

# ***EAGLE***

EASILY APPLICABLE GRAPHICS LAYOUT EDITOR

## **Manual**

**Version 3.55 and later**



**1st Edition**

Copyright © CadSoft 1998

All Rights Reserved

This software and documentation are copyrighted by CadSoft Computer, Inc., doing business under the tradename EAGLE. The software and documentation are licensed, not sold, and may be used or copied only in accordance with the EAGLE License Agreement accompanying the software and/or reprinted in this document. This software embodies valuable trade secrets proprietary to CadSoft Computer, Inc.

All trademarks referenced in this document are the property of their respective owners.

Specifications subject to change without notice.

© Copyright 1996 CadSoft Computer, Inc. All rights reserved worldwide.  
Printed in the United States of America.

OrCAD is a registered trademark of OrCAD, Inc.

## **How to reach us**

Office Hours are:

Mon - Fri: 9 am until 6 pm EST

Phone: (561) 274-8355

Fax: (561) 274-8218

BBS: (561) 278-5749

E-Mail : [Info@CadSoftUSA.COM](mailto:Info@CadSoftUSA.COM)

Web: <http://www.cadsoftusa.com>

**CadSoft Computer, Inc.**

**801 South Federal Highway, Suite 201**

**Delray Beach, Florida 33483-5185**

## **EAGLE LICENSE AGREEMENT**

This is a legal agreement between you, the end user, and CadSoft Computer, Inc., which markets software products under the trademark EAGLE. CadSoft Computer, Inc. shall be referred to in this Agreement as CadSoft. If you do not agree to the terms of this Agreement, promptly return the diskette package and accompanying items (including written materials and containers) to the place you obtained them for a full refund. USE OF THIS PRODUCT CONSTITUTES YOUR ACCEPTANCE OF THESE TERMS AND CONDITIONS AND YOUR AGREEMENT TO ABIDE BY THEM.

### **Grant of License**

CadSoft grants to you the right to use one copy of the accompanying EAGLE software program and any and all updates that you may receive (the Software) on a single computer or workstation. You may, however, install the Software on more than one computer or on a file server provided you do not operate the Software on more than one computer or workstation at a time.

### **Copyright**

The Software is owned by CadSoft and is protected by United States copyright laws and international treaty provisions. Therefore, you must treat the Software like any other copyrighted material (e.g., a book or musical recording). You may not copy the written materials accompanying the Software.

### **Other Restrictions**

You may not rent or lease the Software, but you may transfer your stand-alone copy of the Software and accompanying written materials on a permanent basis provided you retain no copies and the recipient agrees to the terms of this Agreement. Any such transfer must include all updates and prior versions of the Software and accompanying written materials, and notice must be given by you to CadSoft that such transfer has taken place. You may not reverse engineer, decompile, disassemble, or create derivative works based on the Software for any purpose other than creating an adaptation to the Software as an essential step in its utilization for your own use. You acknowledge Cadsoft's claim that the Software embodies valuable trade secrets proprietary to CadSoft; you may not disclose any information regarding the internal operations of the Software to others.

## **LIMITED WARRANTY**

CadSoft warrants the accompanying Software and documentation to be free of defects in materials and workmanship for a period of ninety (90) days from the purchase date. The entire and exclusive liability and remedy for breach of this Limited Warranty shall be, at Cadsoft's option, either (a) return of the price paid or (b) replacement of defective Software and/or documentation provided the Software and/or documentation is returned to CadSoft with a copy of your receipt. Cadsoft's liability shall not include or extend to any claim for or right to recover any other damages, including but not limited to, loss of profit, data or use of the Software, or special, incidental or consequential damages or other similar claims, even if CadSoft has been specifically advised of the possibility of such damages. In no event will Cadsoft's liability for any damages to you or any other person ever exceed the lower of suggested list price or actual price paid for the license to use the Software, regardless of any form of the claim.

TO THE EXTENT PERMITTED UNDER APPLICABLE LAW, CadSoft DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. SPECIFICALLY, CadSoft MAKES NO REPRESENTATION OR WARRANTY THAT THE SOFTWARE IS FIT FOR ANY PARTICULAR PURPOSE, AND ANY IMPLIED WARRANTY OF MERCHANTABILITY IS LIMITED TO THE NINETY-DAY DURATION OF THE LIMITED WARRANTY COVERING THE SOFTWARE AND PHYSICAL DOCUMENTATION ONLY, AND IS OTHERWISE EXPRESSLY AND SPECIFICALLY DISCLAIMED.

THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS; YOU MAY HAVE OTHERS WHICH MAY VARY FROM STATE TO STATE. SOME STATES DO NOT ALLOW THE EXCLUSION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, OR THE LIMITATION ON HOW LONG AN IMPLIED WARRANTY LASTS, SO SOME OF THE ABOVE MAY NOT APPLY TO YOU.

## **GOVERNING LAW AND GENERAL PROVISIONS**

This License and Limited Warranty shall be construed, interpreted and governed by the laws of the State of Florida, U.S.A. If any provision is found void, invalid or unenforceable, it will not affect the validity of the balance of this License and Limited Warranty which shall remain valid and enforceable according to its terms. If any remedy, hereunder, is determined to have failed of its essential purpose, all limitations of liability and exclusions of damages set forth herein shall remain in full force and effect. This License and Limited Warranty may only be modified in writing, signed by you and a specifically authorized representative of CadSoft. All rights not specifically granted in this License Agreement are reserved by CadSoft.

## **U.S. GOVERNMENT RESTRICTED RIGHTS**

The Software and documentation are provided with RESTRICTED RIGHTS. Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights In Technical Data and Computer Software clause at 252.227-7013. Contractor/manufacturer is CadSoft Computer, Inc., 801 South Federal Highway, Suite 201, Delray Beach, Florida 33483-5185, U.S.A.

<b>Introduction</b>	<b>7</b>
<b>Technical Terms</b>	<b>8</b>
<b>EAGLE Modules and Utilities</b>	<b>9</b>
EAGLE Light, Standard, and Professional	9
If you are Using the Layout Editor Only	9
Schematic and Autorouter Module	9
<b>The Control Panel</b>	<b>10</b>
Context Menu	11
Select Icons	11
File Menu	11
Options Menu	12
Window Menu	13
Help Menu	13
<b>The Schematic Editor Window</b>	<b>14</b>
How to Get Detailed Information on a Command	15
The Action Toolbar	16
The Schematic Command Toolbar	17
Commands not Available in the Command Toolbar	20
Mouse Keys	22
<b>The Layout Editor Window</b>	<b>23</b>
Commands in the Layout Toolbar	24
<b>The Library Editor Window</b>	<b>28</b>
Load or Rename Package, Symbol, or Device	29
The Package Editing Mode	30
Symbol Editing Mode	31
Create Actual Components from Symbols and Packages	32
<b>The CAM Processor Dialog</b>	<b>34</b>
Generate Data	35
<b>The Text Editor Window</b>	<b>36</b>
<b>Command Input Alternatives</b>	<b>37</b>
<b>The EAGLE Command Language</b>	<b>39</b>
<b>Grids and the Current Unit</b>	<b>42</b>
<b>Names and Automatic Naming</b>	<b>43</b>
<b>Script Files and Data Import</b>	<b>44</b>
<b>Data Export</b>	<b>45</b>
<b>Individual Configuration of EAGLE</b>	<b>46</b>
<b>The EAGLE User Language</b>	<b>47</b>
<b>Forward&amp;Back Annotation</b>	<b>48</b>
<b>From Schematic to Finished Board</b>	<b>49</b>
Create Schematic	49

---

Points to Note for the Schematic Editor	50
Considerations Prior to Creating a Board	52
Create Board	53
<b>Multilayer Boards</b>	<b>54</b>
Signal Layer	54
Power Supply Layer with One Signal	54
Ground Areas and Supply Layers with More than One Signal	55
<b>Component Design Explained through Examples</b>	<b>56</b>
Resistor Package	57
Resistor Symbol	60
Resistor Device	63
Labeling of Schematic Symbols	65
Mirrored Layer	66
Supply Symbols	67
Component Power Supply Pins	69
Pins with the same Names	71
More about the Addlevel Parameter	72
Relay: Coil and First Contact must be Placed	73
Plug: Some Connection Surfaces can be Omitted	73
Plug with Fixing Hole and Forbidden Area	75
Drawing Frames	76
<b>Preparing the Manufacturing Data</b>	<b>77</b>
Which Files does the Board Maker Need?	77
Set Output Parameters	80
Aperture Configuration File with the Same Units	81
Automating the Output with CAM Processor Jobs	82
Creating your own Device Driver	83
Film Creation using PostScript Files	84
Gerber Files for Photoplotters with Variable Aperture Wheels	85
Gerber Files for Photoplotters with Fixed Aperture Wheels	88
Drill Data and Drill Plan	92
<b>The Autorouter</b>	<b>94</b>
Basic Features	94
What Can be Expected from the Autorouter	94
Controlling the Autorouter	95
Control Parameters and their Meanings	96
What Has to be Defined in the Control File Before Autorouting	97
Grid Considerations	99
Considerations for Preferred Directions	101
Restricted Areas for the Autorouter	101
How the Cost Factors Influence the Routing Process	102
Number of Ripup/Retry Attempts	104
Backup, Interruption of Routing, and Best Data	105
Setting the Control Parameters in the Autorouter Menu	106
Polygons as Supply Layers	108
Information for the User	109

---

## Introduction

---

This manual describes the use of the EAGLE software and its basic principles.

For a quick, hands-on introduction, refer to the EAGLE Tutorial. You will also find a list of EAGLE's key features there.

Detailed information, especially about the EAGLE command language and the EAGLE user language, is available on the Help pages. You can reach a basic understanding very quickly by using this manual, and you can use the convenient search features of the Help function to quickly locate the answers to particular questions.

## Technical Terms

---

**Airwire:** Unrouted connection on a board, displayed in the unrouted *layer* (= rubberband).

**Electrical Rule Check (ERC):** EAGLE can locate the violation of certain electrical rules (e.g. if two outputs are connected) with the ERC.

**Design Rule Check (DRC):** EAGLE can locate the violation of certain design rules (e.g. if two different tracks overlap or are too close) with the DRC.

**Gate:** The term *gate* is used in this manual for a part of a component which can be individually placed on a schematic. This can be one gate of a TTL component, one contact pair in a relay, or an individual resistor from a resistor array.

**Net:** Electrical connection in a schematic.

**Package:** Component footprint stored in a library.

**Pad:** Through-hole pad associated with a package.

**Pin:** Connection point on a schematic symbol.

**Ratsnest:** Command for calculating the shortest airwires.

**Signal:** Electrical connection in a board.

**Symbol:** Schematic representation of a component, stored in a library.

**Via:** Plated-through hole for changing the layer of a track.

**Wire:** Electrical connection in a board, or a line (since lines are drawn with the WIRE command).



## EAGLE Modules and Utilities

---

### EAGLE Light, Standard, and Professional

---

EAGLE is offered in different performance/price categories (editions) called Light, Standard, and Professional. You will find the restrictions of the Light and Standard editions on the Help pages (search for *License*).

### If you are Using the Layout Editor Only

---

The basic EAGLE software package comes with the layout editor, which allows you to design printed circuit boards (pcbs), plus the library editor, the CAM processor, and the text editor. With the library editor you can only design packages (footprints), unless you have the schematic editor as well. The CAM processor is the program which generates the output data for the production of the pcb (e.g. Gerber or drill files).

The basic software package also includes the utility programs XPAD, XCONVERT, and ELC. They allow you to change pads in boards or libraries, to convert netlists into the EAGLE script format, and to convert OrCAD libraries into the EAGLE format. See the \*.DOC files in your EAGLE subdirectory ..\DOC.

### Schematic and Autorouter Module

---

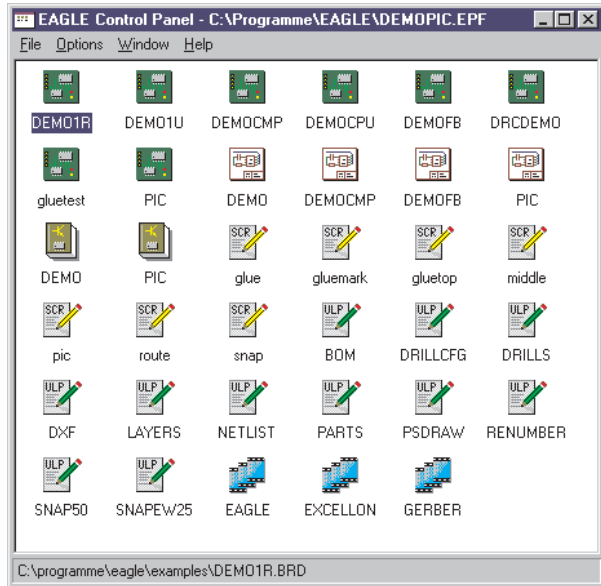
A schematic module and an autorouter module are available for the EAGLE package. The term module is used because EAGLE always behaves like one single program. The user interface is identical for all the program parts.

If you own the schematic module, you can generate the board from the schematic diagram with a single mouse click. EAGLE then changes to the layout editor, where the packages are placed next to an empty board - connected via airwires (rubberbands). From here you can go on designing with the layout editor as usual. Schematic and layout are automatically kept consistent by EAGLE (Forward&Back annotation). Schematic diagrams can consist of up to 99 sheets.

If you own the autorouter module, you can route the airwires automatically. You can choose single nets, groups of nets or all nets for the automatic routing run.

## The Control Panel

The control panel appears when you start EAGLE. It is the main window from where you can open schematics, layouts, libraries, text files, script files, or user language programs (ULP's). The CAM processor can be opened from the control panel as well.



The control panel is similar to a folder, such as will be familiar from the Windows environment, except that the icons represent different objects (which can be located in different directories).

The icons have the following meanings:



**Board**



**Schematic**



**Library**



**CAM Job**



**Script File**



**ULP**

## Context Menu

---

If you click one of these icons with the right mouse button, a context menu appears offering the following choices:

<b>Open:</b>	open file.
<b>Print:</b>	print file.
<b>Copy:</b>	copy file.
<b>Rename:</b>	rename file.
<b>Delete:</b>	delete file.

If you click Help, you will get detailed information about the context menu.

## Select Icons

---

If you click the icon of an object with the left mouse button, the status line shows the file name with the complete path information.

A double click with the left mouse button opens the appropriate editor window or the CAM processor, and loads the file.

## File Menu

---

The *File* menu contains the following items:

### **New**

Generates a file of the type board, schematic, library, CAM job, ULP script, or text.

The default directories for the various file types are defined in the *Options/Directories* menu.

Additionally, the menu item *New* allows you to create a new project file.

CAM jobs are definitions for generating output data, created with the CAM processor.

Script and ULP files are text files containing command sequences in the EAGLE command language or the EAGLE user language. They can be created and edited with the EAGLE text editor or with any other text editors.

### **Open**

Opens an existing file of the types mentioned above.

The default directories for the various file types are defined in the *Options/Directories* menu.

## Save Project as

A project file is saved under a new name.

The default directory for project files is the current directory. If you work on a Windows platform, you can find out the current directory by clicking the EAGLE icon with the right mouse button and selecting *Properties/Shortcut* (see field: *Start in*).

## Save all

All of the currently opened files are saved.

## CAM Processor

The CAM processor is called.

## Exit

The program is terminated. When EAGLE is started again, the last program status is restored, i.e. the windows and other working environment parameters, appear unchanged.

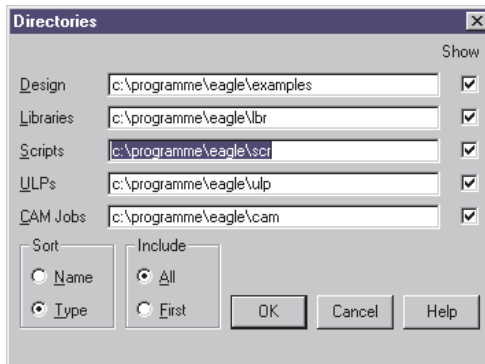
The current status is also saved when you leave EAGLE with Alt-X from any program part.

## Options Menu

---

### Directories

The default directories for certain EAGLE files are entered into the *Directories* dialog box. Several directory entries can be separated by a semicolon. Exception: Only one design directory is allowed. This directory contains the schematics and the boards of a project. The design directory is also the default directory for the text editor.



If the box on the right of a directory field is checked, the related files are shown as icons in the control panel.

With the radio button *All* you can specify that the files in all the paths that have been entered are shown as icons. If *First* is checked, only the first path in a field is relevant for the icon display.

The radio buttons *Name* and *Type* specify whether the icons are sorted according to their name or extension.

### **Backup**

When files are saved, EAGLE creates backup copies of the previous files. The field *Maximum backup level* allows you to enter the maximum number of backup copies (default: 9).

If the option *Automatically save project file* is chosen, your project is automatically saved when you leave the program, provided you have created a project file.

### **User Interface**

The *User interface* dialog allows you to adjust the appearance of the schematic, board, or library windows to your own needs. Changes take effect only after the editor window concerned has been reopened.

Click the *Help* button in this dialog window to get more detailed information.

---

## **Window Menu**

From the *Window* menu you can choose the window (schematic, board, etc.) to be displayed in the foreground. The number on the left is the window number. It allows you to choose a window when combined with the Alt key (e.g. Alt+1 selects window 1).

---

## **Help Menu**

The *Help* menu contains an item for calling the help function, as well as items for installing a new license (*Product registration*) and getting information about the program version etc. (*Product information*).

### **Product registration**

The Registration dialog is called automatically when you start EAGLE the first time. If you want to install an upgrade you must start this dialog from the help menu, and then enter the necessary information according to the *License/Product Registration* section of the help function.

### **Product information**

If you contact our technical support you should provide the information you will find under this menu item .

