specified in the *Custom Ap List* field within the *Defaults* form. Use the *Options/Defaults* command to change this field if necessary. A aperture list used for custom apertures should not itself contain any custom apertures.

Tip It is recommended that you set aside one aperture list dedicated to all your custom apertures.

Chapter 12



Working with text fonts

GerbTool uses a font file containing a list of X-Y coordinate pairs that constitute the "strokes" required to display each character inserted by the *Edit/Text* command. You can have more than one font file but GerbTool will always read the STROKE.FNT file at startup. To use a different font file, rename STROKE.FNT to some other name, then rename your font file to STROKE.FNT. GerbTool allows you to edit existing fonts and create new fonts that are used for text insertion. This chapter details the steps for editing fonts.

Editing a font

Before you edit a font you must convert it into individual Gerber files for each character. To do this, from the system prompt change to the GerbTool fonts directory and type the following command, then touch the ENTER key:

```
f2g ../stroke.fnt
```

This will create an individual Gerber file for each character in the font file. You can now start GerbTool and load one of the provided design files UPCASE.GTD, LWCASE.GTD, NUMBERS.GTD, PUNC1.GTD, or PUNC2.GTD, which cover uppercase, lowercase, numbers, and punctuation characters respectively. The Film Box is set to a 7-mils square, which each character must remain within. You can draw any shape you want as long as you stay in or on the film box and you don't try to add flashes.

Note It is important that the file format of the individual Gerber files for each character remain at Imperial, absolute, 2.3, and no zero suppression.

Once you have finished editing the characters, you can use the following command at the system prompt to create a new font file, then touch the ENTER key.

```
g2f newfile.fnt
```

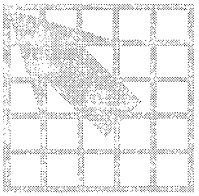
In the above example a new font file would be created with a filename of NEWFILE.FNT. Note that this program does not purge the individual Gerber character files. You may do this manually if you want. Remember that GerbTool will not recognize your new font file unless it is named STROKE.FNT and is in the GerbTool program directory.

Creating a new font

To create a completely new font you can follow the steps detailed in *Editing a font* above, but skip the font file to Gerber file conversion step.

Note It is usually easier (and faster) to modify an existing font than to create one from scratch.

Appendix A



Command ID values

The tables in this appendix contain the command ID values associated with each GerbTool command. You can use these values to program your mouse and function keys.

Command	*ID*
Apertures/Compact	AO
Apertures/Convert	AV
Apertures/Edit	AE
Apertures/Load	AL
Apertures/Merge	AM
Apertures/Report	AR
Apertures/Save	AS
Apertures/Unload	AU
Edit/Add/Arc 3 Pt	EAA3
Edit/Add/Arc Ctr	EAAC
Edit/Add/Circle	EAC
Edit/Add/Draw	EAD
Edit/Add/Flash	EAF
Edit/Add/Polygon	EAP
Edit/Add/Rectangle	EAR
Edit/Add/Text	EAT
Edit/Add/Vertex	EAV
Edit/Align	EA
Edit/Clip	EK

Command ID values (page 1 of 5).

Command	*ID*
Edit/Copy	EC
Edit/Dcode/Expand	EDE
Edit/Dcode/Polarity	EDP
Edit/Dcode/Scale	EDS
Edit/Dcode/Transcode	EDT
Edit/Erase	EE
Edit/Item	ET
Edit/Mirror	EI
Edit/Move	EM
Edit/Origin	EO
Edit/Purge	EP
Edit/Rotate	ER
Edit/Select/Add	ESA
Edit/Select/Invert	EPI
Edit/Select/New	ESN
Edit/Select/Off	ESO
Edit/Select/Remove	ESR
Edit/Undo	EU
Files/Chgdir	FD
Files/Close	FC
Files/Exit	FQ
Files/Export/BARCO DPF	FEB
Files/Export/IPC-D-350	FE350
Files/Export/IPC-D-356	FE356
Files/Format/Drill	FFD
Files/Format/Gerber	FFG
Files/Format/Load	FL
Files/Import/BARCO DPF	FIB

Command ID values (page 2 of 5).

