cadence®

Preparing Manufacturing Data

Product Version 23.1
September 2023

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Contents

1

Preparing Manufacturing Data Overview

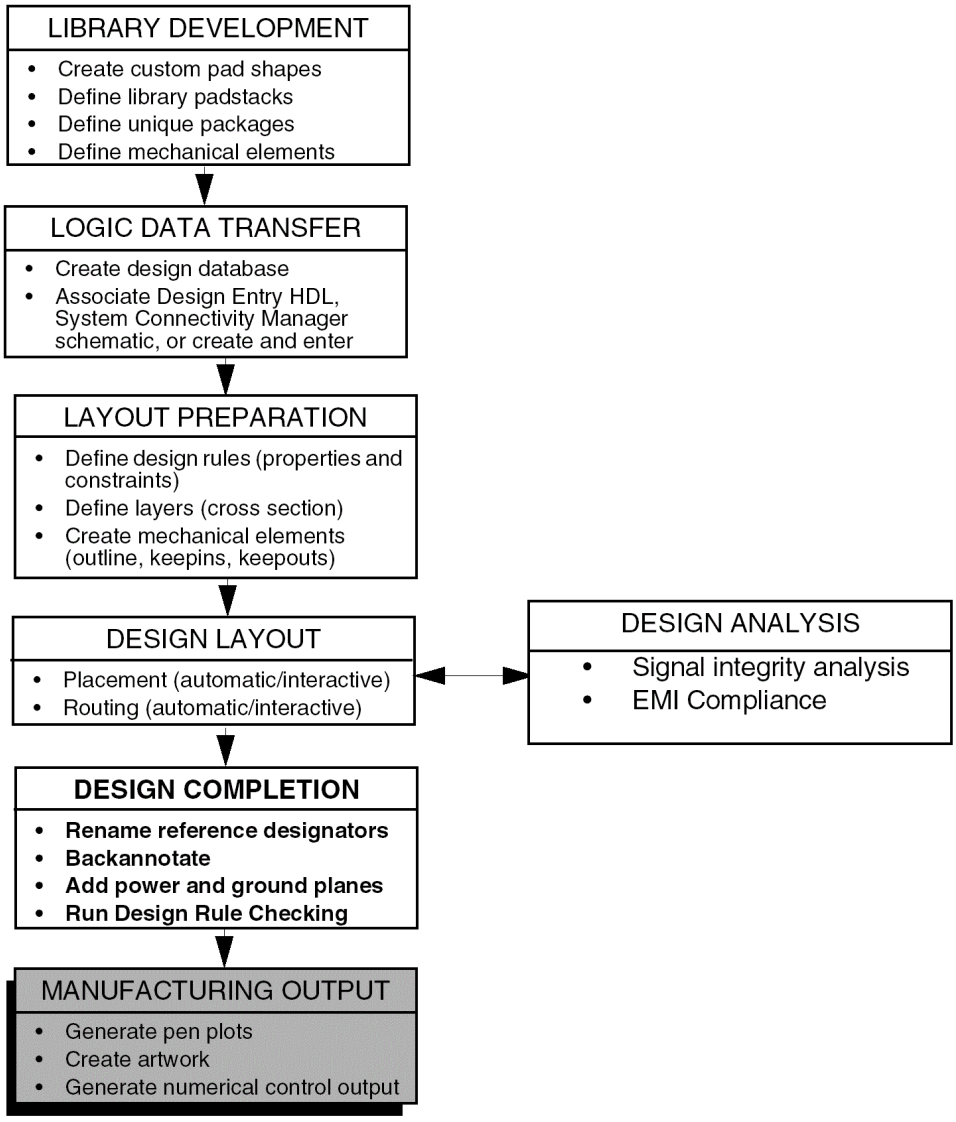
This set user guide describes different manufacturing processes by generating output files for these processes:

- Files for numerically controlled (NC) drills and routers
- Silkscreen files
- Penplotting files
- Artwork (photocopying) files

> Many features are common to all three layout editors: Allegro X PCB Editor, Allegro Package Designer, and System-in-Package tools. When a feature is not common to all editors, it is noted in the heading. If an illustration shows only one of the editors, it is also noted.

The following figure illustrates where manufacturing processes occur in a PCB design flow.

Artwork Generation in a PCB Design Flow



2

PCB Editor and APD: Test Preparation

Testprep creates test probe sites for any type of test fixture using parameters that enable you to control making test locations on a design. You can create test points automatically or interactively, as well as edit all test point locations. You can automatically choose and label appropriate component pin, via, or pad locations as test points. Once you finish generating test points, you can create artwork and NC data files for drilling design test beds.

Testprep Design Rule Check (DRC)

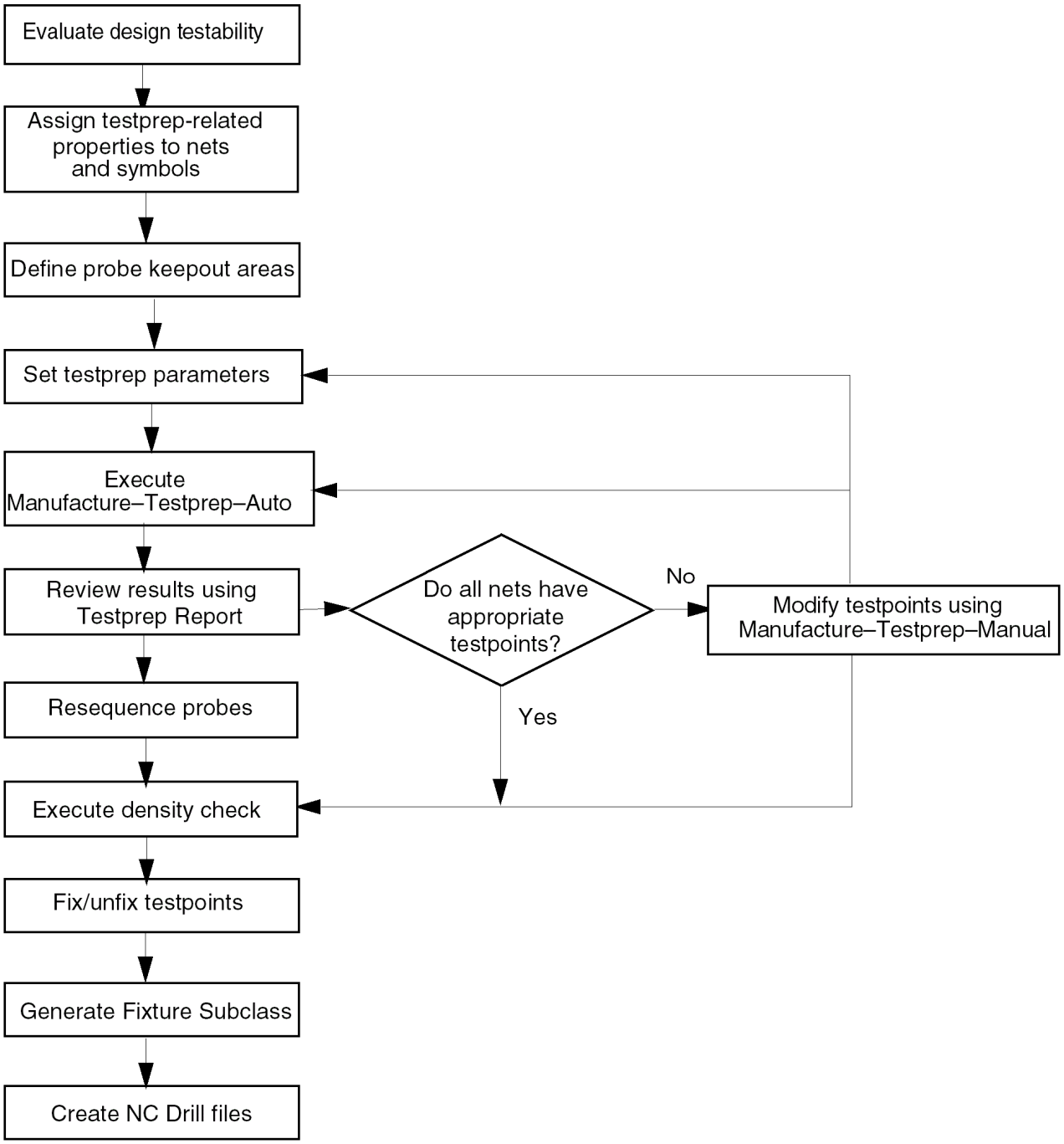
The testprep process scans the database to create test points for each net, according to the criteria that you establish in the Testprep Parameters dialog box, and runs appropriate DRCs. These include test point-to-component spacing DRCs, such as test point-location and test point-pad-to-component outline. When you set the *Allow Under Component* field to *Never,* these DRCs detect a test point that may lie beneath a component as a result of placing or moving a component over an existing test point via. The current setting for *Component Representation (*either ASSEMBLY or PLACE_BOUND) is used. Changing either of these fields causes the layout editor to mark DRCs out of date.

For test point pins and vias too close to a component (package), TC bowtie characters flag test point pins and vias that fail either of the test point-to-component spacing checks. For test point pins and vias under a component, TC bowties also flag DRCs. The DRC subclass layer is top or bottom, corresponding to the side on which the component is placed.

Testprep Process Flow

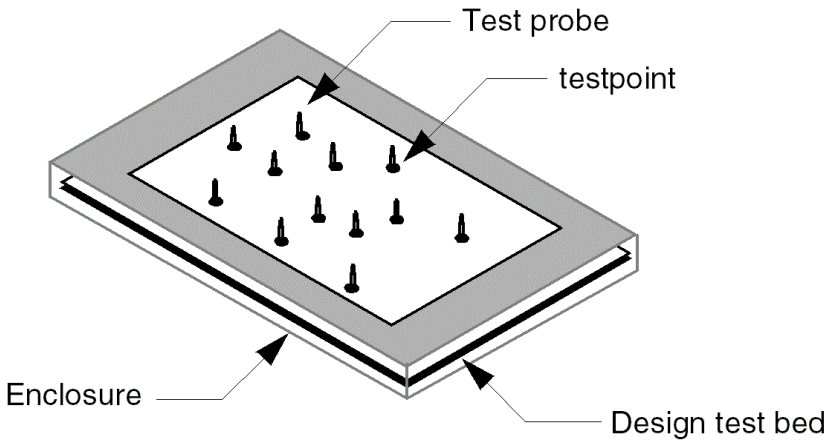
Due to the high density of design elements on PCBs and MCMs, Cadence recommends that you run testprep to optimize testing.

The Testprep Process Flow



It is ideally suited to the requirements of testing on a "bed-of-nails" test fixture as illustrated in the following figure, but it offers the flexibility to accommodate any type of testing requirement, such as bareboard or in-circuit testing.

Parts Associated with a Bed-of-Nails Fixture



Bareboard Testing

A bareboard test consists of an electrical continuity check performed after fabrication of the physical board. Connections between all component pins ensure that no shorts or opens exist. Once the physical board is checked, it is ready for assembly. During a bareboard test, all component pins on each side of the PCB are probed. Doing so requires a fixture containing probe pins that contact the PCB at specific locations. Bareboard testers use a bed of nails, which are spring-loaded pins positioned every 100 mils. Flexible and double-density testers have pins that are closer. These nails are hard-wired to a test box that can recognize any connectivity that may exist between them.

A test fixture is sandwiched between the bed of nails and the PCB. The fixture contains probe pins that adapt the bed of nails to specific contact points on the PCB surface. Most PCB designs are double-sided, while most bareboard test machines are single-sided, which means they have only one bed of nails. Newer two-sided testers can probe both sides of the PCB simultaneously (a bed of nails and separate test fixtures for each side). Instead of using double-sided testers, clamshell test fixtures may be used, which permit both sides of the board to be tested using a single-sided tester. To avoid the cost of clamshell testing, the board may be tested twice, once per side.

In-Circuit Testing

In-circuit testing verifies that the board and components function together as intended, and occurs after board assembly. During in-circuit testing, test engineers require access to each signal by probing a pin or routing via from the bottom side of the board. As in bare-board testing, in-circuit testing requires a test fixture that serves as a receptacle for the probe pins. Yet unlike bareboard testing, the fixture does not adapt a bed of nails to the PCB surface. The pins in the fixture correspond directly to the test locations on the PCB and are custom wired through cable connectors to the test box as no bed of nails exists. The PCB sits on the test fixture bottom side down in a vacuum-sealed enclosure.

Evaluating Testability

Early in the design process, evaluate the testability factors involved in the type of test under consideration, working with the board manufacturer to assess their test environment and data you can supply. Your test engineering group can provide additional in-circuit test information.

For bareboard tests, using two test fixtures (one for each side), place through-hole and most SMT parts so that all pins are on a reasonable test grid that maximizes part-to-part clearances and reduces the density of probe points per square inch. Top-side surface-mount-technology PCBs should be probed from the top, bottom-side and all through-hole parts, from the bottom.

For in-circuit tests using one fixture for the bottom-side only, consider mechanical obstructions or restrictions. Mechanical obstructions in a design may prevent you from adding test points. To accommodate these restrictions, define probe keepouts on the top or bottom of the board in layout mode. To control the proximity of test vias to surrounding tall parts, build a probe keepout area into the library package definitions in symbol mode, using the MANUFACTURING/NO_PROBE_TOP or NO_PROBE_BOTTOM layers of the symbol drawing. Control the clearance from test vias to surrounding conductive areas (wiring, vias, pads, and so on) from the *Spacing Rule* dialog box.

Assigning Test Preparation Properties

During the testprep process, you can assign the NO_TEST, TESTPOINT_QUANTITY, and TESTPOINT_ALLOW_UNDER testprep-related properties on nets or symbols from within the manual testprep environment using *Manufacture – Test Prep – Manual* (`testprep manual` command) and clicking the *Properties* button in the *Options* tab or by choosing *Manufacture – Testprep – Properties* (`testprep properties` command). You can also use *Edit – Properties* (property edit command) to assign testprep-related properties.

To exclude a net from being processed, attach the NO_TEST property to it. Specify the desired number of test points on the net using the TESTPOINT_QUANTITY property.

The `TESTPOINT_ALLOW_UNDER` property can be specified on a symbol to allow test points underneath all its component instances even when the *Allow Under Component* field may be set to *Never*. For example, you may attach this property to a large mechanical component, such as a heat sink on the bottom side, that occupies considerable real estate yet is not inserted until after testing finishes.

The `TESTPOINT_MAX_DENSITY` property can be specified on a symbol to verify the maximum test point allocation beneath a component instance of a symbol but on the opposite side to that on which the component/symbol is placed. This property works in conjunction with the *Component Area Check* on the Testprep Density Check dialog box, available by choosing *Manufacture – Testprep – Density Check* (testprep density command). For example, if a 2000-pin BGA occurs on layer TOP, the layout editor checks for a maximum test point allocation on layer BOTTOM, but within that component's place-bound region.

The *Component Area Check* considers components with multiple place-bound areas on a side. While addressing multiple areas, the *Component Area Check* optionally uses ASSEMBLY data depending on the *Component Representation* setting on the General Parameters tab. ASSEMBLY data is used if it is a SHAPE or RECTANGLE entity, or a single multi-segment LINE entity that forms a closed shape. ASSEMBLY data resembling a rectangle, but actually comprised of four different LINE entities, is not used. Arcs are recognized in a SHAPE or LINE entity.

A test point within multiple areas of the component counts once toward the density check. In the log file, the test point appears multiple times, once for each area that it is within.

See *Manufacture – Testprep – Properties* (testprep properties command) in the *Allegro PCB and Package Physical Layout Command Reference* for procedural information.

Defining Probe Keepout Areas

Testprep lets you define probe keepouts, which are areas in which test points cannot be generated. You can create a probe keepout in the layout, or in a library symbol. For procedural information, refer to *Setup – Areas – Probe Keepout* (keepout probe command) in the *Allegro PCB and Package Physical Layout Command Reference*.

Setting Test Preparation Parameters

You use the *Testprep* Parameters dialog box to determine the output of the automatic or manual testprep process by setting parameters, available by choosing *Manufacture – Testprep – Automatic* (`testprep automatic` command), or *Manufacture – Testprep – Resequence* (testprep resequence command) or *Manufacture – Testprep – Manual* (`testprep manual` command) and clicking the *Parameters* button.

> You can also access the Testprep Parameters dialog box by choosing *Setup – Design Parameters* (`prmed` command), then clicking *Edit testprep parameters* under the *Mfg Applications* tab, or use *Manufacture – Testprep – Parameters* (testprep prmed command).

In-circuit test engineering limits the maximum number of test probes that can be placed within a defined area or under a specific component. Dense test probe placement can damage PCBs.

To verify the test point density within user-definable unit areas, choose *Manufacture – Testprep – Density Check* (testprep density command). You specify the maximum number of test points allowed per unit area in the *Max testpoints per Unit Area* field. Exceeding this value creates rectangular figures that correspond to the user-defined unit areas and overlay the PROBE_DEN_TOP and PROBE_DEN_BOTTOM subclasses of the MANUFACTURING class. The layout editor automatically creates or clears these subclasses as required to verify the test point density within the areas of violation.

The `testprep_density.log` file flags all unit or component areas that exceed the allowable maximum as violations. Click on any hyperlinked x/y coordinates in the report to center that location in the display.

Sample Testprep_density.log **File**

```
(------------------------------------------------------------)
(        Testprep Density Check Log File                     )
(                                                            )
(        Drawing          : exported.brd                     )
(        Software Version : 15.5B1                           )
(        Date/Time        : Thu Apr 14 14:17:22 2005         )
(                                                            )
(------------------------------------------------------------)
Clearing existing PROBE_DEN_TOP subclass ...
Clearing existing PROBE_DEN_BOTTOM subclass ...
Unit area density check settings ...
  Unit area square size:                     1000 MIL
  Unit area square displacement:              500 MIL
  Max testpoints per unit area:                    30
Performing testpoint unit area density check on TOP side ...
Performing testpoint unit area density check on BOTTOM side ...
  Unit area ((3264.50 665.30) (4264.50 1665.30)) centered at (3764.50 1165.30) ... 47 ... VIOLATION.
    Testpoint TP341 at (4099.71 977.52).
    Testpoint TP339 at (4099.71 938.15).
NOTE: Component density checking using component assembly data ...
Performing testpoint component area density check on TOP side ...
Performing testpoint component area density check on BOTTOM side ...
Component U22 located at (7519.00 4285.00) ... 76 (Maximum 1) ... VIOLATION.
    ... area located at (7239.47 4580.28)
    ... area located at (7184.35 3950.35)
    Testpoint TP593 at (7282.78 4048.78).
    Testpoint TP547 at (7204.04 4048.78).
Testpoint TP179 at (7833.96 4442.48).
  Component U77 located at (5719.00 6710.00) ... 46 (Maximum 1) ... VIOLATION.
    ... area located at (5439.47 7005.28)
    ... area located at (5384.35 6375.35)
    Testpoint TP391 at (5482.78 6473.78).
```

Automatically Generating Test Points

Automatic testprep scans each net in the design, compares the pins or vias within each net against the specified parameters, and selects test sites. If no suitable pins or via is found, no testpoint is generated although a new via may be generated (surface-mount pad only). To exclude a net from processing, attach the NO_TEST property to it.

The layout editor determines legal probe sites using the following hierarchical path, beginning with I/O pins.

- If no suitable I/O pin satisfies the parameters you set, then the layout editor selects an IC output pin.
- If no output pin is available, then the layout editor considers any suitable pin as inputs or discrete pins. The layout editor flags these types of pin selection with asterisks in the log file.
- If the layout editor finds no legal pins, all pre-existing vias are examined.

If all these criteria fail, then the layout editor issues a message if testability cannot be achieved. All chosen points are marked with a test point symbol (as defined on the *Probe Types* tab of the Testprep Parameters dialog box) on the MANUFACTURING/PROBE_TOP or PROBE_BOTTOM layers of the layout drawing. The net name or probe number may also be included with each test point choice.

Testprep generates a data file of X/Y coordinates used to drill holes into the test fixture. These coordinates correspond to the test marker locations on the board. Probe pins are inserted that extend through the fixture and contact the surface of the board at a specific location (component pin or via).

Resequencing Test Points

Even though testpoints may have sequential refdes text assigned (such as *TP1, TP2, … TP<n>* when you enable *stringNumeric* in the *Display* field in the Testprep Parameters dialog box), *TP1* may be on one side of the board, *TP2* on the other side etc., such that visually the test points do not appear sequentially.

You can use *Manufacture – Testprep – Resequence* (testprep resequence command) to rename the refdes text of test points to ensure a visually sequential appearance, sorted by X/Y location from left to right and bottom to top on each side, starting with the TOP side first and then the BOTTOM side.

Initially you may create test points with certain *Probe Types* enabled on the *Probe Types* tab of the Testprep Parameters Dialog Box, which is available by choosing *Manufacture – Testprep – Parameters* (testprep prmed command). Then you may subsequently change vias that may be test

points, and need to resequence probe types due to violations. Test points may be too close for the probe type. Or a probe type of 100 may be more appropriate for a test point initially assigned a probe type of 75 given its proximity to another test point, which has since been deleted or moved.

You can use resequencing to detect invalid probe types and resequence them to valid probe types. If a valid probe type does not exist, such a test point can remain as is, but appears as TOO CLOSE in the Testprep report, Testprep Manual Query, or in the query available by choosing *Display – Element* (show element command). The layout editor flags it as an invalid probe type and it appears on the PROBE_TOP/BOTTOM subclasses with the cross (+) figure reserved for such test points.

Evaluating Testprep Results

You can evaluate the results of running testprep using the Testprep Report or the `testprep.log` file.|

This report organizes data regarding the test point coverage of a design, highlighting untestable nets, as well as the percentage coverage, number of nets covered, number of test points, and number/percentage of test points on top/bottom sides. It outlines:

- Nets currently being tested, subdivided into sections for the top and bottom sides
- Untestable nets
- Nets currently flagged with the NO_TEST property

If the test point is on a pin, the *Type* column indicates this as well as the refdes and pin number associated with the pin. An example is Pin (U1.1). Pad Size for a test point is the minimum dimension of the pad's bounding box.

Testprep General Analysis

In addition, the Testprep Report contains a section on testprep general analysis that appears after the report header, which compares coverage between an existing committed test fixture and current coverage, as the following example illustrates.

```
|=============================================================================|
      Testprep Report
     Thu Apr 14 14:37:29 2005
    /hm/pwright/boards/exported.brd



NOTE: An '*' in column 1 of the QUANTITY field for the TESTPOINT_QUANTITY net property value
indicates that the net does not have the required number of testpoints.
A blank QUANTITY field indicates that the net does not have TESTPOINT_QUANTITY property set on it.
|=============================================================================|
Testprep General Analysis ...
  Total number of nets                                ...   1244
  Total number of nets tested                         ...    649
  Total number of nets not tested                     ...    595
  Total number of nets flagged with NO_TEST property  ...      0
  Total number of nets testable (tested + not tested) ...   1244
  Percentage of all nets tested                       ...  52.17 percent
  Percentage of testable nets tested                  ...  52.17 percent
```

```
Nets requiring more than one testprobe:
  (None)
Total number of testprobes on TOP    side           ...      0 (  0.00 percent)
Total number of testprobes on BOTTOM side           ...    649 (100.00 percent)
Total number of testprobes on pins                  ...    198
Total number of testprobes on vias                  ...    451
WARNING: There are    649 testprobes with no assigned probe type.
Minimum pad size for probing                        ... 0 MIL
```

The `Nets Requiring More Than One Testprobe` section details nets with a `TESTPOINT_QUANTITY` property greater than 1, which is the first number shown, followed by the actual number of test points currently on the net.

Other messages include:

`WARNING: There are <n> testprobes with no assigned probe type`	Indicates test points created without active probe types in the Probe Types tab of the Testprep Parameters dialog box or created prior to release 15.5, where the design was upreved to release 15.5.
`WARNING: There are <n> testprobes with the TOO-CLOSE probe type.`	After initially creating test points with particular *Probe Types* enabled, if you resequence without enabling the *Delete probes too close* field, *<n>* test points may be too close for the probe type based on the required spacing and could not be resequenced to a valid type. For instance, a probe type of 100 may be more appropriate for a test point initially assigned a probe type of 75 given its proximity to another test point, which has since been deleted or moved.

The Testprep Report generated by *Tools – Reports* (reports command) provides additional information, which reflects the value of any TESTPOINT_PROBE_TYPE property attached to each reported test point. A *Probe Type* column in the report shows any assigned probe type for that test point.

When you run testprep, the layout editor generates an ASCII file called testprep.log, which summarizes the most recent execution of the testprep program. It lists all parameters, net names, and pin numbers for all test points. Other statistics are warnings, fails, completions, location (top or bottom), ignores (no test nets), and failure reasons. To access this file, click View Log on the Testprep Automatic dialog box or choose File – Viewlog.

Sample Testprep Log File

```
(-------------------------------------------------------)
(        Testprep Automatic Log File                    )
(                                                       )
(        Drawing          : replace_existing_via.brd    )
(        Software Version : 15.5A2                      )
(        Date/Time        : Wed Mar 30 13:30:53 2005    )
(                                                       )
(-------------------------------------------------------)
Probes accessing both sides of the board.
 No restrictions on pad type.
 Pin type restricted to 'VIA'.
 Minimum pad dimension is : 0
 Test pad grid: X: 0 Y: 0
 Minimum test point spacing is : 0
 Test points allowed under components due to bare board testing.
 Component representation is ASSEMBLY data.
 Bare board switch on, test points allowed on component pins on tested side.
 Tested every satisfactory pin.
 No unused pins tested.
 Testing allowed directly on pin.
 No direct trace testing allowed.
 Automatic test point insertion disabled.
 Test pad displacement: Min: 0, Max: 100
 Via replacement allowed.
 Cline bubbling allowed.
 'Padstack Selections' settings are:
      TOP Side      BOTTOM Side
      --------      -----------
SMT Testpad         SMD25_50     SMD25_50B
Thru Via         PAD35CIR25D     PAD35CIR25D
 'Padstack Selections' replacements are:
Existing Via         TOP Side     BOTTOM Side
------------         --------     -----------
VIA1         VIA1_A     VIA1_A
VIA         VIA_A     VIA_A
DISABLED BLVIA5_BOT          BLVIA5_BOT_A
DISABLED BLVIA1_2          BLVIA1_2_A
 Executed in INCREMENTAL mode.
 *** WARNING *** Pre-existing test points may not fit parameters.
 No text displayed on targets.
 Probe text angle: 0 degrees.
 Probe text offset: X: 0 Y: 0
```

```
  NET        PIN                              PROBE TEXT
================================================================
Processing net VCC (1 out of 73),(c=0, f=0, i=0).
 Net VCC NOT ACCESSIBLE from either side.
    (non-via=10)
Processing net TN-9 (2 out of 73),(c=0, f=1, i=0).
 Net TN-9 NOT ACCESSIBLE from either side.
    (non-via=2)
Processing net TN-8 (3 out of 73),(c=0, f=2, i=0).
  TN-8       VIA             @(  4650,  6600)              (bottom)(pre-existing)
  TN-8       VIA             @(  4650,  6500)              (bottom)(pre-existing)
  TN-8       VIA             @(  4275,  6500)              (bottom)(pre-existing)
  TN-8       VIA             @(  3625,  7500)              (bottom)(pre-existing)
(non-via=2)
```

Failure Reasons

The `testprep.log` file may list the following failure reasons.

Net *<name>* NOT ACCESSIBLE from bottom side. (pad undef=N, no pin-escape=N, non-routable=N	N indicates the number of times a potential pin or via on the net could not be made into a test point for the specified reason. pad undef = N: N pins failed as no pad exists on the bottom side no pin-escape = N: N pins failed an existing pin escape could not be found non-routable = N: N pins failed as pin escape could not be inserted
too small	Pin/via pad doesn't meet *Minimum Pad Size*.
under comp	Pin/via pad is under a component where disallowed.
under pin	Test point on the pin on the same side of the board that the component is placed.
off grid	Pin/via is not on an enabled test grid.
pad undef	Padstack has no pad defined on the layer to be tested.
non-smd	Pin/via is not a surface-mount device (SMD); pad stack type is set to SMT.
non-thru	Pin/via is SMD; pad stack type is set to Thru.

via	Via encountered for consideration, but specified Pin type doesn't allow vias.
non-via	Pin encountered for consideration, but'Pin type' doesn't allow pins.
non-output	Pin is not an output pin, Pin type set to Output
non-i/o	Pin is not an IO pin, Pin type set to Input
no pin-escape	No existing() pin escape found on a pin
non-routable	Pin escape could not be routed for a pin
too-close	Test point would create a DRC error, OR is too close to another test point as per Min spacing
not-placed	Unplaced pin encountered
not-node	Pin/via is NOT a node, Test method set to Node
out-of-bounds	Pin/via is within a NO_PROBE area
pad-mismatch	Via replacement padstack for Replace vias does not match the padstack of the via under consideration, such that connectivity would be affected
rep-cause-drc	Replace vias is enabled, and replacement of the via pad is detecting the existence of a DRC error(s), either with or without cline bubbling
pad-cause-drc	Making a pin pad or a via pad (without replacement), a test point as is, detects the existence of DRC errors.

