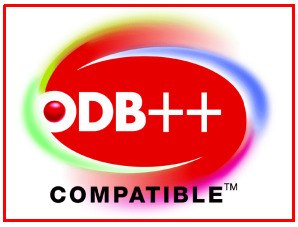


ODB++

Version B.04

Document 0202.0801
Published August 2001

Valor Computerized Systems, Ltd.,
4, Faran Road
P.O.Box 152, Yavne Industrial Area
70600 Yavne, ISRAEL
Voice: +972-8-943-2430
Fax: +972-8-943-2429
Web site: http://www.valor.com
Documentation email: doc@valor.com

Chapter 5 Job>steps Entity

Chapter 6 Job>steps>layers Entity

Chapter 1 *Overview*

Introduction

This book contains the full description of the ODB++ CAD/CAM/DFM data exchange format. ODB++ is widely accepted as a practical de-facto standard within the electronics industry as an efficient way to move printed circuit bare-board, assembly and test data on the manufacturing-engineering level within design/manufacturing supply chains. It is designed as a simple yet comprehensive description of all entities needed in the manufacturing of a printed circuit board.

Originally defined by Valor, ODB++ is now fully used by 3rd-party tool providers within the industry; Valor is committed to a policy of openly supporting any and all organizations to successfully implement the ODB++ format.

Intended Readers

This book is intended for anyone interested in implementing the ODB++ format, for CAD/CAM applications and for interfacing to logistical supply-chain processes. This specification forms a part of the information kit supplied to members of Valor's “Open Systems Alliance”.

ODB++ Versions

The versioning system assigns a version number to each entity in the ODB++ database. This allows updates to be made to each entity separately without affecting other entities. Whenever any entity is updated, a comprehensive ODB++ version number is assigned and set on the front cover of this document.

Version Number Format

Each entity has a file called **`<entity_name>.ver`** saved next to it (such as **`steps.ver`**) where the file contains the version number in the following syntax:

`<major>.<minor>`

Where **`<major>`** is one character **`(A,B,C ...Z).`** A change in **`<major>`** means forward compatibility is not retained (newer versions of an ODB++ database cannot be read by older versions of the application software).

Where **`<minor>`** is two digits **`(01,02,03 ... 99)`**. A change in **`<minor>`** means that forward compatibility is retained.

Recent Updates

This section lists the changes made to ODB++ and to the documentation since February 2000.

February 2000

<job_name>/steps/<step_name>/stephdr

Two new fields added (**AFFECTING_BOM** & **AFFECTING_BOM_CHANGED**). See "stephdr (Step Header)" on page 57.

<job_name>/steps/<step_name>/eda/data

Two new fields added to the PIN Record Structure (**<etype> & <mtype>**). See "Job>Steps>et (Electrical Test)" on page 123.

<job_name>/steps/<step_name>/netlists/cadnet/netlist

New parameters added to the Netlist (x, **e, & by**). See "netlists (Netlists)" on page 59.

<job_name>/steps/<step_name>/layers/<layer_name>/components2

New Job Entity. The **components** file describes the original EDA data for a component, while the **components2** file represents component data after processing with Assembly Merge (BOM Merge, Library Merge and Board Merge).

See "components2" on page 92.

<job_name>/steps/<step_name>/netlists/cadnet/netlist

The following parameters have been appended [<x>] [<e>] and [<by>].

See "netlists (Netlists)" on page 59.

September 2000

<job_name>/steps/<step_name>/chk/<checklist_name>/ actions/<action_num>/res/ sres/<layer_name>/meas

New symbols for **<ftype>** and **<fsym>**. See "meas (Measurements)" on page 120.

<job_name>/steps/<step_name>/eda/data

New net Attributes. See "Job>Steps>et (Electrical Test)" on page 123.

<job_name>/steps/<step_name>/eda/net_prp

New net type clearances. See "<job_name>/steps/<step_name>/eda/ net_prp" on page 77.

Conventions and Terminology

Entity Definitions

Core

Data entities marked as "core" contain data that form an essential part of modelling the Printed Circuit Assembly (including all aspects of the PCB bare-board). In essence, "core" entities contain all the information necessary for CAM systems to prepare PCB fabrication and assembly operations.

Supplementary

Non-core entities (supplementary) are included in the ODB++ format to support certain CAM and DFM functions specific to certain solutions vendors. These supplementary entities are open to all, and are maintained in accordance with the specification, in the same way as the "core" entities.

Chapter 2 Design Principles

File System

Hierarchy

ODB++ uses a standard file system structure. A job in ODB++ is represented by a stand-alone directory tree that can be transferred between systems without any loss of data.

The advantages of a directory tree compared to one large file are apparent when a job is being read from disk or saved to disk. The flexible tree structure allows only a small part of the job to be read/saved, avoiding the overhead of reading and writing a large file.

When a job tree has to be transferred to another system, standard ‘tar’ and compression utilities can be used to convert a directory tree into one flat file.