Command	*ID*
Files/Import/Drill	FIN
Files/Import/HPGL	FIH
Files/Import/IPC-D-356	FI356
Files/Merge/Design	FMD
Files/Merge/Gerber	FMG
Files/New/Auto	FNA
Files/New/Manual	FNM
Files/Open	FO
Files/Plot/HPGL	FPH
Files/Plot/PostScript	FPP
Files/Print	FP
Files/Save	FS
Layers/Colors	LC
Layers/Edit	LE
Options/Arcs 360	OA
Options/Bg Color	OB
Options/Defaults	OD
Options/Filmbox	OF
Options/Grid	OG
Options/KeyCmds	OK
Options/Metric	OM
Options/Ortho	OR
Options/Overlay	OO
Options/Save	OV
Options/Show Errs	OE
Options/Sketch	OS
Options/Undo	OU
Query/Copper	QC

Command ID values (page 3 of 5).

Command	*ID*
Query/Extents	QE
Query/Highlight/Dcode	QHD
Query/Highlight/Net	QHN
Query/Highlight/Off	QHO
Query/Item	QI
Query/Measure/Edge to Edge	QME
Query/Measure/Point to Point	QMP
Tools/Convert/Circles	TCA
Tools/Convert/Pads	TCP
Tools/DRC	TD
Tools/Fix SS	TF
Tools/Lyr Spread	TL
Tools/Macro/Load	TML
Tools/Macro/Run	TMR
Tools/NC Drill/Drawing	TNDD
Tools/NC Drill/Write	TNDW
Tools/Netlist/Generate	TNLG
Tools/Netlist/Write	TNLW
Tools/Pad Removal/Isolated	TPI
Tools/Pad Removal/Stacked	TPS
Tools/Panelize	TP
Tools/Snoman	TS
Tools/Vent	TV
View/All	VA
View/Errors	VE
View/Filmbox	VF
View/Pan	VP
View/Previous	VV

Command ID values (page 4 of 5).

Command	*ID*
View/Recall	VC
View/Redraw	VR
View/Save	VS
View/Window	VW
View/ZoomIn	VI
View/ZoomOut	VO

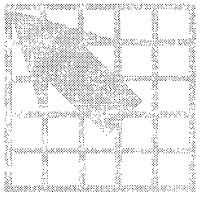
Command ID values (page 5 of 5).

Command	*ID*
View/All	VA
View/Film Box	VF
View/Pan	VP
View/Previous	VV
View/Redraw	VR
View/Window	VW
View/Zoom In	VI
View/Zoom Out	VO

Command ID values assignable to mouse buttons.

Appendix B

Configuration files



This appendix contains a complete listing of all configuration parameters supported by GerbTool. Note that some parameters are specific to a particular operating system platform and are identified as such. If a parameter is not identified as being restricted to a particular platform, then it is applicable to all platforms.

ALL_ARCS_360

Syntax	`ALL_ARCS_360=yes_no`
Description	Normally, GerbTool requires 360° interpolated arcs to be prefixed with a G75 block. Otherwise they are interpreted as quadrant format arcs. This parameter allows you to override this behavior and instruct GerbTool to treat all G02/G03 blocks as 360° interpolated arcs.
Related command	*Files/Format*
Example	`ALL_ARCS_360=YES`

AP_CONV

Syntax	`AP_CONV=filename, description`
Description	This parameter allows you to inform GerbTool of available aperture list converters. As new converters are provided you can "upgrade" GerbTool by adding a line to your GerbTool configuration file.
Related command	None
Example	`AP_CONV=mentr2gt.exe, MENTOR`

AP_CONV_IGNORE

Syntax	`AP_CONV_IGNORE=ext1 ext2 ...`
Description	This parameter allows you to inform GerbTool that files with one of these filename extensions should be ignored by the *File/New/Auto* command. This allows this command to avoid wasting its time on files that are not valid Gerber or Aperture list files, such as executable and batch files.
Related command	*File/New/Auto*
Example	`AP_CONV_IGNORE=BMP DLL DOC WRI INI ACR`

ARCS_MODAL

Syntax	`ARCS_MODAL=yes_no`
Description	Normally, GerbTool considers G02/G03 arc blocks to be modal. This parameter overrides this behavior and instructs GerbTool to require arcs to be prefixed with a G02/G03 command.
Related command	*Files/Format*
Example	`ARCS_MODAL=NO`