Modifying Test Points Interactively

After you generate test points automatically, you can modify them interactively using *Manufacture – Testprep – Manual* (testprep manual command). Prior to using the options outlined below, you may use the *Route – Connect* (add connect command) or *Route – Slide* (slide command) interactive etch editing tools with the appropriate bubble options to permit additional test points.

- *Add* allows a test point to be added to an existing via, pin, or cline
- *Add (Scan and Highlight)* lets you review all untested nets, and each net is highlighted and fits into the window screen
- *Delete* removes the test point symbol from an existing test point via or pin and also reverts the test via to the original via padstack name if the *Replace Via* option was used. It also removes the physical via itself if it resulted from adding a test point to a trace, or from testprep automatic pin escape insertion. Removing a pin escape via removes the associated routing as well.'
- *Swap* exchanges the test point location to another location on the same net
- *Query* provides net, location and symbol property information associated with the test point pin and permits limited querying of PROBE_DEN_TOP/BOTTOM subclass density violation areas.

Fixing/Unfixing Test Points

You can globally fix all test points to prevent further editing or automatic removal of existing test points using *Manufacture – Testprep – Fix* (testprep fix command).

Generating the FIXTURE Subclasses

When the layout editor creates test points, the MANUFACTURING class PROBE_TOP and PROBE_BOTTOM subclasses capture test point locations; however, they change accordingly as you modify test points. A symbol figure (as defined on the *Probe Types* tab of the Testprep Parameters dialog box) appears at the test point location, along with test point identification text. These locations are subsequently used to generate NC drill files for test fixture fabrication for each side of the board, as required.

As test fixtures are expensive, reusing them is desirable. Prior to revisions, you can choose *Manufacture – Testprep – Create FIXTURE* (testprep create fixture command) *to create* the static FIXTURE_TOP and FIXTURE_BOTTOM subclasses to copy PROBE_TOP and PROBE_BOTTOM subclass information to them. The FIXTURE_TOP and FIXTURE_BOTTOM subclasses maintain the information regardless of what test points are added, deleted, or moved during design revisions.

By comparing the current PROBE subclass to the static FIXTURE subclass, you can tailor the revisions to synchronize current test point locations with the original FIXTURE locations. NC drill file generation only uses the PROBE subclass information.

Information on Test Point Markers

Although markers (as defined in the *Figures* column on the *Probe Types* tab of the Testprep Parameters dialog box) appear with enabled PROBE_ subclasses to indicate that the pin or via is a testpoint, these markers do not actually exist on the PROBE subclass. (The circle symbol does exist on the FIXTURE subclass. However, as no association exists in the database between the circle symbol and the text on the FIXTURE subclass, you must be cautious when editing the FIXTURE subclass information.)

Consequently, with only the PROBE_ subclasses displayed, you cannot choose test point markers and use *Display – Element* (show element command) to obtain information about them. Visibility for vias or pins must be enabled as the information given is for the test point pin or via itself.

When you initially create a FIXTURE subclass or update an existing one, the color visibility for the subclass is enabled to highlight the results immediately. Set colors and color priorities for the PROBE and FIXTURE subclasses to subsequently superimpose the symbol figures (as defined on the *Probe Types* tab of the Testprep Parameters dialog box) on the circles.

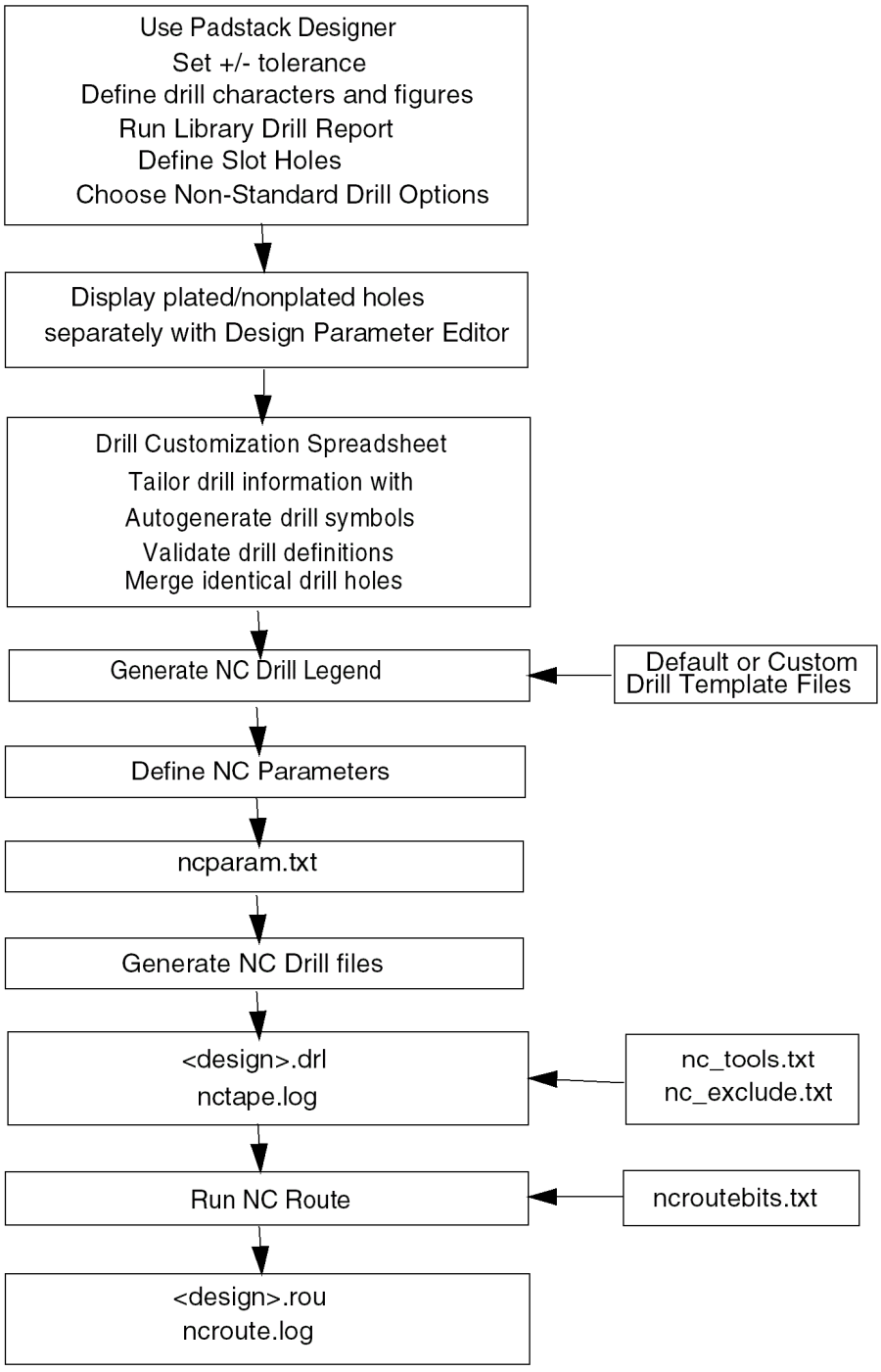
- Symbol figure superimposed on circle means current and old test point location match.
- Circle only means no test point currently exists at an old test point location.
- Symbol figure only means a test point currently exists at a location that had none.

Create NC Drill File or IPC356

3

Creating Numerical Control Data

When you are ready to manufacture your design, the layout editor can generate output files for numerically controlled (NC) drills and routers. These files let you create data in Excellon format.


Use Padstack Designer
Set +/- tolerance
Define drill characters and figures
Run Library Drill Report
Define Slot Holes
Choose Non-Standard Drill Options
Display plated/nonplated holes
separately with Design Parameter Editor
Drill Customization Spreadsheet
Tailor drill information with
Autogenerate drill symbols
Validate drill definitions
Merge identical drill holes
Generate NC Drill Legend
Default or Custom
Drill Template Files
Define NC Parameters
ncparam.txt
Generate NC Drill files
<design>.drl
nctape.log
nc_tools.txt
nc_exclude.txt
Run NC Route
ncroutebits.txt
<design>.rou
ncroute.log

Prior to Generating NC Drill Output Files

Prior to generating output files, you can specify information in Padstack Designer that can make preparing your design for manufacturing significantly easier, such as specifying positive/negative tolerance with non-symmetrical values, such as +0.001/ -0 .005, which typifies a via hole of 5 mils, as vias usually have a positive/negative tolerance that differs from that of component pin through holes.

By associating drilled hole tolerance at the padstack level, this information propagates from the library to both drill legend and drill file output. You can also more clearly delineate the type of hole to be used, including a rectangular, square, or oblong slot hole, at the padstack level. For custom applications in which you must override library data, you can modify drilled hole tolerance at the design level with the *Drill Customization* spreadsheet by using *Manufacture – NC – Drill Customization* (`ncdrill customization` command).

You can define a drill size using up to three characters, which creates significantly more combinations of potential drill symbols. These values may also be customized using the Drill Customization spreadsheet. Automatic drill symbol assignment alleviates library-based dependencies.

You can choose whether to display plated drill holes in your design using the *Display* tab of the Design Parameter Editor, available by choosing *Setup – Design Parameters* (`prmed` command). Plating hole visibility is often not required due to the volume of entities and supporting geometries, such as etch/conductor layer pads. You may want to make non-plated holes visible during placement and routing because they have no padstacks associated with them on etch/conductor layers and are normally invisible. As a result, they may lack proper keepout areas to guide you.

You can also change the default file extension of `.drl` for NC drill output filenames by setting the `ext_drill` environment variable in the User Preferences Editor, available by choosing *Setup – User Preferences* (`enved` command). Cadence recommends changing the extension at the CDS_SITE level to ensure your company uses a common extension. Verify that downstream tools can support the extension you choose.

NC Drill Legends

During the last phase of PCB design, drill legend tables are created that usually appear on a fabrication drawing quantifying the number, type, and tolerance of plated and non-plated holes. It is used to help establish PCB costing. These drawings are automatically created when you run *Manufacture – NC – NC Drill Legend (*`ncdrill legend` command). The layout editor automatically generates or replaces all drill legends for all layers and drill figures in the design whether or not the layers are visible.

NC Parameters File

You then create a parameter file using the *NC Parameters* dialog box, available from *Manufacture – NC – NC Parameters* (`ncdrill param` command). The `nc_param.txt` parameter file specifies the drill coordinate data format and serves as input when generating an NC drill or route file as well as drill legends.

When you execute *Manufacture – NC – Drill Legend,* if the drilling type specified on the NC Drill dialog box differs from that specified on the Drill Legend dialog box, `nc_param.txt` synchronizes them. If `nc_param.txt` does not exist, or contains no drill types, `nctape_full` and `ncdrill legend` scan existing drill legend subclasses to determine the default drilling types specified on the dialog boxes. Otherwise, the default is layer pair drilling.

Related Topics

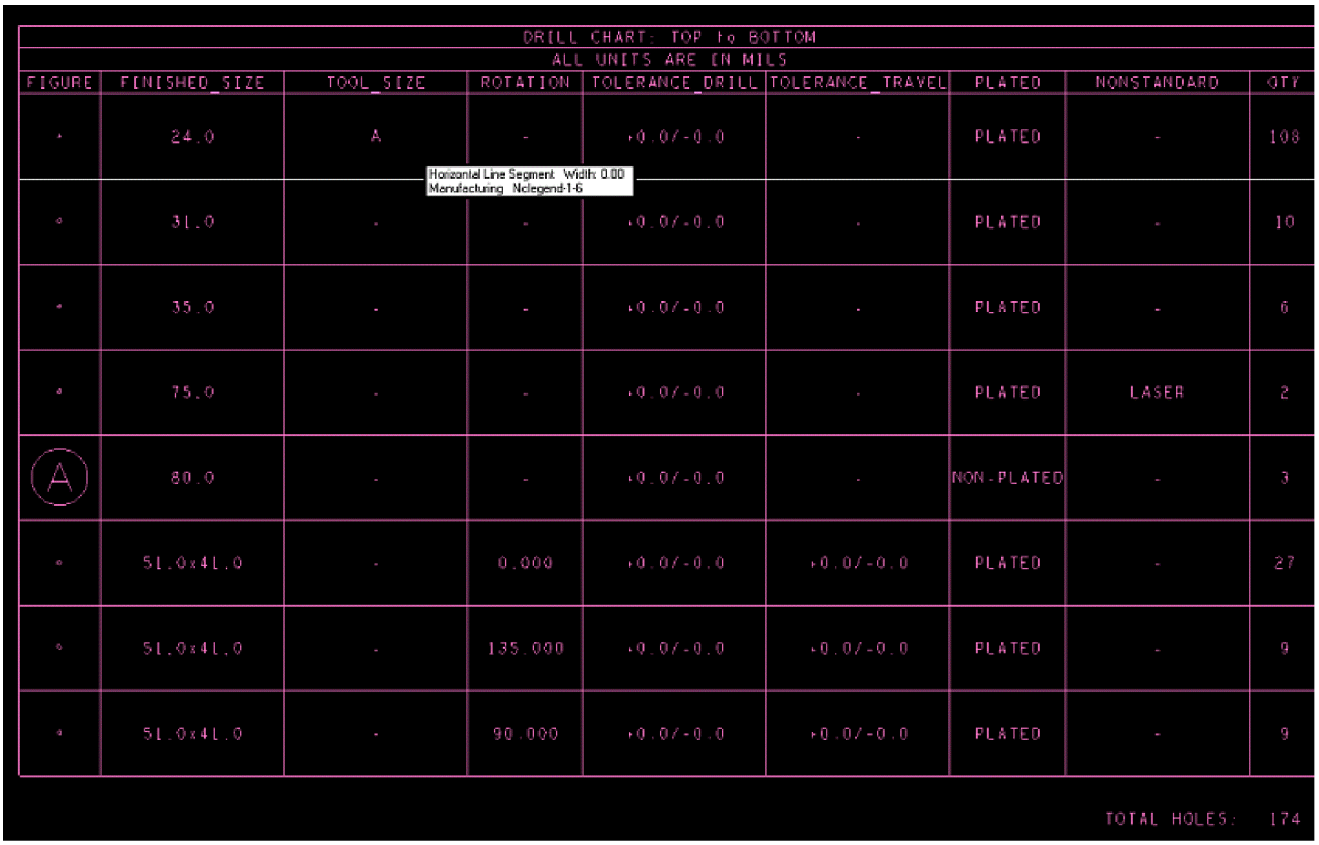
- ncdrill customization
- prmed
- enved
- ncdrill legend
- ncdrill param

Generating Drill Legends

The *Manufacture – NC – Drill Legend* (ncdrill legend command) enables you to create drill drawings. Over each hole to be drilled in the design, the layout editor displays a drill figure with optional drill characters inside. This combination of figure and characters represents both the drill size and the hole-plating information (defined in the padstack). This command also creates drill legends on the design.

The following figure shows a sample legend. Use the Padstack Designer to define the drill figure and characters for each hole size and plating type combination as part of the pin and via padstack data. See Library Padstacks.

NC Drill Legend Format



DRILL CHART: TOP to BOTTOM
ALL UNITS ARE IN MILS

FIGURE	FINISHED_SIZE	TOOL_SIZE	ROTATION	TOLERANCE_DRILL	TOLERANCE_TRAVEL	PLATED	NONSTANDARD	QTY
◦	24.0	A	-	+0.0/-0.0	-	PLATED	-	108
◦	31.0	-	-	+0.0/-0.0	-	PLATED	-	10
◦	35.0	-	-	+0.0/-0.0	-	PLATED	-	6
◦	75.0	-	-	+0.0/-0.0	-	PLATED	LASER	2
Ⓐ	80.0	-	-	+0.0/-0.0	-	NON-PLATED	-	3
◦	51.0x4L.0	-	0.000	+0.0/-0.0	+0.0/-0.0	PLATED	-	27
◦	51.0x4L.0	-	135.000	+0.0/-0.0	+0.0/-0.0	PLATED	-	9
◦	51.0x4L.0	-	90.000	+0.0/-0.0	+0.0/-0.0	PLATED	-	9

TOTAL HOLES: 174

Using Drill Template Files

Cadence supplies template files to control the drill legend table format and let you customize its appearance. The drill template file specifies the number and order of columns, column titles, and custom data for each tool size in the design.

The default `.dlt` drill legend template files include the layer names in the legend table title line. The width of the *Size* and *Tolerance* columns allow for slot holes and positive and negative tolerance, respectively.

Because drill template files contain unit-specific information, Cadence provides five different templates, one for each type of unit:

Design Units	Default Template Filename

mils	`default-mil.dlt`
inches	`default-in.dlt`
mm	`default-mm.dlt`
microns	`default-um.dlt`
cm	`default-cm.dlt`

Directories defined by the NCDPATH environment variable, which points to <*Allegro_install_dir*>`/share/pcb/text/nclegend`, store default template files. The default template files are ASCII text files you can edit with any text editor. The following example shows the `default-in.dlt` file.

```
(make__ncTemplate_struct
?Name                  "default-in"
?Units                 "inches"
;   ?TitleHeight
;   ------------
;
;   Height of the main title line row in units as indicated by ?Units.
;   This height must take into account a title that may in fact be multi-line.
;
 ?TitleHeight           0.25
;
;   Text block id numbers to be used for the actual title text, the column header
;   text, and text appearing in each drill data row, respectively.
;
 ?TitleTextBlock        9
 ?ColumnTitleTextBlock  9
 ?DataTextBlock         9
;
;   Order of holes in the legend table based on plating status.
;
;     "PlatedFirst" or "NonPlatedFirst"
;
 ?PlatingOrder          "PlatedFirst"
;
;   ?Title
;
;   ?Title              Title for 'Layer pair' drill legends, and default title
```

```
;                       for other drill legend types when not specified otherwise.
;
;   ?TitleByLayer       Title for 'By layer'  drill legends.
;
;   ?TitleBackdrill     Title for 'Backdrill' drill legends.
;
;   The appearance of the string "$lay_nams$" indicates where "<layer name> to <layer name>"
;   may appear in a title, while "$lay_nums$" indicates where "<layer number> to <layer number>"
;   may appear in a title.
;
;   The appearance of '|' characters in the title string will cause subsequent text
;   in the title to start on a new line. It is up to the user to change the ?TitleHeight
;   specification accordingly, as it will not automatically be changed.
;
 ?Title                 "DRILL CHART: $lay_nams$"
 ?TitleByLayer          "BY LAYER: $lay_nams$"
 ?TitleBackdrill        "BACKDRILL: $lay_nams$"
;
;   ?UnitsTitle
;   -----------
;
;   The units title line appearing in the 2nd header row.
;   The appearance of the string "$units_id$" in the title line indicates
;   where the actual units identifier is to be located in the string.
;
 ?UnitsTitle            "ALL UNITS ARE IN $units_id$"
;
;   ?SuppressHeaderRow<n>
;   ---------------------
;
;   The first three header rows of each legend title are respectively the above
;   title line, followed by the fixed units line, and then the column headers line.
;   All or any combination of these header rows can be suppressed from appearing
;   with the following options. The default setting is "no", with "yes" indicating
;   the header row is to be suppressed.
;
 ?SuppressHeaderRow1    "no"
 ?SuppressHeaderRow2    "no"
 ?SuppressHeaderRow3    "no"
;
;   ?Precision
;   ----------
;
```

```
;   Precision specifies the number of decimal digits to follow the
;   decimal point in size and tolerance values appearing in the table.
;
;   -1 = as many as are needed ... minimum 1 (DEFAULT)
;    0 = suppress decimal point and decimal digits
;   >0 = always put <n> decimal digits after the decimal point,
;        adding any trailing 0's as necessary
;
 ?Precision             -1
;
;   ?SuppressLeadingZero
;   --------------------
;
;   SuppressLeadingZero specifies specifies that any leading zero before
;   the decimal point of a decimal value for a size or tolerance will be
;   suppressed.
;
;     e.g. '0.100' --> '.100'
;
;   The default setting is "no", while "yes" means suppress the 0.
;
 ?SuppressLeadingZero   "no"
;
;   ?FixedFigureHeight
;   ------------------
;
;   A non-zero FixedFigureHeight will force all figures displayed
;   in the "Figure" column to be displayed with the same specified
;   height, overriding what is defined in individual padstacks. The
;   setting of 0 included below indicates that they will be displayed
;   with the sizes as defined in the padstacks.
;
 ?FixedFigureHeight     0
;
;   ?TableSlotRotations
;   -------------------
;
;   The legend table will normally show separate legend table entries
;   for each rotation found for a slot hole whose figures are otherwise
;   identical. By specifying "no" for TableSlotRotations, the figure
;   shown for the slot hole in the table will have a 0 rotation
;   (e.g. major dimension along the X axis), and the quantity shown
;   will be the sum of ALL rotations for that slot hole figure. It
```

```
;   should be noted that the main figure display for the individual
;   slot figure instances will always show the proper rotation,
;   regardless of what this setting is for the figure in the table.
;   The default setting is "yes".
;
 ?TableSlotRotations    "yes"
;
;   ?ForceToleranceColumn
;   ---------------------
;
;   The "Tolerance" column specified in the ?ColumnDefinitions
;   will be automatically suppressed if it is found that all holes
;   have 0 tolerances. The column can be forced to appear anyway with
;   a setting of "yes" for ?ForceToleranceColumn. The default setting
;   is "no".
;
 ?ForceToleranceColumn  "no"
;
;   ?ShowTotalQuantity
;   ------------------
;
;   The total hole quantity can be requested to appear with the legend
;   table with a setting of "yes" for ?ShowTotalQuantity. The default
;   setting is "no". The quantity appears at the bottom of the "Quantity"
;   column, just outside of the bounding rectangle for the legend table.
;   The numeric quantity is prefixed with the text "TOTAL HOLES:".
;
 ?ShowTotalQuantity     "no"
;
;   ?SeparateSlotHoleLegend
;   -----------------------
;
;   A legend can be created for slot holes that is separate from the legend
;   for the drilling of normal thru holes. The legend is controlled by the
;   MANUFACTURING class NCLEGEND-SLOTS subclass and the legend table graphics
;   group name is NC_LEGEND_SLOTS. A setting of "yes" will create the separate
;   legend, with the default setting being "no".
;
 ?SeparateSlotHoleLegend  "no"
;
;   ?RowHeightExpansion
;   -------------------------------
;
```

```
;   The legend row height will at minimum be the maximum figure height in the
;   legend table expanded by a percentage of that figure height. The default
;   expansion is 50 percent. An expansion of 0 percent could then result in
;   the figure of maximum height touching the top and bottom of its row.
;   The text height used in the rows also factors in to the row height
;   calculation, so this option may in fact have no effect on the height if
;   the text height is larger.
;
 ?RowHeightExpansion 50
;;
;   ?AlternateUnits
;   ---------------
;
;   If ?AlternateUnits is specified, and also "Holesize2", or "Tolerance2", or
;   "HoleSizenTol2" in ?ColumnDefinitions, the columns will appear similar to
;   the standard columns, but in the units specified by ?AlternateUnits.
;   The permissable settings are:
;
;     "mils"  "inches"  "microns"  "millimeters"  "centimeters"
;
;   e.g. ?AlternateUnits "millimeters"
;
;   Where the columns appear and the column header text is left up to the user
;   as usual in ?ColumnDefinitions. How the units of the columns are indicated
;   are also left up to the user. Possibly the column header text (e.g. "SIZE (MM)"),
;   or with an appropriate ?DrillNotes/?BackdrillNotes line as below.
;
;   ?DrillNotes
;   -----------
;   ?BackdrillNotes
;   ---------------
;
;   Lines of notes that are to appear below the legend table rectangle for
;   each drill legend that is generated. ?DrillNotes are for 'Layer pair' and
;   'By layer' legends, while ?BackdrillNotes are for 'Backdrill' legends.
;   Note text is allowed to be in upper and/or lower text, and there is no explicit
;   limit on the number of lines of text. The note text is specified as:
;
;     '(
;        "<line 1 of text>"
;        "<line 2 of text>"
;               .
;               .
```

```
;               .
;         "<line n of text>"
;       )
;
;   The default for each is "'()" which indicates there are no notes.
;
 ?DrillNotes     '()


 ?BackdrillNotes '(
                    "NOTES:"
                    "- DRILL SIZES LISTED IN LEGEND"
                    "  ARE CONSIDERED FINISHED."
                    "- VENDOR IS REQUIRED TO SELECT"
                    "  TOOLING FOR OVERDRILLING."
                    "- LEGEND DOES NOT SPECIFY DEPTH"
                    "  INTO ADJACENT DIELECTRIC LAYER."
                  )
;
?ColumnDefinitions
;   ------------------
; The first field of each definition uniquely identifies the column
;   to appear in the legend table, while the second field provides the
;   user-specified header text for the column. The third field controls
;   the width of the column.
;
;   Each column definition can have an optional 4th field included
;   as well to control the justification of the data displayed
;   within that column. The permitted values are:
;
;      "center", "right", or "left"
;
;   with "center" being the default if the 4th field is not provided,
;   or is provided but is not one of the above permitted values.
;
;   Other columns that can be specified:
;
;     "HoleSizenTol"   ... "Holesize" and "Tolerance" combined in one column
;     "Rotation"       ... If a slot hole, it's rotation, where 0 degrees is
;                          when the major axis of the hole is on the X axis.
;     "User"           ... User-defined column. See ?CustomData below.
;
;     "Holesize2"      ... See ?AlternateUnits.
```

```
;     "Tolerance2"
;     "HoleSizenTol2
;
 ?ColumnDefinitions '(
                       ("Figure"      "FIGURE"        7)
                       ("Holesize"    "SIZE"         15)
                       ("Tolerance"   "TOLERANCE"    15)
                       ("PlateStatus" "PLATED"       10)
                       ("NonStandard" "NONSTANDARD"  15)
                       ("Quantity"    "QTY"           6)
                     )
;   ?CustomData
;   -----------
;
;   A column definition in ?ColumnDefinitions above can have "User" appear
;   in the first field to indicate a column of user-defined data. The
;   data is specified by ?CustomData definitions that are matched to holes
;   appearing in the legend table.
;
;     ?CustomData '(
;                    ( 28          "Plated"     "<User 1 data>")
;                    ( 42          "Plated"     "<User 1 data>")
;                    ( 36          "Plated"     "<User 1 data>")
;                    ( 63          "Plated"     "<User 1 data>")
;                    (109          "Non Plated" "<User 1 data>")
;                    ((130 50    ) "Plated"     "<User 1 data>")
;                    ((130 50 "r") "Plated"     "<User 1 data>")
;                  )
;
;   Matching of data to a hole is done on the basis of the hole size in field 1
;   and the plating status in field 2. For a normal hole, the size field is
;   simply '<size>'. For matching to an oval slot hole, size must be
;   '(<major> <minor>)', with <major> being the major dimension of the oval,
;   and <minor> the minor dimension. For matching with a rectangular slot
;   hole the "r" must be added to size specification. The additional brackets
;   around the size specification for a slot hole MUST be included.As many
;   data strings should appear as there are "User" columns in ?ColumnDefinitions.
;   The order of the data strings is matched to the order of "User" columns
;   in ?ColumnDefinitions
;
)
```

Drill Template File Fields

A drill template file contains the following fields:

Name	Specifies the template file name used only for reference purposes and not by the program.
Units	Specifies mm, mils, inches, um, or cm units. Because a drill template contains unit specific data, (for example, the custom fields for each drill hole size), use the corresponding templates for each type of unit. The layout editor converts the values given in the template file to the design units before using them.
TitleHeight	Specifies the height of the drill legend title block. The height is in the units indicated in the *Units* field of the template.
TitleTextBlock	Specifies the text block number for the drill legend title. Must be a positive number between 1 and 64.
ColumnTitleTextBlock	Specifies the text block number for the drill legend column title. Must be positive number between 1 and 64.
DataTextBlock	Specifies the text block number for the drill legend table entries. Must be a positive number between 1 and 64.
PlatingOrder	Specifies whether to sort drill legend entries by plated or nonplated holes first.
Title	Specifies the title for *Layer Pair* drill legends and default title for other drill legend types when not otherwise specified.
TitleByLayer	Title for *By Layer* drill legends.
TitleBackdrill	Title for *Backdrill* drill legends. The appearance of the string lay_nams indicates where <*layer name*> to <*layer name*> may appear in a title; lay_nums, <*layer number*> to <*layer number*> may appear in a title. For example: *Title:* DRILL CHRT: lay_nams *TitleByLayer*: BY LAYER: lay_nams *TitleBackdrill*: BACKDRILL: lay_nams
UnitsTitle	The units title line appears in the second header row. The string $units_id$ in the title line indicates where the actual units identifier is to be located in the string. ALL UNITS ARE IN $units_id$"

SuppressHeaderRow<n>	Any combination of the above title, the fixed units, and the column headers rows can be suppressed from appearing by specifying `yes`. The default setting is `no`. For example: ? SuppressHeaderRow1 no ?SuppressHeaderRow2 no ? SuppressHeaderRow3 no
Precision	Specifies the number of decimal digits to follow the decimal point for use in size and tolerance values. For example: -1: add as many as are needed with a minimum of 1, which is the default. 0: suppress decimal point and decimal digits >0: add <*n*> decimal digits after the decimal point, plus any trailing zeros as necessary.
SuppressLeadingZero	Removes any leading zero before the decimal point of a size or tolerance decimal value. For example, *0.100* becomes *.100*. The default setting is *no*; *yes* suppresses the 0.
FixedFigureHeight	A non-zero value forces all figures to display with the same specified height, overriding the sizes defined in the padstacks. A zero value uses the padstack size definitions.
TableSlotRotations	Separates each rotation for slot holes with identical figures into its own entry in the legend table if you specify yes, which is the default. Otherwise, the figure shown has a 0 degree rotation (occurring when the slot hole's major axis exists along the X axis) and the quantity shown is a sum of all rotations for that slot hole figure. The main figure for the individual slot figure instances shows the proper rotation, regardless of this setting.