Mandatory / Optional Files

The following list specifies the files that are mandatory, while those not mentioned are optional:

For the Job:

`job/matrix/matrix`

For each Step defined in the Matrix:

`job/steps/<step_name>/stephdr`

For each Layer defined in the Matrix:

`job/steps/<step_name>/layers/<layer_name>/features or`
`job/steps/<step_name>/layers/<layer_name>/features.Z`

There are also links between files that are implicitly defined in the ODB++ definition which create dependencies between one file and another. For example,. the **`/<step_name>/layers/comp_+_top/components`** file contains links to **`/stepname/eda/data`**.

Legal Entity Names

Job name	Layer name	Attribute name
Step name	Symbol name	Attribute string

ODB++ entity names must follow these rules:

- The length of any name should not exceed 64 characters. However, user attribute strings (not names) are determined by the **MAX_LEN, MIN_LEN** fields in the **<job_name>/misc/userattr** Job File (see "<job_name>/misc/userattr" on page 29)
- Use only the following:
  - lower case letters ('a' through 'z')
  - digits ('0' through '9')
  - punctuation - dash (**-**), underscore (**_**), dot (**.**) and plus (**+**)
- Names must not start with dot (**.**), with the exception of attributes which can start with (**.**).

Readable ASCII files

All files in ODB++ are readable ASCII files. This concept provides the advanced user with the capability to read database files for understanding. In contrast, binary databases which are still used in older systems prevent the user from reading database files directly and require a special extraction program to retrieve all or part of the database.

In all files, the **#** character specifies a comment. Lines which start with this character are ignored by the system and are only used for readability.

The line separator can be either <CR> or <CR><LF>, depending on operating system and platform.

Units of Measurement

All units are imperial units (inches, mils).

Large File Compression

One of the reasons vendors have chosen binary databases in the past was the need to conserve space on hard disks. Modern compression techniques are available today and provide excellent compression ratios, especially for ASCII files with repetitive patterns. Large files in ODB++ are saved in standard UNIX compress format. The compression is optional, and any reader of ODB++ database should expect some files to be in either compressed format (.Z suffix) or without compression. The files which are potentially compressed are clearly identified in the following material.

Sum File

Many of the files in ODB++ have an attached hidden file which provides information about them. The name of the attached file is:

.<name>.sum

The file contains the following information:

Size - size of the data file

Sum - checksum of the file (can be enabled/disabled as a configuration parameter)

Date - date in which the file was written, where
format is mm/dd/yy before software version 4.3
format is mm/dd/yyyy after version 4.3

Time - of writing

Version - version of the software in which the file was saved

User - user operating the software when file was last saved

Note No verification of the size and sum is done today when the file is read by the system. This was intended to allow advanced users to modify files manually in extreme cases.

Example of a sum file:

```
SIZE=274
SUM=-1
DATE=05/24/97  (after version 4.3 = 05/24/1998)
TIME=20:05:10
VERSION=03.02 (BUILD 00 FOR HP-UX)
USER=MOSHIK
```

Structured Text files

To improve readability, many of the small files in ODB++ contain expressions of the type:

```
<var>=<value>
```

The main advantages of this structure is readability. The user can open a file and understand its contents without having to refer to external sources.

Example (from the **`stephdr`** file):

```
X_DATUM=0.3
```

A more elaborate structure, which appears in some structured files, describes arrays. Arrays are lists of elements, each one containing several fields. An array element has the following structure:

```
<array_name> {
 <var>=<value>
 <var>=<value>
 ....
}
```

This element will appear a number of times, each time defining an element of the array.

Example (from the `matrix` file):

```
LAYER {
   ROW=1
   CONTEXT=BOARD
   TYPE=COMPONENT
   NAME=COMP_+_TOP
   POLARITY=POSITIVE
   START_NAME=
   END_NAME=
}
```

Line Record Text files

Some of the files in the database are relatively large and saving them as structured text files is impractical. These files are saved as line record text files. Each line contains a multitude of fields, typically separated by space characters. Reading or writing such files without proper reference information is more difficult. Typically, the first character or word in each line defines the type of record which the line describes. In many cases, the line order is important. Certain lines require that the following line will exist in a particular sequence.

The maximum characters in one line are, in general, 500 characters, however there are exceptions. Any line over the defined limit will be truncated.

Example (from the feature file):

```
#
#Feature symbol names
#
$0 r50
$1 r70
$2 r80
$3 r93
$4 ths80x60x0x4x15
#
#Feature attribute names
#
@0 .geometry
@1 .pad_usage
#
#Feature attribute text strings
#
&0 systest_board
&1 term_1
&2 via_1
#
#Layer features
#
L -0.4 -3.6 -0.4 -4 0 P 0;0=0
L -0.4 -4 0 -4 0 P 0;0=0
L -0.4 -1.4 -0.4 -1 0 P 0;0=0
L -0.4 -1 0 -1 0 P 0;0=0
```

Note Hole symbols cannot appear in a feature file.

Angles

Angles are mainly used to position spokes of thermals and to rotate SMDs. The following rules apply:

- Angle values are expressed as integers within the range 0-359, with angle 0 due East with positive values measured counter-clockwise.
- Angles for rectangular thermals can be in 45 degree increments only, whereas they can be other than multiples of 45 degrees in square/round thermals (when not in 45 degrees, the spoke gap will lie along a line extending from the center).
- When the start and end-point of an arc coincide, it is considered a 360 degree arc. There are no single-point arcs in ODB++.