ARCS_SEGMENTED

Syntax	`ARCS_SEGMENTED=yes_no`
Description	Normally, GerbTool enters segmented arcs when adding arcs to a layer with the Edit/Add commands. This parameter allows you to instruct GerbTool to enter interpolated arcs instead. The default is YES.
Related command	*Options/Arcs 360°*
Example	`ARCS_SEGMENTED=NO`

BG_COLOR

Syntax	`BG_COLOR=color`
Description	This indicates the color of the drawing area background.
Related command	*Options/Bg Color*
Example	`BG_COLOR=Black`

BORDER_TEXT

Syntax	`BORDER_TEXT=text`
Description	This parameter allows you to specify the text to appear in the border of check plots generated by the *Files/Plot* command. GerbTool looks for the keywords $DATE, $TIME, $DESIGN, and $PROG. If GerbTool finds any of these keywords, they will be replaced with the appropriate text. All other text specified will be included in the border verbatim.
Related command	None
Example	`BORDER_TEXT=XYZ Company $DESIGN $DATE $TIME $PROG`

CHAR_SET

Syntax	`CHAR_SET=ASCII_EBCDIC_EIA`
Description	This parameter specifies the expected character set for Gerber files.
Related command	*Files/Format*
Example	`CHAR_SET=ASCII`

CHORD_ANGLE

Syntax	`CHORD_ANGLE=n`
Description	This parameter allows you to specify the chording angle used when generating segmented arcs within GerbTool.
Related command	*Options/Defaults*
Example	`CHORD_ANGLE=10`

CROSSHAIR

Syntax	`CROSSHAIR=x, y`
Description	This parameter allows you to control the size of the drawing cursor. Use 0,0 for a full screen crosshair.
Related command	*Options/Defaults*
Example	`CROSSHAIR=26, 24`

DEF_CUSTOM_MAP

Syntax	`DEF_CUSTOM_MAP=aperture_list`
Description	This parameter specifies the aperture list that GerbTool will use for any and all custom apertures loaded. The aperture list specified cannot itself include custom apertures.
Related command	*Options/Defaults*
Example	`DEF_CUSTOM_MAP=CUSTOM.MAP`

DEF_DSN_EXT

Syntax	`DEF_DSN_EXT=design_extension`
Description	This specifies the default extension to be used when dealing with GerbTool design files.
Related command	*Options/Defaults*
Example	`DEF_DSN_EXT=GTD`

DEF_DSN_PATH

Syntax	`DEF_DSN_PATH=dsn_path`
Description	Specifies the default directory for finding design files. If an explicit path is not provided when loading a design file, this path will be used.
Related command	*Options/Defaults*
Example	`DEF_DSN_PATH=C:\ORCADWIN\LAYOUT\GERBTOOL`

DEF_GERB_EXT

Syntax	`DEF_GERB_EXT=gerber_extension`
Description	Specifies the default extension to be used when dealing with Gerber files.
Related command	*Options/Defaults*
Example	`DEF_GERB_EXT=gbr`

DEF_HPGL_EXT

Syntax	`DEF_HPGL_EXT=hpgl_extension`
Description	Specifies the default extension to be used when dealing with HPGL files.
Related command	*Options/Defaults*
Example	`DEF_HPGL_EXT=plt`

DEF_LJ_EXT

Syntax	`DEF_LJ_EXT=laserjet_extension`
Description	Specifies the default extension to be used when dealing with LaserJet files.
Related command	*Options/Defaults*
Example	`DEF_LJ_EXT=lj`

DEF_MAP

Syntax	`DEF_MAP=aperture_list`
Description	This specifies the aperture list that GerbTool will load if no other aperture list has been specified.
Related command	*Options/Defaults*
Example	`DEF_MAP=default.map`

DEF_MAP_EXT

Syntax	`DEF_MAP_EXT=map_extension`
Description	Specifies the default extension to be used when dealing with aperture list files (map files).
Related command	*Options/Defaults*
Example	`DEF_MAP_EXT=map`

DEF_NC_EXT

Syntax	`DEF_NC_EXT=nc_extension`
Description	Specifies the default extension to be used when dealing with NC Drill files.
Related command	*Options/Defaults*
Example	`DEF_NC_EXT=nc`

DEF_PATH

Syntax	`DEF_PATH=path_name`
Description	This parameter specifies the default directory for finding Gerber files and aperture lists.
Related command	*Options/Defaults*
Example	`DEF_PATH=c:\proj5\gerbs`