ColumnDefinitions	Specifies the details for each column in the drill legend. The first field uniquely identifies the column; the second, the user-specified header text; the third, the column width. Each column definition can have an optional fourth field to control the justification of the data displayed within that column. The permitted values are center, right, or left, with center default if the fourth field is not provided, or is not one of the above permitted values. A Rotation column can be optionally added to include slot holes' rotation. A 0 degree rotation occurs when the slot hole's major axis exists along the X axis. Each column has the following format: *Figure* (FIGURE) has the drill figures with a character limit of 7. *HoleSize* (SIZE) has the drill hole size with a character limit of 15. *Tolerance* (TOLERANCE) specifies the positive or negative tolerance associated with the tool. Excluded if all holes have a positive or negative tolerance of zero, even when specified in the `.dlt` file with a character limit of 15. *PlateStatus* (PLATED) specifies whether the hole is plated. with a character limit of 10. *NonStandard* (NONSTANDARD) specifies whether the hole is manufactured with a nonstandard drill method. Excluded if no hole has a non-standard drill setting, even when specified in the `.dlt` file with a character limit of 15. *Quantity* specifies the number of holes on the design with the specified size. Multiple drill holes in multiple-drill vias are counted individually with a character limit of 6.
ForceToleranceColumn	Hides the Tolerance column specified in *?ColumnDefinitions* if all holes have 0 tolerances (default), unless you specify *yes* to show it.
ShowTotalQuantity	Displays the numeric hole quantity prefixed with TOTAL HOLES: at the bottom of the *Quantity* column, outside the bounding legend table rectangle, if you specify *yes*. Otherwise defaults to *no*.
SeparateSlotHoleLegend	Creates a legend for slot holes, controlled by the MANUFACTURING class/NCLEGEND-SLOTS subclass, separate from the drill legend for thru holes if you specify yes. The legend table graphics group name is NC_LEGEND_SLOTS. Otherwise defaults to *no*.
AlternateUnits	If you specify a value here as well as one in Holesize2 or Tolerance2 in ?ColumnDefinitions, the columns resemble the those of Holesize and Tolerance, but in mils, inches, microns, millimeters, or centimeters. The ?ColumnDefinitions dictate column, location, header text and units.

DrillNotes/BackdrillNotes	Notes appear below the legend table with a rectangle for each drill legend in upper- or lowercase text, with unlimited text lines. ? DrillNotes are for Layer pair and By layer legends, ?BackdrillNotes are for Backdrill legends. The note text is specified as: <line 1 of text> <line 2 of text> <line n of text>" The default for each is "'()" which indicates there are no notes. ?DrillNotes '() ?BackdrillNotes '("NOTES:" "-DRILL SIZES LISTED IN LEGEND" "ARE CONSIDERED FINISHED." "-VENDOR IS REQUIRED TO SELECT" "TOOLING FOR OVERDRILLING." "-LEGEND DOES NOT SPECIFY DEPTH" "INTO ADJACENT DIELECTRIC LAYER.")
RowHeightExpansion	Expands the minimum height of a legend row by a percentage of the maximum figure height. The default expansion is 50 percent. Specifying zero percent might cause, for instance, the figure with the greatest height to touch the top and bottom of its row. If the text height used in a row is larger, this option may have no effect. For example: RowHeightExpansion 50
CustomData	Specifies all allowable hole size and plating status combinations that can occur in the design. It also specifies the user data for each of these holes. The layout editor goes through the given list of hole sizes and status to find a match for the hole existing in the design. It places the corresponding data in the User column of the drill legend table. The format for CustomData is: (<Holesize> "<PlateStatus>" "<userdata>")

> The number of user data fields must equal the number of user-defined columns in the *ColumnDefinitions* field.

You can use the Custom Data feature for maintaining data, for example, tolerance that should be added to each hole in a set of designs. The layout editor picks up the custom data only for the holes in the design, you can maintain a file that has all of the allowable sizes of drill holes and their custom data.

To get the custom data for a given hole size in the design, the layout editor uses the following method:

1. If the design units and the units in the template file differ, the layout editor converts the template file units into design units.

   > Roundoff may cause inaccurate results.

2. For each hole size in the drill legend, the layout editor tries match in the drill template file specified. This match is based on both the drill hole size and the plating status. If a match is found, the corresponding custom data is read and placed in the drill legend.

> The custom data is not converted in any way, even though units may differ, and the hole size was converted.

You can create a custom template file by copying a default template and renaming it to your custom legend. For example:

`cat default-in.dlt` becomes *<custom name>*`.dlt`

The following example shows a custom drill template specifying two additional user columns and changing the drill legend column sequence so that *Quantity* is the first column, *Figure* is second, *Holesize* is third, *Tolerance* is fourth, *Plating* is fifth, *NonStandard* is sixth, and two additional *User* columns are last.

```
;     Example custom dlt file
;
;
;     The following example shows a custom drill template specifying
;     two additional user columns and changing the drill legend column
;     sequence so that Quantity is the first column, Figure is second,
;     Holesize is third, Tolerance is fourth, Plating is fifth, and
;     two additional User columns are last.
;
;     The line "?Title" can use either variable $lay_nams$ to add the layer names e.g.
;     Top to Bottom or $lay_nums$ to add the layer numbers e.g. 1 to
;
;     For additional information regarding the data that can be manipulated
;     in .dlt files please refer to the Cadence documentation or the supplied
;     .dlt files located in $CDS_ROOT/shape/pcb/text/nelegend.


(make__ncTemplate_struct


 ?Name                 "custom_drill"
 ?Title                "Title $lay_nams$"
 ?Units                "mils"
 ?TitleHeight          250
 ?TitleTextBlock       5
 ?ColumnTitleTextBlock 3
```

```
 ?DataTextBlock         4
 ?PlatingOrder          "NonPlatedFirst"
 ?ColumnDefinitions     '(
 ;
 ;    You can add user defined columns below. The format must
 ;    follow the guide of "User", Column Entry that you want
 ;    displayed in the drill legend and the column width.
 ;    You can move the order the columns are displayed in
 ;    Allegro by moving the lines below.
 ;
 ;
                          ("Quantity"    "QTY"           6)
                          ("Figure"      "FIGURE"        7)
                          ("Holesize"    "SIZE"         15)
                          ("Tolerance"   "TOLERANCE"    15)
                          ("PlateStatus" "PLATED"       17)
                          ("NonStandard" "NONSTANDARD"  15)
                          ("User"        "OTHER"        20)
                          ("User    "  "OTHER1"       20)
                         )
;
;
;
;      Add any custom data that you would want to appear in the drill
;      legend below.
;
 ?CustomData `(
;
;      This is the column data used under each of the headings
;      above. In order for the data in the User column(s) to
;      be writen in the drill legend the hole must(!) match the
;      data in the "Drill" and(!) "Plating Status" columns below.
;
;               Drill   Plating        OTHER                  OTHER1
;                       Status          column                column
;
;
;
               (156     "Non Plated"    "Text or Numbers"         "B")
               (5      "Plated"     "1"                        "C")
               (18      "Plated"     "2"                       "8")
               (40      "Plated"     "3"                          ".001")
               (36      "Plated"     "A"                        "ABCdef")
```

)

)

Generating NC Drill Files

Using data contained in your `.brd` design, and parameters defined in the `ncparam.txt` file, the layout editor can output data for automatically generating NC Drill files.

You can create this data either with the user interface or in batch mode. Both methods are described under *Manufacture – NC – NC Drill* (nctape_full command) in the *Allegro PCB and Package Physical Layout Command Reference*.

If you choose, you can create the `nc_tools.txt` and `nc_exclude.txt` files with Notepad or another text editor.

Syntax for nc_tools.txt

You can optionally define an *nc_tools.txt* file that contains tool codes for automatic tool selection, specifying each drill size and the *Tnn* code to use for it. If your NC drill equipment supports automatic tool selection, you enable *Auto Tool Select* on the *NC Drill* dialog box to insert *Tnn* tool codes in the Excellon format output file, instead of *M00* stop codes for manual tool changing. The output file excludes drill sizes not in the *nc_tools.txt* file.

The *nc_tools.txt* file is used if it exists; but if it does not, *Tnn* codes automatically generate in sequence (for example, *T01, T02, … Tnn*). `Tnn` tool-diameter specification codes only append to the tool-select code in the header portion of the Excellon-format output file, expanding to a `TnnC.xxx` format to specify the required router bit size, if you enabled *Enhanced Excellon Format* in the *NC Parameters* dialog box. For example, T01C.045 specifies that Tool 1has a 45-mil diameter. For example, T01C.045 specifies that Tool 1has 45-mil diameter, with imperial inch units.

The syntax for each line in the `nc_tools.txt` file is as follows:

`hole_size plating_type tool_number +/- tolerance`

hole_size	Specifies the drill hole size, in the same units as the design.
plating_type	Identifies the type of plating: P (Plated) N (Nonplated) O (Optional)
tool_number	Specifies the tool number for each type of hole your design requires.
+ tolerance	Associates the positive tolerance to the tool. Defaults to 0 if no fourth and fifth fields exist. The tolerances of a design hole must match those of the tool to permit selection of that tool.

- tolerance	Associates the negative tolerance to the tool. Defaults to 0 if no fourth and fifth fields exist. The tolerances of a design hole must match those of the tool to permit selection of that tool.

> Do not place comments in this text file because they are read as errors. If errors exist in this file, you cannot make a drill data output file.

Following is an example of the `nc_tools.txt` file:

```
28 P TO1
35 P TO2
39 O TO3
42 N TO4
```

Syntax for nc_exclude.txt

The text file `nc_exclude.txt` lists drill sizes to exclude from the drill file and the NC drill legend. The syntax for each line in the `nc_exclude.txt` file is as follows:

`hole_size plating_type`

hole_size	Specifies the drill hole size, in the same units as the design.
plating_type	Identifies the type of plating: P (Plated) N (Nonplated) O (Optional)

As with the `nc_tools.txt` file, do not place comments in this text file because they are read as errors that prevent a drill data output file from being created. Here is an example of the `nc_exclude.txt` file:

```
35 P
42 N
```

The nctape.log File

The layout editor creates an ASCII log file named `nctape.log` along with the ASCII output file. This file contains error messages; NC Parameters; the output data file name; and size, plating, and quantity of the holes.

Running NC Route

Running *Manufacture – NC – NC Route* (ncroute command) enables you to generate data for an NC router. The output file is an ASCII file in Excellon Format.

Slot holes detected in the design are included in the NC Route output file. You can create two separate output files for plated and non-plated routing. In contrast, NC Drill output produced by running the *Manufacture – NC – NC Drill* (nctape_full command) only applies to circular drill holes.

When NC Route routes oval and rectangle slot holes, an appropriate tool is chosen from `ncroutebits.txt` using the following guidelines:

- Rectangle Slot: a tool size smaller than the minimum dimension of the rectangle must exist to route a rectangle path with appropriate Excellon tool compensation.
- Oval Slot: a tool size that exactly matches the minor dimension of the oval must exist to route the oval as a single line path. Otherwise, a tool size smaller than the minor dimension must exist to route an oval path with appropriate Excellon tool compensation.

Guidelines

The `ncroute` command executes with the following expectations:

- Each cutting path is drawn as a continuous series of lines and/or arcs.
- There is no limit on the number of paths that may be specified.
- Path direction can be specified.
- If the design is defined in English units, the tool diameter is interpreted to be inches; if Metric, then millimeters.
- Requires nc_param.txt file defined by NCDPATH for controlling additional ncroute settings. If not found uses Cadence defaults.

To specify varying widths for your cutting paths, you must generate a text file called `ncroutebits.txt`. This file cross-references the router tool diameters to Excellon tool codes. Each line of the file contains one diameter followed by a space and a tool code. The two fields can appear anywhere on the line as long as at least one blank space separates them. The following is an example:

```
25.0      T01
```

```
50.0      T02
75.0      T03
```

The tool diameter is interpreted in design units.

If you specify a cutting path at zero width, the layout editor assumes a tool code of T01. Likewise, if a path has a width, but it is not in the `ncroutebits.txt` file, a warning appears in the `ncroute.log` file and a tool code of T01 is assumed.

An ASCII log file called `ncroute.log` is created along with the ASCII output file. This file contains the NC Parameters used, followed by status messages, including warnings and error messages that describe the program execution and outcome.

Generating NC Drill Files for Test Points

After you chose test points automatically or interactively, you can generate an NC Drill file for each side of the design probed by running the *Manufacture – Testprep – Create NC Drill Data* (testprep ncdrill command).

The NC Drill data for top side probes is stored in a file called `top_prob.drl`. The NC Drill data for bottom side probes is stored in a file called `bottom_p.drl`.

4

Drafting and Dimensioning

The drafting and dimensioning features support Electronic Design Automation (EDA) industry standards that enable you to specify the dimensions of every feature on a board. This feature gives you greater control over the release to manufacturing of your design. The layout editor also enables you to customize the dimensioning process to conform to the manufacturing requirements of your site. Drafting and dimensioning normally occurs in the later stages of the design process. The layout editor's drafting features let you:

- Set a wide range of drafting standards
- Create detailed views of layouts

> You can edit drafting and dimensioning parameters by choosing *Setup – Design Parameters* (prmed command), then clicking *Edit drafting parameters* under the *Mfg Applications* tab. In the *Dimensioning Parameters* dialog box, you can modify the drafting and dimensioning parameters. Alternately, you can choose *Manufacture – Dimension Environment (*`dimension edit` command)and right-click to select *Parameters* to access the *Dimensioning Parameters* dialog box.

Setting the Drafting Standard

The layout editor lets you document your designs according to the industry standard that is required at your site. It supports the following standards:

- ANSI (American National Standards Institute) (default)
- BSI (British Standards Institute)
- DIN (German Industrial Normal)
- ISO (International Organization for Standardization)
- JIS (Japanese Industrial Standard)
- AFNOR (French Association for Normalization)

When you choose an industry standard, the layout editor automatically matches the default parameter settings to the standard you have chosen. In addition, the layout editor enables you to modify these settings to accommodate your unique dimensioning style. For example, if you specify AFNOR, the layout editor provides all dimensions in millimeters. You can, however, change the units parameter from millimeters to whichever units are required by your site.

> Choosing dimensioning units has no impact on current database units or accuracy.

Drafting Symbols

The layout editor stores dimensions (leader-oriented, linear, datum, and angular dimensions) in its database as drafting symbols. Like other symbol types, a drafting symbol consists of lines, arcs, and text that can be individually manipulated.

Unlike other symbol types, only dimensioning commands create drafting symbols. No `.dra` files are created. Internally created drafting symbols let you manipulate dimensions within a design (choose move, and delete them, for example).

Drafting symbols created within the symbol editor are decomposed into their base elements (lines, arcs, and text) at *Make Sym* time. No intact drafting symbols exist within the realm of any symbol instance—only lines, arcs, and text.

Creating Detailed Drafting Views

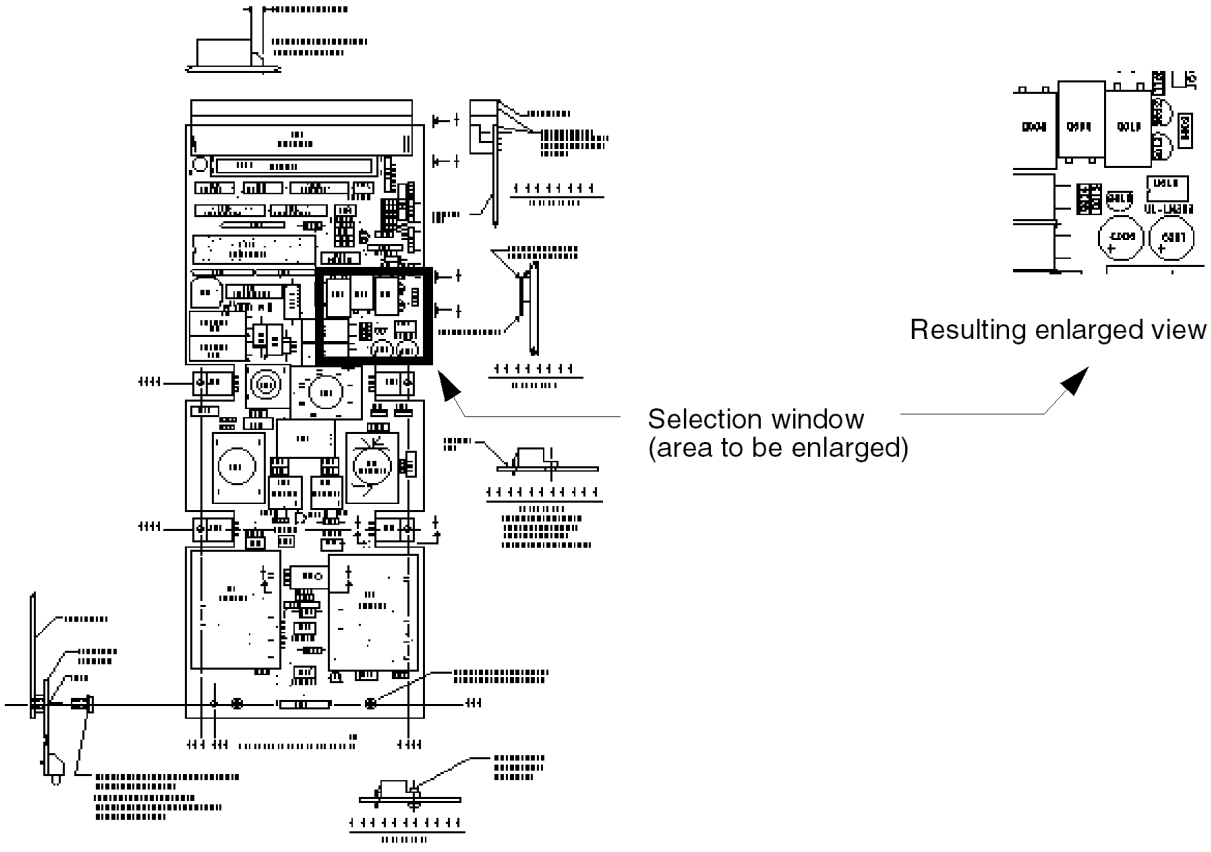
A detailed view is a separate view of an assembly or configuration that typically depicts the assembly or configuration in greater detail. In the layout editor, you create a detailed view by enlarging a chosen area in your design. You can locate a detailed view anywhere on your drawing.

You create detailed views by using *Manufacture – Drafting – Create Detail* (create detail command). Creating a detailed view involves the following basic steps:

1. Choosing an area of the design to be enlarged in either of two ways:
    - Choosing the area with two mouse clicks. The view includes everything in the outlined area.
    - Using the *Group* option from the pop-up menu to outline an area and exclude certain items from the detailed view.

    The layout editor automatically copies all visible elements in the chosen area and enlarges the chosen area by a user-defined scale factor (the scale factor defaults to 2, which means the enlarged view is twice the size of the chosen area).
    The layout editor copies only geometric information, and not electrical, logical, and property information. For example, the layout editor copies only the geometric elements of symbols.
    You can do the following to the chosen area:
    - Deselect (remove) certain elements in the chosen area to exclude them from the detailed view
    - Change the scale factor of the detailed view
    - Rotate the detailed view in 90 degree increments
    - Mirror the detailed view
2. Positioning the enlarged view in another section of the drawing, as the following figure shows:

Example of Enlarged View



Dimensioning Features

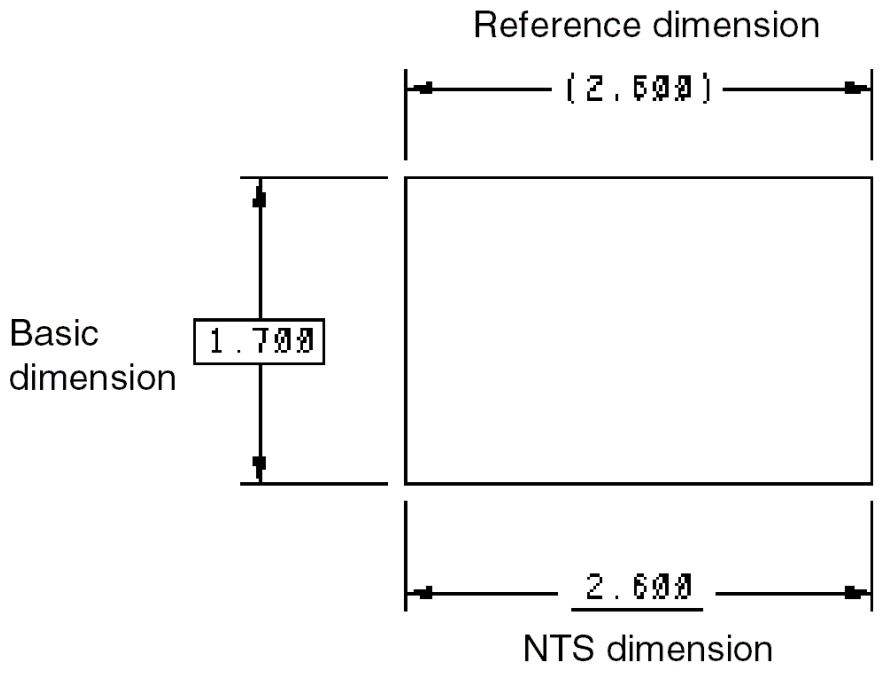
The dimensioning environment enables you to specify default dimensioning parameters that govern the format and syntax of the drafting process. These parameters conform to the ANSI specification for Dimensioning and Tolerancing (Y14.5M-1982) and can be displayed and modified at any time during the drafting session without interrupting the active command. Allegro X PCB Editor provides the following dimensioning features:

- Special dimensioning
- Dual dimensioning
- Coordinate tolerancing and limit dimensioning

Special Dimensioning

You may require special dimensioning during your documentation process. The *Special Handling* field on the *Text* tab in the *Dimension Parameters* dialog box enables you to define these dimensions. When active, special dimensioning features are applied to all dimension types. The following figure shows linear dimensions with special dimensioning features applied.

Linear Dimensioning Using Special Features

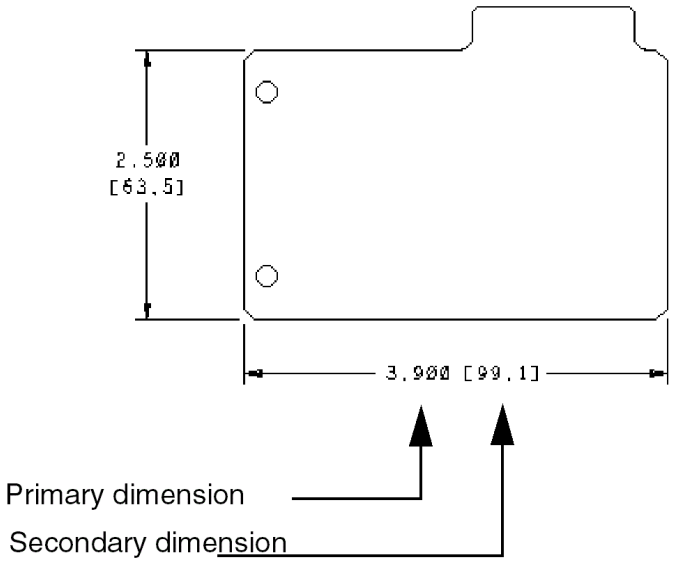


Dual Dimensioning

Dual dimensioning lets you specify primary and secondary units of measure and accuracy. Primary units appear in front of or on top of secondary units, which are enclosed in brackets and appear below or to the right of the primary dimension. The following figure shows the primary dimension in inches and the secondary dimension in millimeters. See the *Text* tab in the *Dimension Parameters* dialog box. Choose *Manufacture – Dimension Environment* (`dimension edit` command) and then right-click to select *Parameters*.

When active, dual dimensioning features are applied to all dimension types.

Primary and Secondary Dimensions

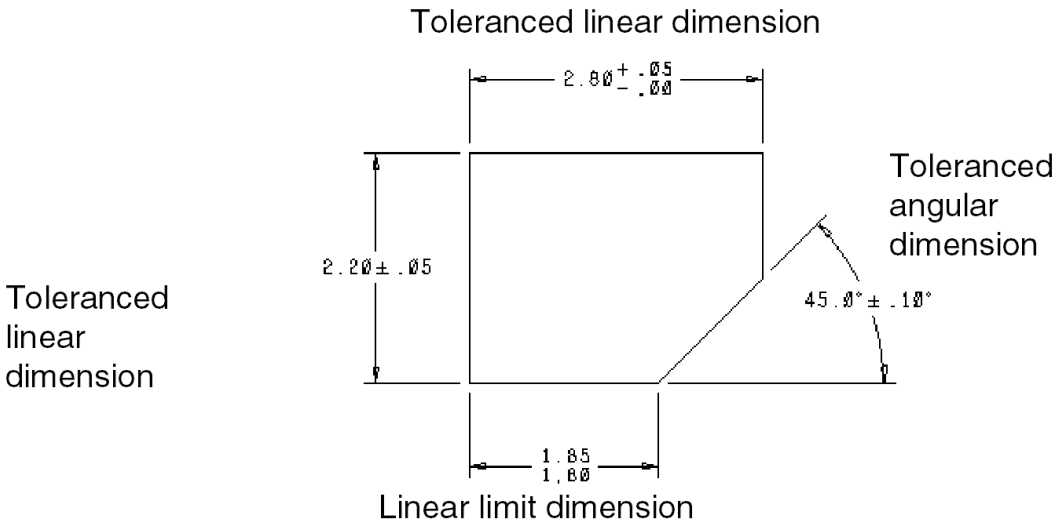


Dimensional Tolerancing

Limit dimensioning and coordinate and angular tolerancing enable you to specify the minimum and maximum amounts by which dimensions can vary during the manufacturing process. You can apply a plus (+) and/or minus (-) tolerance to dimensions. See the *Manufacture – Dimension Environment* (`dimension edi`t command) and then right-click to select *Parameters* section of the *Allegro PCB and Package Physical Layout Command Reference* for procedural details.

The following figure shows some toleranced dimensions.

Examples of Tolerancing Using the ANSI Drafting Standard



When active, tolerances and limits are applied to all dimensions that you create.

Related Topics

- dimension edit

Dimension Associativity and Editing

When a dimension added involving one or more design objects, the identified objects are subsequently forgotten as the objects being dimensioned. Hence, when a dimensioned object is moved or deleted the dimension graphics have no associativity and remain statically fixed. Associativity between the dimension and object would result in more efficiency and effectiveness of dimensions.

With associative dimensioning, objects involved in the initial creation of the dimension continue to remain associated with the dimension symbol. This is accomplished internally in the database by putting the dimension symbol and the objects that it dimensions in a group relation. When an object being dimensioned is either moved or deleted, an associated dimension can be implicitly moved and updated where needed, or deleted.