## The Schematic Editor Window

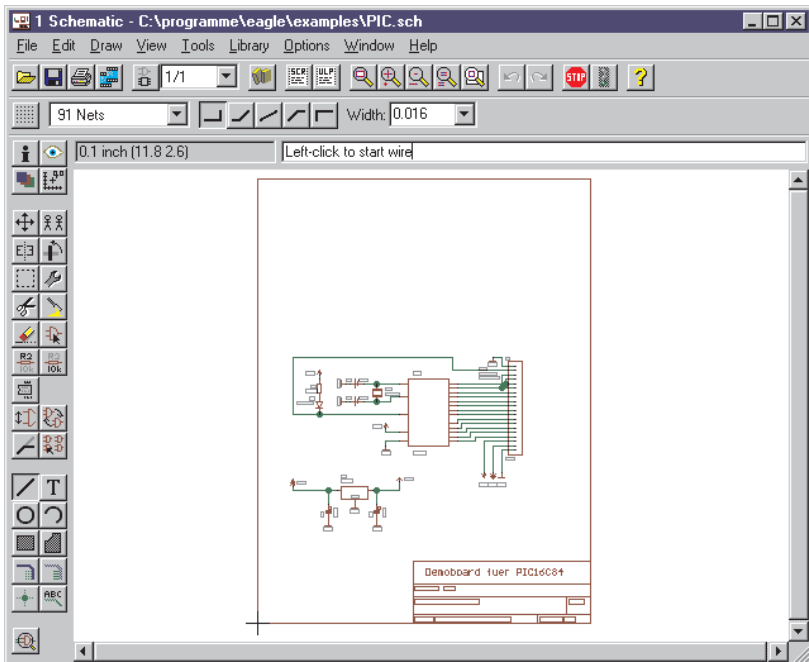
The schematic editor window opens when you load an existing schematic or create a new one .

On top you will see the **title bar**, which contains the file name, and then the **menu bar**, and the **action toolbar**.

Below the action toolbar there is the **parameter toolbar** which contains different icons, depending on the active command.

Above the work space on the left you find the **coordinates display** and, on its right, the **command line**. Hints for the user on what to do next appear in the command line, provided a command has been selected. The command line can also be used for entering commands as text. EAGLE accepts commands in different but equivalent ways: as mouse clicks, text via keyboard, or from command (script) files.

On the left of the work space you find the command toolbar, which contains most of the schematic editor's commands.



---

## How to Get Detailed Information on a Command

---

### Bubble Help or Tool Tips

If the mouse cursor is positioned over an icon for some time, a short explanation appears, starting (in most cases) with the name of the related EAGLE command.

Example: *Wire: draw lines*



### Help Function

If you want to learn more about a command, e.g. the WIRE command, click its icon in the command toolbar, then click the help icon.

As an alternative you can type

HELP WIRE ←

into the command line. The ← character symbolizes the Enter key.

The syntax of the EAGLE command language is explained in its own section.

### Command Parameters

A number of EAGLE commands need additional parameters. Refer to the help pages for a description of the textual parameter entry (via command line or script file).

Most of the parameters can be entered by clicking the appropriate icons in the parameter toolbar, which changes according to the selected command. These icons also show bubble help explanations.

This is how the parameter toolbar appears when the WIRE command is activated.



Left: a selection menu for a set of fixed parameters (drawing layers in this particular case). Center: buttons for the bending angle of the wire (wire style). Right: a selection menu which allows the direct entry of a value.



**GRID command:** This icon is available at any time. It is used to adjust the grid and to select the current unit. In EAGLE, any value relates to the current unit.

## The Action Toolbar

---



Open file, save file, print file, call CAM processor, open/create corresponding board file (BOARD command).



Load/create another schematic sheet.



**USE command:** Load a library for the subsequent placement of parts (from this library) into a schematic (or a board, if the layout editor is active).



**SCRIPT command:** Execute a script file. This enables you to execute any command sequence with a few mouse clicks.



**RUN command:** Start a user language program (ULP).



**WINDOW command:** These icons represent different modes of the WINDOW command: Fit drawing into the screen (Alt-F2), zoom in (F3), zoom out (F4), redraw screen (F2), display new area.



**UNDO and REDO:** These commands allow you to cancel previous commands and to execute commands which have previously been cancelled. Function keys: F9 and F10 (default).



Terminates the execution of EAGLE commands.



Starts the execution of the active EAGLE command. This is only necessary if further parameters could be entered by the user.



---

## The Schematic Command Toolbar

---



### INFO

Provides information about the object to be selected.



### SHOW

Highlights the object to be selected.



### DISPLAY

Select and deselect the layers to be displayed. See *Help* for the meaning of the layers.



### MARK

The following mouse click defines the new (relative) origin for the coordinates display. If you click the MARK icon and then the traffic light icon, the coordinates display will be related to the absolute origin of the drawing.



### MOVE

Move any visible object. The right mouse button rotates the object.

If you move a net over a pin, no electrical connection will be established. If you move the pin of a gate over a net or another pin, an electrical connection (net) will be created.

To move groups of objects: define the group with the GROUP command, click the MOVE icon, then select the group with the right mouse button and move it to the desired location.



### COPY

Copy objects.



### MIRROR

Mirror objects.



### ROTATE

Rotate objects (also possible with MOVE).



### GROUP

Define a group which can then be moved, rotated, or copied (with CUT and PASTE) to another drawing. After the icon has been clicked, a rectangular group can be defined by holding down the left mouse button and dragging the cursor to the diagonal corner of the rectangle. If you want to define a group by a polygon, use the left mouse button to determine the corners of the polygon. Then click the right mouse button to close the polygon.



## CHANGE

Change the attributes of an object, e.g. the width of a line or the size of text. See *Help*.



## CUT

Transfer the objects of a previously defined group into the paste buffer. See PASTE command. Not identical to the Windows Cut function.



## PASTE

Insert objects from the paste buffer into the drawing. Restrictions: see *Help*. Not identical to the Windows Paste function.



## DELETE

Delete visible objects.



## ADD

Add library parts to a drawing. A library must already have been loaded (USE command).



## NAME

Give names to components, nets, or buses.



## VALUE

Provide values for components. Integrated circuits normally get the IC type (e.g. 7400) as their value.



## SMASH

Separates texts for name and value from a component, so that they can be placed individually. The size of smashed texts can also be individually changed.



## PINSWAP

Swap two nets connected to equivalent pins of a component, provided the pins have been defined with the same swaplevel.



## GATESWAP

Swap two equivalent gates of a component, provided the gates have been defined with the same Swaplevel. In the EAGLE terminology, a gate is a part of a component which can be individually placed on a schematic (e.g. one transistor of a transistor array).

**SPLIT**

Insert an angle into a wire.

**INVOKE**

Fetch a particular gate from a component (e.g. gate D before gate C). This command allows you also to add a gate from a component which is located on another sheet. In such a case, type the name of the component (e.g. IC1) into the command line after the INVOKE command has been selected.

**WIRE**

Draw line (this command is called WIRE because it is used to define electrical connections, i.e. wires, in the layout editor).

**TEXT**

Place a text string. Use CHANGE SIZE and CHANGE RATIO to adjust the text height and the width of the lines forming the characters. CHANGE TEXT allows you to change the contents of the text string. Label texts can be changed by changing the name of a net or a bus with the NAME command.

**CIRCLE**

Draw a circle.

**ARC**

Draw an arc.

**RECT**

Draw a rectangle.

**POLYGON**

Draw a polygon.

**BUS**

Draw a bus line. The meaning of a bus is more conceptual than physical - it is only a means to make a schematic easier to read. Only nets define an electrical connection. Nets, however, can be dragged out of a bus.

**NET**

Draw a net. Nets with the same name are connected (even if located on different sheets).

Nets and pins which appear to the eye to be connected are not necessarily electrically connected. Please check with the **SHOW** command, or by exporting a netlist with the **EXPORT** command. See also **MOVE**.



### **JUNCTION**

Place the symbol for a net connection.



### **LABEL**

Place the name of a bus or net as a label. Labels cannot be changed with **CHANGE TEXT** but rather with the **NAME** command.



### **ERC**

Perform an electrical rule check and a consistency check for schematic and board.

---

## **Commands not Available in the Command Toolbar**

---

Menu items already explained in the control panel section are not discussed here.

The following commands can be entered into the command line. Some of them are available as menu items.

### **CLOSE**

Text command for closing the library editor.

### **EDIT**

Text command for loading a file or a library object. You can, for instance, load a board from the schematic editor.

### **WRITE**

Text command for saving the currently loaded file. Please note that, in contrast to *Save as*, the name of the currently edited file is never changed when the **WRITE** command is used.

### **OPEN**

Text command for opening a library for editing. This command is not identical to the *File/Open* menu item of the schematic editor, which only lets you select schematics. You can use the **OPEN** command as an alternative to the *File* menu of the control panel.

### **DIR**

Show the contents of a directory.

### **QUIT**

Quit EAGLE. Identical with the menu item *File/Exit* or **Alt-X**.

**PRINT**

Call up the Windows print dialog. Normally the PRINT command is used to print schematics or check the drawings needed for the pcb production. The actual production data are generated with the CAM processor. If you want to output your drawing in black and white check the *Black* option (and *Solid*, if you don't want colors to be printed as different levels of gray). The caption text is suppressed unless you check *Caption* (available via the *Page* button) . Set *Page limit* to 1 if your drawing is to be fitted on one page.

**EXPORT**

Output lists (especially netlists), directories, or script files.

**ASSIGN**

Assign function keys. The most convenient way of doing this is to use the *Options/Assign* menu.

**MENU**

Define the contents of certain menus.

**SET**

Set system parameters and modes. Best done via the *Options/Set* menu item. Please note that not all of the possibilities are available through this dialog. Presettings can be defined in the script file EAGLE.SCR by using text commands.

**LAYER**

Choose or define drawing layer. When using drawing commands the layer can be chosen in the parameter toolbar.

**REMOVE**

Delete files or schematic sheets.

REMOVE .S3 ←

for instance, deletes sheet 3 of the loaded schematic.

### Mouse Keys

---

The middle and right mouse button have a special meaning for a number of commands.

You can use the middle mouse button only if the operating system knows your mouse is a 3-button mouse, that is your mouse must be installed this way.

The Help section on *Keyboard and Mouse* contains a table showing the commands in which the middle and right button have a special meaning.

#### **Selecting Neighboring Objects**

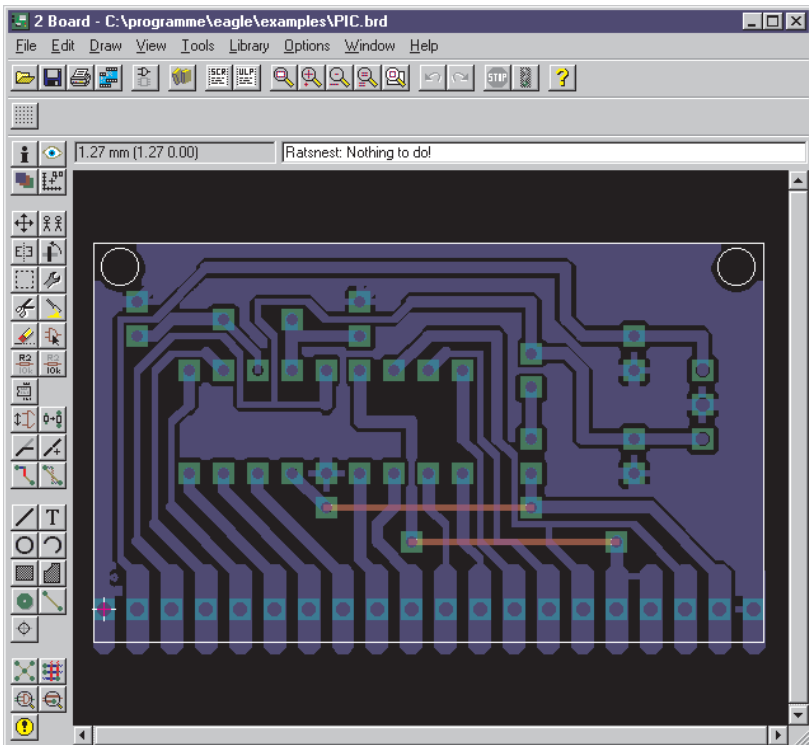
If one of two objects which are very close together is to be selected, the individual objects are highlighted one after the other. The user can select the highlighted object with the left mouse button, or proceed to the next one with the right mouse button. See Help (SET SELECT\_FACTOR).

## The Layout Editor Window

The layout editor window opens when you open an existing board file or when you create a new board. If you own the schematic editor you will normally draw a schematic first and then generate the board file with the BOARD command, or by clicking the “switch to board” icon.

The layout editor window appears very much like the schematic editor window. Even if you don't work with the schematic editor, you should study the previous section, as most of the information there applies to the layout editor, too.

Only the commands in the command toolbar are discussed again, as some commands differ in their use.



## Commands in the Layout Toolbar

---



### INFO

Provides information about the object to be selected.



### SHOW

Highlights the object to be selected.



### DISPLAY

Select and deselect the layers to be displayed. Components on the top side of the board can only be selected if the tOrigins layer is displayed. The same applies to components on the bottom side of the board and the bOrigins layer.

See *Help* for the meaning of the layers.



### MARK

The following mouse click defines the new (relative) origin for the coordinates display. If you click the MARK icon and then the traffic light icon, the coordinates display will be related to the absolute origin of the drawing.



### MOVE

Move any visible object. The right mouse button rotates the object.

The MOVE command cannot connect signals even if a wire (trace) is moved over another wire or a pad. Use ROUTE or WIRE to route signals.

See schematic editor section for moving groups.



### COPY

Copy object.



### MIRROR

Mirror object. Components can be placed on the opposite side of the board by using the MIRROR command.



### ROTATE

Rotate object (also possible with MOVE).



### GROUP

Define group. See schematic editor.



**CHANGE**

Change the attributes of an object, e.g. the width of a track or the size of a text. See *Help*.

**CUT**

Transfer the objects of a previously defined group into the paste buffer. See PASTE command. Not identical to the Windows Cut function.

**PASTE**

Insert objects from the paste buffer into the drawing. Restrictions: see *Help*. Not identical to the Windows Paste function.

**DELETE**

Delete visible objects.

If a group has been defined, it can be deleted with the right mouse button.

If objects cannot be deleted, check whether there are any error polygons left over by the DRC (command DRC CLEAR deletes them). Another reason could be that components cannot be selected because the tOrigins or bOrigins layer is not displayed.

**ADD**

Add library parts to a drawing. A library must already have been loaded (USE command).

**NAME**

Give names to components or signals.

**VALUE**

Provide values for components. Integrated circuits normally get the IC type (e.g. 7400) as their value.

**SMASH**

Separates texts for name and value from a component, so that they can be placed individually. The size of smashed texts can also be changed individually.

**PINSWAP**

Swap two signals connected to equivalent pads of a component, provided the pins have been defined with the same swaplevel.



## REPLACE

Replace a package with another package from any library.



## SPLIT

Insert a bend into a wire.



## OPTIMIZE

Joins wire segments in a signal layer which lie in one straight line.



## ROUTE

Route signal manually. Airwires are converted to wires.



## RIPUP

Convert routed wires (tracks) into unrouted signals (airwires). Change the display of polygons to “outlines”.



## WIRE

Draw line. If used in the layers 1 through 16, the WIRE command creates electrical connections.



## TEXT

Place a text string.

Use `CHANGE SIZE` and `CHANGE RATIO` to adjust the text height and the width of the lines forming the characters. `CHANGE TEXT` allows you to change the contents of the text string.



## CIRCLE

Draw a circle. This command creates restricted areas for the autorouter if used in the `tRestrict`, `bRestrict`, or `vRestrict` layers.



## ARC

Draw an arc.



## RECT

Draw a rectangle. This command creates restricted areas for the autorouter if used in the `tRestrict`, `bRestrict`, or `vRestrict` layers.



## POLYGON

Draw a polygon.

Polygons in the signal layers are treated as signals. They keep an adjustable distance to objects belonging to other signals (copper pouring, flood fill). This enables you to realize different signal areas on the same layer.

Supply layers with one signal are best realized by renaming an inner layer (2..15). If you, for instance, want to define a layer for the signal GND, you can rename e.g. layer 2 to \$GND by using the following command:

```
LAYER 2 $GND ←
```

The POLYGON command creates restricted areas for the autorouter if used in the layers tRestrict, bRestrict, or vRestrict, .

For other possibilities of the POLYGON command see Help.



## VIA

Place a via-hole. Vias are placed automatically if the layer is changed during the ROUTE command. You can assign a via to a signal with the NAME command by changing its name to the name of the signal.



## SIGNAL

Manual definition of a signal. This is not possible if the Forward&Back annotation is active. In that case you have to define the connection with the NET command in the schematic editor.



## HOLE

Define a mounting hole.



## RATSNEST

Calculate the shortest airwires and the “real mode” display of polygons. The polygon calculation can be deactivated with the SET command (*Options/Set/Misc* menu).



## AUTO

Start autorouter.



## ERC

Perform consistency check of schematic and board.



## DRC

Perform Design Rule Check.



## ERRORS

Show errors found by the DRC.

## The Library Editor Window

---

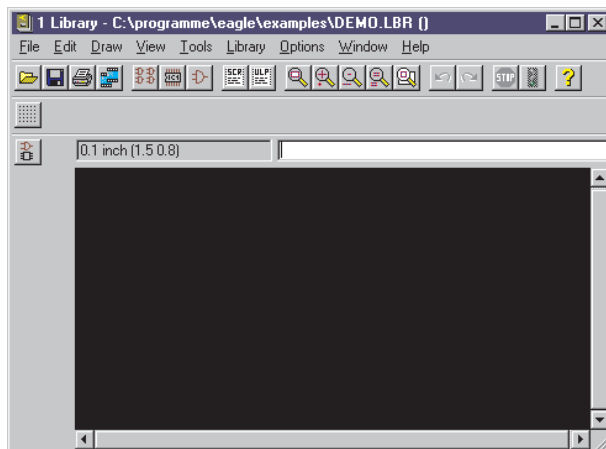
The library editor window opens when you load a library for creating or editing components.