DEF_PS_EXT

Syntax	`DEF_PS_EXT=postscript_extension`
Description	This specifies the default extension to be used when dealing with PostScript files.
Related command	*Options/Defaults*
Example	`DEF_PS_EXT=ps`

DEF_REP_EXT

Syntax	`DEF_REP_EXT=report_extension`
Description	This specifies the default extension to be used when dealing with GerbTool report files.
Related command	*Options/Defaults*
Example	`DEF_REP_EXT=RPT`

END_CAP

Syntax	`END_CAP=pixels`
Description	This parameter specifies when GerbTool should stop attempting to draw end caps on drawn lines. If the thickness of a line (in pixels) is less than or equal to this parameter, no end caps will be drawn. Higher values provide decreased redraw times at minimum zoom levels.
Related command	None
Example	`END_CAP=4`

FILE_FORMAT

Syntax	`FILE_FORMAT=type units m.n mode zeros terminator modal`
Description	This parameter defines the default format expected for input files.
Related command	*Files/Format*
Example	`FILE_FORMAT=Drill Excellon Met 3.3 Inc Trail \n Modal`

FILM_BOX

Syntax	`FILM_BOX=x_size,y_size color`
Description	This parameter indicates the size and color of the film box displayed by GerbTool.
Related command	*Options/Film Box*
Example	`FILM_BOX=18.0000,14.0000 Yellow`

FLAGS

Syntax	`FLAGS=n`
Description	This parameter allows you to control some aspects of GerbTool's low level operations in the field. Typically, you would be instructed by OrCAD Technical Support personnel on how to modify this parameter. The value is entered as a hexadecimal number.
Related command	None
Example	`FLAGS=0x04`

Fn

Syntax	`Fn=hex_command_id`
Description	The *Fn*(*n* = 1 - 12) parameters specify the GerbTool commands assigned to the function keys F1 through F12 respectively. Each *Fn* parameter is assigned a command ID value or macro name. See *Appendix A: Command ID values* for a complete list of command ID values. In the example, the *View/Redraw* command is assigned to function key F1 and the macro BESTDRILL is assigned to F2.
Related command	*Options/Key Cmds*
Example	`F1=VR` `F2=BestDrill`

GRID

Syntax	`GRID=vis snap x_size, y_size`
Description	This parameter specifies the state of the system grid at startup. You specify whether the grid is displayed, if grid snap is on/off, and the size of the grid.
Related command	*Options/Grid*
Example	`GRID=ON SNAP 0.025,0.025` `GRID=OFF NOSNAP 0.050,0.050`

HILI_COLOR

Syntax	`HILI_COLOR=reg selgrp drc`
Description	This parameter allows you to control the colors used by GerbTool when highlighting data. The three color values control the color of regular highlights (e.g., *Query/ Highlight* command), select group highlights (e.g., *Edit/Select*), and DRC generated highlights, respectively.
Related command	*Options/Defaults*
Example	`HILI_COLOR=Highlight Highlight Yellow`

HONOR_CRLF

Syntax	`HONOR_CRLF=yes_no`
Description	Under normal circumstances GerbTool automatically detects the type of block terminator (EOB) used when reading a Gerber file. In the unlikely event that a Gerber file contains inconsistent use of an EOB character, this parameter will allow proper reading of the file if each block contains a carriage return or line feed. Default is NO.
Related command	*Files/Format*
Example	`HONOR_CRLF=YES`

LBUTTON

Syntax	`LBUTTON=view_command_id`
Description	This parameter specifies the viewing command ID assigned to the left mouse button. See *Appendix A: Command ID values* for a list of available commands.
Related command	*Options/Key Cmds*
Example	`LBUTTON=VW`

LOAD_OFFSETS

Syntax	`LOAD_OFFSETS=x_offset,y_offset x_scale,y_scale`
Description	This specifies the offsets and scale used when reading in Gerber files.
Related command	*Files/Offsets*
Example	`LOAD_PARM=2.0000,2.0000  0.500,0.500`

MACRO_FILE

Syntax	`MACRO_FILE=filename`
Description	This parameter allows you to specify the filename of a GerbTool macro file. The specified file will be searched for macros and any found will be added to the list of available macros within GerbTool. There can be multiple MACRO_FILE occurrences.
Related command	*Tools/Macro/Load*
Example	`MACRO_FILE=c:\home\gtwin\gtmac\load1.m`