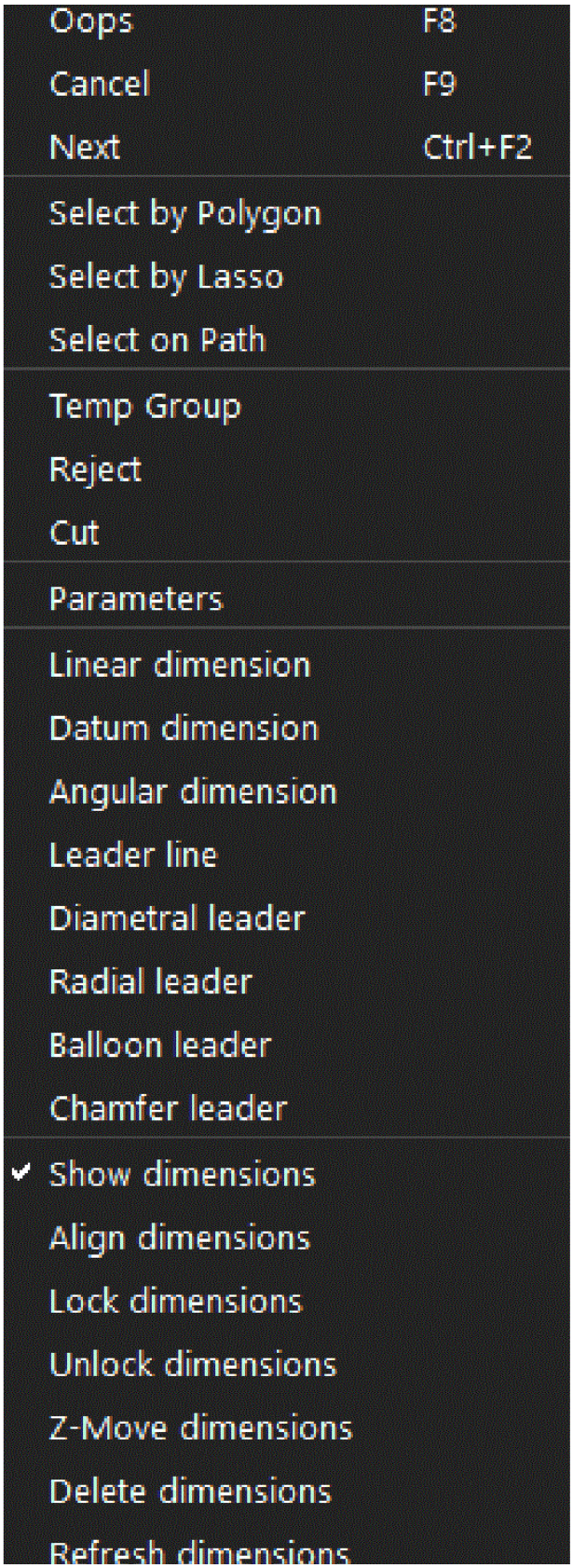
Dimensioning commands pick objects on the design canvas if they are enabled in the Find filter

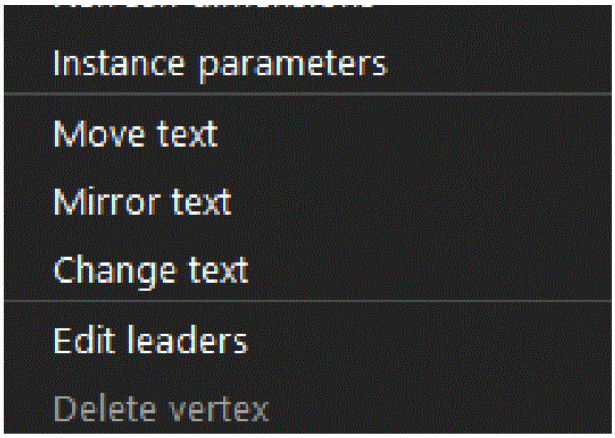
regardless of whether you snap to them or not. To snap dimensions to the grid points, deselect objects under the pick point in the Find filter.

Dimensioning Modes

Dimension modes lets you create and edit all types of dimensions. You can create multiple dimensions within each mode, until the mode is switched to a new one, or *Done* or *Cancel* is selected to terminate the entire `dimension edit` command.

- Parameters
- Linear Dimension
- Datum Dimension
- Angular Dimension
- Leader-Oriented Dimensioning
  - Leader Line
  - Diametral Leader
  - Radial Leader
  - Balloon Leader
  - Chamfer Leader
- Show Dimensions
- Align Dimensions
- Lock Dimensions
- Unlock Dimensions
- Z-Move Dimensions
- Delete Dimensions
- Instance Parameters
- Move Text
- Mirror Text
- Change Text
- Edit Leaders
- Delete Vertex


Oops F8
Cancel F9
Next Ctrl+F2
Select by Polygon
Select by Lasso
Select on Path
Temp Group
Reject
Cut
Parameters
Linear dimension
Datum dimension
Angular dimension
Leader line
Diametral leader
Radial leader
Balloon leader
Chamfer leader
Show dimensions
Align dimensions
Lock dimensions
Unlock dimensions
Z-Move dimensions
Delete dimensions
Refresh dimensions



When you select object for dimensioning, the object gets associated with the dimension. If you move the object the dimension also moves with the object. Similarly, if you move the object that is associated with a reference datum, all the relative datums automatically updates to reflect their new distances from the reference datum. If you delete the object the dimension is also deleted.

If a dimension is not associated with an object but is on a point in mid air, the *Show dimensions* mode shows a warning message.

Parameters

The *Parameters* mode lets you specify the parameters for creating dimension. The parameters are either global or instance-specific. When you change any global parameter it applies to both; to all new dimensions and to all existing dimensions. When you change an instance specific parameter it is applied only to newly created dimensions. The instance-specific parameters of dimensions are saved so that if the object is moved the dimension gets recreated.

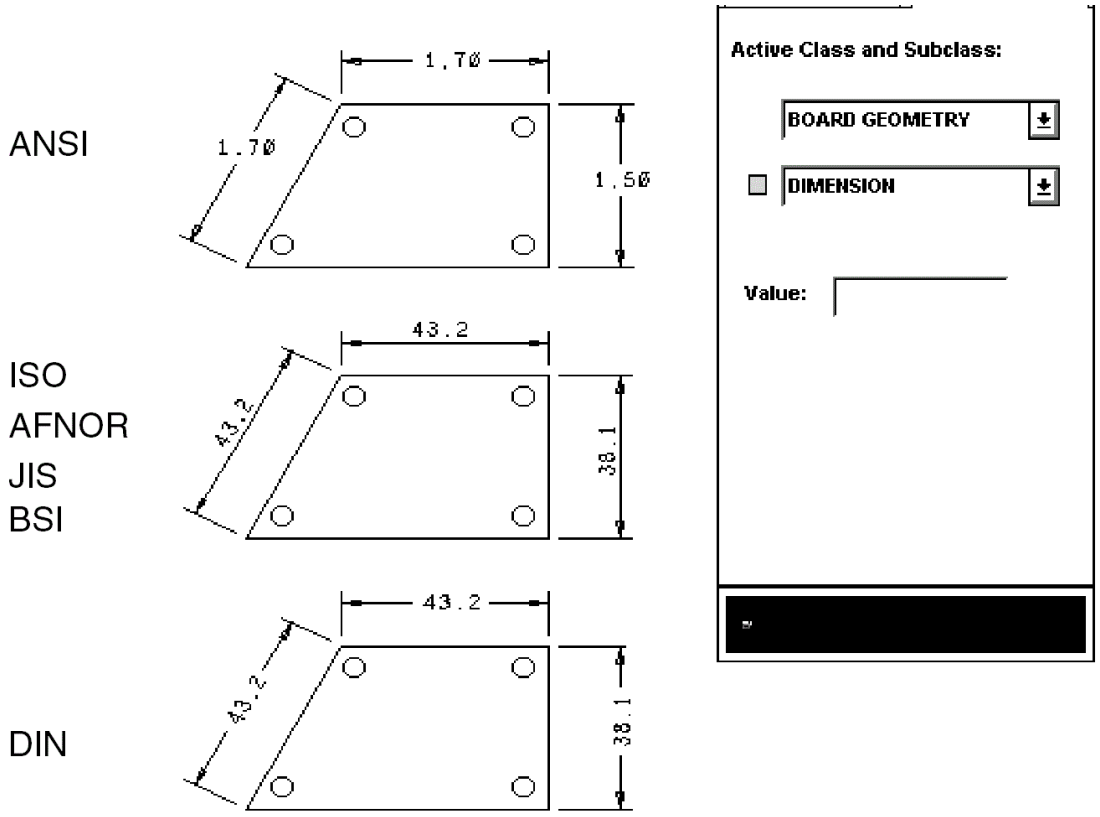
> The dimension environment uses the *Decimal places* settings as specified in *Dimensioning Parameters* dialog and does not adhere to *decimal places* settings that are specified at design level in *Design Parameter Editor* dialog box.

Linear Dimension

Linear dimensioning enables you to add standard linear dimensions of objects or between two user-defined points in a layout. These dimensions can be horizontal, vertical, or at an angle. The following figure shows linear dimensioning using inches in the ANSI standard.

Linear dimensioning enables you to control the element types (lines, arcs, rectangles, pins, and vias) to be dimensioned.

Linear Dimensioning (ANSI Standard)



Datum Dimension

Datum dimensioning lets you specify dimensions relative to a user-defined reference point (datum). The dimensions are placed at the end of a single extension line. The extension line can have multiple vertices, which lets you snake it through complex geometry to the outside of the design outline. This type of dimensioning is recommended for designs requiring a high density of dimensioning.

The first dimension point you add to your layout is defined as x = 0 and y = 0. All other datum dimensions added are relative to the first point. You can override any dimension value by entering another value in the *Value* field in the *Options* tab.

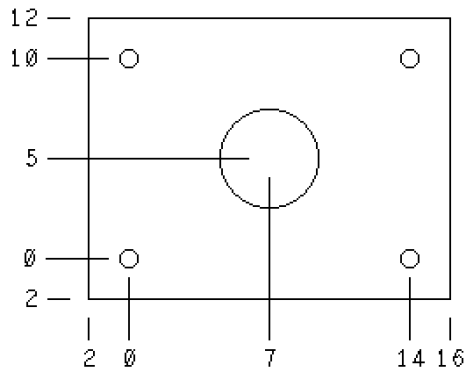
When using the ANSI standard, you can rotate horizontal dimension text 90 degrees by setting the *Align Text with Dimension Line* parameter in the *Text* tab of the *Dimension Parameters* dialog box.

Datum dimensioning enables you to control the element types (lines, arcs, rectangles, symbols, pins, and vias) to be dimensioned.

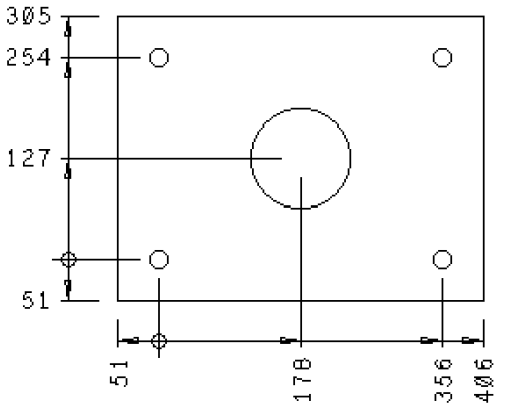
The following figure illustrates the datum dimensioning style for each standard.

Datum Dimension

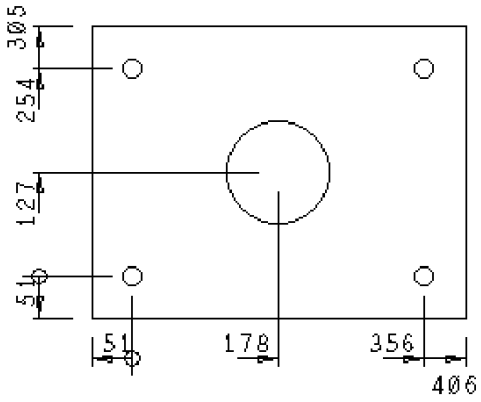
ANSI



JIS
AFNOR
ISO
DIN



BSI



Group/Window Datum Dimensioning

You can choose datum dimensioning objects as a group or by using a window. If you choose the object this way, dimensions are created for all line segment end points, arc and circle centers, symbol-instance origins, via and pin instances, and rectangle vertices according to the current settings in the *Find Filter*.

You can create windows of dimension instances, delete unnecessary instances, or create selective groups of dimensions. You can determine your dimensioning strategy to fit the requirements of each design. For ANSI, define borders for group/window sets by using rubberbanding border axis lines after making a group/window selection. For non-ANSI standards, the dimensions are aligned with reference datum targets.

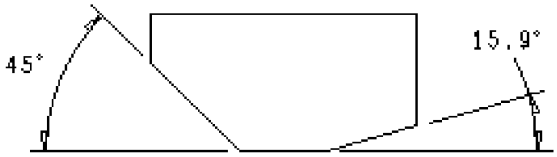
Angular Dimension

Angular dimensioning calculates the angle between two line segments, as follows.

Angular Dimensioning

ANSI

ISO
AFNOR
JIS
BSI



DIN

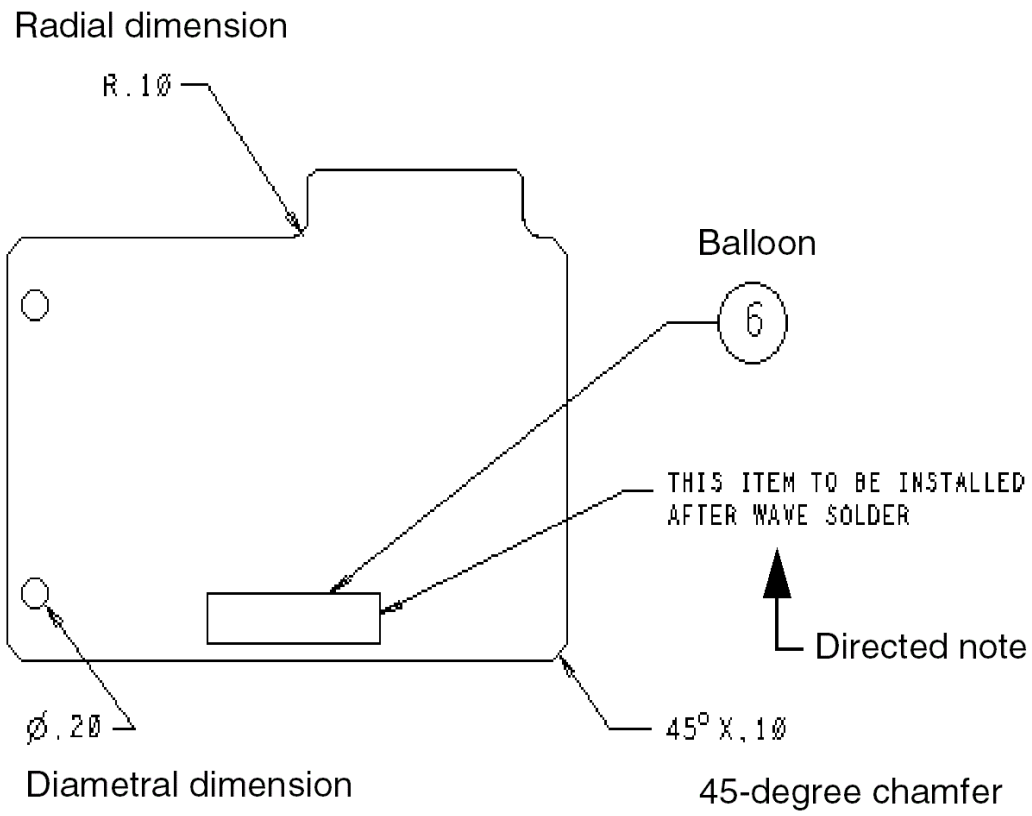
Leader-Oriented Dimensioning

A leader is a note or dimension directed to the drawing feature to which it supplies information. Leaders are used for adding diametral and radial dimensions and balloons. Leaders also supply an alternative method for dimensioning 45-degree chamfers.

Display characteristics for leaders (such as termination type and size) are determined by parameters set in the *Dimension Lines* field in the *Lines* tab of the *Dimension Parameters* dialog box.

You can specify leaders terminated with an arrow, bullet, slash, or without a termination. The following figure shows an example.

Specifying Leaders



Leader Line

To add Leader line dimension as a line, choose *Manufacture – Dimension Environment* (`dimension edit` command) and then right-click to select *Line Leader* .

Diametral Leader

Diametral dimensioning identifies the diameter of a circle. To dimension the diameter of a circle, choose *Manufacture – Dimension Environment* (`dimension edit` command) and then right-clicking to select *Diametral Leader.*

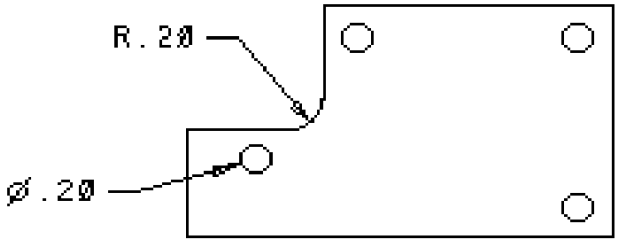
Radial Leader

Radial dimensioning identifies the radius of an arc. To dimension the radiuas of an arc, choose *Manufacture – Dimension Environment* (`dimension edit` command) *a*nd then right-click to select *Radial Leader* .

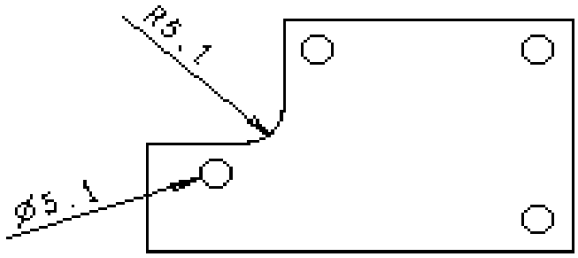
When you choose an arc or circle to dimension, the current diametral or radial value appears in the *Options* tab value field. This value is used in the current dimension. You can modify the value. The following figure shows diametral and radial dimensioning in each standard.

Diametral and Radial Dimension

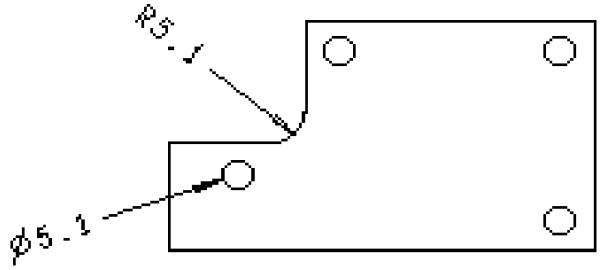
ANSI
BSI



ISO
AFNOR
JIS



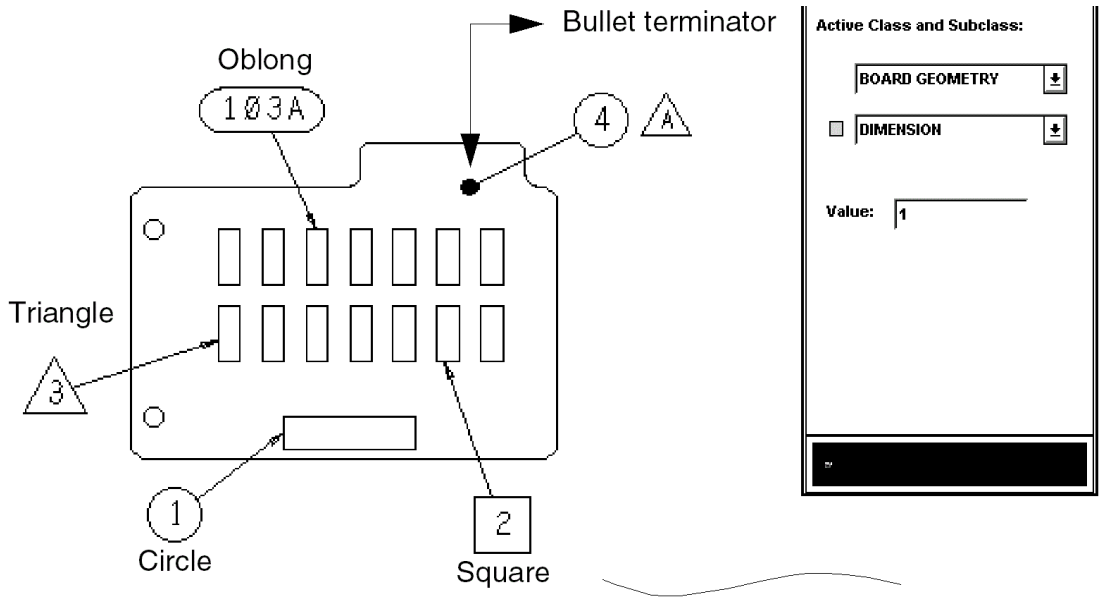
DIN



Balloon Leader

A balloon is a leader with a termination (arrow, bullet, or slash) on one end and a balloon (circle, square, triangle, or oblong) that encloses an alphanumeric character string on the other end. Balloons typically point to a component while the text enclosed in the balloon relates the component to an item in a bill of materials.

Balloon Options



You can add balloons to a design with *Manufacture – Dimension Environment* (`dimension edit` command) and then right-click to select *Balloon Leader* as described in the *Allegro PCB and Package Physical Layout Command Reference*. You can set the value of the balloon characters via the pop-up menu, or they can be incremented automatically.

> Balloons have their own text block field.

The current balloon character string is displayed in the *Options* tab value field. The value in this field is incremented the next time that you use the command, but you can modify this value at any time.

You can also add balloons without leaders to a design for use as drawing markers (for example, revision or note markers and locations) or as multiple balloons that point to a common point (as with attaching hardware stackups—screws, washers, and nuts).

You can set the balloon size and Text block in the *Balloon* tab of the *Dimension Parameters* dialog box.

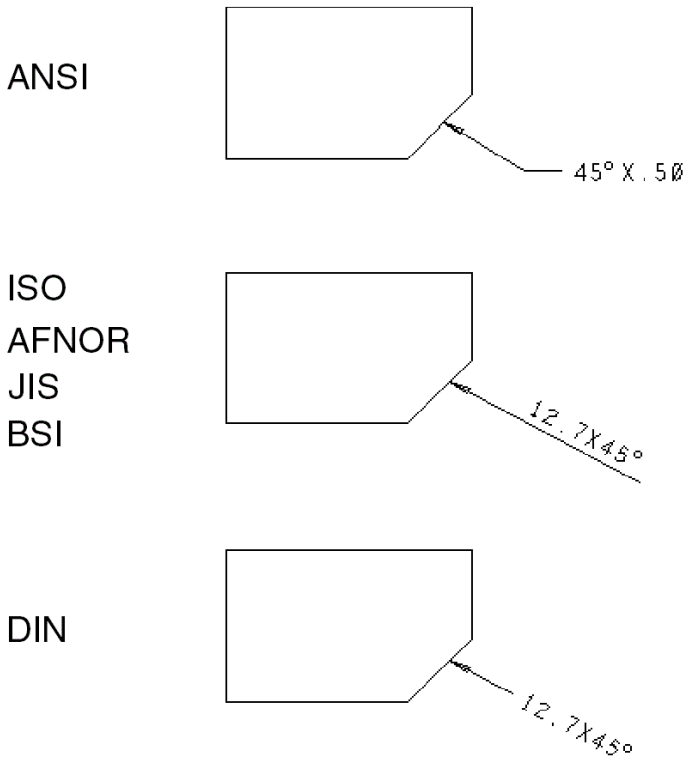
An alternative method is to have Allegro X PCB Editor calculate and set the balloon size for you.

To set the balloon size, set the Text block and maximum number of characters expected in the balloon text and then check the *Auto Size* check box.

Chamfer Leader

Manufacture – Dimension Environment (`dimension edit` command) and then right-click to select *Chamfer Leader* provides an alternative and simpler way of dimensioning 45-degree chamfers than using a combination of linear and angular dimensioning. For details on specific procedures, see *Manufacture – Dimension Environment* and then right-click to select *Chamfer Leader* (`dimension edit` command) in the *Allegro PCB and Package Physical Layout Command Reference*.

Chamfer Leader



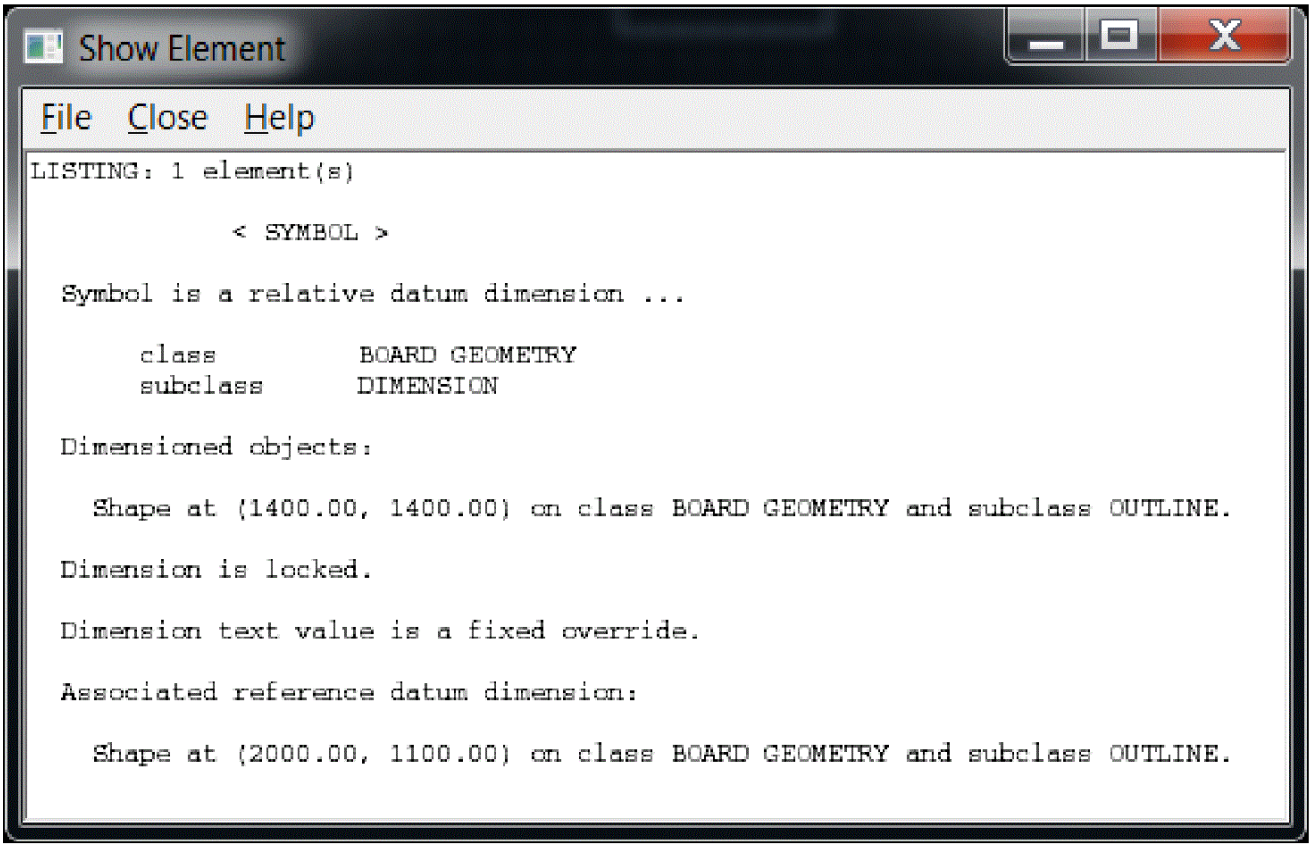
Show Dimensions

The *Show dimensions* mode shows dimension related information of symbol objects. You can select dimensions by single pick, or multiple picks with a window drag, or *Select by Polygon* or *Temp Group* from the right mouse button menu.

Information given for a dimension symbol includes:

- type of dimension
- origin
- class and subclass
- object(s) the dimension is associated with
- if the dimension is on a point in mid air
- if the dimension is locked
- if the dimension value fixed

For a relative datum dimension, information is given on the reference datum object and is listed for each associated relative datum dimension object.



Align Dimensions

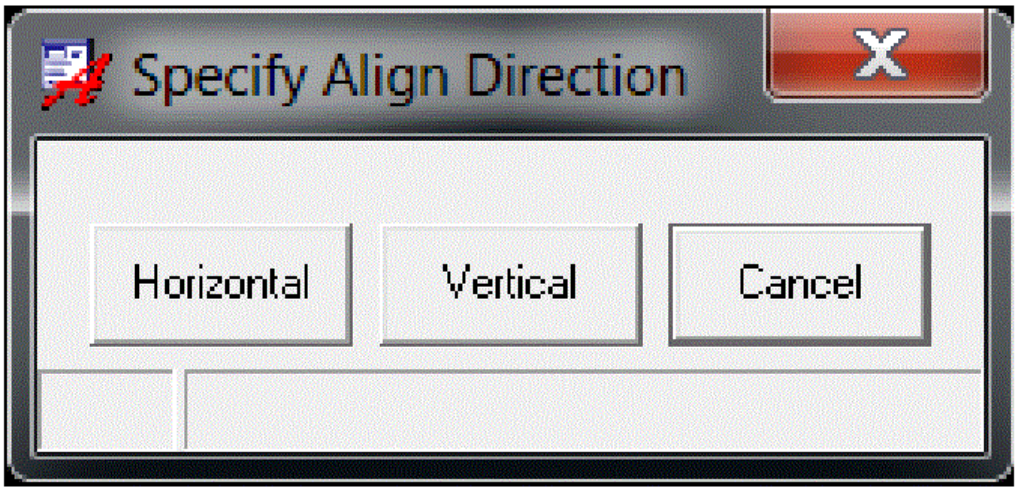
The *Align dimensions* mode lets you align the dimensions with respect to the master dimension. The first dimension selected is a master dimension and remain fixed. The dimensions selected subsequently are aligned to the master dimensions. The dimension can be selected with a single pick, window drag or *Select by Polygon*, and *Temp Group* modes.