If you don't have the schematic editor, you can only edit packages which are the footprints of components. If you do own the schematic editor, you can also edit symbols and devices.

A device is usually a completely defined component which can contain one or several (identical or different) symbols along with a package.

A library need not contain only real components. Ground or supply symbols as well as drawing frames can also be stored as devices in a library.

When you open a library the following window appears:



## Load or Rename Package, Symbol, or Device

---



### EDIT

Load device or package (if you only have the layout editor) for editing.



Load device, load package, load symbol.

### REMOVE

Delete device/package/symbol from library. Available only through the *Library* menu or the command line. See *Help*.

### RENAME

Rename device/package/symbol. Available only through the *Library* menu or the command line. See *Help*.

## The Package Editing Mode

---

The icons available in the command toolbar are equivalent to the identical icons of the schematic or layout editor.

### Design New Package

Place pads (through-hole contacts) or SMDs (SMD contact areas) with the following commands which are only available in the package editing mode.



#### PAD

Place the pad of a conventional component. See the section on *Automatic Naming* (page 43).



#### SMD

Place an SMD pad. See the section on *Automatic Naming*(page 43).

You can change the name of the pads or SMDs with the NAME command.

Use the WIRE, ARC, etc. command to draw

- the symbol for the silkscreen on the tPlace layer,
- additional graphical information for the documentation print into the tDocu layer.

Draw restricted areas for the autorouter, if needed, in the layers tRestrict, bRestrict, or vRestrict, by using the commands CIRCLE, RECT, or POLYGON.

Place mounting holes with the HOLE command, if needed.

Use the TEXT command to place

- the string >NAME in the tNames layer, serving as a text variable containing the name of the component,
- the string >VALUE in the tValues layer, serving as a text variable containing the value of the component.

### Design a New Package from an Existing One

Load the existing package, display all layers, define a group (GROUP command) containing all of the objects, click the CUT icon and then the origin of the drawing area (coordinates 0, 0).

Open a new package for editing, click the PASTE icon and then the origin of the drawing area.

Edit the objects in whatever way you want.

---

## Symbol Editing Mode

---

Defining a symbol means defining a part of a device which can be placed individually. In the case of a 7400 this could be one NAND gate and the two power pins, defined as another symbol. In the case of a resistor, the device contains only one symbol which is the representation of the resistor.

The icons available in the command toolbar are equivalent to the identical icons of the schematic or layout editor.

### Design a New Symbol

Use the commands WIRE, ARC, etc. to draw the schematic representation of the symbol into the symbols layer.

Place the pins by using the following command (which is only available in the symbol editing mode):



### PIN

Place pins.

You can adjust the pin parameters (*name, direction, function, length, visible, swaplevel*) while the PIN command is active, or with the CHANGE command. See *Help* for the meaning of the pin parameters.

Use the TEXT command to place

- the string >NAME in the tNames layer, serving as a text variable containing the name of the component,
- the string >VALUE in the tValues layer, serving as a text variable containing the value of the component.

### Design a New Symbol from an Existing One

Load the existing symbol, display all layers, define a group (GROUP command) containing all of the objects, click the CUT icon and then the origin of the drawing area (coordinates 0, 0).

Open a new symbol for editing, click the PASTE icon and then the origin of the drawing area.

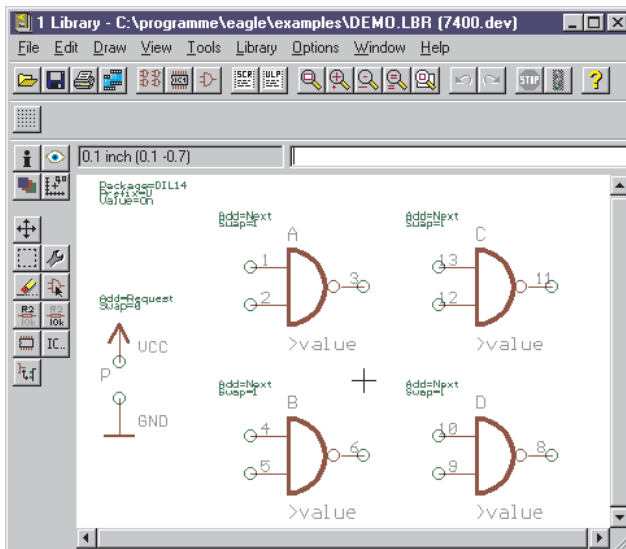
Edit the objects in whatever way you want.

## Create Actual Components from Symbols and Packages

Components are defined as devices. In the device editing mode you do not draw anything, but you define the following:

- which package is used,
- which symbol(s) is/are used (called gate within the device),
- which names are provided for the gates (e.g. A, B),
- if there are equivalent gates which can be interchanged (swaplevel),
- how the gate behaves when added to a schematic (addlevel),
- the prefix for the component name, if a prefix is used,
- if the value of the component can be changed or if the value should be fixed to the device name,
- which pins relate to the pads of the package.

The figure shows the complete defined TTL 7400 component containing four NAND gates and one power “gate”.





For these tasks you can use the following commands.



## PACKAGE

Choose the package.



## ADD

Add a symbol to a device. Gate name, swaplevel, and addlevel can be defined in the ADD command, or redefined later with the CHANGE command.

The swaplevel specifies whether there are equivalent gates.

The addlevel defines, for instance, if a gate is to be added to the schematic only on the users request. Example: the power “gate” of an integrated circuit which is normally not shown on the schematic.



## NAME

Change gate name.



## CHANGE

Change swaplevel or addlevel.



## PREFIX

Provide prefix for the component name in the schematic (e.g. R for resistors).



## VALUE

On: Component value can be changed in the schematic. Off: Component value is set to the device name and cannot be changed (makes sense with IC's). The libraries delivered with the EAGLE software normally default to VALUE ON.



## CONNECT

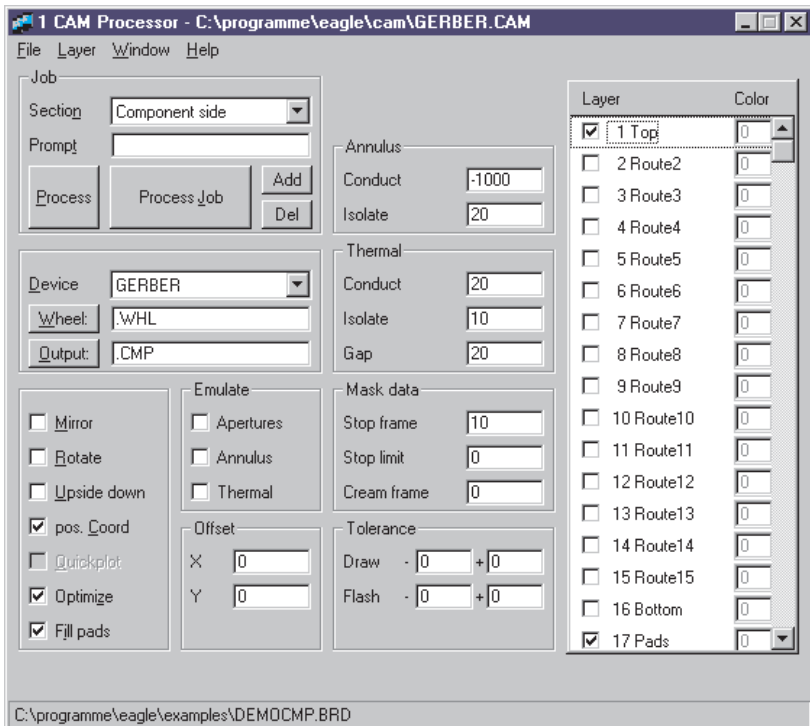
Define which pins (gate) relate to which pads (package).

## The CAM Processor Dialog

The CAM processor allows you to print out schematics or board layouts, or to output the data needed for manufacture of the pcb. While the PRINT command uses the Windows printer drivers, and therefore is not capable of generating manufacturing data, the CAM processor uses proprietary EAGLE drivers.

These drivers are defined in the EAGLE.DEF file. This can be edited with a text editor.

The EAGLE license terms allow you to give away the CAM processor to your pcb house. All you have to do is to send a copy of the demo, since this contains the working CAM processor. Please do not give away the original software including the Installation Code. This is a violation of the license agreement and will be prosecuted.



## Generate Data

---

### Load Job File

A job defines the sequence of several output steps. You can, for example, use a job to generate individual files containing the Gerber data for several pcb layers.

A job is loaded with the *File* menu of the CAM processor or via the control panel.

Output can be generated without defining a job.

### Load Schematic or Board

Before you can generate an output you must open the *File* menu and load a schematic or board.

### Output Parameters

If a job file is loaded, the output parameters are already adjusted. A job can contain several sections with different parameter sets. The various peripheral devices accept different parameters.

If no job is loaded, set the parameters according to your needs (see *Help*).

### Start Output

If you want to execute the job which has been loaded, click the *Process Job* button. If you just want to get an output using the currently visible parameter settings, click the *Process* button.

### Define New Job

Perform the following steps to define a new job:

1. Click *Add*, to add a new section .
2. Set parameters.
3. Repeat 1. and 2. if necessary.
4. Save job with *File/Save job*.

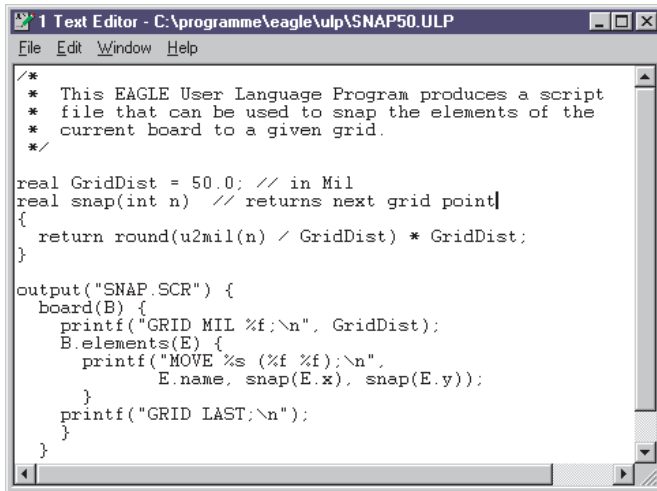
## The Text Editor Window

---

EAGLE contains a simple text editor.

If you load a script file or a user language program using the control panel, the default directories defined in the *Options/Directories* dialog of the control panel are used. The *Design Directory* is the default directory for other text files.

In the text editor, the right mouse key calls up a context menu.



```
1 Text Editor - C:\programme\eagle\ulp\SNAP50.ULP
File Edit Window Help
/*
 * This EAGLE User Language Program produces a script
 * file that can be used to snap the elements of the
 * current board to a given grid.
 */
real GridDist = 50.0; // in Mil
real snap(int n) // returns next grid point
{
  return round(u2mil(n) / GridDist) * GridDist;
}
output("SNAP.SCR") {
  board(B) {
    printf("GRID MIL %f;\n", GridDist);
    B.elements(E) {
      printf("MOVE %s (%f %f);\n",
            E.name, snap(E.x), snap(E.y));
    }
    printf("GRID LAST;\n");
  }
}
```

---

## Command Input Alternatives

---

As an alternative to the mouse, the EAGLE schematic, layout, and library editors allow you enter all of the commands:

- on a command line by typing text commands,
- via function keys,
- via script files.

In any case it is necessary to understand the syntax of the EAGLE command language which is described in the following section. The commands are described in detail on the *Help* pages.

### Command Line

When entering commands into the command line you may abbreviate key words as long as they cannot be mistaken for another key word, or you may use small or capital letters (the input is not case sensitive). The following command line entries are equivalent:

```
CHANGE WIDTH 0.024
```

is equivalent to

```
cha wi 0.024
```

The actual unit for values is set in the GRID menu.

### History Function

You can recall the most recently entered commands by pressing Crsr-Up (↑) or Crsr-Down (↓) and edit them. The Esc key deletes the contents of the command line.

### Function Keys

Texts may be allocated to the function keys and to combinations of those keys with Alt, Ctrl and Shift). If a function key is pressed, this corresponds to the text being typed in via the keyboard. Since every command is capable of being entered as text, every command, together with certain parameters, can be assigned to a function key. Even whole sequences of commands can be assigned to a function key in this way.

The command

```
ASSIGN; ←
```

displays the current assignments of EAGLE commands to the function keys. The best way to make changes here is via the *Options/Assign* menu in the schematic or Layout editor.

### Script Files

Script files are a powerful tool. They can contain long sequences of commands, such as the specification of specific colors and

fill-patterns for all layers. On the other hand they might contain netlists converted from the data of other programs.

The SCRIPT command is used to execute scripts.

ELC, a program included in the EAGLE package, for example, generates a script file from an external library which creates an entire EAGLE library.

EAGLE can itself create a script file for an entire library with the EXPORT command. This file can be edited with a text editor, after which it can be once more imported. General alterations to a library can be carried out quite easily in this way.

See also *Help*.

### **Mixed Input**

The various methods of giving commands can be mixed together.

You can, for instance, click the icon for the CIRCLE command (which corresponds to typing CIRCLE on the command line), and then type the coordinates of the center of the circle and of a point on the circumference in this form

(0 0) (0 1) ←

On the command line.

The values used above would, if the unit is currently set to *inch*, result in a circle with a radius of one inch centered on the coordinate origin. It is irrelevant whether the CIRCLE command is entered by icon or by typing on the command line.

---

## The EAGLE Command Language

---

You only need a knowledge of the EAGLE command language if you want to make use of the alternative input methods discussed in the previous section.

Precise descriptions of the commands are to be found on the associated *Help* pages.

The syntax of the EAGLE command language will be discussed in this section, and typographical conventions, which are important for understanding the descriptions, will be specified.

### Enter Key and Semicolon

If EAGLE commands are entered via the command line they are finished with the enter key. In some cases a command must have a semicolon at the end, so that EAGLE knows that there are no more parameters. It is a good idea to close all commands in a script file with a semicolon.

The use of the enter key is symbolized at many places within this handbook with the

←

sign.

However in the following examples neither the enter key sign nor the semicolon are shown, since all of these commands can be used both on the command line and within script files.

### Bold Type or Upper Case

Commands and parameters shown here in UPPER CASE are entered directly. When they are entered, there is no distinction made between upper and lower case. For example:

Syntax:

GRID LINES

Input:

GRID LINES

grid lines

### Lower Case

Parameters shown here in lower case are to be replaced by names, numbers or keywords. For example:

Syntax:

GRID grid\_size grid\_multiple

Input:

GRID 1 10

This sets the grid to 1 mm (assuming that the current unit is set to mm). Every tenth grid line is visible. The figures 1 and 10 are placed into the command instead of the placeholders *grid\_size* and *grid\_multiple*.

## Underscore

In the names of parameters and keywords the underscore sign is often used in the interests of a clearer representation. Please do not confuse it with an empty space. As can be seen in the example above, “grid\_size” is a single parameter, as is “grid\_multiple”.

If a keyword contains an underscore sign, such as “COLOR\_LAYER” does in the command

```
SET COLOR_LAYER layer_name color_word
```

then the character is to be typed in just like any other. For example:

```
SET COLOR_LAYER BOTTOM BLUE
```

## Spaces

Wherever a space is permissible, any number of spaces can be used.

## Alternative Parameters

The | character means that the parameters are alternatives. For example:

Syntax:

```
SET BEEP ON|OFF
```

Input:

```
SET BEEP ON
```

or

```
SET BEEP OFF
```

The beep, which is triggered by certain actions, is switched on or off.

## Repetition Points

The .. characters means either that the function can be executed multiple times, or that multiple parameters of the same type are allowed. For example:

Syntax:

```
DISPLAY option layer_name..
```

Input:

```
DISPLAY TOP PINS VIAS
```

More than one layer is made visible here.

## Mouse Click

The following sign

-



usually means that at this point in the command an object is to be clicked with the left mouse key. For example:

Syntax:

```
MOVE • •..
```

Input:

```
MOVE ← (or click on the icon)
```

```
Mouse click on the first element to be moved
```

```
Mouse click on the destination
```

```
Mouse click on the second element to be moved  
and so on.
```

You can also see from these examples how the repetition points are to be understood in the context of mouse clicks.

### Entering Coordinates as Text

The program sees every mouse click as a pair of coordinates. If you want to enter the command in text form, the coordinates can be entered via the keyboard in the following form:

```
(x y)
```

where x and y are numbers representing units as selected by the GRID command. The textual input method is necessary in particular for script files.

The coordinates of the current cursor position can be fetched with (@). For example:

```
WINDOW (@);
```

An example of coordinate entry in text form: you want to enter the outline of a circuit board with precise dimensions.

```
GRID MM 1;  
LAYER DIMENSION;  
WIRE 0 (0 0) (160 0) (160 100) (0 100) (0 0);  
GRID LAST;
```

If the commands are being read from a script file, each one must be closed with a semicolon. In the above cases the semicolons can be omitted if the commands are being entered via the keyboard and each is being closed with the enter key.

A 1 mm grid size is first selected. A transfer is then made to the dimension layer. In the following WIRE command, a line width of 0 is first selected and a rectangle is drawn with the aid of the four coordinates. The last command returns the grid size to whatever had been previously selected, since a grid based on inches is usually used for the design of circuit boards.

## Grids and the Current Unit

---

EAGLE performs its internal calculations using a basic grid size of 1/10 000 mm (0.1 micron). Any multiple of that can be set as the working grid.

Microns, mils, inches and mm can be used for the unit. The current unit as set with the GRID command applies to all values.

The preset 0.1 inch grid size should always be used for circuit diagrams.