MAP_STRICT

Syntax	`MAP_STRICT=yes_no`
Description	During aperture list merging and compaction, GerbTool normally requires all aspects of two apertures to be exactly the same, to be considered duplicates. Setting this parameter to NO allows GerbTool to relax this requirement and only compare the size and shape. The default is YES.
Related command	None
Example	`MAP_STRICT=NO`

MAX_LAYER

Syntax	`MAX_LAYER=n`
Description	This parameter allows you to control the number of layers that GerbTool can handle. The valid range of values is 16 - 999. Use the minimum value that satisfies your requirements to conserve memory.
Related command	None
Example	`MAX_LAYER=128`

MBUTTON

Syntax	`MBUTTON=view_command_id`
Description	This parameter indicates the viewing command assigned to a mouse middle button click. See *Appendix A: Command ID values* for a list of available commands.
Related command	*Options/Key Cmds*
Example	`MBUTTON=VA`

OVERLAY_MODE

Syntax	`OVERLAY_MODE=yes_no`
Description	This parameter specifies whether overlay mode is enabled at startup.
Related command	*Settings/Ov*
Example	`OVERLAY_MODE=NO`

PLANE_RES

Syntax	`PLANE_RES=n`
Description	This parameter allows you to specify the dots-per-inch (DPI) resolution of the bitmap created when processing a power/ground plane during netlist generation. To allow maximum speed, keep this value to a minimum. Default is 150 DPI.
Related command	None
Example	`PLANE_RES=150`

RBUTTON

Syntax	`RBUTTON=view_command_id`
Description	This parameter indicates the viewing command assigned to a mouse right button click. See *Appendix A: Command ID values* for a list of available commands.
Related command	*Options/Key Cmds*
Example	`RBUTTON=VP`

SKETCH_MODE

Syntax	`SKETCH_MODE=yes_no`
Description	This parameter specifies whether sketch mode is enabled at startup.
Related command	*Settings/Sk*
Example	`SKETCH_MODE=NO`

TOOLBAR

Syntax	`TOOLBAR=yes_no`
Description	This parameter specifies whether the GerbTool toolbar should be initially displayed. Note: GerbTool stores this parameter in GT.CFG.
Related command	*Options/Toolbar*
Example	`TOOLBAR=YES`

TOOLBARn

Syntax	`TOOLBARn=command_id`
Description	This parameter allows you to control the order and number of tool icons that appear in the GerbTool toolbar up to a maximum of 18 (Replace the "n" with a number between 1 and 18.) You can use the same command ID values as used by the *Options/Key Cmds* command. To disable a particular tool icon, assign the value of NONE. Note: GerbTool stores this parameter in GT.CFG.
Related command	None

Example	`TOOLBAR1=EAF`
	`TOOLBAR13=NONE`
	`TOOLBAR18=FPH`

UNDO

Syntax	`UNDO=yes_no`
Description	This parameter specifies whether undo should be initially on upon startup.
Related command	*Settings/Un*
Example	`UNDO=YES`

USERMENUn

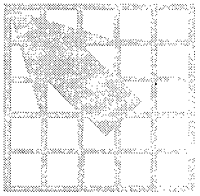
Syntax	`USERMENUn=menu_text, macro_or_cmdid`
Description	This parameter allows you to program the *User* menu. Replace the `n` with a number between 1 and 32, which represents the position within the pull-down menu. The `menu_text` parameter is the text that will displayed in the menu. A character prefixed with an ampersand (&) will be the menu item shortcut key. The `macro_or_cmdid` parameter is the actual macro name or command ID that will be executed when this *User* menu item is selected.
Related command	None

Example	`USERMENU1=&Ship,MyShipMacro`
	`USERMENU2=&Add Draw,EAD`
	`USERMENU3=Best &Drill,BestDrill`