Aligning Linear and Datum dimensions

On aligning the text/endpoint locations of the dimensions are aligned. So, for linear dimensions in X direction the Y-locations of the dimension text are made same as the master dimension and they are aligned in a row. Similarly for linear dimensions in Y-directions the e X- locations are made same as the master dimension to align them in a column. Linear dimensions in X cannot be aligned to a master linear Y dimension, and vice versa. The same rules for aligning linear dimensions in X and Y also similarly apply to datum dimensions in X and Y.

Aligning Leader Type of Dimensions

You can also align the leader dimensions. When you select the dimensions for aligning a menu appears as shown in the figure below to select the direction for alignment.



If you select horizontal , the text/endpoint locations of the leaders maintain their X-locations and are aligned horizontally to the new Y-locations of the master dimension and vice-versa.

You can lock the dimensions after aligning them using *Lock dimensions* mode. These dimensions maintain their alignments with general editing of the objects they are associated with. You can aligned the locked dimensions also. In such case warning messages appear for locked dimensions, and they are re-locked at the new alignment locations.

> If any dimension selected for alignment is incompatible with the master dimension a warning message is issued and dimension is ignored.

```
W- (SPMHDI-54): Dimension is not of the same general type for alignment ... ignored.
```

Lock Dimensions

The *Lock dimension* mode lets you can fix the text location or leader end point on the board by locking the dimension. The dimension can be selected with a single pick, window drag or *Select by Polygon*, and *Temp Group* modes. The leader types of dimensions are locked in both X and Y directions. For linear and datum dimensions, an X-axis dimension is locked in Y, and Y-axis dimension is locked in X. If a dimension is already locked, a warning message is issued and the dimension is ignored.

> The dimension is locked only for general editing. Editing of the dimension itself within *dimension edit* ignore the lock, issues a warning message, and determine an appropriate new lock point.

Unlock Dimensions

The *Unlock dimension* mode lets you unlock the dimensions. Selected dimensions are unlocked and their current text end points are floating and moving with the dimension.

Z-Move Dimensions

The *Z-move dimension* mode lets you move the selected dimensions to the new class and/or subclass. You can select dimension with a single pick, window drag, *Select by Polygon*, and *Temp Group* selection modes.

Delete Dimensions

The *Delete dimension* mode disassociates the selected dimensions from their objects and delete them from the design. You can select dimension with a single pick , window drag, *Select by Polygon*, and *Temp Group* selection modes.

> If a reference datum dimension is deleted, all of its associated relative datum dimensions are automatically deleted.

Disband Dimensions

The *Disband dimension* mode disbands a dimension into individual unassociated objects (text, lines, and shapes) for editing. This option can be used for legacy (non-associative) as well as for associative dimensions.

Refresh Dimensions

The *Refresh dimension* mode refreshes all the associative dimensions in a design. This option restores dimension elements that are no longer visible in the design and allows you to modify or remove them.

Instance Parameters

The *Instance Parameters* mode brings up the *Dimensiong Parameters* dialog box, and allow you to change only instance-specific parameters that are only applied to the selected dimension. The instance-specific settings initially shown reflect the last settings for the selected dimension. On closing the *Dimensiong Parameters* dialog box you can select the other dimension for changing its instance-specific parameters.

Move Text

The *Move text* mode moves the text of a dimension to a new location and the dimension is recreated accordingly to accommodate the new text location. If you move the locked dimension following waring message is issued that a new lock point is being set and you can move the locked dimension.

```
W- (SPMHDI-69): Linear dimension is currently locked and will have a new lock point set
for it.
```

This mode only works with single pick at a time.

Mirror Text

The *Mirror text* mode mirrors the text of a dimension. You can also mirror the locked dimension.

This mode only works with single pick at a time.

Change Text

The *Change text* mode lets you change the dimension text by entering the new value in the *Options* tab. This mode only works with single pick at a time. If the current text value is fixed and you pick the dimension text to change following warning message is issued. A fixed text value is reset to the actual computed value by entering '0' as the new text value in the value field in *Options* tab.

```
W- (SPMHDI-83): Dimension value is currently a fixed override. Change text with a value
of 0 to reset to automatic value determination.
```

Edit Leaders

The *Edit leaders* functionality applies only to the leader types of dimensions and datum dimensions. This mode works with single pick at a time. You can either edit an existing leader type of dimension or create a new vertex at the pick point. When you select the pick point on the leader a rubberband graphics are attached to the cursor. You can select a new location for the vertex, and the dimension is recreated.

Delete Vertex

The *Delete vertex* mode lets you delete the current vertex from the dimension leader. dimension. This is a sub-mode of the *Edit leaders* mode and becomes temporarily selectable on the right-click when an existing or new vertex is attached to the cursor.

Related Topics

- dimension edit

5

Silkscreening

Lines, arcs, and text used to create a silkscreen for a design are usually stored in several classes. Each of the following classes has SILKSCREEN_TOP and SILKSCREEN_BOTTOM subclasses that are used for this purpose:

- BOARD GEOMETRY
- COMPONENT VALUE
- DEVICE TYPE
- PACKAGE GEOMETRY
- REF DES
- TOLERANCE
- USER PART NUMBER

The layout editor generates a complete silkscreen as a composite of the graphics from the subclasses.

A manufacturing problem can occur when a silkscreen goes across a hole on a design that must be plated through. The ink from the silkscreening process can seep into the hole and prevent the solder from flowing through correctly. If a silkscreen line or arc crosses a hole, the portion of the line or arc that crosses the hole normally is trimmed away. If a text string crosses a pad, the text usually is moved away from the pad.

The layout editor automates silkscreen editing by first copying all silkscreen graphics that you have chosen to the MANUFACTURING class on either the AUTOSILK_TOP or AUTOSILK_BOTTOM subclasses. It then edits any duplicated graphics that interfere with a hole.

Automatic Silkscreen Updating

Once you run Auto Silkscreen on a layer, the layout editor automatically and incrementally updates silkscreen information associated with design objects when you change the design. These changes occur at the existing AUTOSILK subclass level by attaching line and text information to associated symbol instances. The actions listed below automatically update silkscreen information:

- Component movement or placement
  - Silkscreen information accompanies the component. The updated silkscreen clears adjacent component and via pads at the new location and deletes the information from the old location.
  - Existing silkscreen information derived from component or board geometry that does not clear the pads of the newly placed or moved component is updated to a new location.
  - Existing silkscreen information that had previously been updated to accommodate a placed/moved component is newly updated if the component is moved or deleted.
- Via movement or placement
  - Existing silkscreen derived from component or board geometry that is in proximity to the placed/moved via is updated.
  - Existing silkscreen information that had previously been updated to accommodate a placed/moved via is newly updated if the via is moved or deleted.
- Export libraries: *File – Export – Libraries* (dlib command)
  - AUTOSILK subclass silkscreen information is stripped out
- Import/export sub-drawings: *File – Import – Sub-Drawing* (clppaste `command`)/*File – Export – Sub-Drawing* (clpcopy command)
  - AUTOSILK subclass silkscreen information is ignored, and the information regenerated as appropriate
- Update symbols: *Place – Update Symbols* (refresh symbol command)
  - AUTOSILK subclass silkscreen information is regenerated

> Automatic silkscreen updating occurs on a layer only after you have run Auto Silkscreen on it.

Because automatic updating depends on information contained on the AUTOSILK subclass layers, you should not edit this layer manually unless absolutely necessary to avoid DRC violations or to maintain legibility. Changes at the AUTOSILK subclass level are lost when Auto Silkscreen is run. Wherever possible, change the SILKSCREEN subclasses.

Note also that automatic silkscreening functionality removes and regenerates elements dynamically for symbol elements but only removes silkscreen elements for line, text, and arcs.

Setting Silkscreen Parameters

You can edit silkscreen parameters by choosing *Setup – Design Parameters* (prmed command), then clicking *Edit silkscreen parameters* under the *Mfg Applications* tab. In the Auto Silkscreen dialog box, you can modify the parameters that apply to silkscreen generation. You can also access the Auto Silkscreen dialog box by choosing *Manufacture – Silkscreen* (`silkscreen param` command).

6

Plotting Functionality

This section describes plotting functionality, including the following:

- Plotting from UNIX and Windows
- Creating plot and intermediate plot (IPF) files and control files from a layout database, Gerber file, or Excellon drill file for plotting
- Previewing IPF files before plotting
- Plotting IPF files using the allegro_plot program

> Because of structural differences in the way UNIX and Windows handle plotting, some information in this chapter only pertains to one or the other operating system. Functionality supported only on Windows or UNIX is highlighted.

Plots can be created from the following:

- Open design
- Gerber photoplot files
- Excellon drill files (UNIX only)

Plotting Methods on a UNIX Workstation

You create a plot on a UNIX workstation using one of these methods:

- Using the `plot` command at the console window prompt.

This involves setting up your plotting parameters in the Plot Setup dialog box using File – Plot Setup (plot setup command). Then you print the plot with File – Plot (plot command).

- Running the allegro_plot program from the operating system prompt. This program is the forms-driven plotting interface that uses the Cadence corporate plotting package `plotServ`.

To do this, you create IPF and control files from any of the following:

  - Current, active layout designs using the IPF section of the Plot Setup dialog box and the `create plot` command
  - Gerber photoplot files using the gbplot command
  - Excellon drill files using the explot command

Then you run allegro_plot.

Setting up Prerequisite `.cdsplotinit` Plotter Configuration File on UNIX

When you run File – Plot (plot command) or the allegro_plot command on a UNIX workstation, the `.cdsplotinit` plotter configuration file, which lists available printers/plotters, must reside in:

- `<install_path>/tools/plot`
- the current working directory
- or your home directory

The `.cdsplotinit` file contains vital device-specific information, including:

- Output device name
- Output format
- Available paper sizes
- Maximum number of allowed pages
- UNIX commands for spooling jobs to the queue
- Checking queued jobs Removing queued jobs

When searching for the `.cdsplotinit` file, the plot program always searches the home directory first and adds the printers in it to the list.

> The plot program does not stop when it encounters the first occurrence of the `.cdsplotinit` file in any directory.

Using CDSPLOTINIT to locate `.cdsplotinit`

Both File – Plot (plot command) and allegro_plot may also use the UNIX-level environment variable CDSPLOTINIT if it is set, which supports a set of directories to examine.

> CDSPLOTINIT is a UNIX-level environment variable, not an Allegro environment variable.

If the CDSPLOTINIT environment variable is set, the plot program then not only searches the home directory for the `.cdsplotinit` file, but also every directory the variable specifies.

If you do not set the CDSPLOTINIT environment variable, the plot program then not only searches the home directory for the `.cdsplotinit` file, but also the current working directory and the `/cds` system directory.

If a `.cdsplotinit` file resides in multiple directories, the program evaluates each path in turn and appends all printers found in each file in each directory to the list.

Sample `.cdsplotinit` File Entry

The following is a sample `.cdsplotinit` file entry:

```
bos1|Apple LaserWriter II NT/NTX: \
:manufacturer=Apple Computer: \
:spool=lpr -Pbos1: \
:query=lpq -Pbos1: \
:remove=lprm -Pbos1 $3: \
:type=postscript1: \
:maximumPages#30: \
:resolution#300: \
:paperSize="A" 2400 3150 75 75: \
:paperSize="A4" 2332 3360 60 60:
```

For detailed information on setting up the `.cdsplotinit` file, refer to the *Plotter Configuration User*

Guide, available on Cadence Online Support.

Using APLT_MULTILINE APLT_USE_WINDOW_EXTENTS to Improve Plot Print Quality

You may also need to set two environment variables to improve the quality of the printed plot:

- APLT_MULTILINE
- APLT_USE_WINDOW_EXTENTS

APLT_MULTILINE

A common problem when using plotters has been the slow performance resulting from the plotting of large solid filled lines. To remedy this problem, the allegro_plot program plots solid filled lines as individual line segments, filling each segment separately. The drawback to this approach is that the overlapping portions of the line segments are filled twice, producing lines that have darker fills at the intersections of the line segments. To avoid this, you can set the environment variable APLT_MULTILINE before invoking the allegro_plot program. When this variable is present, the solid filled lines do not break into segments. If this variable is set, and a filled line crosses over itself, some plotters leave the intersecting areas of the line unfilled.

APLT_USE_WINDOW_EXTENTS

Another common problem is related to hardware text. When hardware text is present in an IPF file, it is often difficult or impossible to determine the text size produced on the plot. If text occurs on or near the borders of the plot data, it may be clipped in the resulting plot. To rectify this, set the environment variable APLT_USE_WINDOW_EXTENTS before invoking the allegro_plot program. When this variable is present, the extents of the plot conform to the size of the viewing window used when the create plot command was run.

Plotting Methods on a Windows PC

You create a plot on a Windows PC using one of these methods:

- Using File – Plot (plot command) at the console window prompt.

This involves setting up your plotting parameters in the Plot Setup dialog box using File – Plot Setup (plot setup command). Then you print the plot with File – Plot (plot command).

- Creating IPF and control files from current, active designs and running them on a UNIX workstation with the allegro_plot program or third-party plotting software.

First, you create IPF and control files using the IPF section of the Plot Setup dialog box and the create plot command. Then you transfer the IPF and control files to a UNIX workstation that runs allegro_plot or to third-party plotting software. Finally, you run the plots.

- Creating IPF and control files from Gerber photoplot files and running them on a UNIX workstation with the allegro_plot program or third-party plotting software.

First, you create IPF and control files using the `gbplot` command. Then you transfer the IPF and control files to a UNIX workstation that runs allegro_plot or to third-party plotting software. Finally, you run the plots.

Plotting a Design Using the plot Command

The layout editor plots according to the settings in the Plot Setup dialog box that appears when you run File – Plot Setup (plot setup command). In conjunction with the Plot Setup dialog box, you use Display – Color/Visibility (color192 command), any of the Display/Rats commands, and the View/Zoom commands to obtain the visibility and scaling of the plot.

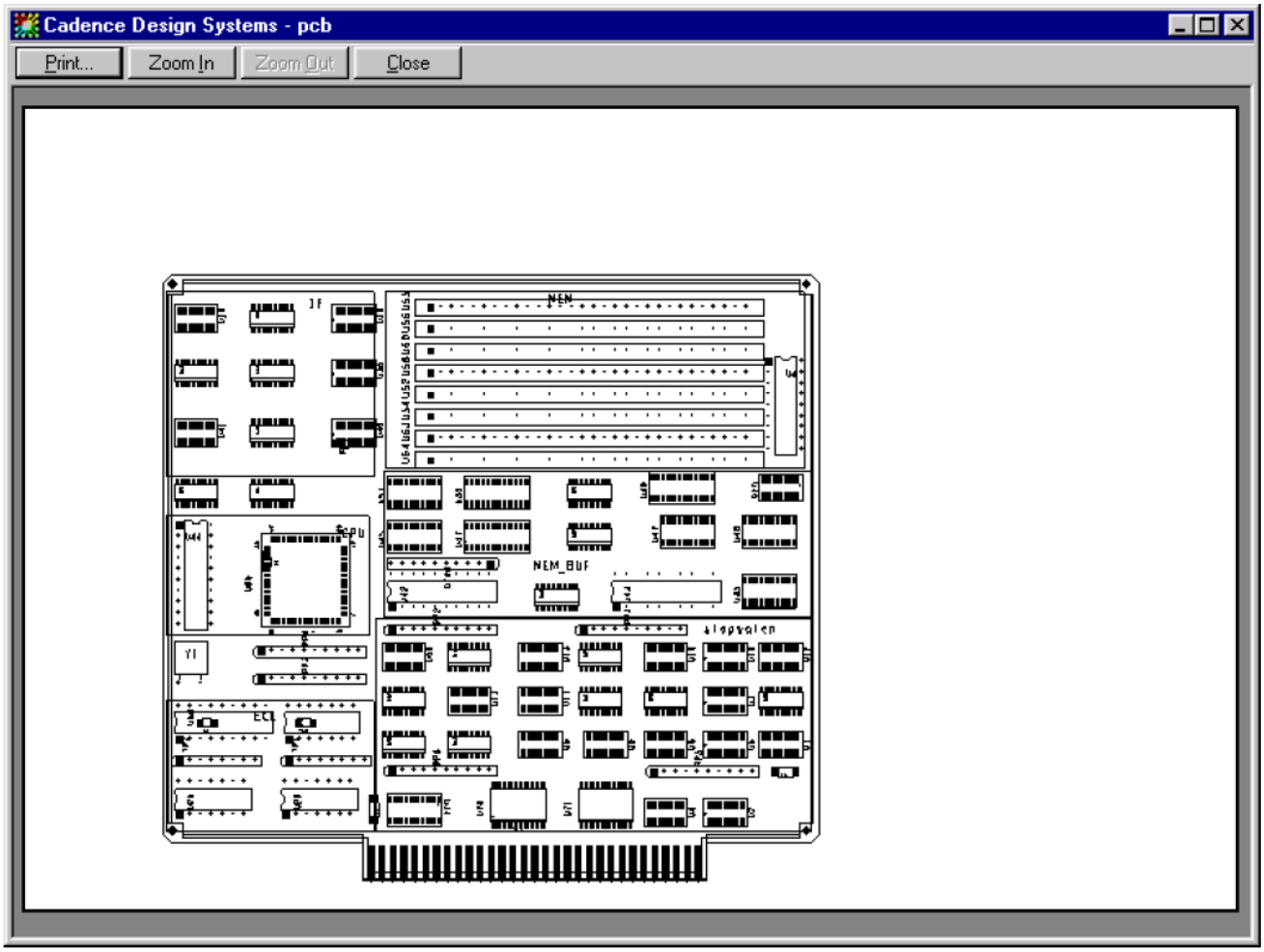
For detailed information on setting plotting parameters, see File – Plot Setup (plot setup command) in the the layout editor.

You can produce a plot on Windows or UNIX by running File – Plot (plot command) and preview your plot prior to doing so (Windows only). See Previewing a Design File (Windows Only) for details. See File – Plot (plot command) in the the layout editor for procedural information.

Previewing a Design File (Windows Only)

You can preview how an active design appears when sent to a printer by using File – Plot Preview (plot preview command). The preview conforms to the settings in the Plot Setup and the Windows Print Setup dialog boxes. This feature temporarily replaces the layout editor with a representation of the plotted design.

Plot Preview of a Design



You can zoom in to a specific area of the design preview by positioning the cursor at the desired location. To enlarge the display without regard for cursor position, click Zoom In . Click Zoom Out to reduce the display size.

To close the preview window and return to the the layout editor, click Close . Do not use the

Microsoft control button to close the window; doing so exits from the layout editor.

Scripting with File – Plot on Windows

You can include plot commands in scripts on Windows and UNIX platforms; however, script files containing plotting are not compatible between operating systems. When you record a script on Windows, the Print quality and Print to file settings from the Print dialog box are recorded. The script records the following settings in the Print Setup page of the dialog box:

- Printer name
- Paper size
- Paper source
- Orientation
- Copy count
- Duplex printing
- Scaling
- Color appearance

> The Print dialog box does not appear while the script replays.

Creating Intermediate Plot and Control Files

The intermediate plot file (IPF) (`filename.ipf`) and control file (`filename.ctl`) are automatically created simultaneously and provide necessary data to generate a plot of the design. Because they are both text files, you can view and edit them.

You can also transfer IPF and control files to other computers. For example, you can create the files on a Windows PC and transfer them to a UNIX workstation on which the allegro_plot program is installed and that is attached to a plotter.

Intermediate Plot File

The IPF contains all of the plotting data—the dimensions and location of all graphic elements to be plotted. The IPF is an ASCII file that you can use for plotting and also as a medium to transfer database geometry to non-Allegro systems. The format and content of the IPF are described later in this section. This file has the extension `.plt`.

Control File

The control (`.ctl`) file is automatically generated, but your use of it is optional. It identifies each color to use when plotting the drawing. On UNIX, pens to which you assign colors (described in the section below) determine color selection.

> Pen assignments are not supported on Windows. Although create plot generates a control file from the layout editor on a Windows PC, to use it, you must copy the file to a third-party tool that generates plots or to a UNIX workstation that can run allegro_plot.

Note that Gerber files use only one color. The control file's format and content are described later in this section.

For the following information, see the create plot command in the the layout editor:

- Prerequisites for Creating Control Files (UNIX Only)
- Creating Files from an Active Design
- Creating Files for Negative Plane Layers

Creating Files from Gerber Files

You use the gbplot command from the operating system prompt to create penplot files from Gerber files. This command is compatible only with the Gerber 4x00 and Gerber 6x00 artwork output formats. The gbplot command requires that the artwork aperture and parameter files be accessible through the ARTPATH environment variable.

Each graphic item in the Gerber data with a station code defining a standard geometry (for example, line, circle, or square) is plotted as that geometry. If the station code is defined as FLASH, the item is plotted as a butterfly figure.

A log file named `gbplot.log` lists the aperture table and photoplotter parameters used. Any errors and warnings encountered are also listed.

For procedures on creating penplot files from Gerber files, see the gbplot command.

Creating Files from Excellon Drill Files (UNIX Only)

The NCdrill parameter file should be accessible through the NCDPATH environment variable. If it is not found, a default set of parameter values is used. Each drill is plotted as a circle of the specified diameter. The command outputs a summary of numbers of each drill size and estimated tape length at the terminal.

For procedures on creating IPF and control files from Excellon Drill files, see excellon processing command.

Previewing Plot Files

You can view the contents of plot files before plotting them by opening the files in a design window. This feature can save plotting time because you can display an IPF file on your workstation before generating a hard copy. You can also:

- Add plot data to an open design.
- Combine IPF files into one design.
- Scale, rotate, and/or mirror each IPF file independently.
- Position and view the design before the final plot.
- Create a new IPF file from the new design.

Each element in the IPF file is added to the design database in the specified position and scale in the manufacturing class, pen number subclass. The layout editor automatically creates a subclass for each pen as needed under the manufacturing class (for example, subclass pen1, pen2, and so on). This default can be changed by changing the class in the Options tab before placing the rectangle.

Subclasses are made visible. Add as many IPF files as needed to the design by repeating the procedure for each IPF file.

Text is entered in the size defined in the IPF file and by the corresponding size in the text size table in the drawing. If that text size is not in the table, a new text size entry is added if a slot is available. If no slots are available, the closest text size is used. In the event of a tie between two text sizes, the smallest is used. Depending on the text size used, scaling of text may not always be the same as the user-defined scale.

Related Topics

- Previewing IPF Files

Plotting Data Using allegro_plot (UNIX Only)

Before running allegro_plot, make sure the `.cdsplotinit` file exists and is correctly configured. For information about this file, see Setting up prerequisite `.cdsplotinit` plotter configuration file on UNIX.

Once created, IPF files may be output to a variety of plotters by using the allegro_plot program, which is based upon the Cadence corporate plotting package, plotServ .

On a UNIX workstation, you can run the allegro_plot program either from a dialog box or as a batch command from an operating-system prompt. Multiple plotting formats are available, as well as the option to output directly to a plotter or to a disk file. The various plotting options may be chosen through command line switches, dialog box buttons, parameter files, or any combination of the three. Plotter queue data can be displayed directly from the dialog box-driven interface. These capabilities combine to provide you with a single environment from which to configure, submit, and track a plot.

For a complete listing of plotter types that are supported by `allegro_plot`, refer to the plotServ documentation. For procedural information on running allegro_plot.

The stipples text file correlates the pen numbers specified in the IPF file created with the create plot command and the colors, line patterns, and fill patterns the plotting device uses. The stipples file may be specified in allegro_plot using the Stipples File field in the Plotter Setup section of the allegro_plot Options dialog box or by specifying a parameter file containing the stipple file designation.

A sample stipple file, `aplot_stipples`, is supplied in

```
<install_path>/share/pcb/text/plot/aplot_stipples
```

and is typically located in a directory such as `/cds/tools`. The file can be used with allegro_plot, but is not required to run it.

File Format

The stipples file is a standard text file, can have any name, and can reside anywhere. This allows you to define multiple stipple files. You may, for example, want to define a different stipple file for each of your available plotters. The stipples file may define colors and stipples for up to 24 different pens. The format of a given stipple definition is shown below. The stipple definition for each pen is broken into four parts.

```
|line color for pen n|

|line stipple pattern for pen n|

|fill color for pen n|

|fill stipple pattern for pen n|
```

Each line in the stipple file must be surrounded by two vertical bar characters (‘|’). Also, there may be no blank lines in the stipple file.

Colors are represented by a series of three integers, each of which must be in the range of 0 to 1000, that represent the red, green, and blue intensities of the color, respectively.

For example, the color bright red would be represented by

```
|1000 0 0|
```

bright yellow would be represented by

```
|1000 1000 0|
```

and dark blue would be represented by

```
|0 0 250|
```

The stipple patterns are represented by a 16x1 matrix in the case of line patterns, and a 16x16 matrix in the case of fill patterns. Each element in the matrix represent a pixel in the plot. Element values may be either a space character or an ‘x’ character where a space indicates that the pixel should be turned off and an ‘x’ indicates that the pixel should be turned on. For example, a solid line pattern would be represented by

```
|xxxxxxxxxxxxxxxx|
```

and a dashed line pattern could be represented by

```
|xxxx  xxxx    |
```

In the example shown below, pen number one is set for a black solid line draw and a red striped fill pattern:

```
|0 0 0|               line color for pen #1 (black)
|xxxxxxxxxxxxxxxx|    line pattern for pen #1
|1000 0 0|            fill color for pen #1 (red)
|x   x   x   x   x   x   x   x   | fill pattern for pen #1
| x   x   x   x   x   x   x   x  |
|  x   x   x   x   x   x   x   x |
|   x   x   x   x   x   x   x   x|
|x   x   x   x   x   x   x   x   |
| x   x   x   x   x   x   x   x  |
|  x   x   x   x   x   x   x   x |
|   x   x   x   x   x   x   x   x|
|x   x   x   x   x   x   x   x   |
| x   x   x   x   x   x   x   x  |
|  x   x   x   x   x   x   x   x |
|   x   x   x   x   x   x   x   x|
```

```
|x   x   x   x   x   x   x   x   |
| x   x   x   x   x   x   x   x  |
|  x   x   x   x   x   x   x   x |
|   x   x   x   x   x   x   x   x|
```

Characters after the second vertical bar (‘|’) are ignored and may be used for comments.

The pens are assigned in order of occurrence. In other words, the first stipple in the file is assigned to pen #1, the second assigned to pen #2, and so on up to the twenty-fourth pattern. If no stipple is defined for a given pen, or if no stipple file is given, then the program defaults to all black colors and all solid stipple patterns.

Stipples File and Electrostatic Plotters

Only the first eight unique colors in the stipples file are used when plotting to an electrostatic plotter. If more than eight are specified, the layout editor matches the new color to the closest of the eight previously defined colors.

The fill_ipf Command (UNIX Only)

The batch command fill_ipf on UNIX workstations fills lines segment by segment by generating an even number of passes for each segment. The first two passes fill the segment's external contour and round its ends in the same way a photoplotter draws lines using a circular aperture. This technique produces lines with good definition of the corners, especially when the line thickness requires many passes of the pen.

After the first two passes, other passes are generated, if required, to fill the internal space of the segments. Arcs with non-zero width are filled with multiple arc passes the same way line segments are filled, except that the ends of the arcs are not rounded.

The fill_ipf command requires a command file called `fill_ipf.cmd` that controls processing. The `fill_ipf.cmd` file is a text file that must exist in the directory where fill_ipf is run. For procedures on how to run fill_ipf, refer to the fill_ipf command in the the layout editor.

fill_ipf .cmd Parameters

The fill_ipf.cmd file comprises the following parameters:

FILL_PEN	Indicates the pen used when drawing and filling all objects that the fill_ipf program are artificially fills. This includes lines, arcs, frectangles, and some figures. Currently, the only figures that can be filled are circles, squares, oblongs, and rectangles. The integer value used for this directive indicates the pen number.
ZERO_WIDTH_PEN	Indicates the pen to be used when drawing zero-width lines and arcs. The integer value used for this directive indicates the pen number.
TEXT_PEN	Indicates the pen to be used when drawing text. The integer value used for this directive indicates the pen number.
UNFILLED_FIGURE_PEN	Indicates the pen to be used for figures that do not get filled. The integer value used for this directive indicates the pen number.
ROUND_FILLED_LINE	Indicates whether or not the lines and arcs are drawn with rounded ends. The Boolean value of Y (Yes) indicates that lines and arcs are drawn with rounded ends. The Boolean value of N (No) indicates that lines and arcs are drawn with square ends.

PEN_WIDTH	Indicates the width of the pens being used to fill the lines, arcs, and so on. The wider the pen, the fewer passes needed to fill a given element. This directive is defined by a floating point value expressed in mils.
PEN_TOLERANCE	Indicates the tolerance of the pen width indicated by the PEN_WIDTH directive. This directive is defined by a pair of floating point values expressed in mils.
PEN_OVERLAP_MIN PEN_OVERLAP_MAX	The PEN_OVERLAP_MIN and PEN_OVERLAP MAX directives indicate the minimum and maximum overlap between fill lines used to fill a given element.The values for both of these directives are floating point values, expressed in mils.
END	Indicates the end of the file directives and is recommended, but not required.