When starting the design of circuit boards or libraries it pays to give prior thought to the question of which grid size (or sizes) will be used as a basis. For example, it is only the origin of a package that will be pulled onto the board's placement grid. All other objects constituting the package (such as pads) are placed relative to that point on the board, just as it was defined in the library.

The basic rule for boards is: always make the grid as big as possible and as small as necessary.

Various grid sizes can be preset in the EAGLE.SCR file for different types of operation (see page 46 ).

---

## Names and Automatic Naming

---

### Length

Device names may have a length of up to ten characters, layer names nine characters, and all other names eight characters.

### Forbidden Characters

Spaces and full stops are not allowed in any names. Inverted commas and semicolons are best avoided.

### Automatic Naming

If a name is given together with one of the commands PIN, PAD, SMD, NET, BUS or ADD, then other names will be derived from it as long as the command is still active. The examples illustrate how automatic naming functions:

```
ADD DIL14 'U1' . . . .
```

fetches three DIL14 elements to the board and names them U1, U2 and U3 (• corresponds to a mouse click).

```
PAD OCT '1' . . . .
```

places four octagonal pads with the names 1, 2, 3, and 4.

If the name consists of only one character from A...Z, then the following objects receive the following letters of the alphabet as names:

```
ADD NAND 'A' . . . .
```

fetches four NAND gates with the names A, B, C and D. If the generated name reaches Z, then names with the default prefix will again be generated (e.g. G\$1).

## Script Files and Data Import

---

The Script command makes a universal tool available to the EAGLE user for data import.

Since every EAGLE operation can be carried out with the aid of text commands, you can import all types of data with the aid of a script file.

The prerequisite for the development of your own script files is that you understand the EAGLE command language. Please therefore refer to the appropriate section in this handbook. You will find the precise functioning and the syntax of the individual commands in the EAGLE *Help* pages.

The file DRCSET.SCR, which is included in the EAGLE package, provides an example of a useful script file. Adjust it to suit your own needs, so that you can set particular design rules with a mouse click.

The file EURO.SCR, which draws the outline of a Eurocard and gives it limiting corners, is another example.

If a netlist is to be imported into a board design which already contains the appropriate components, then a script file of the following form is necessary:

```
SIGNAL GND IC1 7 IC2 7 J4 22;  
SIGNAL VCC IC1 14 IC2 14 J4 1;
```

You can get an impression of the power of importing if you export a library with the EXPORT command (see page 45) and then read the resulting script file back into an empty library.

## Data Export

---

EAGLE offers two different ways to export data:

- using the EXPORT command,
- with user language programs (ULPs).

The user language is more flexible, but calls for the creation of a suitable program. Further details are to be found in the *EAGLE User Language* section.

The EXPORT command has the following modes:

### **EXPORT DIRECTORY**

Outputs a list of the contents of the currently loaded library.

### **EXPORT NETLIST**

Outputs a netlist for the currently loaded schematic or board in an EAGLE-specific format. Use a ULP to obtain other formats.

### **EXPORT NETSCRIPT**

Outputs a netlist of the currently loaded schematic in the form of a script file.

### **EXPORT PARTLIST**

Outputs a component list for the schematic or board.

### **EXPORT PINLIST**

Outputs a pin/pad list for the schematic or board, listing the connected nets.

### **EXPORT SCRIPT**

Outputs the currently loaded library in the form of a script file.

## Individual Configuration of EAGLE

---

### Project Files

EAGLE keeps a note of the options selected and of the windows which were open at the time the program was closed in a project file (\*.EPF), assuming that in the control panel, under *Options/Backup*, the *Automatically save project file* option has been activated. This state is restored when the program is next started.

You can save a particular working environment under a specific name with *File/Save project*. If you load this file again with *Open/Project* you will recreate that environment.

### Configuration Commands

ASSIGN: Display and change function key assignments.  
SET: Alter system parameters.  
CHANGE: Set object properties.  
GRID: Set grid and current unit.

### EAGLE.SCR

The EAGLE.SCR script file is automatically executed when EAGLE starts. EAGLE first looks for it in the directory which is entered in the first text-field of the *Options/Directories* dialog. If no file of that name is found there, EAGLE looks in the program directory.

You can put into this file all the commands that you want to have executed when an editor window (other than the text editor) is opened.

The *SCH*, *BRD* and *LBR* labels indicate those segments within the file which are only to be executed if the schematic, layout or library editor window is opened.

The *DEV*, *SYM* and *PAC* labels indicate those segments within the file which are only to be executed if the device, symbol or package editor mode is activated.

---

## The EAGLE User Language

---

EAGLE contains an interpreter for a C-like user language which permits manipulation of any EAGLE files. User language programs are written with a text editor which does not add any special control codes, and they are executed with the RUN command. A precise description of the language is found in the EAGLE help pages.

The program examples supplied (\*.ULP) can give you an impression of the power of the user language.

A frequent application is for the output of data (netlists etc.) in specific formats.

In order to manipulate EAGLE files, we often create a script file containing the corresponding commands for the data manipulation.

The following example is executed when a board is loaded. It generates the *SNAP.SCR* script file. It contains the necessary commands to pull all the components on the board to the nearest 50-mil grid point. The board must not be altered between the execution of the ULP and the execution of the script file.

```
/*
 * This EAGLE user language program produces a script
 * file that can be used to snap the elements of the
 * current board to a given grid.
 */

real GridDist = 50.0; // in Mil, can be changed to other values

real snap(int n) // returns next grid point
{
    return round(u2mil(n) / GridDist) * GridDist;
}

output("SNAP.SCR") {
    board(B) {
        printf("GRID MIL %f;\n", GridDist);
        B.elements(E) {
            printf("MOVE %s (%f %f);\n",
                E.name, snap(E.x), snap(E.y));
        }
        printf("GRID LAST;\n");
    }
}
```

## Forward&Back Annotation

---

A schematic file and the associated board file are logically linked by automatic Forward&Back annotation. This ensures that the schematic and the board are always consistent. A precise description of the technical background is found in the EAGLE help pages.

As a user, it is not necessary to worry about the details of the mechanism. All that is necessary is to avoid working on a schematic if the associated board file is closed, and vice versa.

If, nevertheless, you have worked separately on the board and the schematic, the electrical rule check (ERC) will check the consistency of the files as they are loaded.

Certain operations, such as the placing of a new net, are only permitted in the schematic. The layout editor will not allow these operations, and issues an appropriate message.



## From Schematic to Finished Board

---

### Create Schematic

---

It is assumed that libraries containing the required components are present. The creation of libraries is described elsewhere.

See also the sections which describe the schematic and layout editor windows.

#### Open Schematic

Open a new or existing schematic via, e.g., the control panel's *File*-menu.

Create more schematic sheets if needed.

#### Load Drawing Frame

If a frame is wanted, load the *FRAMES.LBR* library (USE) and position a frame (ADD).

#### Place Circuit Symbols (Gates)

Select library (USE) and place "gates" (ADD), give them names and values (NAME, VALUE), perhaps reposition them (MOVE) or remove them (DELETE).

#### Draw Busses (BUS)

Busses are given names which govern the signals that can be led out from them. They do not create an electrical connection.

#### Draw Nets (NET)

The connection between the pins of various elements is defined with the NET command. Nets are displayed in the net layer. Connections between nets which cross are made with the JUNCTION command.

#### Check and Correct Schematic.

Carry out an electrical rule check (ERC) and use the messages to correct any errors. You may want to output a netlist or pinlist (EXPORT). Use the SHOW command to follow nets on the screen.

You may also want to print the schematic with the PRINT command or with the CAM processor.

## Points to Note for the Schematic Editor

---

### **Grid**

The grid should under all normal circumstances be 0.1 inch, i.e. 100 mil.

### **Nets**

Nets must be drawn with the NET command and not with the WIRE command.

Nets having the same name are connected, even if they are located on different sheets.

### **False Connections**

If the MOVE command is used to move a net over another net, or over a pin, no electrical connection is created.

To check this, you can click the net with the SHOW command. All the connected pins and nets will light up. If a gate is moved, the nets connected to it will be dragged along.

### **Superimposed Pins**

If the connection points for pins are placed directly over one another (without an explicit net line), they are connected.

### **Open Pins when MOVEing**

If an element is moved then its open pins will be connected to any nets or other pins which may be present at its new location. Use UNDO if this has happened unintentionally.

### **Gates from the same Component on more than one Sheet**

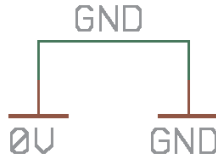
If you want to distribute gates from one component over an number of schematic sheets, you must activate the INVOKE command, and type the name of the element on the command line (see page 14).

### **Power Supply**

Pins defined as having the direction "Pwr" are automatically wired up. This is true, even if the associated power gate has not explicitly been fetched into the schematic. For every Pwr-pin there must be at least one pin with the same name but the direction "Sup" (a supply pin). There must be one on every sheet. These Sup-pins are fetched into the schematic in the form of power supply symbols, and are defined as devices in a library (see SUPPLY1.LBR and SUPPLY2.LBR). These devices do not have a package, since they do not represent components.

If nets are connected to a component's Pwr-pins, then these pins are not automatically wired. They are joined instead to the connected net.

Various supply voltages, such as 0 V or GND, which are to have the same potential (GND, let's say), can be connected by adding the corresponding supply symbols and connecting them with a net. This net is then given the name of that potential (e.g. GND).



If a supply pin (a pin with direction *Sup*) is placed on a net (ADD, MOVE) then the net segment will receive the name of the supply pin.

### Considerations Prior to Creating a Board

---

#### Checking the Component Libraries

The EAGLE component libraries are developed by practicing engineers, and correspond closely to today's standards. The variety of components available is however so wide that it is impossible to supply libraries which, without modification, are suitable for every user.

There are even different packages which are supplied by various manufacturers using the same identification! Manufacturers recommend very different sizes for SMD pads, and these depend again on the soldering procedure being applied.

In short: You can not get away without checking the components, in particular the package definitions, being used when laying out.

#### Agreement with the Board Manufacturer

If you plan to have your pcb films made with a Gerber photoplotter, then, if you have not already done so, now is the time to inquire at your board maker whether he stipulates any particular values for the following parameters:

- track width
- shape of solder lands
- diameter of solder lands
- dimensions of SMD pads
- text size and thickness
- drill hole diameters
- distance between different potentials.

You will save yourself time and money if you take these stipulations into account in good time.

You will find more details on this in the section on the output of manufacturing data.

## Create Board

---

After you have created the schematic, click the switch-to-board icon.

An empty board is generated, next to which the components are placed, joined together with airwires. Supply pins are connected by those signals which correspond to their name, unless another net is explicitly joined to them.

The board is linked to the schematic by the Forward&Back annotation. This ensures that the two are in agreement, as long as neither of the files is edited at a time when the other is not loaded.

### **Without the Schematic Editor**

If you work without a schematic editor, you must generate a new board file, place the packages with the ADD command and define the connections with the SIGNAL command. You must, additionally, draw the board outline with the WIRE command in the dimensions layer.

The remaining procedures are identical for users with and without the schematic editor.

### **Specify the Board Outline and the Placement**

In some cases you may wish to alter the size or shape of the empty board (MOVE, SPLIT). Put the elements into their desired positions (MOVE), and check whether or not the placement is favorable (RATSNEST).

### **Define Forbidden Areas**

If desired, areas forbidden to the autorouter can be drawn as rectangles, polygons or circles in the tRestrict, bRestrict and vRestrict layers (see RECT, POLYGON, CIRCLE).

### **Routing**

Airwires are changed into tracks with the ROUTE command. This task can also be left to the autorouter. If there is no longer any room for an individual signal to be routed, other tracks are pushed aside (MOVE, SPLIT, CHANGE).

### **Check and Correct Schematic**

Carry out design rule check (DRC) and correct any errors. You may want to output a netlist, component list or pin list (EXPORT).

### **Create Manufacturing Data**

The CAM processor is used to create the manufacturing data.

## Multilayer Boards

---

Multilayer boards can be developed with EAGLE. These use not only the Top and Bottom layers, but also one or more inner layers (Route2 to Route15). You include these layers when routing.

### Signal Layer

---

You use the ROUTE command as before to place tracks in those inner layers which are provided for signals. Eagle will itself ensure that the tracks are connected by way of plated-through holes to the appropriate signals in the outer layers. The CAM processor will output the inner signal layers if the inner layer concerned is activated together with the *Pads* and *Vias* layers.

### Power Supply Layer with One Signal

---

A power supply layer with one signal is implemented by renaming one of the Route2...15 layers in such a way that the new name consists of the corresponding signal name prefixed by a \$ sign. If, for example, the signal called GND is to be realized as a power supply layer, then a layer is specified as having the name \$GND. The command corresponding to this example would be:

```
LAYER 2 $GND
```

This specifies that layer number 2 (previously known as Route2) is henceforth known as \$GND, and that it will be treated as a power supply layer. The preferred direction for the autorouter is to be set to 0 for power supply layers. This will cause the autorouter not to use this layer.

Pads are connected to power supply layers with what are known as thermal symbols, or are isolated with annulus symbols. Thermal symbols usually just have four thin bridges as a conductive connection to the through-plated hole. They are used because the high thermal conduction of a continuous copper plane would result in the pad being no longer solderable.

Supply layers are inverse plotted. You should draw a wire in such layers around the edge of the board to keep it free from copper. This will prevent the possibility of shorts between neighboring (power supply) layers. If you use the autorouter however, this wire must not be drawn until after the routing. As described above, power supply layers are not available to the autorouter, but this in fact only applies to layers which do not contain any signals (wires).

The autorouter includes the inner layers, and so delivers the full set of patterns for multilayer boards. It connects SMD pads to inner layers with vias.

### **Ground Areas and Supply Layers with More than One Signal**

Areas of the board can be filled with a particular signal (e.g. ground) using the POLYGON command. The associated pads are then automatically connected using thermal symbols. A specified minimum separation is maintained from all other potentials.

In this way you can also generate layers filled with different signals in several areas. Possible application: a layer with several power supply signals.

Please note that such cases are not the kind of layer that can be defined with "\$name..". For this reason such layers are not inverse plotted. With Gerber plotters this can lead to huge plot times, since large areas have to be filled. So don't set too low a stroke thickness for the polygons! This is of no significance when the output will be on a PostScript recorder.

As soon as there is a polygon in a signal layer, the autorouter can no longer place any vias inside the polygon area. For this reason it is helpful only to define polygons after other signals have been routed. If you want to use the autorouter even though power supply polygons have already been defined, please follow the procedure described in the autorouter chapter under *Polygons as power supply layers*.

## Component Design Explained through Examples

---

When developing circuits with EAGLE, components are fetched from libraries and placed into the schematic or, if the schematic editor is not being used, into the layout. All the component information is then saved in the schematic or board file. The libraries are no longer needed for continued work with the data. An alteration in a library has no effect on a schematic or board.

The most important procedures for the design of components (devices) is explained starting from page 30. Some practical examples follow, from which the effective application of the relevant commands and parameters will be seen.

First we will take the example of a resistor and go through the whole process of designing a simple component. After that we shall discuss the special features which have to be taken into account with more complicated components.



## Resistor Package



Select the package editing mode, and enter the package name *R-10* in the *New* field. Answer the question “Create new package ‘R-10’?” with *Yes*. Later when creating a new symbol and a new device you will again have to answer the corresponding questions with *Yes*.



Use the GRID command to set an appropriate grid size for the pad placement. 0.05 inch (i.e. 50 mil) is usual for standard components with lead wires.



For a resistor with lead-wires, select PAD, and set the pad shape, the diameter and the drill diameter in the parameter toolbar. Then place two pads at the desired distance. The origin of the drawing will later be the identifying point with which a component is selected. For this reason it should be somewhere near the center of the device.



For an SMD resistor, select SMD, and set the pad dimensions in the parameter toolbar. You can either select one of the offered values, or directly type the length and breadth into the entry field.

Select *Top* as the layer, even if the component will later be placed on the underside of the board. SMD components are located on the other side of a board using the MIRROR command. This moves the elements in all the t..-layers into the corresponding b..-layers.

Place the two SMD pads (which in EAGLE are just called SMDs) at the desired distance. You can rotate the SMD with the right mouse button.



You can now enter the names, such as ‘1’ and ‘2’ , for the pads or SMDs using the NAME command.

A different procedure is however recommended for components with many sequentially numbered pads:

Select the PAD command, type in the name of the first pad, e.g. “1” (the inverted commas must also be entered), then place the pads in sequence. See also page 43.

### Silkscreen and Documentation Print

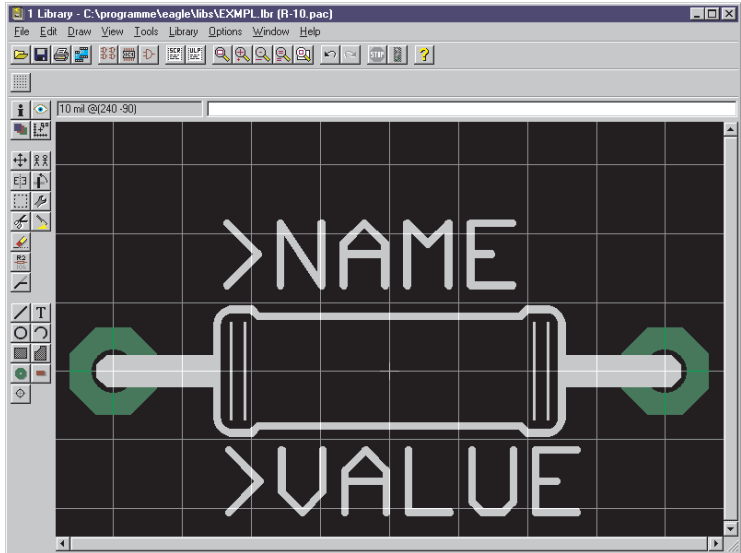


Now use the WIRE command to draw the silkscreen symbol in the tPlace layer. This layer contains what will be printed on the board. It is up to you how much detail you give to the symbol. Set a finer grid size if it helps.