Rotation / Mirroring

- Feature pads are oriented at 90 degree increments, rotated clockwise.
- Mirroring is only on the X axis (left to right, changing X coordinates).
- Diagonal square lines look like rotated rectangles; the endpoints are also rotated (they are not orthogonal).

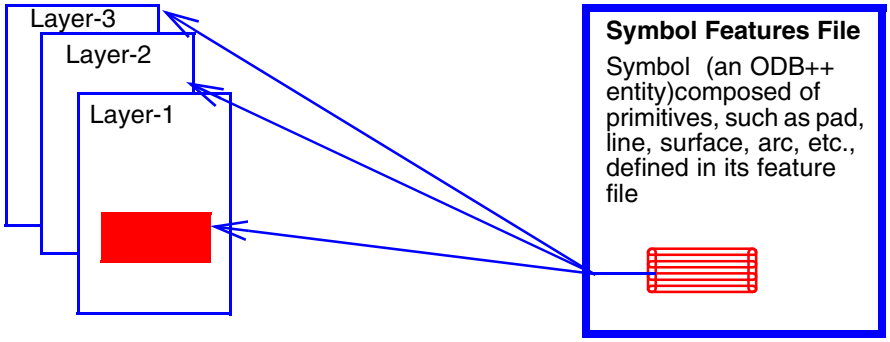
Coordinates

- Coordinate units in feature and symbol files are given in inches with a decimal point.
- When you specify an x,y location for a text string the bottom left of the first character is positioned at the coordinates.
- Point coordinates in a netlist file represent the center of pads.

Symbols

Symbols define a wide variety of shapes (see below) that are mostly used to draw pads. A symbol is an ODB++ entity that is defined once and used many times in order not to repeat the definition of a group of features in a layer. A symbol contains a ‘features’ file that has a number of primitive features (such as pad, line, surface, arc, etc.) that compose the symbol in a layer.

A symbol can be referenced from a number of layers in the job at different coordinates. Changing the symbol definition will automatically cause all its representations in the layer/s to change accordingly.



Symbol can be referenced in many layers. When changed, symbols will automatically change in all layers where defined.

ODB++ supports the following types of symbols:

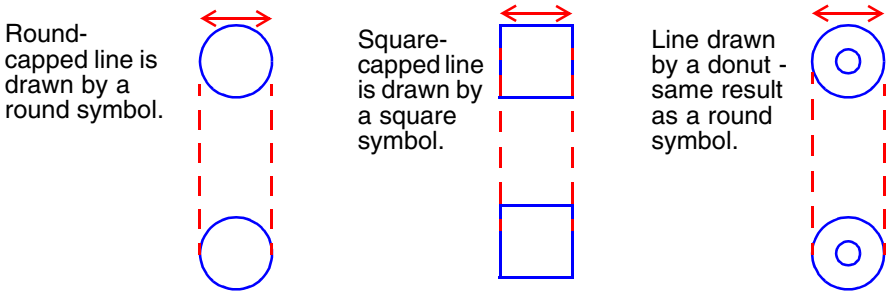
- Standard
- Special (User-Defined)

Standard Symbols

Standard symbols are generated dynamically by the system from their names. They do not require a special graphic symbol entity to be saved in the database. They are round, square or parametric shapes.

For lines, symmetric symbols (where width=height) draw lines with width equal to the width of the symbol, as in figure below:

Width of a line is the width of the symmetric symbol used to draw it.



For example, **`r30`** is automatically generated as a circle feature with a diameter of 30 (mils), **`s200`** is a square with a 200 (mils) diameter (side of square).

Arcs can be drawn with round symbols only.

Pads can be drawn with any symbol. Examples of symbols to draw pads are rectangles = rect width x height (e.g., **`rect100x200`**), ovals = oval width x height (e.g **`oval77x90`**), octagons = hex_s width x height x corner size (e.g. **`hex_s30x50x12`**), and many more.

Units are in imperial units (inches mils).

For example to define a round capped line of width 10 mils, use the symbol '**r10**'. To define the same type line but with a width of 20 mils, specify '**r20**'. To define a square capped line with a width of 10 mils, use '**s10**'.

Drawing a line with an asymmetric symbol generates a one-pixel line ending with the symbol at both ends.

Standard symbols are all positive filled shapes. Holes in symbols are see-thru by definition. The internal implementation of complex symbols uses arcs or contour data with cutouts.