User comments or annotation that precede an END directive require a pound sign (#) at the beginning of each line to identify them as comments. Any comments following an END directive are ignored.

Sample `fill_ipf.cmd` File

Spaces, tabs, commas, or semicolons separate the keyword from the value. The # in the first position of the line identifies comment lines. A carriage return terminates each line. The last line of the `fill_ipf.cmd` file must contain the statement: END.

```
# Sample fill_ipf.cmd file
FILL_PEN     1
ZERO_WIDTH_PEN    1
TEXT_PEN     1
UNFILLED_FIGURE_PEN    1
# The following values are mils
PEN_WIDTH    12.0
PEN_TOLERANCE    -1 +1
PEN_OVERLAP_MIN    2.0
ROUND_FILLED_LINE    Y
END.
```

Control File Format

The format of the control file specifying color-to-pen-to-priority correspondence is as follows. There is a record for each color number (there are up to 31 color numbers, depending on your specific hardware configuration). Each record consists of three fields—the first is the color number, the second is the corresponding pen number, and the third is the color priority number. Part of a typical `.ctl` file is shown below (the column labels are comments and not part of the file).

Color Number	Pen Number	Color Priority Number
1	1	2
2	2	3
3	3	5
4	3	4
5	3	6

In this case, the operator has mapped the last three colors to pen three. You can edit the pen number for each color to any pen number available on your plotter. Note that Gerber IPF files (created using `gbplot`) create a monochrome `.ctl` file with the single record:

Color Number	Pen Number	Color Priority Number
1	1	1

7

Generating Artwork

Artwork is the film, usually mylar, that contains an accurately scaled representation of each layer of a printed circuit/substrate design. In the layout editor, creating artwork is the process of generating files that are used by the manufacturer to physically put the printed circuit design onto film.

The manufacturers of printed circuit boards/substrates use both positive and negative artwork film. In positive film, all graphical elements, such as connect lines, pads, and shapes are dark on a clear background. In negative film, graphical elements are clear on a dark background.

The two artwork processes are vector-based and raster-based. Each uses a different generation of photoplotters. Both produce positive and negative artwork film. The layout editor supports both vector-based and raster-based photoplotter formats.

Input and Output Files in the Artwork Process

When you choose *Manufacture – Artwork* (`film param` command) and then click *Create Artwork* in the Artwork Control Form dialog box or use the batch command `artwork`, the layout editor looks at the ARTPATH environment variable to find the `art_aper.txt` and `art_param.txt` files. If you move these files to a local directory, the layout editor reads the files from the directory specified by the ARTPATH variable, but writes to your local working directory.

The following table summarizes the input files involved in the artwork process. These files are necessary for the manufacturing process.

Artwork Input Files

Type of Input File	Name of Input File	Description	Created When You...
Aperture	`art_aper.txt` (applies to vector only)	A file that associates the size and shape of each aperture used with machine tool code for vector-based artwork only. You cannot change the file name. Be sure to store this file with the board/substrate for manufacturing.	Choose *Manufacture – Artwork* and then click *Apertures* in the dialog box or use the `aperture` command (vector only).
Parameter	`art_param.txt`	A file that describes machine-related parameters. Initially, the layout editor uses default values. When you change the values by modifying the General Parameters tab in the Artwork Control Form dialog box, they are saved in your working directory. You cannot change the file name. Be sure to store this file with the board/substrate for manufacturing.	Choose *Manufacture – Artwork* (`film param` command) and complete the General Parameters tab of the Artwork Control Form dialog box.
Film Control	`<drawing_name>.brd` (stored in database)	Individual records of particular elements of the design to be included together in an output file.	Choose *Manufacture – Artwork* (`film param` command) and complete the *Film Control* tab of the *Artwork Control Form* dialog box.

The following table summarizes the artwork output files.

Artwork Output Files

Type of Output File	Name of Output File	Number of Files Created
`artwork.log`	`photoplot.log`	1
`artwork file`	`<film name>.art` where *film name* is the name of the artwork film provided in the *Artwork Control Form* dialog box.	*n* (same number as artwork film control records)

Changing the Default Artwork Film Filename Extension

You can change the default file extension of `.art` for artwork film filenames by setting the `ext_artwork` environment variable in the User Preferences Editor, available by choosing *Setup – User Preferences* (`enved` command). For example, you might change the extension to `.gbr, gbx, ger,` or `pho,` for Gerber device types. Cadence recommends changing the extension at the CDS_SITE level to ensure your company uses a common extension. Verify that downstream tools can support the extension you choose.

In conjunction with `ext_artwork`, use the `ads_sdart` environment variable to control the the directory to which to save artwork files. Cadence recommends a relative path be used.

Contents of the Aperture List

The `art_aper.txt` file lists the size and shape of each aperture according to aperture wheel. It uses one of the following types of aperture records:

- Single-size geometry records that specify line, circle, and square apertures
- Two-dimensional geometry records that specify rectangular and oblong apertures
- Flash records that specify nonstandard apertures, such as moire patterns, custom pads or flash names representing thermal reliefs

Single-Size Geometry Record Syntax

Single-size geometry records have the following syntax:

```
LINE <size> <machine-code>
CIRCLE <size> <machine-code>
```

```
SQUARE <size> <machine-code>
```

size	A single dimension that describes the size of the aperture in current units (English or metric).

machine-code	The tool code written to the photoplot file to choose this aperture. For the Gerber 6200, the codes are D10 through D29 and D70 through D73. You can enter any string up to 9 characters (with no blanks), depending on the codes that are required by your photoplotter. For example:
	`CIRCLE  250  D70`
	`SQUARE   50  D10`

Two Dimensional Geometry Record Syntax

Two-dimensional geometry records have the following syntax:

```
RECTANGLE <x-size> <y-size> <machine-code>

OBLONG <x-size> <y-size> <machine-code>
```

x-size	Describes the size in the x-direction of the aperture in current units (English or metric).
y-size	Describes the size in the y-direction of the aperture in current units (English or metric).
machine-code	The tool code written to the photoplot file to choose this aperture. In the case of the Gerber 6200, for example, the codes are D10 through D29 and D70 through D73. You can enter any string up to 9 characters (with no blanks), depending on the codes that are required by your photoplotter.

The following syntax shows examples for the Gerber 6200:

```
RECTANGLE  50 100  D20

OBLONG   250 100 D70
```

Flash Record Syntax

Flash records have the following syntax:

```
FLASH <name> <machine-code>
```

name	A string of up to 20 characters that identifies the name of the shape flashed by the machine code. When you use the `artwork` batch command, it matches this name with the names specified in padstacks to find the machine code to use for each flash-type pad specified.
machine-code	The tool code written to the photoplot file to choose this aperture. For the Gerber 6200, for example, the codes are D10 through D29 and D70 through D73. You can enter any string up to 20 characters (with no blanks), depending on the codes required by your photoplotter.

Sample art_aper.txt File

```
WHEEL         1
LINE        5   D10
LINE        6   D11
LINE        8   D12
LINE        10   D13
CIRCLE        5   D17
CIRCLE        6   D18
SQUARE        100   D22
CIRCLE        75   D23
RECTANGLE       75 100  D24
OBLONG        62 80  D25
SQUARE        62   D26
CIRCLE        55   D27
SQUARE        55   D28
FLASH      THRM-REL   D29
FLASH      TARGET    D72
FLASH      MOIRE    D73
```

Sample art_param.txt File

```
DEVICE-TYPE           GERBER6X00
OUTPUT-UNITS          INCHES
FILM-SIZE             2400000 1600000
FORMAT                3.5
ABORT-ON-ERROR        NO
SCALE                 1
G-CODES               NO
OPTIMIZE              YES
STATIONS-PER-WHEEL    999
COORDINATES           ABSOLUTE
SUPPRESS-LEAD-ZEROES  YES
SUPPRESS-TRAIL-ZEROES NO
SUPPRESS-EQUAL        YES
```

MDA Format Output Files

For films that include antipads and thermal flashes, the McDonald Dettwiler (MDA) format needs two artwork files.

The second artwork file has an *_s* suffixed to the file name. For example, when you specify a film named *5v* for a layer that contains antipads or thermal flashes, The layout editor generates the following files:

```
5v.art

5v_s.art
```

The MDA format uses paint and scratch commands. The *_s* suffix is for the file with the scratch commands.

Related Topics

- film param
- artwork
- aperture
- enved

Vector-Based Artwork

Vector-based artwork is the older artwork process. In vector-based artwork, the photoplotter contains a wheel holding different apertures in the photohead. The photohead beams light through one of the apertures and moves across a sheet of photographic film, drawing lines or flashing aperture geometries at specific locations.

The vector-based photoplotter reads a file that specifies how the photoplotter moves its photohead and selects its apertures. The contents of this file are traditionally called Gerber data.

> Because this section covers non-Gerber formats, the term artwork is used unless specifically referring to a Gerber format.

These photoplotters draw lines by selecting an aperture that matches the thickness of the line and shining a beam of light through that aperture as the photohead moves over the length of the line. They draw pads by moving the photohead to the location of the pad and flashing a beam of light through the aperture that is the right size for the pad. They draw shapes by first outlining the shape with a beam of light through a small aperture, then using larger apertures to draw strokes to fill the shape on the film. In Gerber data, all filled-in areas require a series of Gerber data commands and coordinates to draw the lines for filling the areas.

In vector-based artwork process, you may encounter the following problems in plane layers:

- Positive or complex shapes cause longer processing time and the Gerber data can be very long.
- Graphical elements can be so close to each other that the photoplotter has no aperture small enough to draw the space between these elements.
  The layout editor calls this a "can’t fill shape" problem.

You can solve these problems in plane layers by using negative artwork. If you use positive artwork, you can solve these problems by editing the layout so the graphical elements are not too close to each other. These problems do not exist in raster-based artwork.

Vector-Based Pad-Type Behavior

In determining pad-type behavior, the layout editor uses either regular, thermal, or antipad for pins and vias during the artwork process. For vector-based artwork, the layout editor makes a decision based on the film mode: positive or negative. For positive film, the layout editor always uses the regular pad-type. For negative film, it uses either the thermal or antipad depending on whether the pin or via is connected to the shape.

For vector or pad-type behavior in raster formats, use the `-p` switch when using the batch command, `artwork`. The Vector-based pad behavior field is enabled by default in the Film Control tab of the Artwork Control Form dialog box.

Vector-Based Plotter Types

The layout editor supports two vector-based photoplotter format types:

- Gerber 6x00 – vectorizes arcs
- Gerber 4x00 – supports arcs

Prerequisites for Loading Vector-Based Data

Before loading the film for Gerber 6x00 and Gerber 4x00 photo plotter format types:

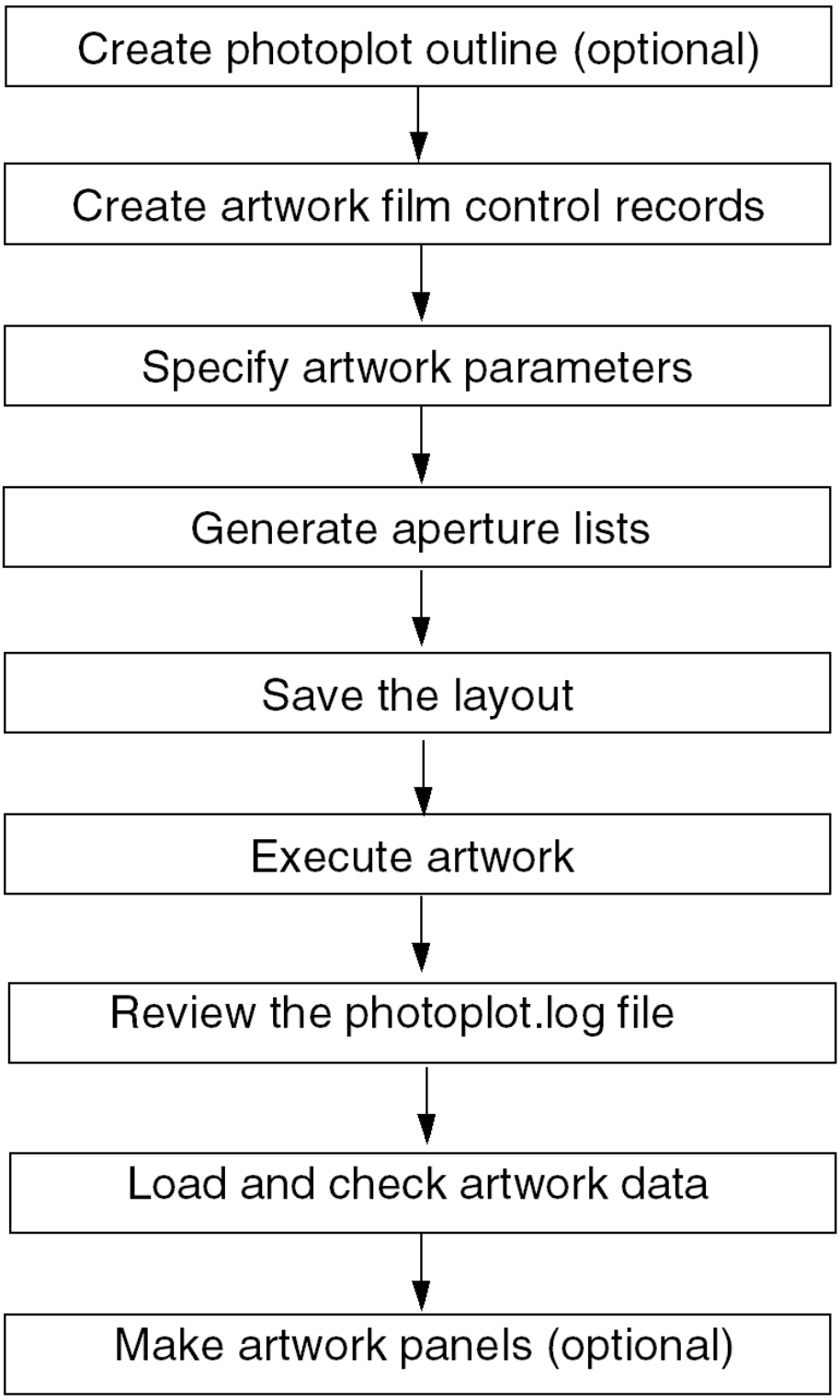
1. Make sure that the appropriate artwork aperture file (`art_aper.txt`) and parameter file (`art_param.txt`) are present.
2. Open a drawing that is at least the same size as the original drawing from which the Gerber file was created. The units and accuracy of the board/substrate should be equal to the units and accuracy of the artwork file.
3. Create a Manufacturing class with a subclass of film names.

Loading data onto *ETCH/CONDUCTOR* subclasses causes DRC for all elements imported to that subclass. For improved performance in artwork review, load the data onto non-*ETCH/CONDUCTOR* subclasses.

The Vector-Based Artwork Process

The following figure shows the vector-based artwork process in the Allegro layout editors.

Vector-Based Artwork Process



To begin the vector-based process:

1. Create the photoplot outline (optional).
2. Create the film control records by specifying the following information:
   - The names of the Gerber data files that the layout editor produces for each layer of the board/substrate
   - Whether the Gerber Data is negative or positive
   - The offset coordinates
   - The classes and subclasses of the graphical elements that the photoplotter draws on the film
3. Specify the artwork parameters by choosing *Manufacture – Artwork* (`film param` command) and complete the General Parameters tab of the *Artwork Control Form* dialog box.
   These parameters tell the layout editor how to prepare the Gerber data files for the photoplotter. Parameter information includes the photoplotter model for which you want Gerber data written, the size of the films, and other printing information.

   > When dynamic shapes are out-of-date, the layout editor displays a *Dynamic Shapes Need Updating...* button on the *Artwork Control Form* dialog box.

   If you try to use the *Create Artwork* button on the *Artwork Control Form* dialog box, an error message appears: "Dynamic Shapes are out of date, please update them." Click *Dynamic Shapes Need Updating...* to open the *Status* tab of the *Status* dialog box, which becomes active, blocking any use of the *Artwork Control Form* dialog box until you update dynamic shapes and/or DRCs before proceeding with artwork.
4. Generate the aperture list. Choose *Manufacture – Artwork* and then click *Apertures* in the *Artwork Control Form* dialog box or use the `aperture` command.
   This list specifies the size and shape, the rotation, and other characteristics of the apertures in the aperture wheel.
5. Save the layout.
6. Execute artwork by clicking *Create Artwork* on the *Film Control* tab of the *Artwork Control Form* dialog (`artwork` command) box and generate the Gerber data files. Also check your design database to ensure that data is valid.
7. Review the photoplog.log file.
8. Load the Gerber data files into a layout to see what the artwork film for a layer will look like.

9. Panelize the layout (optional).

Related Topics

- Steps in the Artwork Process
- artwork
- Artwork Control Dialog Box
- film param
- aperture
- Shapes and Vector-Based Artwork

Raster-Based Artwork

Raster-based artwork is the newer artwork process in which the photoplotter manipulates an image bitmap in memory, then pulses a laser on and off as that laser scans the film. The laser pulses on and off according to the values in each pixel in the bitmap.

The raster-based photoplotter reads an artwork file that specifies the locations of dark and clear areas. These photoplotters can compose layers of dark and clear over each other, for example, a dark shape with a clear void within it. In its bitmap, the photoplotter composes the etch/conductor that remains after it processes the dark shape and the clear void data.

The process of composing etch/conductor in a bitmap from dark and clear layers, then pulsing the scanning laser over the film takes much less time than moving a photohead back and forth over the film to draw connect lines and stroke fill shapes. the layout editor's artwork data files for raster-based artwork are much smaller because they do not contain the strokes the plotter needs to fill shapes, just the outlines of shapes and voids within the shapes. Shape-fill problems no longer exist because there are no physical apertures. Raster-based photoplotters can fill areas less than one mil in size.

Raster-Based Pad-Type Behavior

In determining pad-type behavior, the layout editor uses either regular, thermal, or antipad for pins and vias during the artwork process. Each film record has a Vector-based pad behavior field, which is enabled as the default.

For raster-based artwork, if the default is not enabled and a shape does not contain a pin or via, the layout editor uses the regular pad. If a shape does contain a pin or via (a void is not a shape), the layout editor uses either thermal or antipad. The layout editor makes this decision based on connectivity to that shape.

Cadence recommends enabling the Vector-based pad behavior field with the exception of requirements for donut pads on negative planes. Disabling this field causes Allegro to create composite pads based on padstack information for that subclass. Padstacks can be set up to cause donut pads on planes.

Caution: There is no check of regular pad to antipad sizes. Therefore, on negative planes requiring donuts, all antipads must be larger than regular pads to have clearance between pad and plane.

Raster-Based Plotter Types

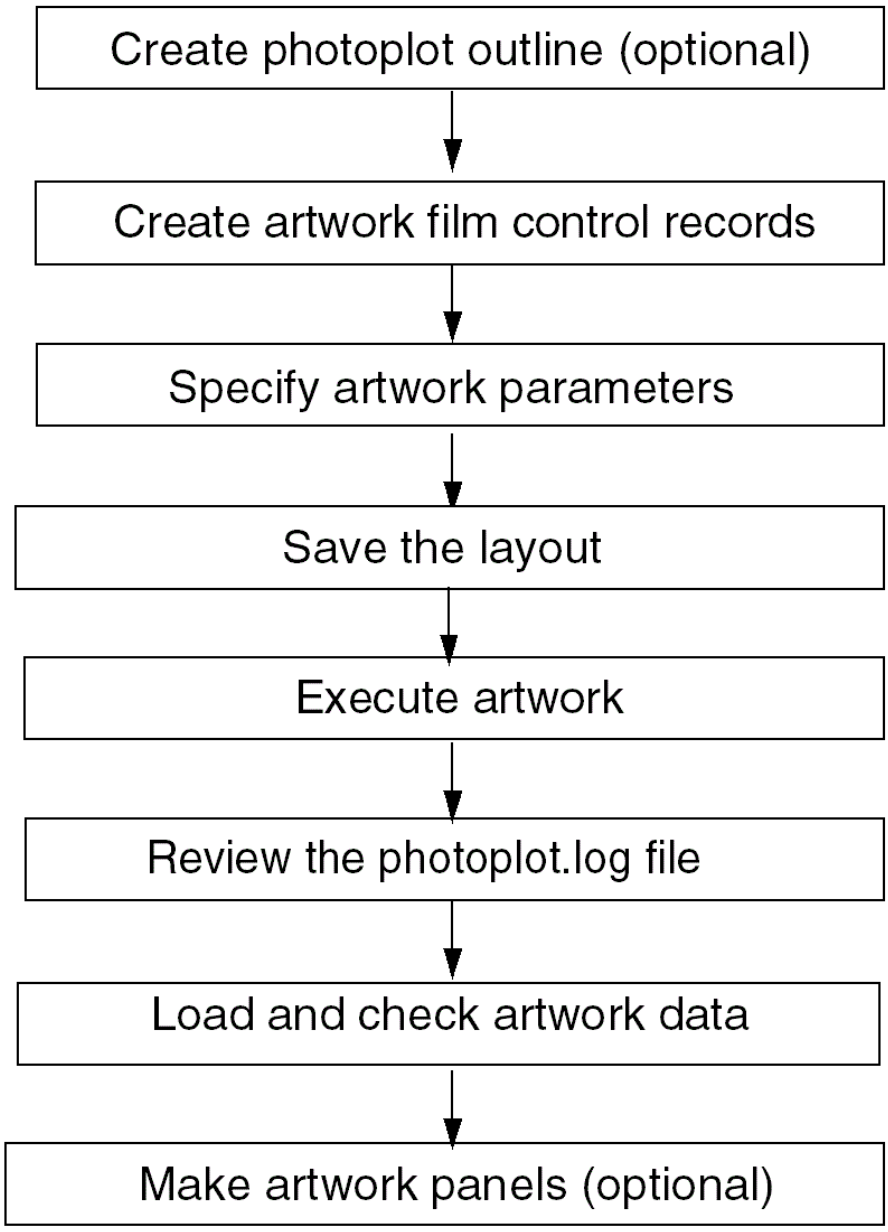
the layout editor supports three raster-based photoplotter format types:

- Barco DPF
- Gerber RS-274X
- McDonald Dettwiler MDA

The Raster-Based Artwork Process

The following figure shows the raster-based artwork process in the Allegro X layout editor.

Raster-Based Artwork Process



To begin raster-based design:

1. Create the photoplot outline.
2. Create the film control records by specifying the following information:
    - The names of the artwork data files that the layout editor produces for each layer of the board/substrate

- Whether the artwork data is negative or positive
- The offset coordinates
- The classes and subclasses of the graphical elements that the photoplotter draws on the film

3. Specify the artwork parameters in the *Artwork Control Form* dialog box.
   These parameters tell the layout editor how to prepare the artwork data files for the photoplotter. Parameter information includes the photoplotter model for which you want artwork data written, the size of the films, and other printing information.

   > When dynamic shapes are out-of-date, the layout editor displays a *Dynamic Shapes Need Updating...* button on the *Artwork Control Form* dialog box.

   If you try to use the *Create Artwork* button on the *Artwork Control Form* dialog box, an error message appears: "Dynamic Shapes are out of date, please update them." Click *Dynamic Shapes Need Updating...* to open the *Status* tab of the *Status* dialog box, which becomes active, blocking any use of the *Artwork Control Form* dialog box until you update dynamic shapes and/or DRCs before proceeding with artwork.
4. Save the layout.
5. Click *Create Artwor*k on the *Film Control* tab of the *Artwork Control Form* dialog box or using the `artwork` batch command to execute artwork. Also check the *Check database before artwork* field to ensure that data is valid.
6. Load the artwork data files into a layout to visually inspect artwork film for a layer.
7. Review the *photoplot.log* file for errors and warnings.
8. Panelize the layout if required.

Related Topics

- artwork
- Steps in the Artwork Process

Steps in the Artwork Process

This section provides additional information for the steps involved in the artwork process.

Panelization is an extension of the artwork process. Once you create artwork files, you can open a new drawing and load as many of these files as needed. Then, generate new artwork data from this file. The fab vendor typically handles panelization. If you want to panelize data, search Cadence Online Support for documents addressing this issue. Exercise caution because panelization reduces DRCs, requires modification of aperture files, and necessitates excessive memory to load data if it is large. Errors can occur when unit/accuracy is not maintained in all steps.

Shapes and Vector-Based Artwork

Shapes determine whether you should use positive or negative artwork for a layer of the board/substrate in vector-based artwork.

In positive vector-based artwork, shapes are filled by a number of strokes by the photoplotter's photohead. Graphical elements, such as thermal-relief pads or antipads inside shapes, increase the number of strokes. A large number of strokes makes the artwork data file large and the photoplotter takes a long time to produce the artwork film. As a result, consider negative artwork for layers that contain embedded or split planes.

In negative vector-based artwork, the background between all shapes and other graphical elements that are outside the shapes, such as pads and connect lines, is filled by strokes of the photohead. The shapes and other graphical elements remain clear. The more of these graphical elements there are outside the shapes, the more strokes required, so consider positive artwork for layers where shapes coexist with other elements.

Raster-based artwork does not use strokes to fill shapes or the background. Choosing either positive or negative artwork for any layer does not make a difference in artwork data file size or photoplotter time to produce the artwork film for raster-based artwork.

Vector-Based Negative Artwork

For negative artwork, the artwork process does the following:

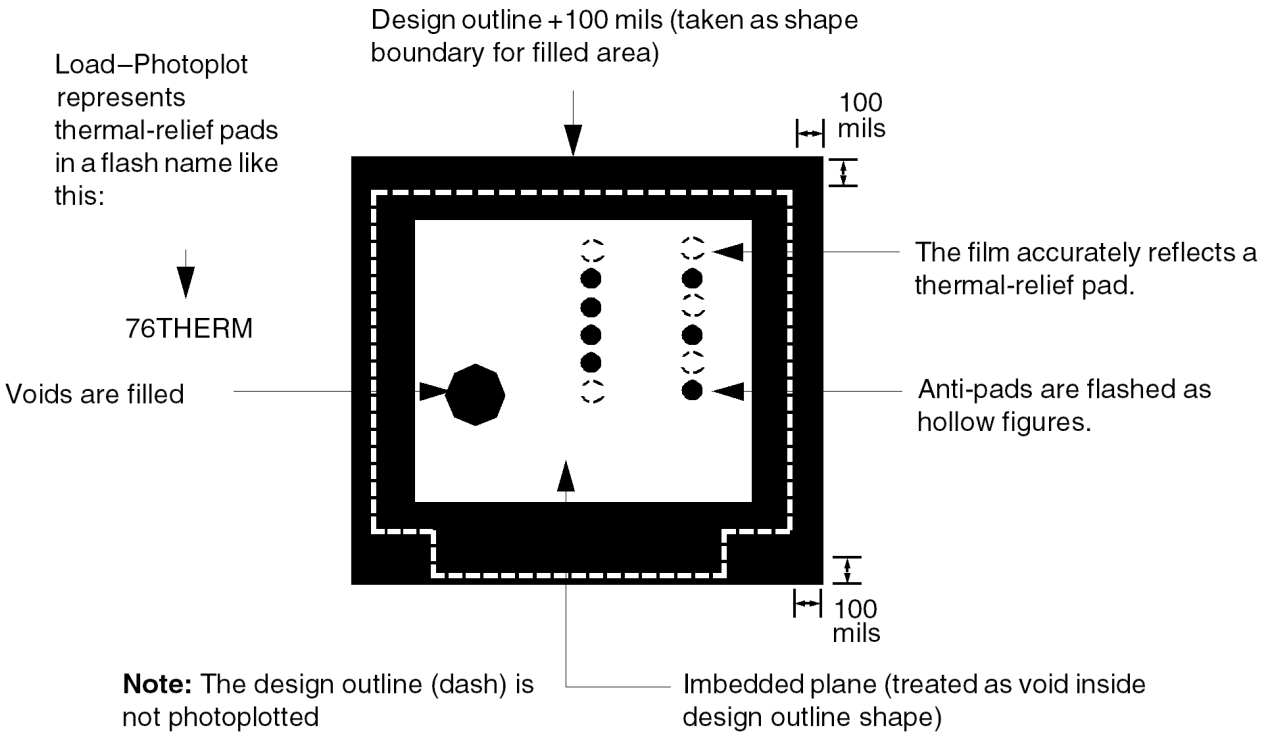
- Adds another outline around the layout outline.
  This new artwork outline extends, by default, 100 mils in all directions beyond the layout outline.
- Fills the area between the artwork outline and the shape(s).
- Leaves all shapes as clear.
  In vector-based artwork, all connect lines, pads, and text that are between the shapes and inside the artwork outline are lost when the photoplotter fills the area between shapes inside the outline.
- Draws or flashes thermal-relief pads for pin or via connections to shapes.
- Draws or flashes antipads for pins and vias that go through shapes and do not connect to the shapes.