Take the information provided in LIBRARY.DOC as a guideline for the design of components.

You can also use the ARC, CIRCLE, RECT and POLYGON commands to draw the symbols for the silkscreen.

The tDocu layer is not used to print onto the board itself, but is a supplement to the graphical presentation which might be used for printed documentation. Care must be taken in the tPlace layer not to cover any areas that are to be soldered, but in the tDocu layer a more realistic appearance can be given which is not subject to this limitation. In the example of the resistor, the symbol, the symbol can be drawn in the tPlace layer, but the wires, which go over the pads, are drawn in the tDocu layer.



## Labeling



With the TEXT command you place the texts >NAME and >VALUE in those places where in the board the actual name and the actual value are to appear. 0.07 inch for the text height (size) and 10% for the ratio (relationship of stroke width to text height) are recommended.

SMASH and MOVE can be used later to change the position of this text relative to the package symbol on the board.

The value for ICs is the component type (e.g. 7400).



The CHANGE command can be used at a later stage to alter object properties such as the stroke thickness, text height, or the layer in which the object is located.

If you want to change the properties of several objects at one go, define a group with the GROUP command, click the CHANGE

command, select the parameter and the value, and click on the drawing surface with the right mouse button.

Example: Use GROUP to define a group that contains both pads, then select CHANGE and SHAPE/SQUARE. Click on the drawing surface with the right mouse button. The shape of both pads changes.

## Resistor Symbol

---



Select the symbol editing mode, and enter the symbol name *R* in the *New* field. This name only has a meaning internal to the program, and does not appear in the schematic.

Now check that 0.1 inch is set as the grid size. The pins in the symbol must be placed on this grid, since this is what EAGLE expects.



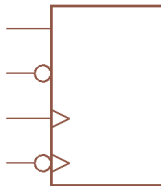
Select the PIN command. You can now set the properties of these pins in the parameter toolbar, before placing them with the left mouse button. All these properties can be changed at a later stage with the CHANGE command. Groups can again be defined (GROUP) whose properties can then be altered with CHANGE and the right mouse button. See also page 58.

### Orientation

Set the direction of the pins (*Orientation* parameter) using the four left-hand icons in the parameter toolbar or, more conveniently, by rotating with the right mouse button.

### Function

The Function parameter is set with the next four icons on the parameter toolbar. This specifies whether the symbol is to be shown with an “inversion circle” (Dot), with a clock symbol (Clk), with both (DotClk) or simply as a stroke (None). The diagram illustrates the four representations on one package.



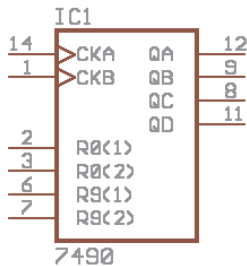
### Length

The next four icons on the parameter toolbar permit setting of the pin length (0, 0.1 inch, 0.2 inch, 0.3 inch). The 0 setting is used if no pin-line is to be visible, or if, as in the resistor symbol, a pin shorter than 0.1 inch is desired. In that case the pin is to be drawn with the WIRE command as a line in the *symbols* layer.

### Visible

The next four icons in the parameter toolbar specify whether the pins are to be labeled with pin names, pad names, both or neither. The diagram illustrates an example in which pin names are shown inside

and pad-names outside. The placement of the label in relationship to the pin is fixed. The text height is also fixed (60 mil).



### Direction

The *Direction* parameter specifies the logical direction of the signal flow:

NC	Not connected
In	Input
Out	Output
I/O	Input/output
OC	Open Collector or Open Drain
Hiz	High impedance output
Pas	Passive (resistors, etc.)
Pwr	Power pin (power supply input)
Sup	Power supply output for ground and power supply symbols

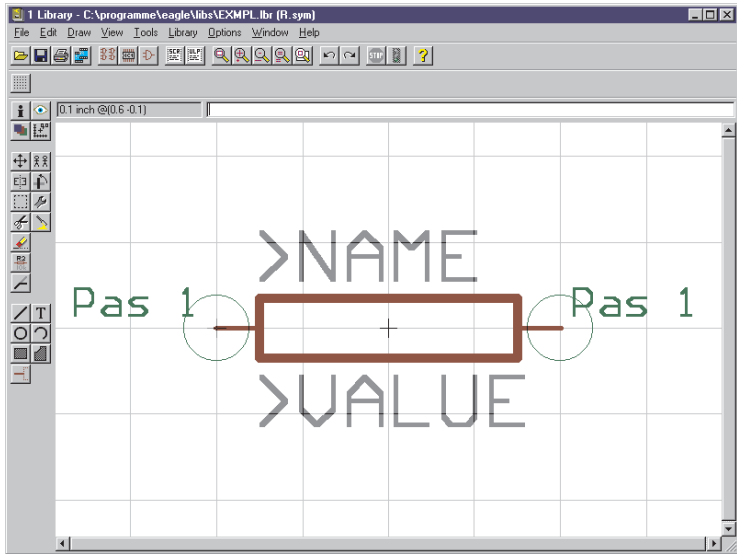
The electrical rule check is based on these parameters. It will flag, for example, if two pins with the *Out* direction are connected. Please note that the ERC can only offer warnings. You must interpret the messages yourself.

The *Pwr* and *Sup* directions are used for the automatic connection of supply voltages (see [page 67](#)).

### Swaplevel

The *swaplevel* is a number between 0 and 255. The number 0 means that the pin cannot be exchanged for another pin in the same gate. Any number bigger than 0 means that pins can be exchanged for other pins which have the same *swaplevel* and are defined within the same symbol. The pins can be swapped in the schematic or in the board with the `PINSWAP` command.

The two pins of a resistor can have the same swaplevel (e.g. 1), since they are interchangeable. The connections to a diode may not be swapped, and are therefore both given the swaplevel 0.



If the Pins layer is being displayed, the connection points on nets are shown with green circles. The direction and swaplevel parameters moreover (here *Pas* and *1*) are displayed in this layer.

### Pin Names

The `NAME` command allows you to name pins after they have been placed. The automatic name allocation, as described on pages 43 and 57 also operates.

### Schematic Symbol

The schematic symbol is drawn in the symbols layer using `WIRE` and the other drawing commands. Place the texts `>NAME` and `>VALUE` in the *Names* and *Values* layers. Place them where the name and value of the component are to appear in the schematic.

## Resistor Device

---



Create the new device *R-10* with this icon. When you later use the ADD command to fetch the component into the schematic, you will select it by using this name. It is only a coincidence that in this case the name of the package and the name of the device are the same.



With the PACKAGE command you select the (previously defined) package for the device. This package can later be replaced by another in the board (REPLACE). In the case of the resistor, select R-10.



The PREFIX command is used to specify a prefix for a name. The name itself will initially be automatically allocated in the schematic. For a resistor this would, naturally enough, be *R*. The resistors will then be identified as R1, R2, R3 etc.. The names can be altered at any time with the NAME command.



You can specify with the VALUE command whether the device's value can be altered in the schematic or in the board. This is absolutely necessary for resistors (*On*). For other components it can be useful to set the value to *Off*, as might well apply to power supply symbols (see 67). In general, the value *On* (default) is to be recommended.

When a device is fetched into a schematic, it initially receives the device name as its value.



The previously defined resistor symbol is fetched into the device with the ADD command. If a device consists of several schematic symbols which can be placed independently of one another in the circuit (in EAGLE these are known as *gates*) then each gate is to be individually brought into the schematic with the ADD command.

Set an addlevel of *Next* and a swaplevel of *0* in the parameter toolbar, and then place the gate near the origin. (There are further explanations about addlevel in the following sections.)

The swaplevel of a gate behaves very much like the swaplevel of a pin. The value of *0* means that the gate cannot be exchanged for another gate in the device. A value greater than *0* means that the gate can be swapped within the schematic for another gate in the same device and having the same swaplevel. The command required for this is GATESWAP.

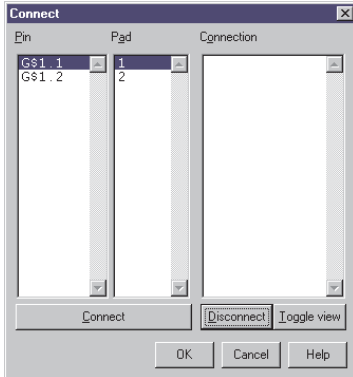


You can change the name of the gate or gates with the NAME command. The name is unimportant for a device with only one gate, since it does not appear in the schematic. For devices with more than one gate, the particular gate-name is added to the name of the component.

Example: The gates are called A, B, C and D, and the name of the component in the schematic is IC1, so the names which appear are IC1A, IC1B, IC1C and IC1D.



With the CONNECT command you specify which pins are taken to which package pads.



In the present example the resistor gate was automatically called G\$1 so that the pins 1 and 2 of this gate appear in the *Pin* column. The two connections for the package are listed in the *Pad* column. Mark a pin and the associated pad, and click *Connect*. If you want to undo a connection, mark it under *Connection*, and click on *Disconnect*. The *Toggle view* button changes the sorting sequence of the fully defined connection.

The resistor is now defined, and can be fetched into the schematic.



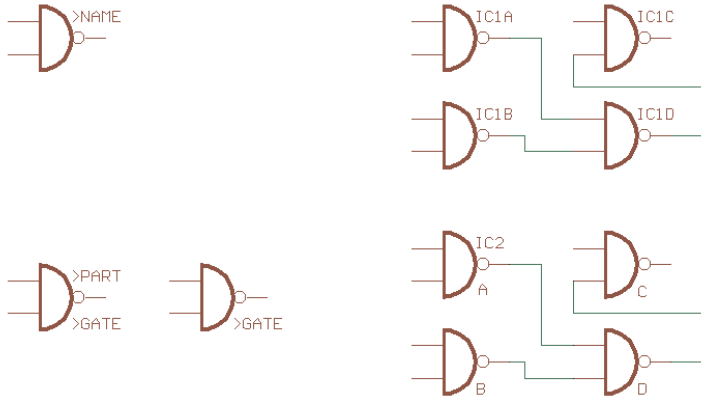
You can alter the *addlevel* and *swaplevel* parameters retrospectively with the CHANGE command.



## Labeling of Schematic Symbols

The two text variables  $>NAME$  and  $>VALUE$  are available for labeling packages and schematic symbols. Their use has already been illustrated. There are two further methods that can be used in the schematic:  $>PART$  and  $>GATE$ .

The following diagram illustrates their use in contrast to  $>NAME$ .



In the first case all the symbols are labeled with  $>NAME$ . In the second case, the symbol of the first gate is labeled with  $>PART$ , the other three with  $>GATE$ .

SMASH cannot separate  $>PART$  and  $>GATE$  from their symbols for individual movement.

### **Mirrored Layer**

---

SMD components can be placed on the top or bottom layers of a board. For this reason EAGLE makes a set of pre-defined layers available which are related to the top side (Top, tPlace, tOrigins, tNames, tValues etc.) and another set of layers related to the bottom side (Bottom, bPlace etc.).

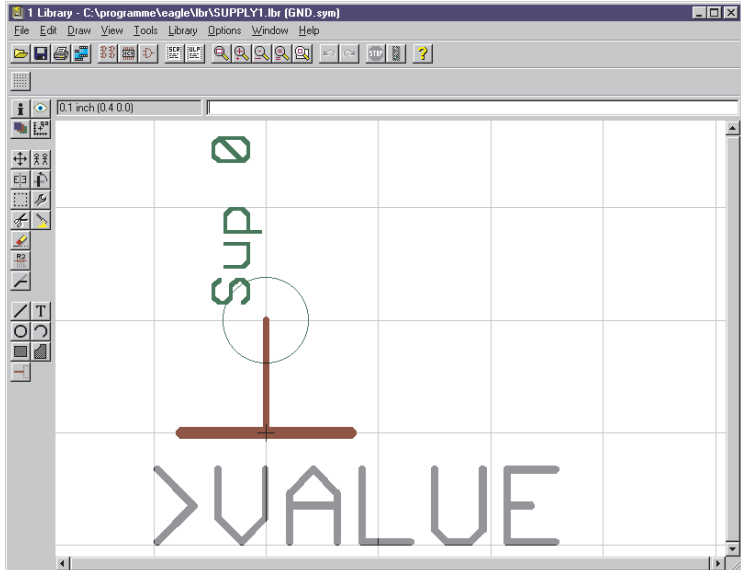
SMD components are always defined in the layers associated with the top. In the board, a component of this sort is moved to the opposite side with the MIRROR command. This causes objects in the Top layer to be reflected into the Bottom layer, while all the elements in the t.. layers are reflected into the corresponding b.. layers.

Films and drawings related to the bottom side are printed with reflection (*Mirror* option).

## Supply Symbols

Supply symbols, such as might be used in the schematic for ground or VCC, are defined as devices without a package. They are needed for the automatic wiring of supply nets (see page 50).

The diagram shows a GND symbol as it is defined in one of the supplied EAGLE libraries.



### Pin and Device Names should Agree

The pin is defined with direction *Sup* and has the name *GND*. This specifies that the device contained by this symbol is responsible for the automatic wiring of the GND signal. The text variable for the value (>VALUE) is chosen for the labeling.

The device also receives the name *GND*. Thus the label GND appears in the schematic, since by default EAGLE uses the device name for the value.

It is very important that the labeling reproduces the pin names, since otherwise the user will not know which signal is automatically connected.

The pin parameter visible is therefore set to *Off*, since otherwise the placing, orientation and size of the pin name would no longer be freely selectable.

Directly labeling with the text *GND* would also have been possible here. With the chosen solution however, the symbol can be used in various devices (such as for *DGND* etc.).

### **Device**

The associated device is defined with addlevel *Next*. If you set the value to *On*, you can be sure that the labeling will not be changed by accident. On the other hand, they are more flexible if the value is *Off*. You can change the labeling if you have, so to speak, a second ground potential. You must however explicitly connect the net for the second ground.

## Component Power Supply Pins

In the symbol definition, component's power supply pins are to be defined with direction *Pwr*. They are automatically connected (even without an explicit net line) if there is a supply symbol on the same sheet that has a pin with the same name.

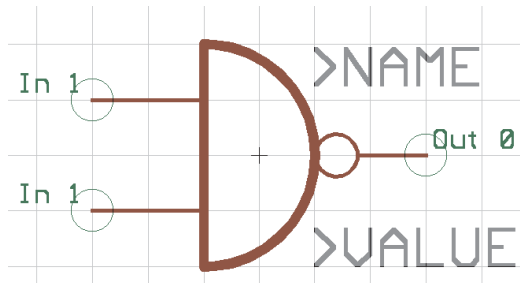
The power supply pins in the EAGLE libraries are generally identified as GND and VCC (although they do have other names in some cases). This means that the automatic connection only functions with the corresponding supply symbols. If you use the supply symbol with, say, the +5V label instead of VCC, then you will have to connect the supply pins to the supply net explicitly (or else use devices with +5V as the name for the supply pins).

### Invisible Supply Pins

We do not want as a rule to draw the supply connections for logic components or operational amplifiers in the schematic.

In such a case a specific symbol containing the supply connections is defined. This can be demonstrated with the example of a 7400 TTL component.

You first define a NAND gate with the name *7400*, and the following properties:



The two input pins are called *I0* and *I1* and are defined as having Direction *In*, swaplevel *1*, visible *Pin* and function *None*.

The output pin is called *O* and is defined with direction *Out*, swaplevel *0*, visible *Pin* and function *Dot*.

Now define the supply gate with the name *PWRN*, and the following properties:



The two pins are called *GND* and *VCC*. They are defined with direction *Pwr*, swaplevel *0*, function *None* and visible *Pad*.

Now create the *7400* device.

Specify the package with *PACKAGE* (which must already be present in the library) and use *PREFIX* to specify the name prefix as *IC*.

Use the *ADD* command to place the *7400* symbol four times, with addlevel being set to *Next* and swaplevel to *1*. Then label the gates as *A*, *B*, *C* and *D* with the *NAME* command. The addlevel of *Next* means that as these gates are placed into the schematic, they will be used in that sequence, i.e., the sequence in which they were fetched into the device.

Then place the *PWRN* symbol once, using addlevel *Request* and swaplevel *0*. Name this gate *P*.

Addlevel *Request* specifies two things:

- The supply gate will only be fetched into the schematic if requested, i.e. with the *INVOKE* command. The *ADD* command will only be able to place *NAND* gates.
- The supply gate will not be included when names are allocated to the schematic. Whereas an *IC* with two *Next*-gates appears in the schematic as something like *IC1A* and *IC1B*, an *IC* with one *Next*-gate and one *Request*-gate will only be identified as *IC1*.

## **Pins with the same Names**

---

If you want to define components having several pins of the same name, then proceed as follows. Let us suppose that three pins are all to be called GND. During the symbol definition the pins are given the names GND@1, GND@2 and GND@3. Only the characters in front of the “@” are visible in the schematic, and the pins are treated there as if they were all called GND. However these pins are not necessarily internally connected.

### More about the Addlevel Parameter

---

The addlevel of the gates that have been fetched determines the manner in which these gates are fetched into the schematic, and under what conditions it can be deleted from the schematic.

A summary of all the possibilities follows. Various practical examples are described in the next section.

**Next:** For all gates that should be fetched in sequence (e.g. the NAND gates of a 7400). This is also a good option for devices with a single gate. The ADD command first takes unused next-gates from components which exist on the current sheet before “opening” a new component.

**Must:** For gates which must be present if some other gate from the component is present. Typical example: the coil of a relay. Must-gates cannot be deleted before all the other gates from that component have been deleted.

**Can:** For gates which are only used as required. In a relay the contacts may be defined with addlevel “Can”. In such a case the individual contacts can be specifically fetched with INVOKE, and can later be deleted with DELETE.

**Always:** For gates which as a general rule will be used in the schematic as soon as the component is used at all. Example: contacts from a multi-contact relay, of which a few are occasionally left unused. These contacts can be removed with DELETE, provided that they were defined with addlevel “Always”.