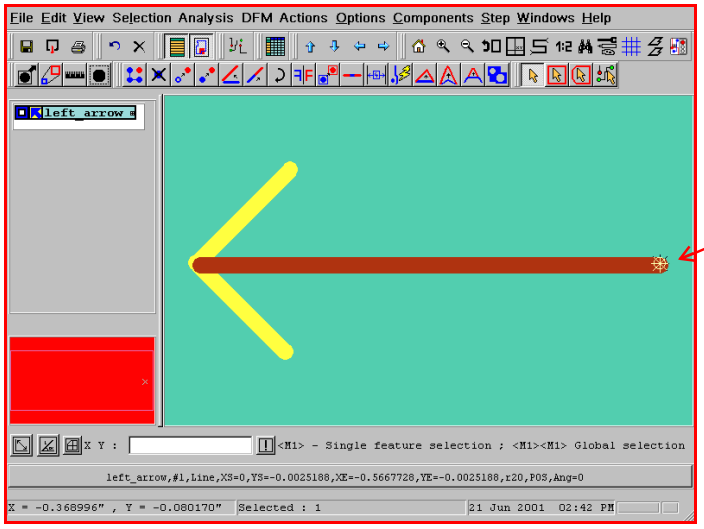
Special (User-Defined) Symbols

Special symbols are user-defined symbols which have a full graphical description stored in feature files in the job's symbols subdirectory. They can contain any number of features. Special symbols are defined for a job usually for shapes not found among the standard symbols. Special symbol names cannot be identical to those reserved for standard symbols.

User-defined symbols can be saved within the system and used when needed. The system recalls the graphic shape defined by the user.

Special symbols are not scalable, such as standard symbols. The reason is that a specific feature file definition is created for each special symbol that defines its shape. Therefore, you need to create a new symbol for each set of parameters. It is preferable to name the symbol to indicate its dimensions.

The figure shows an arrow whose origin (0,0) is at the tip of the butt.



This special symbol is drawn from lines (standard symbols)

When inserting this symbol, the insertion point is at 0,0. This means it is a left arrow

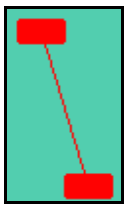
Symbol Characteristics

Asymmetric vs. Symmetric

When asymmetric symbols (such as rectangles) are used to draw diagonal lines, the lines are single-pixel lines whose end-points are the symbols used to draw the lines (see figure below).

When drawn orthogonally (horizontal: y_start= y_end, or vertical: x_start=x_end) the output line has the height/width of the asymmetric shape from start point to end point (the line is created by "dragging" the symbol from one end of the line to the other in the same orientation as it was placed).

For example, if the symbol in the figure is dragged vertically, the width of the line will be the width of the symbol. If dragged horizontally, the width will be the height of the symbol.



Note Single-pixel line width is expressed in the internal software resolution of 10160 pixels = 1 mil, or 400 Pixels = 1 micron, meaning that single-pixel lines are 1/400 of a micron wide.

If the symbol used to draw a line is symmetrical (square or round) the generated line is the side or diameter of the symbol. A round symbol (whose name starts with r) generates round line ends, a square symbol (s) generates square line ends.

For arcs, only round symbols can be used.

Holes in surface features are transparent (empty).

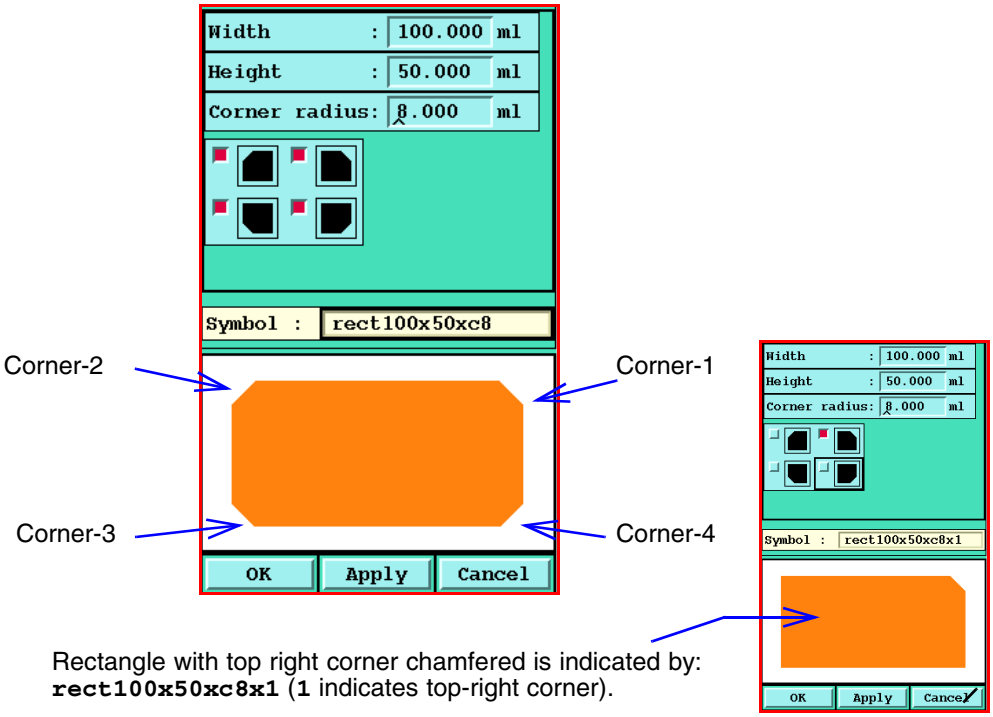
Dimensions of standard symbols can be in non-integers with resolutions up to 1/100 mils

Symbol **`r0`** is a legal entity represented by a single pixel width.

The standard Octagon symbol corner size is the distance between the bounding box corner and the vertex.