In vector-based negative artwork, the layout editor does not include some design data in the negative artwork data file. This lost design data includes:

- Pads that are not inside shapes
- Connect lines between shapes
- Text covered by the negative artwork background
- Tooling corners and crop marks in the negative artwork outline
- Any shape inside a void (because the `artwork` command fills all voids.)
  In vector-based artwork, you can solve shape loss in a void by suppressing the shapefill algorithm that fills the background and voids when you choose *Manufacture – Artwork* (film param command).

For a negative artwork layer, define a thermal flash name to represent connections to the copper plane. In negative artwork, the thermal and antipad definitions in the padstack determine all flashes.

Negative Film



Negative artwork files plot all images on the assumption that there is a photographic reversal of the plotted film for design manufacture. In the figure, everything that is black represents an absence of copper.

When you choose *File – Import – Artwork* (load photoplot command), which verifies artwork, the layout editor displays thermal-relief pads as the flash name with a triangle through it. The film shows the real thermal relief. Give your defined flash names to your photoplotting facility and describe their geometry. See Loading Data and Verifying Gerber Artwork for more information about thermal-relief pads and flash name representations when you choose *File – Import – Artwork* (load photoplot command).

Vector-Based Positive Artwork

Because a large number of strokes makes the artwork data file large and lengthens the time the photoplotter requires to produce the artwork film, consider using negative artwork for layers with embedded or split planes. However, you must use positive artwork in vector-based photoplotting when the layer of a design includes graphical elements between the shapes or in voids inside shapes.

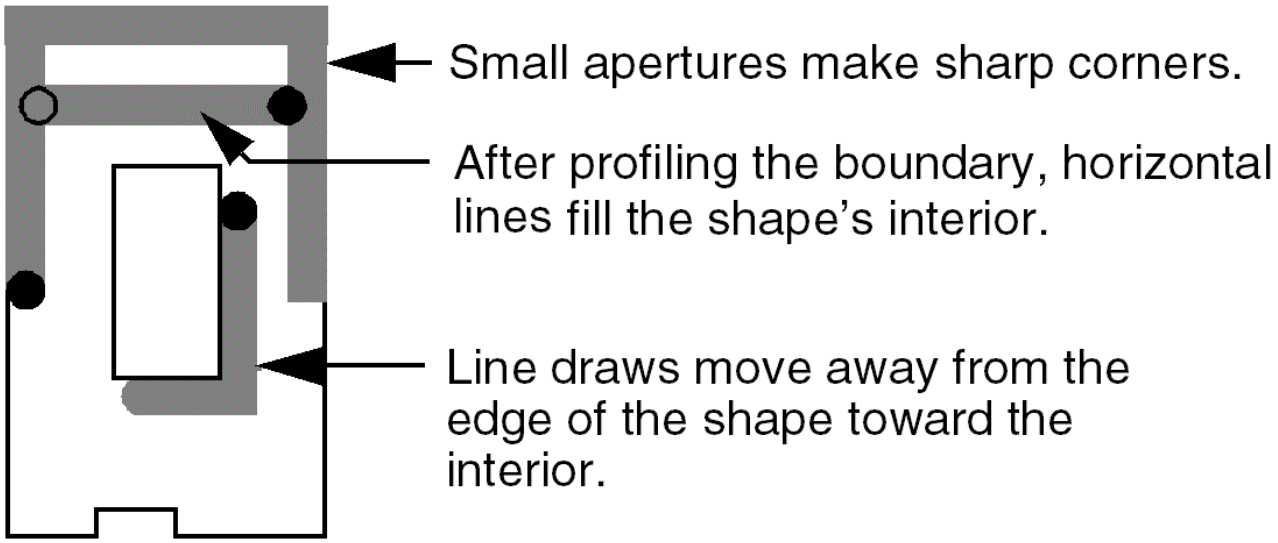
Positive Artwork Profile Lines

When you create a positive artwork file, lines are drawn to cover all interior areas within shapes. The shapefill algorithm produces an accurate outline and an efficient fill pattern. The layout editor uses an even aperture size larger than 3 mils while APD uses the smallest available line aperture to draw a profile along the inside of the shape border and the outside of the voids. The center of this profile line is offset from your shape boundary by half the line-aperture width, so that its edge falls exactly on the boundary that you defined when you added the shape.

After this initial profile line is drawn, the algorithm tries to draw several more profile lines. Each line is drawn with a larger line aperture and is offset further until the largest aperture is selected, or the next largest aperture selected does not fit within the area that must be profiled.

Once the profiling portion of the shapefill is finished, a scan fill tries to fill the remaining unfilled areas using the largest aperture selected in profiling. The following figure shows this technique.

Artwork Shapefill Algorithm



The *Edit Aperture Stations* dialog box creates multiple line widths that can fill most shapes.

Positive artwork files plot all images on the basis that there is no photographic reversal of the plotted film for design manufacture. All etch/conductor lines, pads, filled rectangles, and shapes are plotted as flashes, fills, and lines respectively (and are therefore black on the film), while all voids and open areas are clear on the film. In other words, the plotted areas define where copper is to exist.

Positive Photoplotting

Positive photoplotting does the following:

- Draws all lines, text, and arcs.
- Flashes all positive pads.
- Draws shapes as filled, voids as not filled.
- Draws thermal-relief pad connections based on shape instance parameters.
- Draws antipads as voids around pins that are not connected to the shape being filled.

Controlling Vector Artwork File Size

Even though a positive artwork file is larger than a negative one, running an artwork check can reduce file size necessitated by limited disk space.

The layout editor performs a shape check on static shapes when you choose *Shape – Manual Void – Element* (shape void element command) that examines the shape for any narrow regions that the `artwork` command might have difficulty filling based on the aperture size for shape check listed in the shape parameters. If it encounters a problem, it adds a circular figure to a subclass of the MANUFACTURING class called SHAPE PROBLEMS.

The circular figures that identify problem areas are the same size and color as DRC markers.

> This program prompts you for the smallest aperture in your aperture table (*shapefill* is based on the smallest aperture). A check occurs based on this size.

8

PCB Editor: Backdrilling

To pass high-frequency signals over a backplane, you minimize the effect of plated through hole (PTH) stubs using one of the following methods:

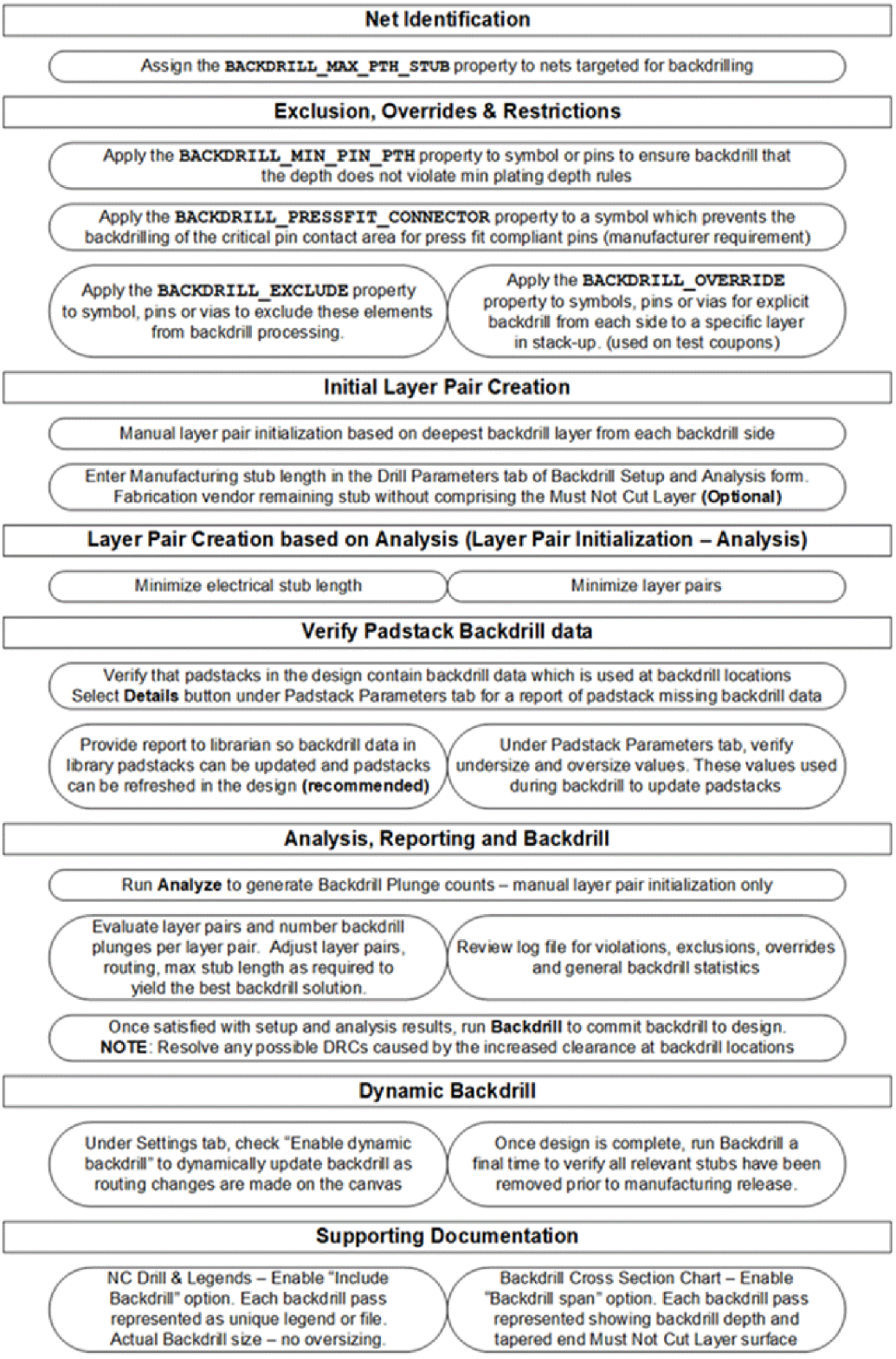
- The full barrel length for signal layer transitions, thus minimizing stubs
- Buried or blind vias
- Backdrilling, a manufacturing process also known as counterboring

In backdrilling, the conductive path of the unused section of the plated hole is drilled out to a controlled depth. The secondary drill diameter must be larger than the primary drill to ensure removal of all deposited metal. Backplanes used in high-end communication systems typically require backdrilling.

The signal-integrity engineer identifies high-speed nets as backdrilling candidates. These nets are most likely constrained with other electrical constraints driven from an ECSET. Vertical stub-length violations determine the backdrilling of plated pins or vias. Knowing constraints on nets allows the designer to optimize routing and reduce stub length from a correct-by-construction approach. For example, a signal may route to a new layer to cut stub length.

The board fabricator backdrills after plating the board. The OEM provides an NC list of plated hole sites, including either the depth to which to drill or explicit layer pairs. In the design flow, layer pairs are used to transfer backdrilling information.

Backdrilling Process



Before You Begin

Backdrilling in Allegro is a flow related application. Nets targeted for potential backdrilling require the property BACKDRILL_MAX_PTH_STUB. The value of this property, which can be applied at the schematic level or inside Allegro X PCB Editor, is the maximum allowable vertical stub in units of length. Prior to backdrilling, do the following:

- Define stackup information for copper and dielectric thickness on the Layout Cross Section dialog box, available by choosing *Setup – Cross-section* (xsection command).
- Assign the BACKDRILL_MAX_PTH_STUB property to nets targeted for backdrilling. Cadence recommends using the General Properties worksheet in Constraint Manager to assign this property. Or you can use *Edit – Property* (property edit command).
- Assign the BACKDRILL_EXCLUDE property to symbols, pins, or vias to exclude them from backdrilling.
- Assign the BACKDRILL_MIN_PIN_PTH property to symbols or pins to ensure the backdrill depth does not violate the minimum plating rules you specify. You can then control how much vertical depth is required for a pin to be plated properly.
- Assign the BACKDRILL_OVERRIDE property to pins or vias to ensure backdrilling always occurs for specific objects to user-defined layer spans; for example, to force backdrilling of a pin or via normally excluded due to being a testpoint on that side.
- Assign the BACKDRILL_PRESSFIT_CONNECTOR property to pressfit-connector component symbols to ensure they can be backdrilled from both board sides with respect to their contact range in the plated thru hole.

To graphically display any of these properties for ease of identification, use the Show Property dialog box's *Graphics* tab, available by choosing *Display – Property* (show property command).

Identifying Nets for Backdrilling

The net-based property BACKDRILL_MAX_PTH_STUB determines which vertical stub-length violations to backdrill. Backdrilling may not be necessary on nets where pins and vias are within the max stub allowance. A value of 0 essentially means drill out the entire stub. The property's length value in database units defaults to all vias and thru-hole pins on the net., although stub-length requirements may differ among net classes. For backdrilling, a pin or via must be on a net with the BACKDRILL_MAX_PTH_STUB property and no BACKDRILL_EXCLUDE property.

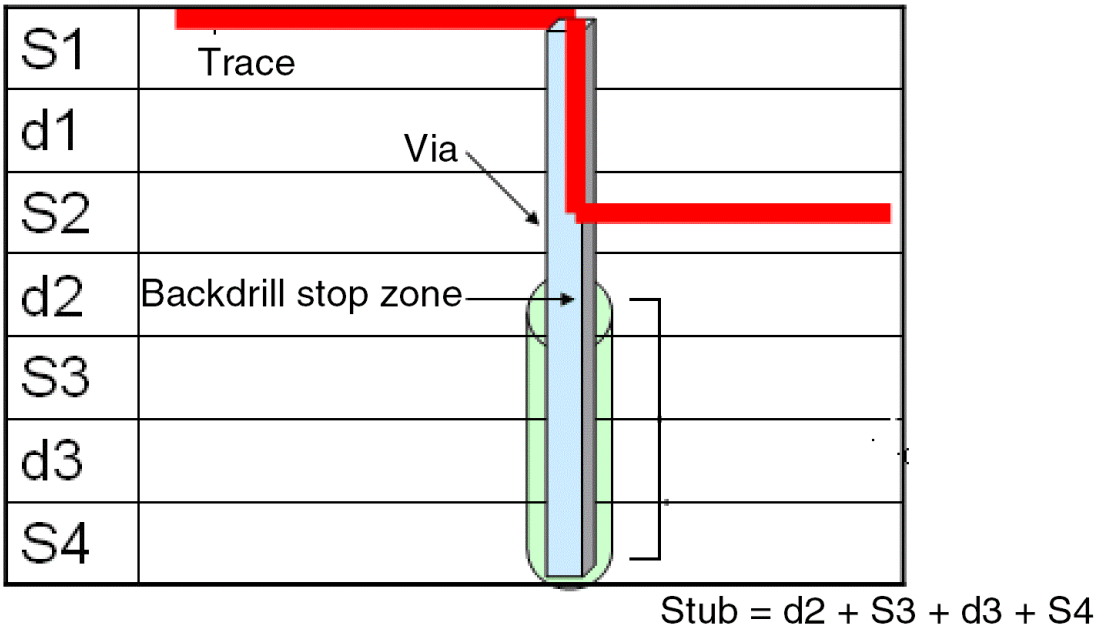
Using stackup information for copper and dielectric thickness from the Layout Cross Section dialog box, the layout editor calculates the plated thru hole stub violation as the distance from the board

top or bottom to the dielectic layer adjacent to the signal trace's padstack entry and exit point.

For a top-side connector, the layout editor calculates the stub calculated from the bottom side inward to the dielectric layer adjacent to the trace entry and exit layer.

For example, on a 4-layer board, a signal traversing layers S1 and S2 produces a stub length that comprises the thicknesses of layers S3 and S4, plus the dielectric thicknesses of d2 and d3. To remove the entire stub, the fabricator programs the drill depth to stop within the dielectric material (d2) between layers S2 and S3, considered the buffer zone. Drilling beyond this point can result in an open circuit

Calculating Stub Length



For vias, a stub may exist from both the top and bottom sides. A via on a 16-layer board that traverse layers 6 and 7 results in a stub from layers 1 to 5, which might be acceptable, and 8 to 16, which is not.

The stub check occurs on each pin or via object in the net, so the view log file displays the net-based object and its descendents (pins and vias).

Excluding Elements from Backdrilling

You may suppress certain objects in a net from backdrilling, such as vias in a BGA field or all pins on a soldertail connector, to preclude backdrilling. Typically, press fit connectors are used in applications where backdrilling occurs. A soldered connector may lose its ability to be soldered to the board if the hole barrel is drilled out. Symbols, pins, or vias with the BACKDRILL_EXCLUDE property are ignored during stub calculations.

Even if the stub length exceeds the maximum allowed, a pin or via hole is exempt from backdrilling from a particular side when:

- designated as a testpoint
- its symbol exists on the same side of the board
- no connections exist for it

Assign the BACKDRILL_MIN_PIN_PTH property to symbols or pins to ensure the backdrill depth does not violate the minimum plating rules you specify.

Configuring Backdrill Definitions

When backdrill configurations are created, an analysis occurs to ensure the most aggressive backdrill pass results in enough plated depth to satisfy the manufacturer's requirements.

Choose *Manufacture – NC – Backdrill Setup and Analysis* (backdrill setup command) to define the backdrilling passes on the Backdrill Setup and Analysis Dialog Box.

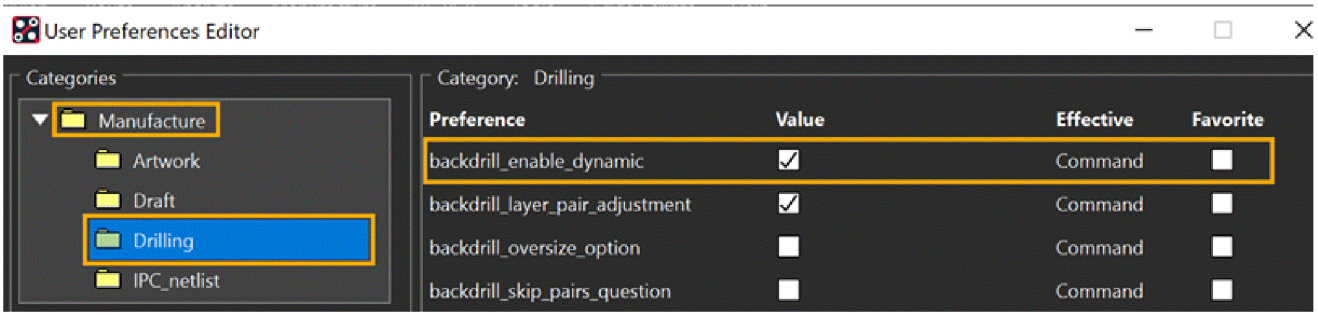
Two system default passes let you quickly assess the result of backdrilling all pins and vias to the maximum depth permitted. All layer combinations are used.

You determine the board side on which to allow backdrilling in the *From Side* column, and the type of holes to consider as backdrilling candidates in the *Objects* column. Define the number of passes from the side top or bottom to the target layer in the *Passes* column.

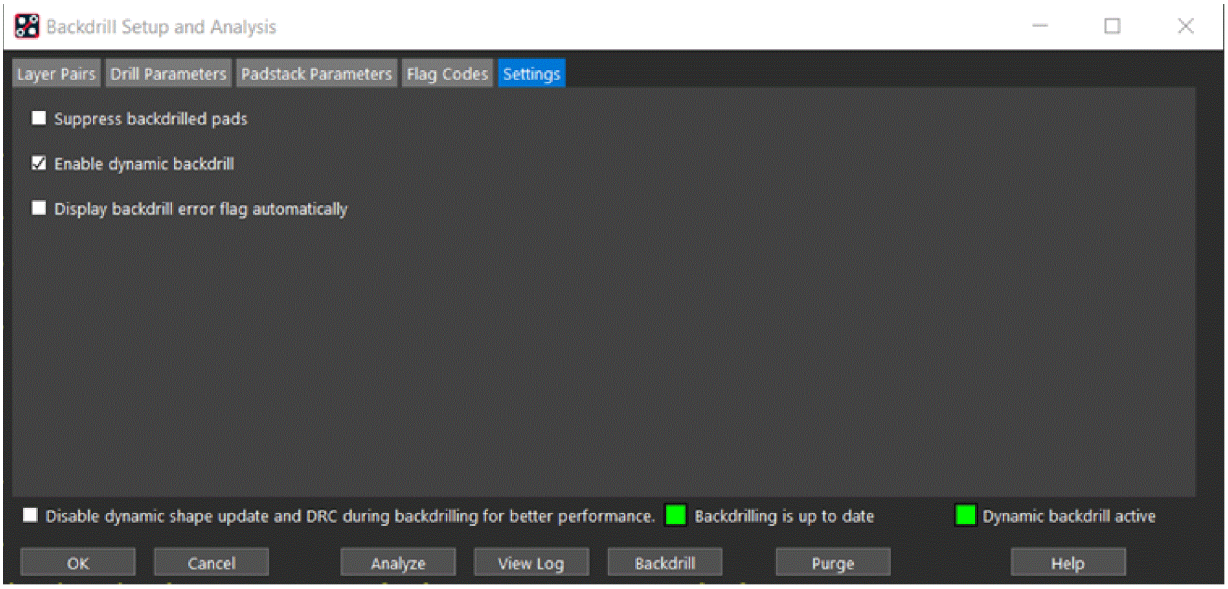
Choose a layer to which to backdrill in the *To Layer* column. For pass-set members, the etch layer names from the specified *From Side* for the initial pass-definition row sequentially populate this column. If you choose no layer, backdrilling occurs to any layer using the specified *From Side.* Then verify in the Depth field the depth of the required dielectric layer beyond the etch specified in the *To Layer* column. Enable the board side and object definitions that guide backdrilling.

Enabling Dynamic Backdrilling

Interactive routing or placement may impact pins and vias, hence make backdrill out of date. By enabling dynamic backdrill mode, the backdrill information gets updated in real-time whenever an edit has been made to a design. To enable this mode, set *backdrill_enable_dynamic* in User Preferences Editor under the *Manufacturing – Drilling* category.



The *Settings* tab of the Backdrill Setup and Analysis dialog starts showing the *Enable dynamic backdrill* option. Select this option to enable dynamic backdrill mode.



When dynamic backdrill is active, the backdrill information is dynamically updated on sliding slide vias, moving components, or changing connection layers. The backdrill process selectively maintains and updates backdrill without re-running full backdrill executions.

Analyzing Backdrilling Results

Experiment with different configurations to determine the layer passes necessary for satisfying the stub allowance requirements. The approach may be aggressive or conservative. If aggressive, the assumption is to drill all combinations of layers to eliminate the maximum amount of stub possible. If a conservative approach is taken, a limited number of passes eliminates stubs to just meet margins set by the BACKDRILL_MAX_PTH_STUB property.

You can generate the `backdrill_analysis.log` report detailing backdrill data by clicking *Analyze* on the Backdrill Setup and Analysis dialog box. This preview lets you evaluate the backdrilling impact of enabled pass definitions. Use the log file to review the number of pins or vias backdrilled from respective sides of the board, excluded objects, stub violations unresolved by altering the backdrill pass definitions, and BACKDRILL_MIN_PTH_PIN violations.

Using Backdrill Legends

Separate legends are created for each layer that is backdrilled to from each side. While dielectric layers are used in analysis calculations, legends are based on conductor layers only. Drill sizes shown in the legend tables are finished sizes only, prior to any backdrilling.

When you choose *Manufacture – NC – Drill Legend,* you can enable the *Include Backdrill* option on the Drill Legend dialog box to create backdrill legends. For backdrill legends, an NCBACKDRILL-<*L1*>-<*L2*> subclass generates on the MANUFACTURING class and groups legend graphics as DRILL_LEGEND_BD_<*L1*>_<*L2*>, in which <*L1*> indicates the from-side layer number; <*L2*>, the to-layer.

In contrast, for *Layer-pair* drill legends, an NCLEGEND-<*L1*>-<*L2*> subclass automatically generates on the MANUFACTURING class and groups the legend's graphics as DRILL_LEGEND_<*L1*>_<*L2*>, where <*L1*> and <*L2*> are the layer numbers of the drilled layers. For *By Layer* drill legends, an NCLEGEND-BL-<*L1*>-<*L2*> subclass generates on the MANUFACTURING class, where *-BL* indicates *By Layer* drilling and groups legend graphics as DRILL_LEGEND_BL_<*L1*>_<*L2*>.

Legend Type	Subclass	Group Naming
Layer Pair	NCLEGEND-<*L1*>-<*L2*>	DRILL_LEGEND_<*L1*>_<*L2*>
By Layer	NCLEGEND-BL-<*L1*>-<*L2*>	DRILL_LEGEND_BL_<*L1*>_<*L2*>
Backdrill	NCBACKDRILL-<*L1*>-<*L2*>	DRILL_LEGEND_BD_<*L1*>_<*L2*>

If backdrilling only to the top or to the bottom layer, then the second layer <*L2*> is suppressed.

Backdrilling on the bottom side of a 4-layer board, for example, results in *-4, -4-3,* and *-4-2* outputs, where <*L1*> is greater than <*L2*>, unlike layer-pair or by-layer drilling where <*L2*> is greater than <*L 1*>.

For Layer Pair and By Layer legends, as you move or delete a pin or via, the drill symbol instance graphics do as well. This does not apply to backdrill legends, and the legends must be re-generated to update the drill symbol instance graphics. Backdrill drill symbol instances are true figure elements in the database that are unattached to the pin or via, which is not the case for the other legends.

The figure-display legend for backdrill violations and exclusions on the BACKDRILL-FLAG-TOP or -BOT subclasses, which generate on the fly when you click *Analyze* on the Backdrill Setup and Analysis dialog box, is as follows:

Backdrill Violation Codes

SQUARE and C	Pin excluded as its component is placed on this side
SQUARE and T	Pin or via excluded as it is a testpoint on this side
SQUARE and X	Pin or via excluded by BACKDRILL_EXCLUDE property
SQUARE and P	Minimum pin length violation
SQUARE and S	Maximum stub length violation
SQUARE and R	Pin or via excluded as it has no routed connections
SQUARE and O	Pin or via has BACKDRILL_OVERRIDE property
SQUARE and W	Pin or via override has etch violation warning (for example, backdrill override extends into a layer with an etch connection)

Using Backdrill NC Drill Files

As with backdrill legends, separate NC drill files are created for each layer that is backdrilled to from each side. While dielectric layers are used in analysis calculations, NC drill files are based on conductor layers only. It is assumed fabricators control the depth to stop somewhere in the dielectric layer, which may vary from vendor to vendor. Drill sizes shown in the files are finished sizes with no attempt to list actual drill bits for backdrilling. The overage varies among vendors.

Drill Type	File Names
Layer Pair	<*name*>-<*l1*>-<*l2*>.drl
By Layer	<*name*>-bl-<*l1*>-<*l2*>.drl
Backdrill	<*name*>-bd-<*side*>-<*l2*>.drl

<*side*> denotes the board top or bottom from which backdrilling occurs; <l2> is the board layer to which backdrilling occurs.

9

IPC2581 Spec Definitions

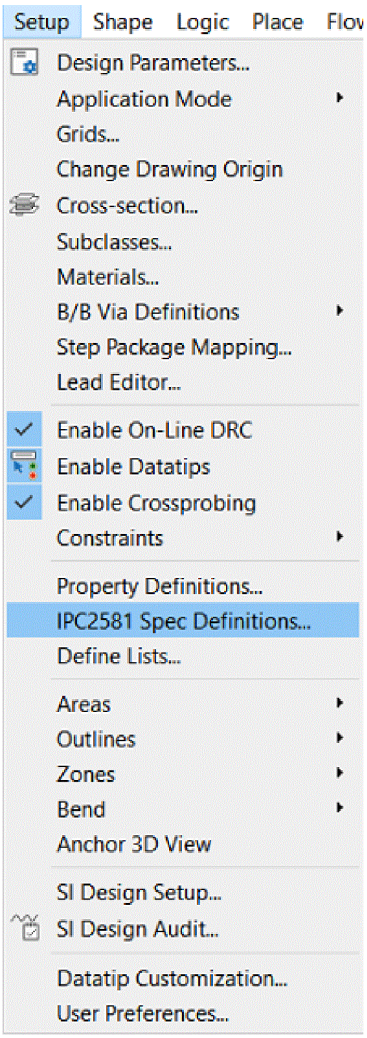
Board fabrication process details are usually contained within notes added to the board fabrication drawing. Assembly drawing also include notes associated to specific parts that describe the assembly process. These fabrication and assembly notes are exchanged through plot images or paper/electronic documents. This process of transferring information between designers and manufacturing or assembly facility can be simplified by using capabilities of IPC2581 data exchange format.

IPC2581 is a manufacturing data file that contains information required to fabricate, assemble and test a PCB design. The Spec element type of IPC2581 data is used to define characteristics of stack-up materials, drilled holes, backdrill, and other miscellaneous attributes for graphical and electrical elements. This Spec element can be used to pass fabrication notes or assembly instructions. Allegro layout editors provides the ability to define the Spec details as IPC2581 data that can be directly read by IPC2581 viewing tools and does not require additional documents.