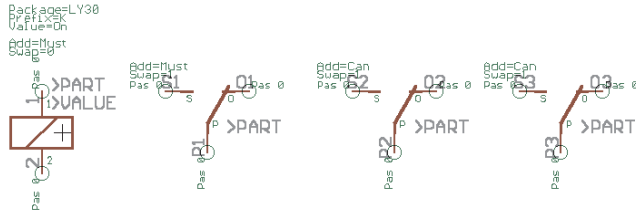
**Request:** Only for components’ supply gates. The difference from “Can” is that they are not counted in name allocation. So a device with a Next-gate and a Request-gate will not be named IC1A and IC1B.



## Relay: Coil and First Contact must be Placed

A relay with three contacts is to be designed, of which typically only the first contact will be used.

Define the coil and one contact as their own symbols. In the device, give the coil and the first contact the addlevel *Must*. All the other contacts are given the addlevel *Can*.

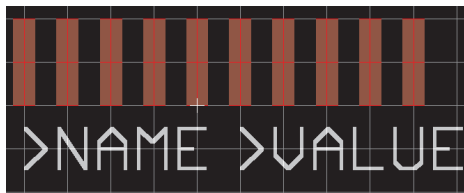


If the relay is fetched into the schematic with the ADD command, the coil and the first contact are placed. If another contact is to be placed, this can be done with the INVOKE command. The coil can not be deleted on its own. It disappears when all the contacts have been deleted (beginning with those defined with *Can*).

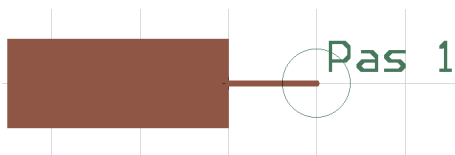
## Plug: Some Connection Surfaces can be Omitted

A pcb plug is to be designed in which normally all the contact areas are present. In some cases it may be desirable for some of the contact areas to be omitted.

Define a package with 10 SMDs as contact areas, giving the SMDs the names 1 to 10.



Now define a symbol representing one contact area. Set visible to *Pad*, so that the names 1 to 10, defined in the package, appear in the schematic.



Then fetch the symbol ten times into a newly created device, setting the addlevel in each case to *Always*, and use the CONNECT command to create the connections between the SMDs and the pins. When you now fetch this device into a schematic, all the connections appear as soon as it is placed. Individual connections can be removed with DELETE.

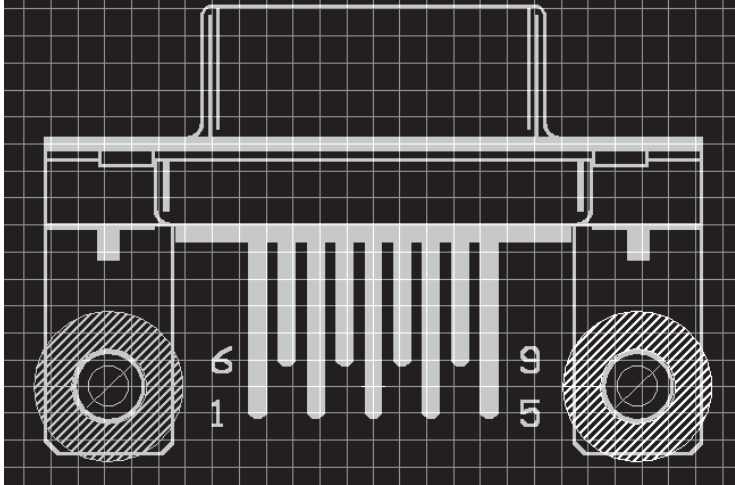
```
Package=C-PAD
Prefix=
Value=0n
Add=Always
Swap=1
1 Ras 0
Add=Always
Swap=1
2 Ras 0
Add=Always
Swap=1
3 Ras 0
Add=Always
Swap=1
4 Ras 0
Add=Always
Swap=1
5 Ras 0
Add=Always
Swap=1
6 Ras 0
Add=Always
Swap=1
7 Ras 0
Add=Always
Swap=1
8 Ras 0
Add=Always
Swap=1
9 Ras 0
Add=Always
Swap=1
10 Ras 0
```

---

## Plug with Fixing Hole and Forbidden Area

---

A plug is to be defined having fixing holes. On the solder side (bottom), the autorouter must avoid bringing tracks closer to the holes than a certain distance.



The drill holes are placed, with the desired diameter, on the package using the `HOLE` command. The drilling diameter can be retrospectively changed with `CHANGE DRILL`.

The forbidden area for the autorouter is defined in the `bRestrict` layer using the `CIRCLE` command. For reasons of representational clarity the circle is shown here with a non-zero *width*. Circles whose width is 0 are filled. In this case it has no effect on the autorouter, since it may not route within the circle in either case.

## Drawing Frames

---

It may be true that drawing frames are not components, but they can be defined as devices with neither packages nor pins. Such devices in EAGLE's FRAMES library contain a symbol consisting merely of a frame of the appropriate size, and a documentation field, which is also defined as a symbol.

In both cases the identification point is located at the bottom left, so that neither the frame nor the documentation field will be unintentionally selected while working within the drawing area.

TITLE: >DRAWING_NAME	
Document Number:	REV:
Date: >LAST_DATE_TIME	Sheet: >SHEET

The text variables >DRAWING\_NAME, >LAST\_DATE\_TIME and >SHEET are contained, as well as some fixed text. The drawing's file name, date and time of the last change appear at these points together with the sheet number in the schematic (e.g., 2/3 = sheet 2 of 3).

In addition, the variable >PLOT\_DATE\_TIME is available. It contains the date and time of the last printout.

All of these text variables can be placed on the schematic, and (with the exception of >SHEET) on the board.

The frame is defined in the device with addlevel *Next*, and the documentation field with addlevel *Must*. This means that the documentation field cannot be deleted as long as the frame is present.

## Preparing the Manufacturing Data

---

Since EAGLE is used more often in Germany than any other layout program, there are many pcb firms there who need only the board file in order to manufacture films or prototypes. You will find links to such firms on our Internet pages.

If however your board maker is not set up to process EAGLE board files directly, you will have to supply him with a set of files. You generate these with the aid of the CAM processor.

You can obtain a parts list by loading your board into the editor and using the RUN command to execute the BOM.ULP user language program (name.BOM will contain the parts list).

You will find more useful ULPs on our Internet pages, such as for the generation of glue masks.

### Which Files does the Board Maker Need?

---

The following list contains the files which should be generated for the four-layer board DEMO.BRD. Layer 2 is the normal inner layer with various signals. Layer 3 is a supply layer for the GND signal, which the user has renamed to \$GND in accordance with EAGLE's naming convention for supply layers. The filenames are suggestions.

You generate these files with the CAM processor. You will find further details in the following sections.

The options given in the list are recommendations, which in some cases can be changed.

#### **Drill Information**

For the drilling information, the drill format (e.g. SM1000 or Excellon) should be selected as the device.

If through-plated holes are to be drilled separately from non through-plated holes, two files must be generated, one with the drills layer activated and one with the holes layer.

The *Rotate* option, given in the list, is useful if the board appears in "landscape" format in the layout editor.

## Files Generated with the CAM Processor

File	Active layers	Comments / recommended options
<b>DEMO.CMP</b>	Top Via Pad	Component side. Options: pos. coord., optimize, fill pads.
<b>DEMO.LY2</b>	Route2 Via Pad	Multilayer inner layer. Options: pos. coord., optimize, fill pads.
<b>DEMO.LY4</b>	\$GND	Multilayer supply layer. Is automatically output inverse. Options: pos. coord., optimize.
<b>DEMO.SOL</b>	Bottom Vias Pads	Solder side. Options: mirror, pos. coord., optimize, fill pads.
<b>DEMO.PLC</b>	tPlace Dimension tName	Component side silkscreen. Options: pos. coord., optimize.
<b>DEMO.PLS</b>	bPlace Dimension bName	Solder side silkscreen. Only needed if there are components on the underside Options: mirror, pos. coord., optimize.
<b>DEMO.DCC</b>	tPlace Dimension tName tValue tDocu	Component side documentation print. Visually effective display for your own documentation. Names and values, to taste.
<b>DEMO.DCS</b>	bPlace Dimension bName bValue bDocu	Solder side documentation print. See DEMO.DCC. Only needed if there are components on the underside Options: mirror.
<b>DEMO.STC</b>	tStop	Component side solder mask. Options: pos. coord., optimize.
<b>DEMO.STS</b>	bStop	Solder side solder mask. Options: mirror, pos. coord., optimize.
<b>DEMO.DRP</b>	Drills Holes Dimension	Print with the positions of the drill holes.
<b>DEMO.DRD</b>	Drills Holes	Drilling data for NC drilling machines. Options: mirror, rotate, pos. coord., optimize.

**Additional Information for the Board Manufacturer**

You must also supply the service company with the configuration files for apertures (*name.whl*) and drills (*name.drl*). You can also include the board file *name.brd*. This can help to avoid problems with what can become time-consuming questions. In general, alignment marks (which are for example defined in the Reference layer) or an information field (in, for example, the Documentation layer) can be included.

A text file, such as DEMO.DOC, should contain instructions for the board manufacturer.

**Rules that Save Time and Money**

- Each layer should without fail be uniquely identified (e.g. CS for Component Side).
- For cost reasons you should, if at all possible, avoid track that narrows to below 0.2 mm.
- Only angles should be drawn at the corners to delimit the board. Closed borders can lead to manufacturing difficulties.
- Always maintain a 2 mm margin free from copper at the edge of the board. This can be achieved in the supply layers of multilayer boards, which are inverse plotted, by drawing a wire around the edge of the board.

## Set Output Parameters

---

This section describes the setting of the parameters for the output of a drawing or a file, which will then be started with the *Process*-button. The parameters for a section, as described below, are set in the same way.

Load the schematic or board file from the CAM processor's *File/Open* menu, and set the following parameters:

- Select the driver for the desired output device in the *Device* combo-box.
- Enter the output interface or file in the *Output* field.  
If, under *Output*, *Wheel* or *Rack*, you only give the dot and the file extension, the entry will be expanded with the name of the schematic or layout file that is loaded.  
Example: “.CMP” is expanded to “boardname.CMP”.  
See also the section on *Names of the output files*.
- Select the output layers by clicking the appropriate checkboxes.
- Set device-specific parameters (aperture wheel etc.).
- If a board is loaded, set the board parameters (annulus, thermal, mask data).
- If the loaded schematic has more than one sheet, select the sheet.
- Set flag options (mirror, etc.).

### Names of the Output Files

The extensions for the output files given in the table are only suggestions. If you are defining a job for Gerber data, which generates all the files with these extensions, the information file (\*.gpi) will be overwritten with each succeeding section. If you want to supply information files to your photoplotting service, proceed as follows:

For output, enter either “*name.xx#*” or “*.xx#*”.

Here, *name* stands for any name, and *xx* stands for any characters (that are allowed in file names).

The output file generated will then be called *name.xxX* or *boardname.xxX*.

As an information file *name.xxI* or *boardname.xxI* will be generated.

Concrete example:

The board “MYBOARD.BRD” is loaded. “.CP#” is entered into the output field. The output file is called “MYBOARD.CPX”, and the information file is called “MYBOARD.CPI”.

Please ensure when defining a job that the extensions for each section are unique and therefore distinguishable.



## Aperture Configuration File with the Same Units

---

When automatically generated with the GERBERAUTO driver, a single aperture table can contain both inch and mm values. If your pcb manufacturer insists on consistent units, you can achieve this by altering the GERBER or GERBERAUTO drivers.

In order to do this use a text editor (one that does not introduce any control codes) to edit the EAGLE.DEF file. Look for the line

```
[GERBER]
```

or

```
[GERBERAUTO]
```

and at the end of that section add the lines

Units = or Decimals =.

Examples:

```
Units =    inch  
Decimals = 3
```

To avoid problems arising from rounding errors during the conversion, a tolerance of between +0.01 and -0.01 should be allowed.

## Automating the Output with CAM Processor Jobs

---

The CAM processor provides a job mechanism, with the aid of which the whole process of generating the output files for a board can be automated.

### Define Job

A job consists of one or more sections. A section is a group of settings, as described above, which defines the output of *one* drawing or file. In this way you can use a job to generate all the drawings and files that are necessary for a project.

You define a job as follows:

- Click on the *Add* button and enter a name for the *section* which is to be defined.
- Make a complete set of parameter settings.
- In the *Prompt* field enter a message which is to appear on the screen before execution (such as *Change paper*), should you so wish.
- Define further sections in the same way using different names. Very important: First use *Add* to create a new section, then set the parameters. If a name appears in the *Section* field, the parameter settings or alterations will affect that section.
- Delete a section if needed by clicking the *Delete* button.
- Save all the sections of a job under a name of your choosing (with *File/Save job as*).

You can load a job via the *File* menu in the control panel or the CAM processor, or by clicking the corresponding icon in the control panel. To generate the output for a particular schematic or board, the file must be loaded via the CAM processor's File menu.

All the sections of the job will be executed if you click the *Process Job* button. A specific section will be executed if you click the "Process" button.

## **Creating your own Device Driver**

---

Output device drivers are defined in the EAGLE.DEF text file. Here you will find all the information that is needed for the creation of your own device driver. The best way is to copy the block for an output device of the same general category, and then alter the parameters where necessary.

Please use a text editor that does not introduce any control codes into the file.

## Film Creation using PostScript Files

---

Whereas until a few years ago Gerber files almost exclusively had to be generated for the manufacture of professional pcb films, there are today PostScript raster image recorders which offer a high-quality alternative which is simple to handle and economical.

With the "PS" driver, the CAM processor generates files in PostScript format. These can be processed directly by appropriate service companies (most of which operate in the print industry).

For PostScript recorders the *Width* and *Height* parameters should be set to very high values (e.g. 100 x 100), so that the drawing is not spread over several pages.

The *EPS* driver generates Encapsulated PostScript files, and these can be further processed with DTP programs.

---

## Gerber Files for Photoplotters with Variable Aperture Wheels

---

Nowadays most pcb manufacturers use Gerber photoplotters with variable aperture wheels. In contrast to photoplotters with fixed aperture wheels, no agreement about the aperture table is required between you and the film maker.

Please inquire which format is desired. The simplest way is the relatively new RS274-X format, in which the aperture table is integrated into the output file. The generation of the data is just as easy as it is for PostScript or any other printer. Use the GERBER\_RS274X driver for this.

RS-274D, or subsets of it, is the commonest format. In this case a file with the associated aperture table must be supplied, in addition to the files with the plot data. All the further explanations in this section are based on this format.

### GERBER.CAM Job for Two-Layer Boards

The *GERBER.CAM* CAM job uses the GERBERAUTO and GERBER drivers. These generate files in the RS-274D format. It is set up for two-layer boards which are to receive a solder stop mask on the component and on the soldering side, and is to receive a silkscreen print.

Load the job by clicking the GERBER icon in the control panel.

In the first step an aperture table *name.WHL* is automatically generated.

Data for the following layers is subsequently output:

<b>name.CMP</b>	component side
<b>name.SOL</b>	solder side
<b>name.PLC</b>	Silkscreen
<b>name.STC</b>	Solder stop mask, component side
<b>name.STS</b>	Solder stop mask, solder side

If other layers are also to be generated, e.g. silkscreen for the underside, or a solder cream mask, the Gerber job can be extended as required.

You simply have to add extra sections to the job in order to generate the appropriate layers. It is important to activate newly added layers in the first section (*Generating a wheel file*), since this produces a common aperture table for all the Gerber files. In the first section all the layers which will later be used to generate the individual Gerber files must therefore be simultaneously included.



Then save the newly created job in the CAM processor via *File/Save job* using a new name. Check once more whether all the necessary layers for the creation of the aperture table are active in the first section.

If you want to alter the pre-set values for the over-sizing of the solder stop mask or the solder cream, you must do this in both the first section for the aperture table generation, and also in the corresponding sections for the solder stop mask and the solder cream!

The output file generated in the first section cannot be used. For this reason, the file

**name.\$\$\$**

should be deleted.

The listing on page 78 can guide you in naming the output files.

### **Gerber Files for Photoplotters with Fixed Aperture Wheels**

---

Files for Gerber photoplotters with fixed aperture wheels are generated with the GERBER driver. It is essential to confer with your photoplot service bureau in good time, so as to adjust EAGLE to the available apertures.

There are various types of aperture. They differ in size and shape. The most common are circle, octagon, square, thermal and annulus symbols. The drawing aperture used for tracks is normally the round aperture.

You must specify the aperture configuration before you can generate files for a fixed aperture wheel photoplotter.

To do this, enter the configuration file for apertures e.g. with the EAGLE text editor, and load this file into the CAM processor by clicking the "Wheel" button.

The CAM processor normally chooses the aperture from the configuration file appropriately for each object. For each line, a draw aperture must be present with the appropriate diameter. For each symbol (e.g. pad) an aperture with the correct shape and size is required. If this is not available, no output file will be produced.

#### **Info File**

The apertures which have not been found are then listed in the file:

**name.gpi**

where name is the name chosen for the output file.

You can then change your board in such a way that the existing apertures can be used. After generating plot or drill files you should always check the related info files.



---

D010	annulus	0.004 x 0.000
D011	annulus	0.005 x 0.000
D015	annulus	0.055 x 0.000
D016	annulus	0.059 x 0.000
D017	annulus	0.063 x 0.000
D020	round	0.004
D021	round	0.005
D032	round	0.055
D033	round	0.059
D040	square	0.004
D041	square	0.006
D067	square	0.055
D052	square	0.059
D054	thermal	0.090 x 0.060
D055	thermal	0.100 x 0.063
D056	thermal	0.115 x 0.075
D057	thermal	0.120 x 0.080
D104	oval	0.030 x 0.090
D105	oval	0.090 x 0.030
D100	rectangle	0.060 x 0.075
D101	rectangle	0.075 x 0.060
D102	rectangle	0.050 x 0.030
D103	rectangle	0.030 x 0.050
D110	draw	0.004
D111	draw	0.005

**Aperture configuration file example:** all values default to inches, unless a unit is added (e.g. 0.010in or 0.8mm). Comments are marked with semicolons at the beginning of a line, or with a semicolon following a blank character.