Rounded/Chamfered Rectangles

The corners of rounded/chamfered rectangles can be specified in ascending order counter-clockwise, starting from the top-right corner, as in the following figure:



Rectangle with top right corner chamfered is indicated by: **`rect100x50xc8x1`** (**`1`** indicates top-right corner).

Shape

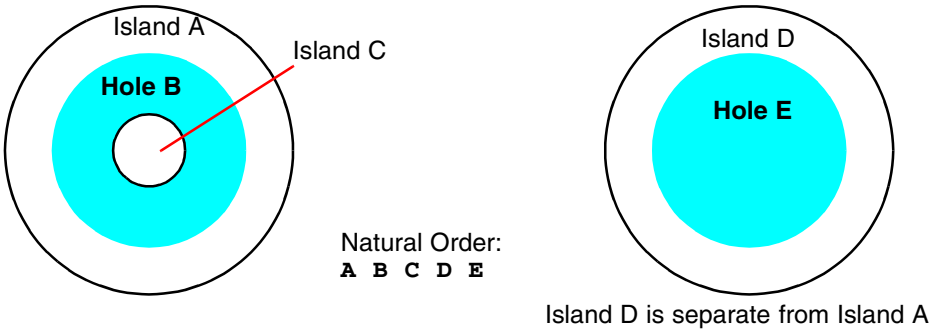
An internal geometrical entity (that may consist of a number of features) used by the system during algorithmic operations. Shapes are always positive. They include points, segments, curves, lines, arcs, squares, rectangles and contours(g). For example, a shape can be a contourized(g) shaved pad that consists of a pad and a feature that shaves it.

Shapelist

An internal data structure which, during analysis, is created for a layer upon demand. The shapelist simplifies the representation of a layer within the system by dealing with multiple polarities, odd shape symbols, etc. A Shapelist can be deleted to improve memory usage; it will be rebuilt by the system automatically when needed.

Order of Holes/Islands in Surfaces

The order of containment of holes and islands within surfaces determines their natural order. The outermost island comes first. Islands precede holes that are contained in them. Holes precede islands that are contained in them. Take, for example, the following containment order:



Chapter 3 Job Tree Charts

Job Overview

A Database job is one directory which is composed of the following subdirectories:

Steps, which are multi-layer entities (e.g. a single image, a sub panel array, a production panel or a multi layer coupon). Each step contains a collection of layers. Layers are two-dimensional sheets, containing graphics, attributes and annotation. Layers express physical board layers, mask layers, NC drill and rout layers and miscellaneous drawings. All steps in one job have the same list of layers, albeit the contents may be totally different.

A **Matrix**, in which the rows are the job layers and the columns are the job steps. The matrix contains for each row additional information such as the type, polarity and context. The matrix is also crucial in defining the physical order of the layers and the relation of drill layers (through, blind, buried, etc.).

Symbols, single layer graphic entities which can be referenced from within any graphical layer in a step.

Stackups, allow the user to develop the optimal 3D buildup, for any given PCB design, in a convenient and efficient manner. This module assures that the demands dictated by the PCB design are strictly met.

Work Forms, user defined collection of fields (textual and graphical) and buttons.

Work Flows, user-defined procedures, composed of stages, conditions and switches in a hierarchical manner.

Attributes, user-defined attributes to facilitate automation.

Wheels, aperture tables created in the Wheel Editor Popup.

Input, automatically identifies the format type of the incoming data (Gerber, Excellon drill, etc.) and interprets the Gerber wheel based on predefined wheel templates.

Output, multiple format translators to choose the output device.

User, where user can store his own files.

Extension, used for third party data files.

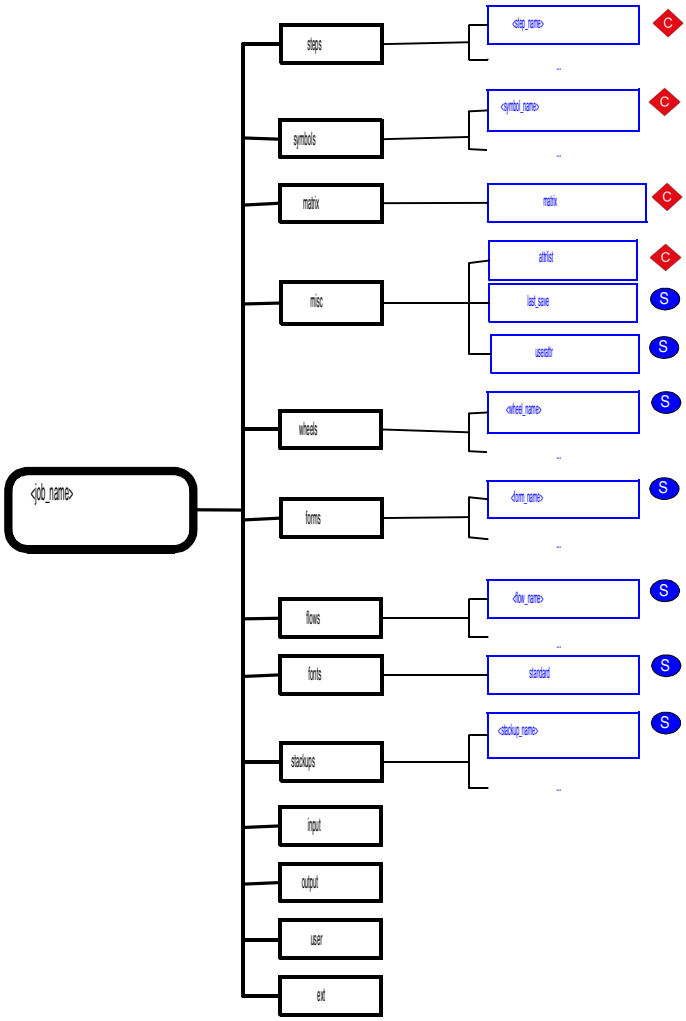
Log, intended for job specific log files.