The following is a sample configuration file showing the required format:

```
FILE_FORMAT=Gerber RS274X Imp 2.3 Abs Leading \r\n NOMODAL
DEF_PATH=/usr/gerbs
DEF_DSN_PATH=/usr/designs
DEF_MAP=default.map
DEF_CUSTOM_MAP=custom.map
DEF_MAP_EXT=map
DEF_GERB_EXT=gbr
DEF_NC_EXT=nc
DEF_TOOL_EXT=tf
DEF_HPGL_EXT=hpgl
DEF_PS_EXT=ps
DEF_LJ_EXT=lj
DEF_DSN_EXT=gtd
DEF_REP_EXT=rpt
GRID=OFF NOSNAP 0.025,0.025
LOAD_OFFSETS=0.0000,0.0000 0.0000,0.0000
FILM_BOX=20.0000,16.0000 White
UNDO=YES
END_CAP=4
SKETCH_MODE=NO
OVERLAY_MODE=YES
BG_COLOR=Black
HILI_COLOR=Highlight Highlight Highlight
MAX_LAYER=36
MACRO_FILE=demo.mac
ARCS_SEGMENTED=NO
CHAR_SET=ASCII
MAP_STRICT=YES
CROSSHAIR=36, 34
CHORD_ANGLE=10
PLANE_RES=150
AP_CONV=algro2gt,Allegro
AP_CONV=mentr2gt,Mentor
SPOOL_DIR=/usr2/ps,spoolps.sh
SPOOL_DIR=/usr2/text,lpr -r
LBUTTON=VW
MBUTTON=VA
RBUTTON=VO
F1=VR
F2=VE
F3=VV
F4=LC
F5=LE
F6=AE
F7=AR
F8=OK
F9=QI
F10=QM
F11=ESA
F12=TMT
```

Appendix C



Aperture list file format

This appendix describes the format of a GerbTool aperture list and provides an example of an aperture list.

Aperture lists are stored as simple ASCII files. There are nine fields in each line of the file. Each line defines one D-Code. The fields consist of the following:

Field	*Possible values*
D-Code	10 - 4095
Shape	Round, Square, Rectangle, Oblong, Donut, Diamond, Octagon, Thermal, Therm45, Target, Complex, or a filename prefixed by a "%"
Width	0.0 - 9.9999
Height	0.0 - 9.9999 When referring to Donuts or Thermals, this field represents the diameter of the inner hole. When referring to Targets, it refers to the diameter of the inner ring of the Target.
Type	SM (surface-mount) or TH (through-hole)
Tool	0 - 999 Specifies the Tool used to drill this D-Code.
Tool Size	0.0 - 9.9999 Specifies the size of the above Tool number.
Legend	10 - 4095 Specifies the D-Code to use in place of this D-Code when creating a Drill Drawing using *Tools/NC Drill/Drawing*.
R90	10 - 4095 Specifies the D-Code to substitute for this D-Code when rotating 90 or 270 degrees. This field exists only for compatibility with older versions of GerbTool, as newer versions perform the D-Code substitutions automatically.

Aperture list field definitions.

All fields are separated by white space. Lines that begin with a "#" are treated as comments. Although the author and data comments are not required, they are generally included as an aid for other users. The header of a GerbTool aperture list may contain a format line proceeded by a "%." This line contains either IMPERIAL or METRIC followed by a version number. If IMPERIAL is specified, all sizes are in inches. If METRIC is specified, they are in millimeters. If no format line is provided, IMPERIAL is assumed. The version number is for documentation purposes only. An excerpt from an aperture list showing the required format follows.

```
# Format, Version
%IMPERIAL, V3.0
#
# Author: GerbTool V1.0 (c) 1992 WISE Software
Solutions, Inc.
# Date:   Wed Oct  7 13:28:46 1992
#
#    Shape      Width   Height Type Tool  Size Legend R90
#
D12  Round      0.0100 0.0100 TH   0      0.0  0      0
D21  Square     0.0200 0.0200 TH   2      0.0  0      0
D22  Rectangle  0.0220 0.0180 SM   3      0.0 85      0
D23  Oblong     0.0220 0.0180 TH   3      0.0  0      0
D24  Diamond    0.0240 0.0240 TH   4      0.0  0      0
D25  Target     0.1800 0.1600 TH   0      0.0  0      0
D26  %FIDUCIAL  0.0000 0.0000 TH   0      0.0  0      0
D70  Octagon    0.0240 0.0240 TH   5      0.0  0      0
D71  Thermal    0.0240 0.0200 TH   0      0.0  0      0
```

Sample aperture list file.