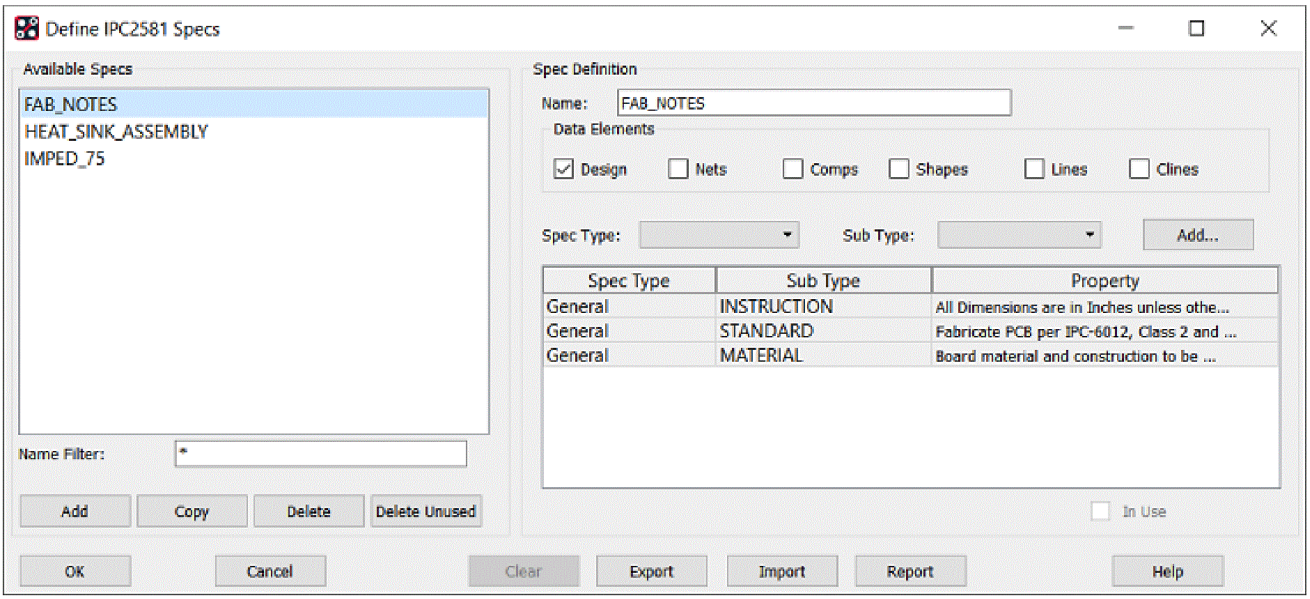
The IPC2581 spec is an XML element and may contain one or more items. The spec can be attached to the board drawing, components, or any design element that requires the passing of design intent for manufacturing. Once created, these specs can be imported into other designs.

Defining IPC 2581 Spec

To create a spec, choose *Setup – IPC2581 Spec Definition* or run `define ipc spec` command.



The command invokes Define IPC2581 Specs dialog to add, edit and delete spec definitions. A spec can have multiple entries for different requirements and can be assigned to multiple design objects. For example, fabrication notes, assembly instructions on a part, impedance value for clines, and so on.



First specify a name and select design elements for which the spec is being created. The IPC2581 standard contains a specific set of spec types. The spec types and associated sub types are listed in the following table:

IPC-2581 SPEC Types and Subtypes

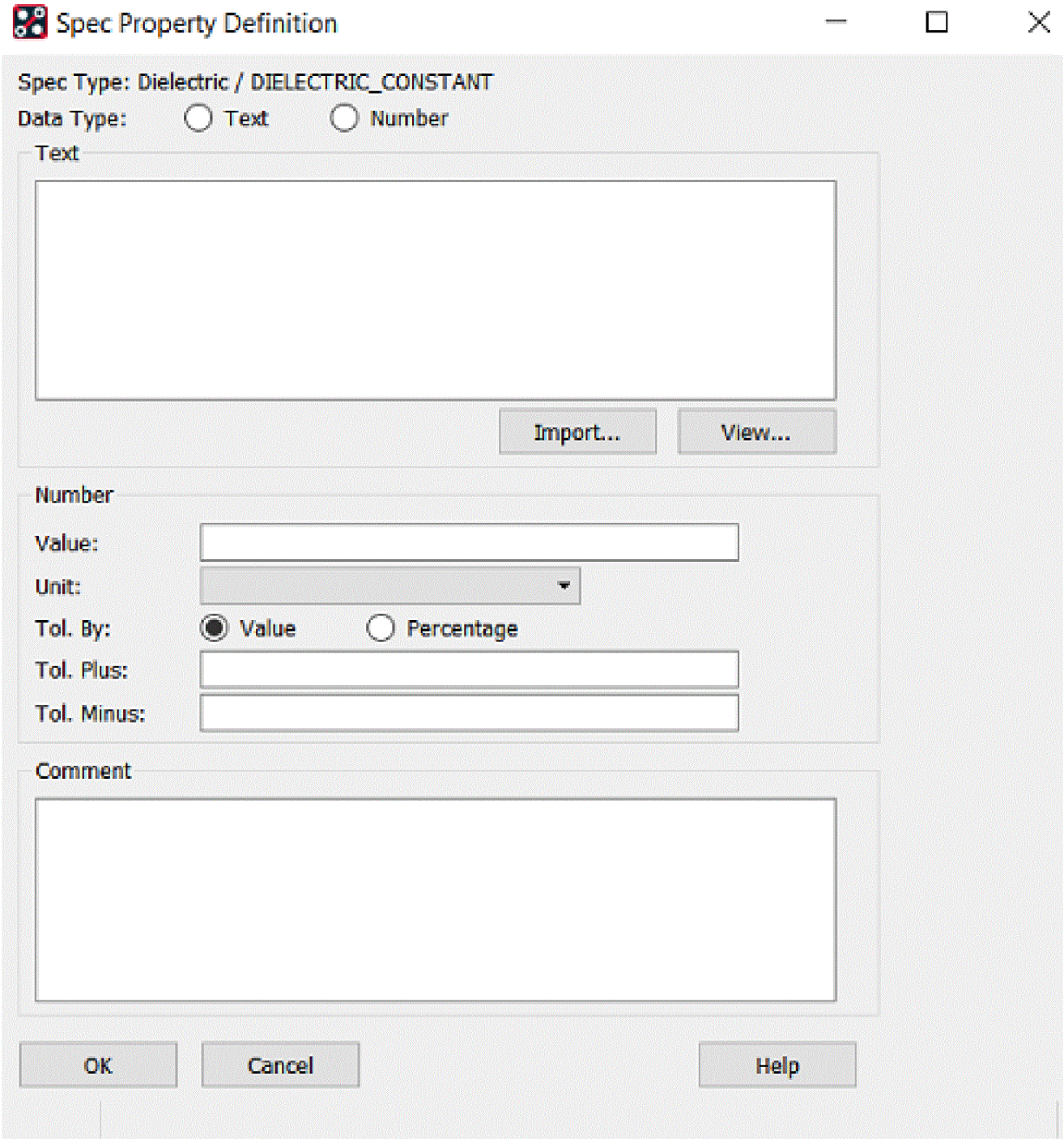
Spec Type	Spec Subtype
Compliance	ROHS
	Conflict_minerals
	WEEE
	REACH
	Halogen_free
	Other
Conductor	Surface_roughness_upfacing
	Surface_roughness_downfacing
	Surface_roughness_treated

	Etch_factor
	Finished_height
	Other
Dielectric	Dielectric_Constant
	Loss_tangent
	Glass_type
	Glass_style
	Resin_content
	Processability_temp
	Other
EdgeChamfer	Angle
	Width
	Side
General	Electrical
	Thermal
	Material
	Instruction
Impedance	Impedance
	Line_width
	Spacing
	Coplanar_Ground_Spacing
	REF_PLACE_LAYER_ID
	OTHER
Technology	RIGID

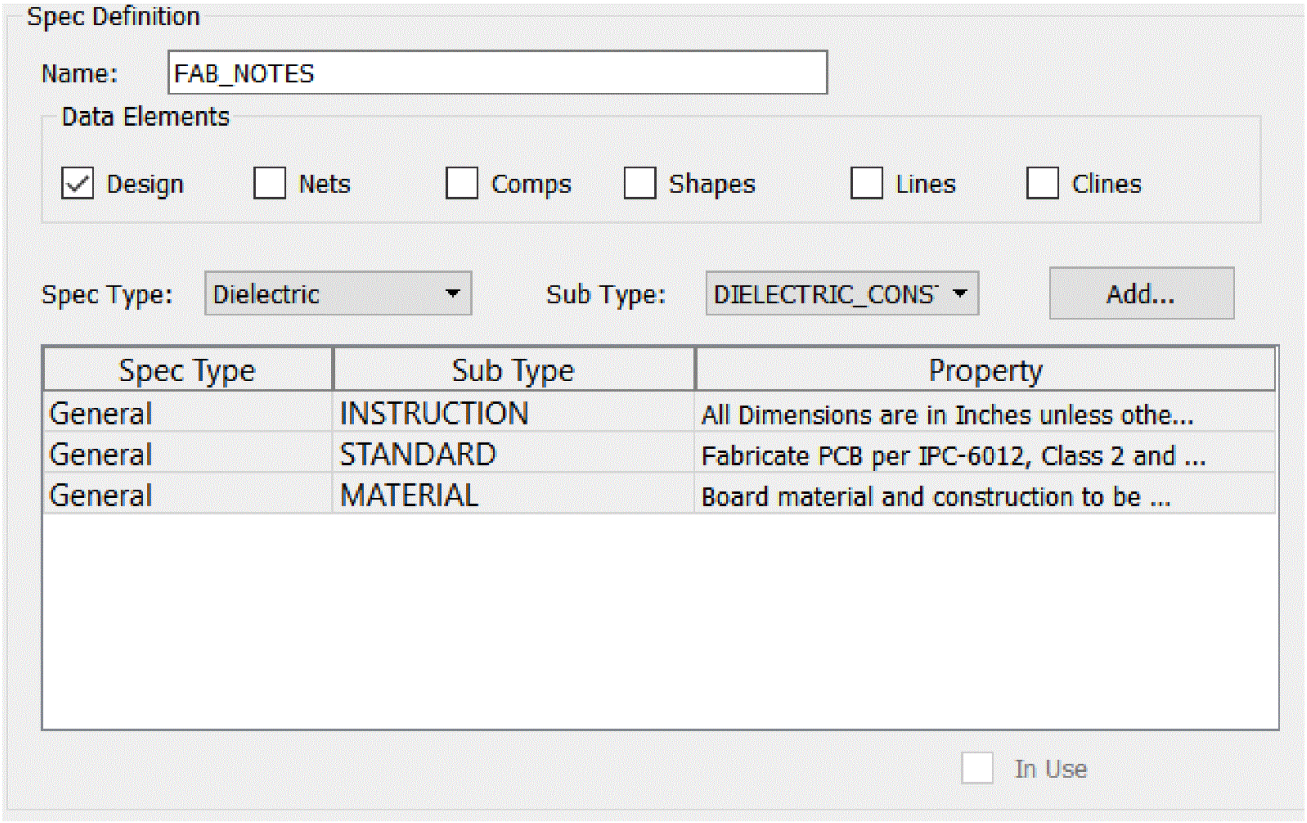
	RIGID_FLEX
	FLEX
	HDI
	Embedded_Component
	Other
Temperature	Thermal_Delamination
	Expansion_Z_Axis
	Expansion_X_Y_Axis
	other
Tool	Carbide
	Router
	Laser
	Flatnose
	Extension
	V_CUTTER
Thieving	Keep_In
	Keep_Out
V_Cut	Angle
	Thickness Remaining

For more information on each attribute, see IPC-2581 specification.

After selecting spec type click the *Add* button to Property Definition dialog. You can add the details as text string or numerical value.

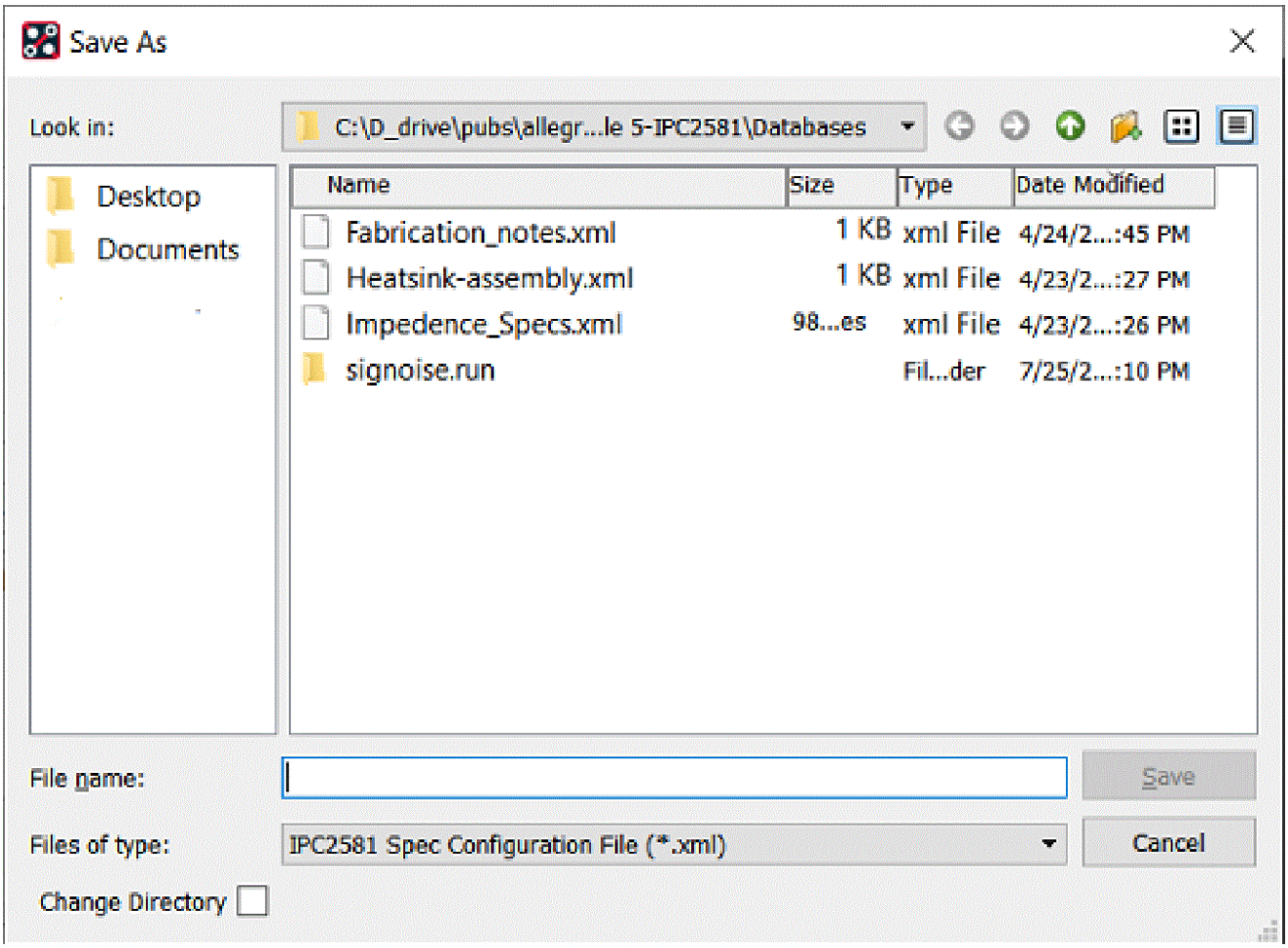


Once added the, spec property becomes associated with the spec and can be modified by the selecting it.

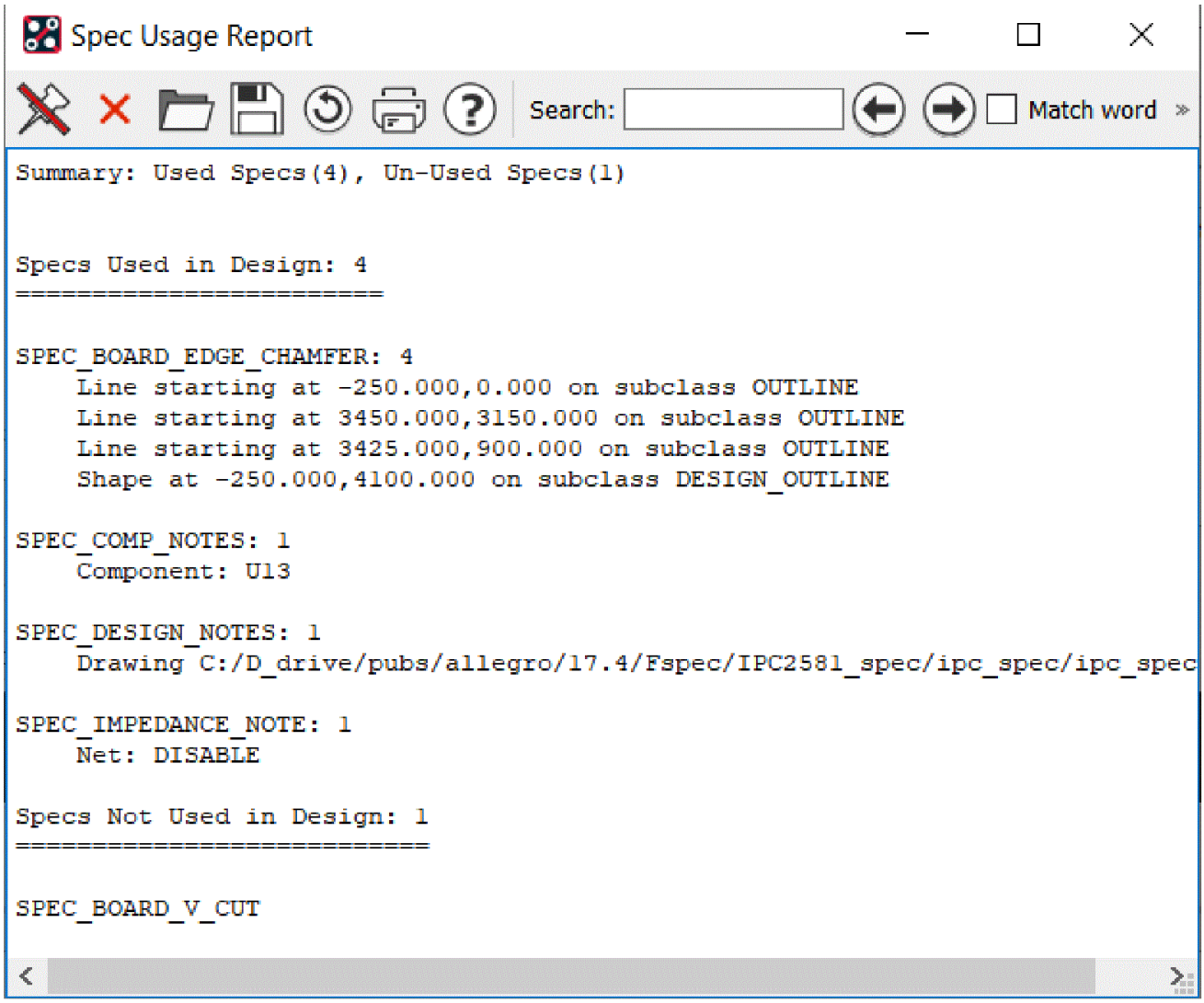


Exporting and Importing IPC 2581 Spec

Specs are saved as IPC2581 Spec Configuration file as an `.xml` file at the location specified by the environment variable *ipc2581spec_path*. To save the spec, click *Export* and enter a name to save the file. Once saved, the spec definitions can be imported into other designs.

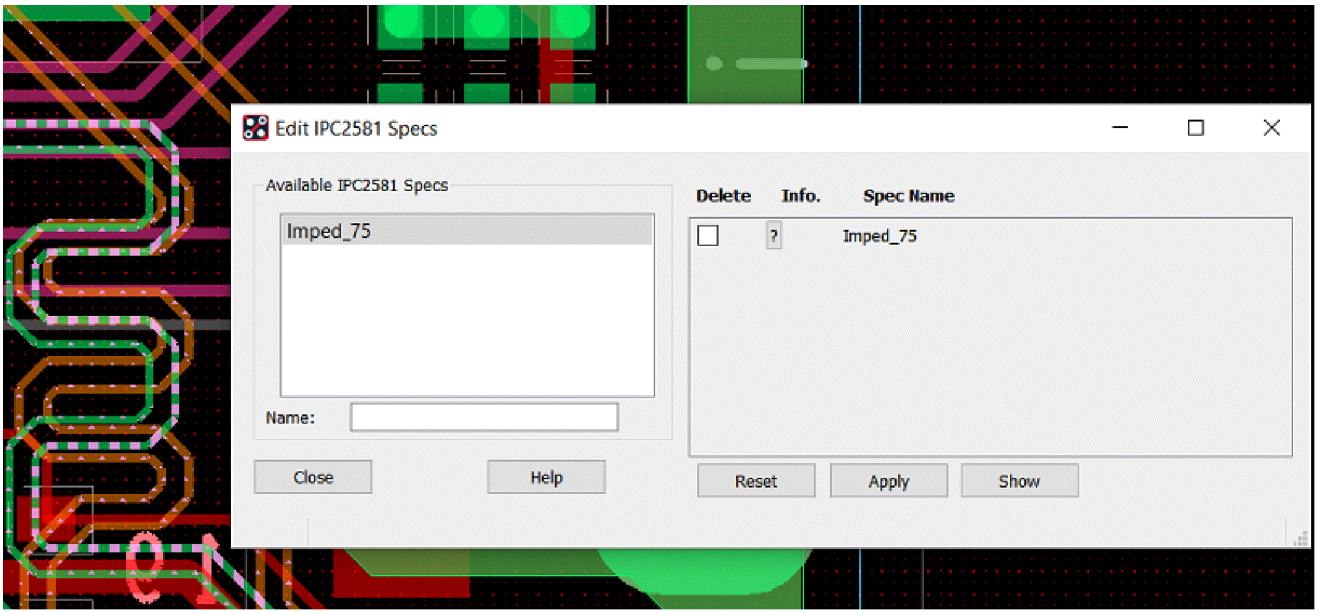


You can also review the status of all the spec definitions for a current design using *Reports* option. A Spec Usage Report shows the name and state of each spec.

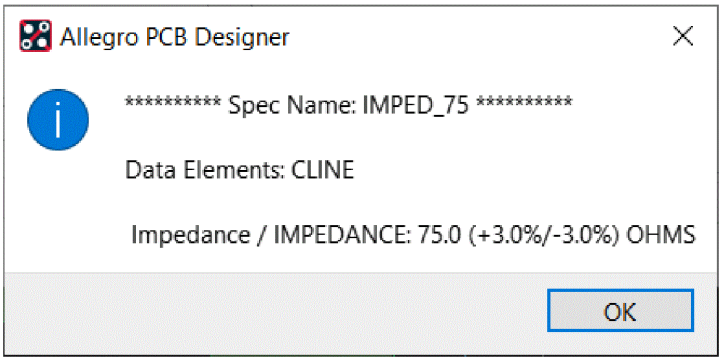


Adding IPC2581 Specs to Design Elements

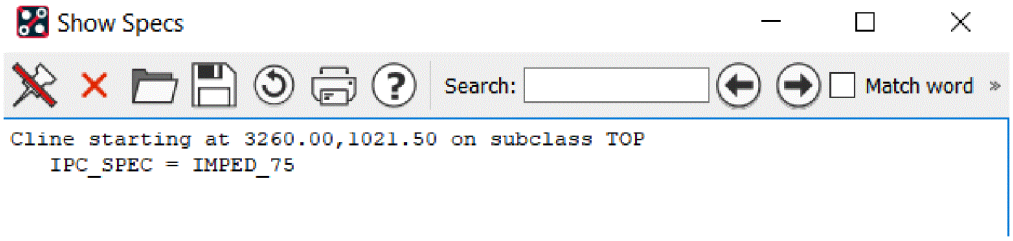
To assign IPC 2581 specs to design objects, choose *Edit – IPC2581 Specs*. Set the *Find* filter and select an object in the design canvas. The Edit IPC2581 Specs dialog is displayed showing all the specs created for the selected object.



You can view the spec definition before adding. Click *Info.* button to view the spec definition.



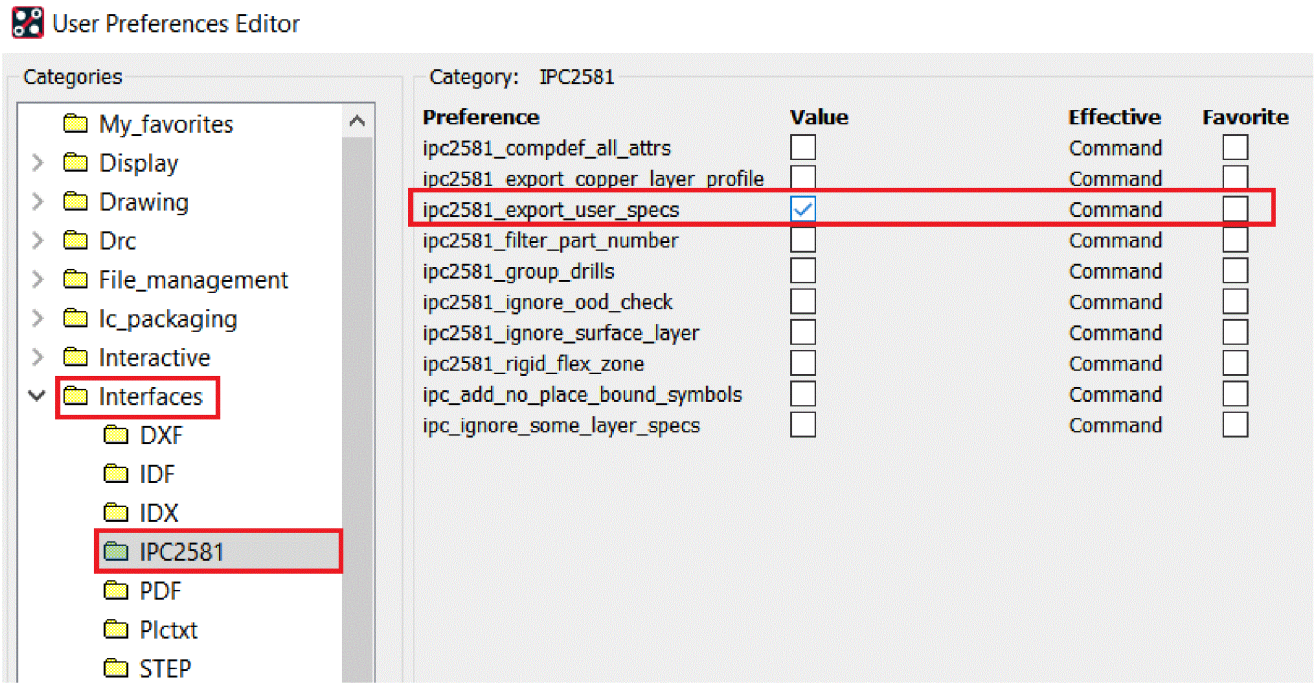
On applying the spec, the Show Spec window appears displaying the spec property name and value.



To delete a spec property from an object, select the *Delete* checkbox in the Edit IPC2581 Specs dialog. If you want to delete all spec references from objects in a design, use *Clear* button in the Define IPC2581 Specs dialog.

Exporting Specs to IPC 2581 Output

To enable export of IPC2581 Specs during the IPC 2581 export process, set an environment variable *ipc2581_export_user_specs* in the *Interfaces – IPC2581* category in the User Preferences Editor.



Can be exported only with IPC2581 Rev B or higher

Spec Examples Defined in the IPC2581 Output File

```
Spec Definitions in IPC-2581:
<Spec name="FAB_NOTE">
        <General type="INSTRUCTION">
          <Property text="All Dimensions are in Inches unless otherwise specified"/>
          </General>
        <General type="STANDARD">
          <Property text="Fabricate PCB per IPC-6012, Class 2 and conform to IPC-A-600 Class 2
Workmanship using current revisions"/>
          </General>
        <General type="MATERIAL">
          <Property text="Board material and construction to be U.L. Approved and marked on finished
board"/>
          </General>
        </Spec>
      <Spec name="IMPEDANCE_50">
        <Impedance type="IMPEDANCE">
          <Property unit="OHMS" value="50.00000" tolPlus="3.00000" tolMinus="3.00000"
tolPercent="true"/>
          </Impedance>
</Spec>
SPEC assignment to the Design database in the IPC-2581 Output:
<Step name="2581-Spec_Workshop">
        <SpecRef id="FAB_NOTE"/>
SPEC assignment to a trace in the IPC-2581 Output:
<Set net="HDMI_TXC-">
            <SpecRef id="IMPEDANCE_50"/>
            <Features>
              <Location x="0.0" y="0.0"/>
              <Polyline>
                <PolyBegin x="3.23000" y="1.02150"/>
                <PolyStepSegment x="3.23000" y="0.99800"/>
                <PolyStepSegment x="3.22500" y="0.99300"/
```