### Aperture Emulation

If objects exist in a drawing which are not compatible with the available aperture sizes, it is possible to emulate the desired dimensions by selecting the *Emulate Apertures* option.

The CAM processor then selects smaller apertures to emulate dimensions which are not matched by aperture sizes. The file *name.gpi* indicates the need for emulation and which apertures must be emulated.

Note: emulation results in longer plot times and higher costs, so it should be avoided whenever possible.

### Aperture Tolerances

If you enter tolerances for draw and/or flash apertures, then if the aperture with the exact value is not available the CAM processor will use apertures within the tolerances. Please bear in mind that your design rules might not be kept as a result of allowing tolerances!

Refer to file *name.gpi* for tolerances used.

### Thermal and Annulus Symbols

Thermal and annulus symbols are only emulated if the *Emulate Thermal* and *Emulate Annulus* options are specified. If deactivated, the next (larger) aperture is taken. The file *name.gpi* tells you which one was used.

### Defining the Aperture Configuration

The CAM processor distinguishes DRAW apertures and FLASH apertures. The first type is used to draw objects (e.g. tracks). The second type is used to generate symbols (e.g. pads) by a light flash. Only if draw apertures are defined can the plotter draw lines. Therefore, if the plotter doesn't distinguish between draw and flash apertures, you must additionally define round or octagonal apertures as draw apertures. The following apertures are available:

<b>Name</b>	<b>Dimension</b>
Draw	diameter
Round	diameter
Square	length
Octagon	diameter
Rectangle	length-X x width-Y
Oval	diameter-X x diameter-Y
Annulus	outside diameter x inside diameter
Thermal	outside diameter x inside diameter

### Use of Aperture Shapes in the CAM processor:

Draw	draws wires and emulates apertures
Round	draws round pads and vias
Square	draws square pads, SMD's and vias
Octagon	draws octagonal pads and vias with the same X- and Y-dimensions
Rectangle	draws rectangles and SMD's
Oval	draws octagonal pads with different X- and Y-dimensions
Annulus	draws isolation rings in a supply layer
Thermal	draws connections in a supply layer

## Calculating Annulus and Thermal Symbols

EAGLE calculates the required dimensions for annulus and thermal apertures based on the pad's drill diameter and the annulus and thermal parameters set in the CAM processor:

Annulus:

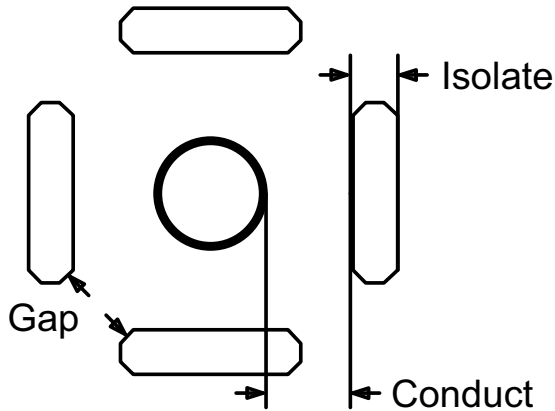
```
Inside diam. = Pad drill diameter + 2 * Conduct
Outside diam. = max(Pad drill dia., inside dia.)
                + 2 * Isolate
```

Default: Conduct = -1000 (filled), Isolate = 20 (Mil).

Thermal:

```
Inside diam. = Pad drill diameter + 2 * Conduct
Outside diam. = inside diameter + 2 * Isolate
Gap: gap of the conducting path
      (only relevant with emulation)
```

Default: Conduct = 20 (Mil), Isolate = 10 (Mil), Gap = 20 (Mil).



Meaning of the Gap, Isolate and Conduct parameters.

## Drill Data and Drill Plan

---

The generation of drill data is very similar to the generation of photoplot data. The aim is to create files which are sent to the board manufacturer. The appropriate device drivers are SM1000, SM3000, and EXCELLON. A drill configuration has to be defined before an output can be generated. The manufacturer must know this configuration, otherwise he cannot drill the proper holes. It is therefore advisable to send the drill *rack* configuration file along with the drill *data* file(s) to the manufacturer.

EAGLE generates drill holes for the objects *pads*, *vias*, and *holes*. The appropriate data for pads and vias (plated-through holes) are generated if the Drills layer is active during the CAM processor output session. The data for holes (with no contact between different layers) are generated if the Holes layer is active. Both layers must be active if all holes should be drilled together. If they should be drilled separately, either the Holes or the Drills layer must be active, and two separate files created.

A tolerance can be specified for drills. In such cases please make sure that your design rules are maintained.

### Info File

The file *name.dri* contains the missing drill diameters and other important information. It is written into the same directory as the output file.

### Drill Data

To generate the drill data for a board, carry out the following steps:

- Load the board into the layout editor.
- Execute the DRILLCFG.ULP user language program with the RUN command. It generates the *name.DRL* drill configuration file.
- Load the board into the CAM processor (*File/Open/Board*).
- Load the predefined job EXCELLON.CAM, e.g. with *File/Open/Job*.
- Check the parameters and change them to fit your needs (e.g. choose a different device driver like SM1000 or SM3000). A tolerance of  $\pm 0.025$  makes sense to avoid problems due to internal unit conversions.
- Save the file via *File/Save job*.
- Execute the job (*Process Job*).

Send the files *name.drl* and *name.drd* (perhaps *name.dri* as well) to your boardhouse.

```
T01  0.010
T02  0.016
T03  0.032
T04  0.040
T05  0.050
T06  0.070
```

**Drill configuration file example:** All values default to inches, unless a unit is added (e.g. 0.010in or 0.8mm). Comments are marked with semicolons at the beginning of a line, or with a semicolon following a blank character.

## Drill Plan

A drill plan can be generated which enables you to check the drill holes and their diameters. It shows an individual symbol for each diameter. EAGLE uses 19 different symbols: 18 of them are assigned a certain diameter; one ( $\emptyset$ ) appears if no symbol has been defined for the diameter of this hole.

The diameter symbols appear in the Drills layer at the positions where pads or vias are placed, and in the Holes layer at the positions where holes are placed. These layers must therefore be active when the drill plan is generated.

The relation between diameters and symbols is defined via the *Options/Set/Drill* dialog of the layout editor.

The figure shows the predefined drill plan symbol numbers and their related symbols.

1	+	10	✕
2	✕	11	▽
3	□	12	△
4	◇	13	◁
5	⊗	14	▷
6	⊠	15	⊕
7	⊚	16	⊗
8	⊛	17	⊞
9	✖	18	⊞

## The Autorouter

---

### Basic Features

---

- Any routing grid (min. 4 mil)
- Any placement grid (min. 0.1 micron)
- SMDs are routed on both sides
- The whole drawing area can be the routing area (provided enough memory is available)
- The strategy is selected via control parameters
- Multilayer capability (up to 16 layers can be routed simultaneously, not only in pairs)
- The preferred track direction can be set independently for each layer: horizontal and vertical, true 45/135 degrees (important for inner layers!)
- Ripup and retry for 100 % routing strategy
- Optimization passes to reduce vias and smooth track paths
- Prerouted tracks are not changed

### What Can be Expected from the Autorouter

---

The EAGLE autorouter is a “100 %” router. This means that boards which, in theory, can be completely routed will indeed be 100 % routed by the autorouter, provided — and this is a very important restriction — the autorouter has unlimited time. This restriction is valid for all 100 % autorouters whatsoever. However, in practice, the required amount of time is not always available, and therefore certain boards will not be completed even by a 100 % autorouter.

The EAGLE autorouter is based on the ripup/retry algorithm. As soon as it cannot route a track, it removes prerouted tracks (ripup) and tries it again (retry). The number of tracks it may remove is called ripup depth which is decisive for the speed and the routing result. This is, in principle, the previously mentioned restriction.

Those who expect an autorouter to supply a perfect board without some manual help will be disappointed. The user must contribute his ideas and invest some energy. If he does, the autorouter will be a valuable tool which will greatly reduce routine work.

Working with the EAGLE autorouter requires that the user places the components and sets control parameters which influence the routing strategy. These parameters must be set carefully if the best results are to be achieved. They are therefore described in detail in this section.

## Controlling the Autorouter

---

The autorouter is controlled by a series of parameters which are defined in the DEFAULT.CTL or *boardname*.CTL file (Table 9-2). This file is searched for initially in the current directory, and then in the directory where EAGLE.EXE is located. The default values will be used if no such file is found. The control file can be modified by means of the autorouter menu.

In principle, the routing process consists of several steps which are initialized by “labels” in the control file.

### Bus Router

Normally the bus router (whose parameters are set such that the bus structures can be optimally routed) is the first pass. This step may be omitted. The bus router is activated only if there is a layer with the appropriate preferred direction.

### Routing Pass

The actual routing pass is then started, using parameters which make a 100 % routing as likely as possible. A large number of vias are deliberately allowed to avoid paths becoming blocked.

### Optimization

After the main routing pass, any number of optimization passes can be made. The parameters are then set to remove superfluous vias and to smooth the track paths. In the optimization passes tracks are removed and rerouted one at a time. This can, however, lead to a higher degree of routing, since it is possible for new paths to be freed by the changed path of this track.

### Routing Examples

There are control files (\*.ctl) optimized for some of the example boards delivered with the EAGLE software. These parameter sets are applied automatically if the name of the control file is identical to the name of the board file.

## Control Parameters and their Meanings

---

Parameter	Default	Meaning
RoutingGrid	= 50	Grid used by the autorouter for tracks and via-holes (see "Considerations for the Grid")
		Minimum distances between autorouted wires and...
1)		
mdWireWire	= 8	...Wires
mdWirePad	= 8	...Pads
mdWireDimension	= 40	...Lines (wires) in Dimension layer
mdWireVia	= 8	...Via-holes
mdWireRestrict	= 8	...Restricted areas
		Minimum distances between autorouted vias and...
1)		
mdViaPad	= 8	...Pads
mdViaDimension	= 40	...Lines (wires) in Dimension layer
mdViaVia	= 8	...Via-holes
mdViaRestrict	= 8	...Restricted areas
		Cost factors for...
cfVia	= 8	...Vias
cfNonPref	= 5	...Not using preferred direction
cfChangeDir	= 2	...Changing direction
cfOrthStep	= 2	...0 or 90 deg. step
cfDiagStep	= 3	...45 or 135 deg. step
cfExtdStep	= 0	...deviation 45 deg. against preferred direction
cfBonusStep	= 1	...Step in bonus area
cfMalusStep	= 1	...Step in handicap area
cfPadImpact	= 4	...Pad influence on surrounding area
cfSmdImpact	= 4	...SMD influence on surrounding area
cfBusImpact	= 4	...Leaving ideal bus direction
cfHugging	= 3	...Wire hugging
cfAvoid	= 4	...Previously used areas during ripup
cfBase.1	= 0	Basic costs for a step in the given layer
cfBase.2	= 1	
...		
cfBase.15	= 1	
cfBase.16	= 0	
		Maximum number of...
mnVias	= 20	...Vias per connection
mnSegments	= 9999	...Wire segments per connection
mnExtdSteps	= 9999	...Steps 45 deg. against preferred direction
mnRipupLevel	= 10	...Ripups per connection
mnRipupSteps	= 100	...Ripup sequences per connection
mnRipupTotal	= 20	...Ripups at the same time
		Track parameters for...
tpWireWidth	= 16	...Wire width
tpViaDiameter	= 40	...Via diameter
tpViaDrill	= 24	...Via drill diameter
tpViaShape	= Round	...Via shape (round or octagon)
PrefDir.1	=	Preferred direction in the given layer
PrefDir.2	= 0	Symbols: 0 - /   \ *
...		0 : Layer not used for routing
PrefDir.15	= 0	* : No preferred direction
PrefDir.16	= -	- : X is preferred direction
		: Y is preferred direction
		/ : 45 deg. is preferred direction
		\ : 135 deg. is preferred direction

All values in mil

1) The following preconditions occur:

```
tpViaDiameter + 2 * mdWireVia > tpWireWidth + 2 * mdWireWire
tpViaDiameter + 2 * mdViaVia > tpWireWidth + 2 * mdWireVia
tpViaDiameter + 2 * mdViaPad > tpWireWidth + 2 * mdWirePad
```



## **What Has to be Defined in the Control File Before Autorouting**

---

### **Design Rules**

The design rules must be set according to the board complexity and the manufacturing technologies to be used. Modify the corresponding parameters (mdXXX or tpXXX) in the control file. Do not forget to set the parameters for the DRC command appropriately. The best way is to prepare a script file which has the same values.

### **Grid**

The design rules determine the routing and placement grid. See also “Considerations for the Grid”.

### **Layer**

Layers on which the autorouter must not place tracks are defined with PrefDir.X=0 (X = number of layer).

If you wish to design a two-sided board, select Top and Bottom as route layers. For a single-sided board, only the bottom layer should be used. With inner layers we recommend you use the layers from the inside to outside, i.e. first 2 and 15 and so on. Inner layers become supply layers when you rename them with \$name, where name is a valid signal name.

Boards which are too complex to be routed on two sides should be planned as multilayers. In this case the costs for the inner layers should be very high.

### **Preferred Directions**

On the two outside layers the preferred directions are normally set to 90 degrees from each other. For the inner layers it may be useful to choose 45 and 135 degrees to cover diagonal connections. See also “Considerations for Preferred Directions”. Before setting the preferred direction it is well worth examining the board (based on the airwires) to see if one direction offers advantages for a certain side of the board. This is particularly likely to be the case for SMD boards.

**Please also follow the preferred direction when pre-placing tracks. The defaults are vertical for the Top (red) and horizontal for the Bottom (blue).**

### **Bus Router**

Only activate the bus router if there are bus structures on the board. Buses are only routed if there is a layer with the corresponding preferred direction.

### **Cost Factors**

The default values for the cost factors are set in line with our experience to produce the best results. Therefore we recommend

that, with the exception of via costs, they remain unchanged. With all other parameters, even small modifications may have considerable effects.

### **Further Control Parameters**

The remaining control parameters such as `mnRipupLevel`, `mnRipupSteps`, etc. are set in line with our experience to produce the best results.

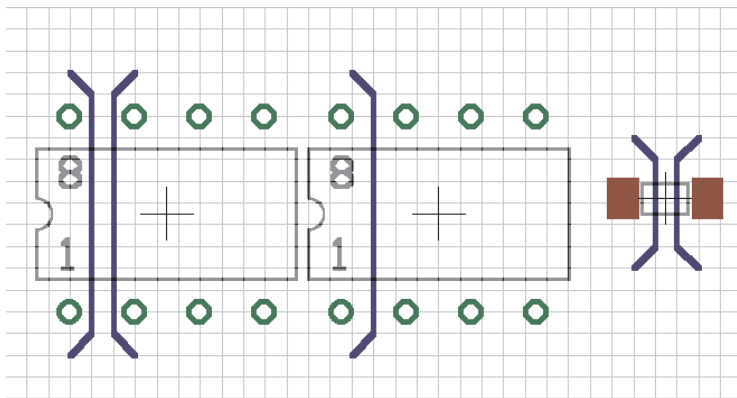
## Grid Considerations

### Placement Grid

Although the autorouter does permit any placement grid, it is not sensible to place the components on too fine a grid. As a rule:

- The placement grid should not be finer than the routing grid.
- If the placement grid is larger than the routing grid, it should be set to an integral multiple of the routing grid.

These rules make sense if, for example, you consider that it might be possible, within the design rules, to route two tracks between two pins of a component, but that an inappropriate relationship between the two grids could prevent this (see diagram).



### Routing Grid

Bear in mind that for the routing grid the time demand increases exponentially with the resolution. Select therefore as large a grid as possible. The main question for most boards is how many tracks are to be placed between the pins of an IC. To answer this question, the selected design rules (i.e. the minimum spacing between tracks and pads or other tracks) must of course also be considered.

The result is:

**The two grids must be selected so that component's pads are located on the routing grid.**

There are of course exceptions, such as with SMD's to which the opposite may apply, namely that a position outside of the routing grid leads to the best results. In any event the choice of grid should be carefully considered in the light of the design rules and the pad spacing.

The example may clarify the situation:

For the component on the left, the pads are placed on the routing grid. Two tracks can be routed between two pads. The pads of the component in the middle are not on the routing grid, and therefore only one track can be routed between them.

On the right you see the exception from the rule shown for SMD pads, which are placed between the routing grid lines so that one track can be routed between them.

When choosing the grid please also ensure that each pad covers at least one grid point. Otherwise it can happen that the autorouter is unable to route a signal, even though there is enough space to route it (message: *Unreachable SMD at x y*).

## Considerations for Preferred Directions

---

Experience has shown that for small boards with mainly SMD's it may be useful to route the tracks without preferred directions (cfNonPref = 0 and PrefDir.xx = \*). The router will then produce a useful result much faster. Boards which are to be routed on one side only must be routed without a preferred direction.

## Restricted Areas for the Autorouter

---

If the autorouter is not supposed to route tracks or place vias within certain areas, you can define restricted areas by using the commands RECT, CIRCLE, and POLYGON in the layers tRestrict, bRestrict, and vRestrict.

**tRestrict:** Restricted areas for the Top layer.

**bRestrict:** Restricted areas for the Bottom layer.

**vRestrict:** Restricted areas for via-holes.

Such areas can be defined either in the board or in the library (package), e.g. around a component's mounting holes.

## How the Cost Factors Influence the Routing Process

---

For each cost factor (cfXXX) values between 0..99 are possible, but the full range is not useful for all parameters. Sensible values are therefore given with each parameter. We would, however, like to stress again that we recommend you use only the values given in the DEFAULT.CTL file (except for cfVia).