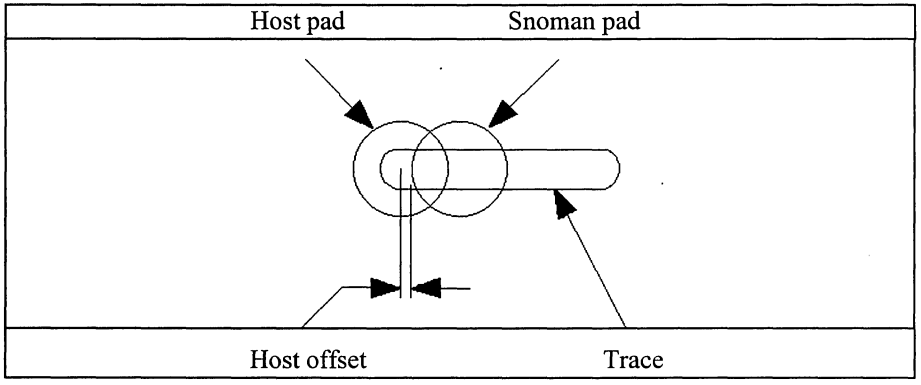
In the above example, D26 is specified as a Custom aperture with a filename of FIDUCIAL.CUS. The "%" is required, to notify GerbTool that what follows is a custom aperture filename.

Appendix D

Snoman concepts

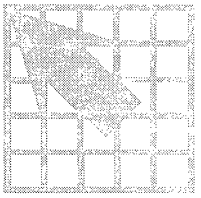
Snoman is a tool designed to create a *maximum material condition* at the point where a trace segment enters a pad, thereby eliminating the possibility of pad/trace separation (breakout). This is accomplished by examining a Gerber file (layer) and outputting pad flashes at the correct locations, and of the correct size, to provide the most material where a trace enters a pad. Automatic adjustments are made to the size and location of the generated Snoman pads to eliminate design rule spacing violations.

The following illustration shows the original pad and trace, as well as the resultant Snoman pad.



Snoman concept.

The distance maintained between the host pad center and the edge of the generated Snoman pad (see *Host offset* in illustration above) is adjustable. Negative values allow the Snoman pads to closely hug the host pads.



A

absolute mode When all X-Y coordinates are referenced to a common origin (0,0).

active layer The layer that all items added to the database will go to.

aperture list A list of Gerber D-Code definitions.

ASCII Acronym for American Standard Code for Information Interchange. This is a standard that relates characters to specific code numbers.

B

block size The size of a coordinate value in characters. Also known as *m.n* format.

breakout Pad and trace separation during manufacturing.

C

checkable button A small square button (box) that appears in a form and that can be selected or cleared. When the check button is selected, a checkmark, or similar symbol, appears in the button.

clicking Pressing and releasing a mouse button.

D-H

desktop The screen background for GerbTool on which Gerber data, menus, icons and dialog boxes appear.

design file A file containing information about the layer structure of a single PCB design. This file also stores various information about the GerbTool operating environment.

double clicking Pressing a mouse button twice in rapid succession.

DRC Acronym for Design Rules Check.

I-L

incremental mode When each X-Y coordinate is a displacement from the previous coordinate.

isolated pads Pads that do not have a trace connected to them.

M

mouse A hand-held pointing device attached to a computer.

mouse cursor An icon that indicates the current mouse position.

N-O

NC drill Refers to files produced to drive Numerically Controlled drilling machines.

netlist A file containing groups of pad X-Y locations that are connected by traces.

P-R

pad removal The act of removing isolated or stacked pads.

pan Moving the location of the viewing window without changing its size.

panelize Placing multiple copies of a PCB on one piece of film. The multiple copies are then manufactured on a single panel, thereby reducing manufacturing costs.

point A X-Y location within the drawing area.

S-U

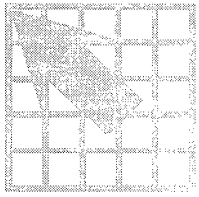
scroll bar A box within a form used to scroll the contents of the form. Move the mouse over the box and press the left mouse button. Without releasing it, move the box up or down by moving the mouse. When you release the mouse button, the form will scroll.

V

virtual memory A combination of hardware and software that allows an application to address all memory that the CPU is capable of addressing, even when there is less actual memory. The virtual memory manager swaps data back and forth to the disk and remaps memory addresses to provide applications with virtually unlimited memory. Available disk space becomes the limiting factor.

W-Z

wildcard specification A method of specifying more than one filename at a time. Use the asterisk (*) character to match any character or group of characters. Use a question mark (?) to match a single character. For example: *.GTD represents all files that end with the .GTD extension.



D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

Z

MN-01-5048 CDS 3K 6/96 Recyclable