Each Step entity contains, in addition to general information and the list of layers, several other important subentries:

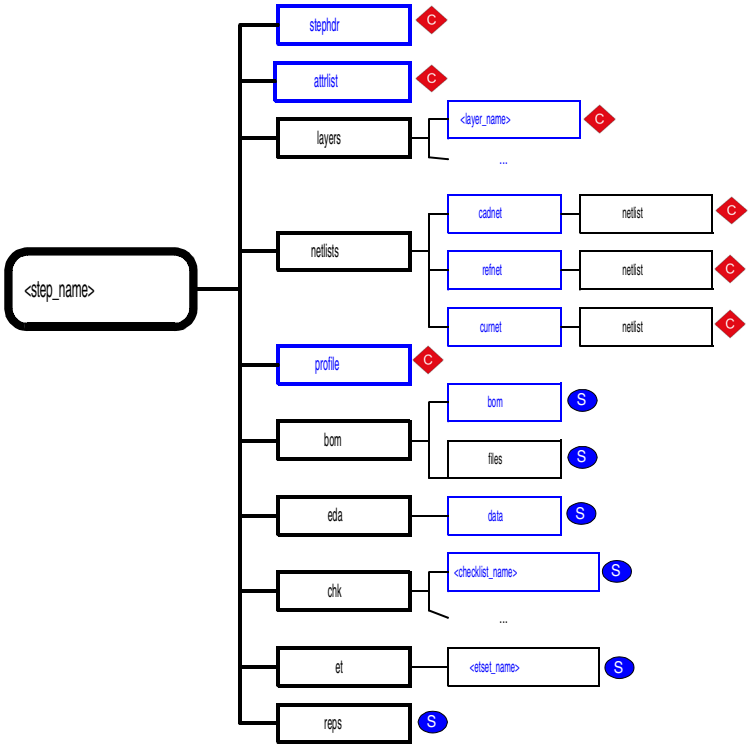
- Step & repeat information (in the stephdr file), specifying any previous steps which are included in this one and their relative location and orientation.
- Up to three netlists of the step (CAD netlist, reference netlist and current netlist).
- An EDA object, containing data regarding the component packages and pins. It also contains information about the relation of features in the board layers to specific design nets and properties imported from the EDA system.
- An unlimited number of checklists, each one is composed of analysis or DFM actions. An action contains the definitions (parameters to run with) and the results (measurements) of the last successful run.
- A profile which is a schematic border around the step.