### **cfVia: 0..99**

Controls the use of vias. A low value produces many vias but also allows the preferred direction to be followed. A high value tries to avoid vias and thus violates the preferred direction. Recommendation: low value for the routing pass, high value for the optimization.

### **cfNonPref: 0..10**

Controls that the preferred direction is followed. A low value allows tracks to be routed against the preferred direction, while a high value forces them into the preferred direction.

### **cfChangeDir: 0..10**

Controls how often the direction is changed. A low value means many bends are allowed within a track. A high value produces virtually straight tracks.

### **cfOrthStep, cfDiagStep**

Causes the rule that, in a right-angled triangle, the hypotenuse is shorter than the length of the other two sides. The default values are 2 and 3. That means that the costs for the route using the two other sides are 2+2 (i.e., 4), as against 3 for the hypotenuse. These parameters should never be changed!

### **cfExtdStep: 0..10**

Controls the avoidance of track sections which run at an angle of 45 degrees to the preferred direction, and which would divide the board into two sections. A low value means that such sections are allowed while a high value tries to avoid them. To allow for the possibility of smoothing 90 degree track corners with short 45 degree sections, this parameter only becomes effective for track sections of the length set with "mnExtdSteps". This parameter is only relevant to layers which have a preferred direction. Recommendation: higher value for the routing pass, lower value for the optimization.

### **cfPadImpact, cfSmdImpact: 0..10**

Pads and SMDs produce "good" and "bad" sections or areas around them in which the autorouter likes (or does not like) to place tracks. The "good" areas are in the preferred direction (if defined), the "bad" ones perpendicular to it. This means that tracks which run in the

preferred direction are routed away from the Pad/SMD. With high values the track will run as far as possible in the preferred direction, but if the value is low it may leave the preferred direction quite soon.

**cfBonusStep, cfMalusStep: 1..3**

Controls the use of “preferred” areas and the avoidance of “bad” areas on the board. With high values the router differentiates strongly between “good” and “bad” areas. With low values the influence of this factor is reduced.

**cfBusImpact: 0..10**

Controls whether the ideal line is followed for bus connections (see also cfPadImpact). A high value ensures that the direct line between start and end point is followed. Only important for bus routing.

**cfHugging: 0..5**

Controls the hugging of parallel tracks. A high value allows for a strong hugging (tracks are very close to each other), a low value allows for a more generous distribution. Recommendation: higher value for routing, lower value for the optimization.

**cfAvoid: 0..10**

During the ripup, areas are avoided from which tracks were removed. A high value means strong avoidance.

**cfBase.xx: 0..5**

Base costs for one step on the corresponding layer. Recommendation: outside layers (Top, Bottom) always 0, inside layers greater than 0.

## Number of Ripup/Retry Attempts

---

Due to the structure of the autorouter there are some parameters which influence the ripup/retry mechanism. They are set in such a way that they offer a good compromise between time demand and routing result. The user should therefore not change the values for mnRipupLevel, mnRipupSteps and mnRipupTools.

As a rule, high parameter values allow for many ripups but result in increased computing times.

To understand the meaning of the parameters you need to know how the router works.

To begin with the tracks are routed one after the other until no other path can be found. As soon as this situation occurs, the router removes up to the maximum number of already routed tracks (this number has been defined with mnRipupLevel) to route the new track. If there are 8 tracks in the way, for example, it can only route the new track if mnRipupLevel is at least eight.

After routing the new track, the router tries to reroute all the tracks which were removed. It may happen that a new ripup sequence must be started to reroute one of these tracks. The router is then two ripup sequences away from the position at which, because of a track which could not be routed, it started the whole process. Each of the removed tracks which cannot be rerouted starts a new ripup sequence. The maximum number of such sequences is defined with the mnRipupSteps parameter.

The parameter mnRipupTotal defines how many tracks can be removed simultaneously. This value may be exceeded in certain cases.

If one of these values is exceeded, the router interrupts the ripup process and re-establishes the status which was valid at the first track which could not be routed. This track is considered as unroutable, and the router continues with the next track.



## **Backup, Interruption of Routing, and Best Data**

---

As, with complex layouts, the routing process may take several hours, a backup is carried out at intervals (approx. every 10 minutes). The file *name*.JOB always contains the last status of the job. If the job is interrupted for any reason (power failure etc.) the computer time invested so far is not lost, since you can recall the status saved in *name*.JOB. Load the board with "EDIT", and then enter "AUTO". Answer the prompt as to whether the autorouter should recall ("Continue existing autorouter job!") with "yes". The autorouter will then continue from the position at which the job was last saved (max. 10 minutes may be lost).

If the autorouting is interrupted with "Ctrl-Break", the *name*.JOB file also remains intact and can be recalled. This may be useful when you have started a complex job on a slow computer and want to continue with it on a fast computer as soon as one is available.

Please note that changing the parameters before recalling will not influence the job, since it will have been saved together with the parameters which were valid at the time of the initial autorouter start.

When the autorouter has finished, the routed board is saved as *name*.B\$\$\$. You can rename and use it, for instance, if a power failure occurred after the autorouting run (and therefore the *name*.JOB file has been deleted). This file is deleted automatically after the board has been saved.

## Setting the Control Parameters in the Autorouter Menu

If you don't want to start the autorouter with the default values, you can change the control parameters directly within a menu. Changing the control parameters within the menu while the board *name*.BRD is loaded creates the control file *name*.CTL in the directory where the board is stored. This file has a higher priority for the board *name*.BRD than the DEFAULT.CTL file. The settings are, of course, only valid for this board.

Please note that EAGLE looks for the file *name*.CTL first. If there is no such file, the program will look for DEFAULT.CTL. If this file doesn't exist either, the default settings are used.

The menu is displayed when you select AUTO in the command menu, or if you enter it via the keyboard (terminated with return but without semicolon). Then you can set the parameters for the individual passes, one after the other. Select the entries "Busses", "Route", "Optimize1...3" with the mouse, and enter the values for the selected pass. The mark in the check box next to "Busses" etc. indicates that this pass will be executed.

Layer	Costs	Maximum	Minimum Distance
1 Top	Via: 8	Vias: 20	Wire: 0.00800
2 Route2	NonPref: 5	Segments: 9999	Pad: 0.00800
3 Route3	ChangeDir: 2	ExtSteps: 9999	Via: 0.00800
4 Route4	OrthStep: 2	RipupLevel: 10	Dim: 0.04000
5 Route5	DiagStep: 3	RipupSteps: 100	Restr: 0.00800
6 Route6	ExtStep: 0	RipupTotal: 100	
7 Route7	BonusStep: 1		
8 Route8	MalusStep: 1		
9 Route9	PadImpact: 4		
10 Route10	SmdImpact: 4		
11 Route11	BusImpact: 0		
12 Route12	Hugging: 3		
13 Route13	Avoid: 4		
14 Route14			
15 Route15			
16 Bottom			

Pass:

- Busses
- Route
- Optimize1
- Optimize2
- Optimize3

Track:

- Grid: 0.05000
- Wire Width: 0.01600
- Via Diameter: 0.04000
- Via Drill: 0.02400
- Via Shape: Round

Continue existing job

Buttons: Create Job, Select, Start, Cancel, Help

On the left of the menu you can enter the preferred directions for any layer (first entry) and their basic costs (second entry). If you want the autorouter **not** to use a layer, enter "0" into the preferred direction field.

All parameters are global, except the *Costs* and *Maximum* groups. These can be different for any routing run.

All values are given in the current unit.

Create Job: Creates an autorouter job with the current parameters.

Select: You may determine certain signals for autorouting according to the syntax of the AUTO command.

Start: Starts the autorouter for all not yet routed signals.

Cancel: Cancels all changes.

End Job: Quits the current job and loads the best result accomplished so far. This button doesn't show up unless a job is in progress.

## Polygons as Supply Layers

---

You will find more about multilayer boards, ground areas, and supply layers on page 54. If you use the autorouter in supply layers realized with polygons, please bear the following points in mind.

It is possible, with polygons, to create supply layers, which contain more than one supply voltage and/or additional wires. In such cases the layers are not regular “supply layers” anymore, which would have been marked with a “\$name”. They become regular signal-routing layers.

If a polygon is located on a signal-routing layer, the autorouter can no longer place vias within the polygon. Because of this, it does not make sense to define a polygon until all other signals have been routed.

If you still want to use the autorouter, even if supply-polygons have already been defined, the following procedure is recommended:

- assume that a supply polygon exists for GND in layer 2
- load the board file (polygons are not calculated yet)
- configure the autorouter so that layer 2 is NOT available
- route everything, except GND (“Auto !GND;” or click Select and type “!GND;”)
- activate layer 2 for the autorouter and route GND

Now the autorouter can work on the other layers (and penetrate the GND-polygon with vias), but doesn’t put signals within the GND-polygon (which would cause the GND-polygon to fragment).

---

## Information for the User

---

### Status Display

During the routing, the autorouter displays information on the actual routing result. The displayed values have the following meaning:

```
Routing: result % (hitherto maximum, best data)
Vias:    Number of vias
Conn.:   Connections total/total/total not routable
Ripup:   No. of ripups/act. RipupLevel/act. RipupTotal
Signals: Signals found/signals handled/signals prepared
```

Connections means 2-point connections.

### Log File

For each routing pass the autorouter generates a file called name.PRO which contains useful information.

Example:

```
EAGLE AutoRouter Statistics:

Job           : C:\EAGLE\DEMO2.BRD
Strategy      : C:\EAGLE\DEMO2.CTL

Start at     : 12:12:51 ( 1/1/1997)
End at       : 12:16:49 ( 1/1/1997)
Elapsed time : 00:03:56

Signals      : 48   RoutingGrid: 50 mil Layers: 2
Connections  : 95   predefined: 11 ( 0 vias )

Router memory : 20164

Passname      :      Busses      Route Optimize1 Optimize2 Optimize3
Time per pass : 00:00:11 00:00:45 00:01:05 00:00:58 00:00:57
Number of Ripups : 0 0 0 0 0
max. Level : 0 0 0 0 0
max. Total : 0 0 0 0 0

Routed : 30 84 84 84 84
Vias : 0 87 23 19 19
Resolution : 43.2 % 100.0 % 100.0 % 100.0 % 100.0 %
```

# Index

## A

Action Toolbar	14, 16
ADD	18, 25, 33
Addlevel	72
Airwire	8
Alt-X	12
Always	72
Annulus	
Aperture	88
Calculate Dimensions	91
Annulus Symbol	90
Aperture	
Configuration	81
Tolerance	81
Aperture Configuration	
Defining	90
Aperture Emulation	89
Aperture Wheels	
Fixed	88
Variable	85
Apertures	
Configuration	79
Draw	90
Flash	90
Forms	90
ARC	19, 26
ASSIGN	21
AUTO	27
Automatic Naming	43
Autorouter	94
100 % Router	94
Backup	105
Bus Router	97
Controlling	95
Cost factors	102
Cost Factors	98
Design Rules	97
Examples	95
Features	94
Further Control Parameters	98

Grid	97, 99
Interrupting	105
Layer	97
List of Control Parameters	96
Log File	109
Placement Grid	99
Preferred Direction	54, 94, 97, 101
Restarting	105
Restricted Areas	101
Ripup/Retry	104
Routing Grid	99
Setting the Control Parameters	106
Supply Layers	86
Autorouter Module	9

## B

Backup	13
Bend	
Inserting	26
BOARD	16
bRestrict	75, 101
Bubble Help	15
BUS	19
Bus Router	95

## C

CAM Job	
Gerber	81
Output	82
CAM Processor	12, 21, 34, 49
Aperture Tolerances	90
Drill Data	92
Drill Plan	93
Gerber Output	88
Photoplot Output	88
Can	72
CHANGE	18, 19, 25, 33, 46, 58
CIRCLE	19, 26
Clk	60
CLOSE	20
Command Language	39

Command Line	37	Name	76
Command Parameters	15	DRC	8, 27
Command Toolbar	14	Drill	
Component Design	56	Configuration	92
Component List	45	Data	92
Configuration		Diameter	75, 93
Commands	46	Hole Diameters	52
EAGLE	46	Information	77
CONNECT	33	Plan	92, 93
Consistency	20	Symbol	93
Context Menu	11	Tolerance	92
Control Panel	10	Drill Data	92
Coordinates		Drill Info File	92
Display	14	DRILLCFG.ULP	92
Entering	41	Drivers	
Copy		Output	83
File	11	<b>E</b>	
COPY	17, 24	EAGLE.SCR	46
Current Directory	12	Presettings	21
Current Unit	42	EDIT	20, 29
CUT	18, 25	Edition	9
<b>D</b>		ELC	9
Default Directories	11	Enter Key	15, 39
Delete		Environment	46
File	11	EPS	84
from Library	29	ERC	8, 20, 27
DELETE	18, 25	ERRORS	27
Design Rule Check	27	EXCELLON	77, 92
Desktop Publishing	84	Exit	12
Device	68	EXPORT	21, 45
Device Driver	83	DIRECTORY	45
Diameter of Lands	52	NETLIST	45
Dimensions of Pads	52	NETSCRIPT	45
DIR	20	PARTLIST	45
Direction	50, 61	PINLIST	45
Directories	12	SCRIPT	45
DISPLAY	17, 24	<b>F</b>	
Distance between Potentials	52	False Connections	50
Documentation Print	57	File	
Dot	60	Commands	11
DotCk	60	Menu	11
Drawing		New	11
Frames	76		

Open	11	Commands	37
Save all	12	Inverted Commas	57
Save Project as	12	INVOKE	19
Film Maker	85	<b>J</b>	
Fixing Hole	75	JUNCTION	20
Forbidden Areas	53	<b>L</b>	
Forward&Back Annotation	48	LABEL	20
Function	60	Labeling	58
Function Keys	16, 21, 37	Last change	76
<b>G</b>		LAYER	21
Gate	8	Layout Editor	23
GATESWAP	18	Layout Toolbar	24
Gerber Information File	88	Length	60
GERBER.CAM	85	Library	
GERBER_RS274X	85	Closing	20
GERBERAUTO	81, 86	Editor	28
GND symbol	67	Opening	20
Grid		Light Edition	9
Autorouter	99	Line	8
Current Unit	42	Log File	
GRID	15	Autorouter	109
LINES	39	<b>M</b>	
Ground Layer	54	MARK	17, 24
GROUP	17, 24	MENU	21
<b>H</b>		MIRROR	17, 24
Help	7	Mirrored Layer	66
Function	15	Mixed Input	38
Menu	13	Modules	
History Function	37	EAGLE	9
Hiz	61	Mouse Button	11
HOLE	27	Mouse Click	40
<b>I</b>		Mouse Keys	22
I/O	61	MOVE	17, 24
Icons		Multilayer Boards	54
Select	11	Must	72
Identifying point		<b>N</b>	
of a component	57	NAME	18, 25, 33
Import	44	NC	61
In	61	Neighboring Objects	
INFO	17, 24	Selecting	22
Input		Net	8, 50



NET	19	EXPORT	45
NETSCRIPT		PINSWAP	18, 25
EXPORT	45	Plug	73
Next	72	POLYGON	19, 26
None	60	PostScript Files	84
<b>O</b>		Power	
OC	61	Supply	50
Open		Supply Layer with Signal	54
File	11	PREFIX	33
OPEN	20	Print	
Optimization	95	File	11
OPTIMIZE	26	PRINT	21
Options Menu	12	Printout Time	76
OrCAD	9	Product Information	13
Orientation	60	Product Registration	13
Out	61	Professional Edition	9
Outline	44	Project Files	46
Output	80	Pwr	50, 61
See also Export		<b>Q</b>	
as CAM job	82	QUIT	20
Device	80	<b>R</b>	
Drawings	21	Ratsnest	8
Drivers	83	RATSNEST	27
Files	80	RECT	19, 26
Parameters	35	REDO	16
<b>P</b>		REMOVE	21, 29
Package	8	Rename	
Editing	30	File	11
Replacing	26	RENAME	29
PACKAGE	33	REPLACE	26
Pad	8	Request	72
PAD	30	Resistor	
Parameter Toolbar	14	Device	63
PARTLIST		Package	57
EXPORT	45	Symbol	60
Pas	61	RIPUP	26
PASTE	18, 25	ROTATE	17, 24
Pin	8	ROUTE	26
Names	62	Routing	53
Superimposed	50	Pass	95
PIN	31	RUN	16
PINLIST			

## S

Schematic	
Command Toolbar	17
Editor	14
Module	9
Symbol	62
Symbol, Labeling	65
Screen	
Redraw	16
SCRIPT	16
EXPORT	45
Script Files	37, 44
Select	
Icons	11
Semicolon	39
SET	21
Shape of Lands	52
Sheet Number	76
Sheets	9
SHOW	17, 24
Signal	8
SIGNAL	27
Signal Layer	54
Silkscreen	57
Single-sided Routing	97
SM1000	92
SM3000	92
SMASH	18, 25
SMD	30
Solder Stop	85
SPLIT	19, 26
Standard Edition	9
Status Display	109
Sup	50, 61
Superimposed Pins	50
Supply Layer	
Polygons as	108
Supply Pins	
Invisible	69
Supply Symbols	67
Swaplevel	61
Symbol	8
Symbol Editing	31

## T

Technical Terms	8
Termination	
of Command	16
TEXT	19, 26
Text Editor	36
Text Size and Thickness	52
Thermal	54
Calculate Dimensions	91
Symbol	90
Track Width	52
tRestrict	101

## U

UNDO	16
USE	16
User	
Interface	13
Language	47

## V

VALUE	18, 25, 33
Via	8
VIA	27
Visible	60
vRestrict	101

## W

Window	
Menu	13
Number	13
Wire	8
WIRE	19, 26
WRITE	20

## X

XCONVERT	9
XPAD	9