Charts

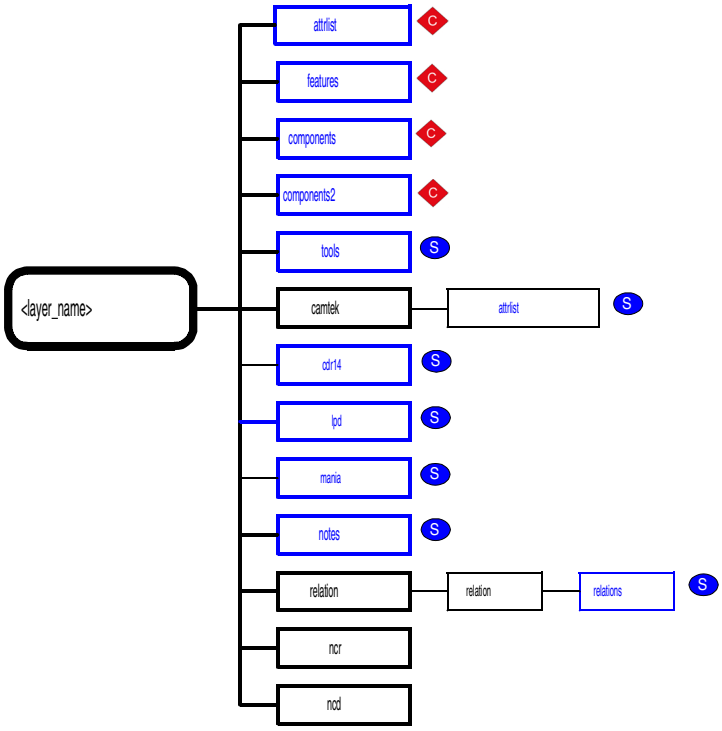
Job Chart



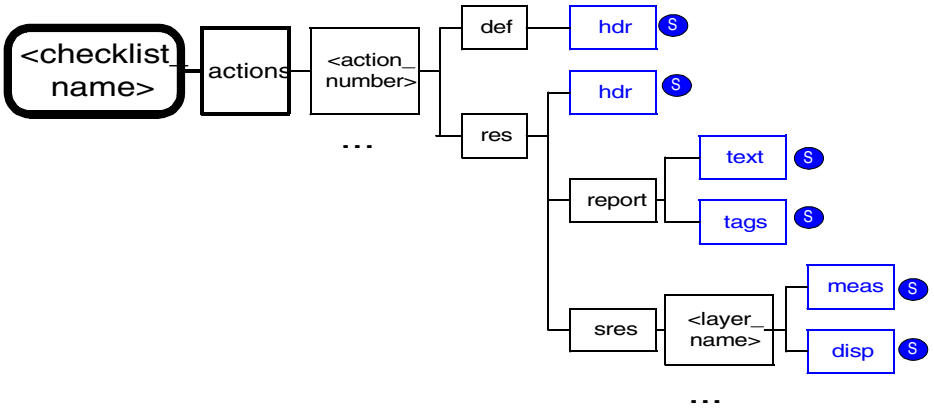
Job>steps



Job>steps>layers

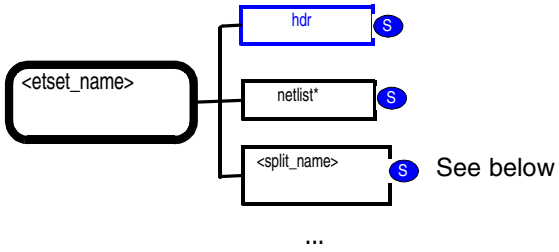


Job>steps>chk (Checklist)

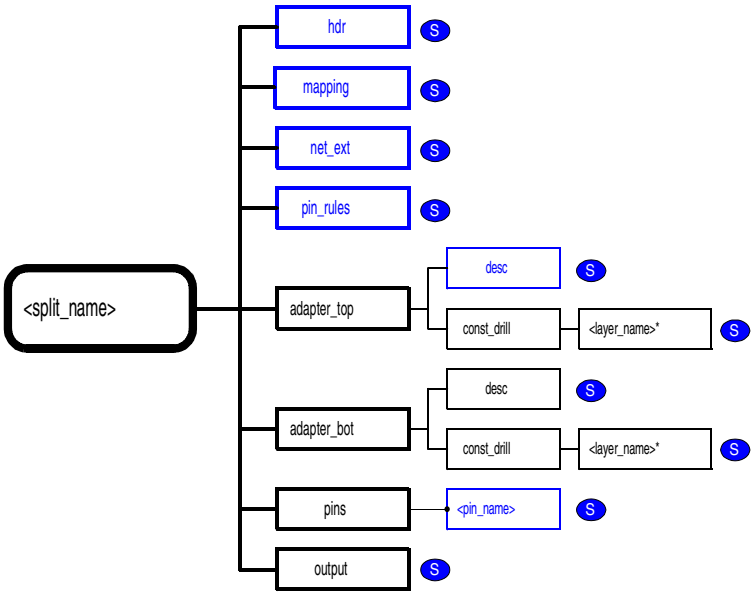


Job>steps>et (Electrical Test)

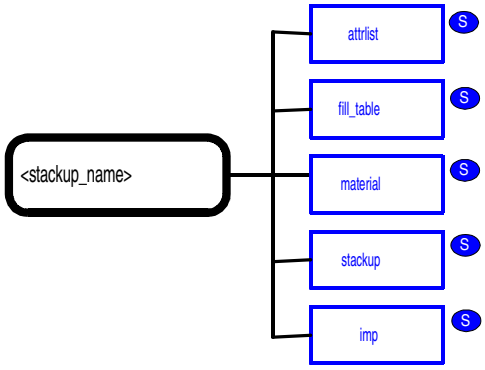
* Netlist is identical in type to the other netlists.



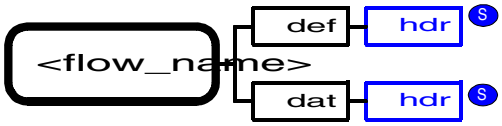
Job>steps>et>split_name



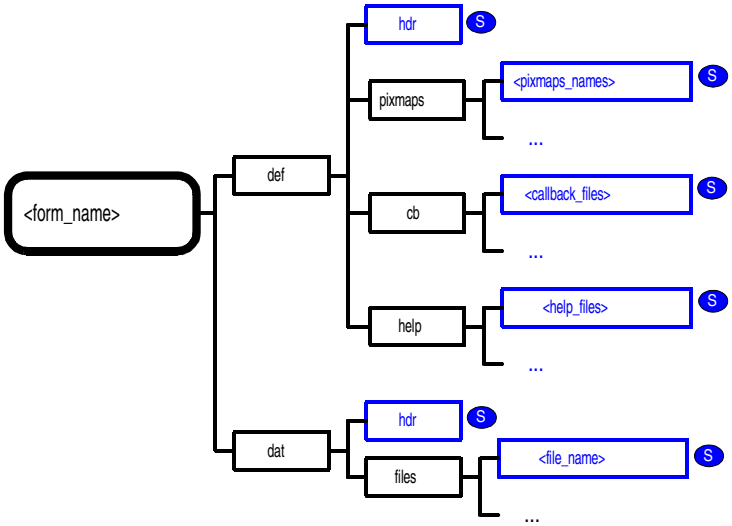
Stackup



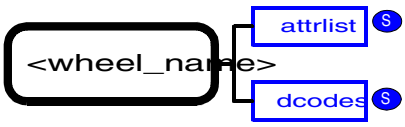
Flows



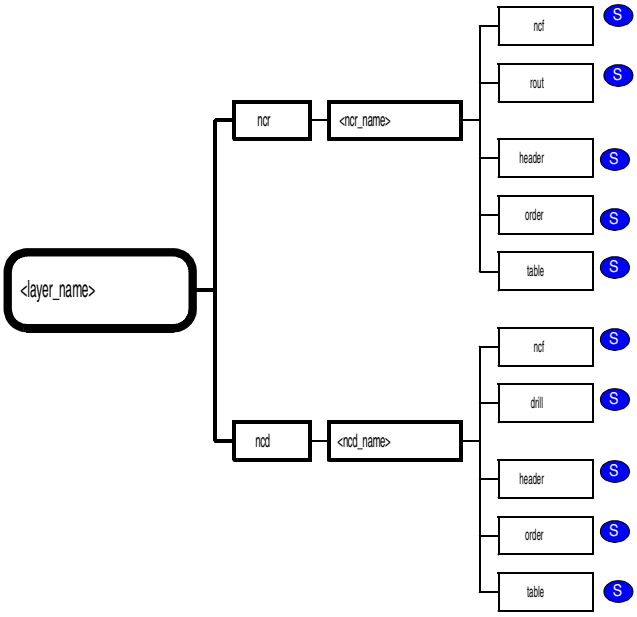
Forms



Wheels



NCR - NCD



Note The NCR and NCD entities are not documented in this edition of the manual.

Chapter 4 Job Entity Database

This chapter describes in detail each element of the Job Entity database.

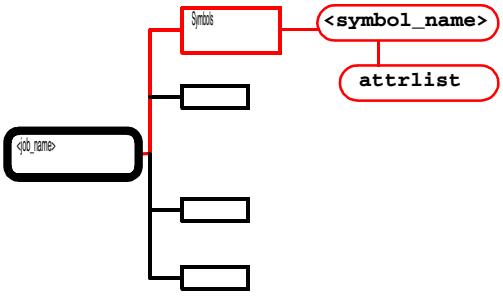
steps (See Chapter 5)

symbols (System and User Symbols)

attrlist (Attribute List)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/symbols/<symbol_name>/attrlist**

This file contains the values for attributes (system and user) of a symbol.



Example

```
.out_break = no
.out_scale = no
.break_away = no
.fill_dx = 0.100000
.fill_dy = 0.100000
.image_dx = -1.000000
.image_dy = -1.000000
connector = no
target = no
component =
comment =
hole_type = plated
serial_number = 15
```

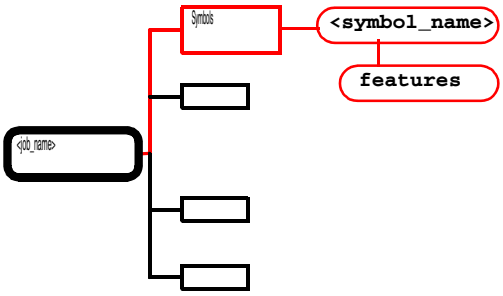
features (Symbol Features)

Type:	Line Record Text
Compression:	Yes
Sum file:	Yes
Path	**<job_name>/symbols/<symbol_name>/features**

The symbol features file describes the graphical shape of the symbol. It is similar in structure to the layer features file.

Example

See **<job_name>/steps/<step_name>/layers/<layer_name>/features**

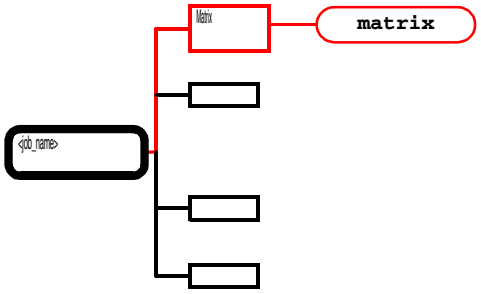


matrix (Job Matrix)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/matrix/matrix**

This file contains all the information which represents the Job Matrix. The Job Matrix is a two-dimensional array, where columns are **steps** - multi-layer entities (such as single images, sub panel arrays, production panels and coupons) and rows are **layers** - sheets on which elements are drawn for plotting, drilling and routing or assembly.

Each job can contain only one matrix file. The library job can contain several matrices.



Example

```
STEP {
   COL=1
   NAME=PCB
}
STEP {
   COL=2
   NAME=PANEL
}
...
LAYER {
   ROW=1
   CONTEXT=BOARD
   TYPE=COMPONENT
   NAME=COMP_+_TOP
   POLARITY=POSITIVE
   START_NAME=
   END_NAME=
}
```

```
LAYER {
   ROW=2
   CONTEXT=BOARD
   TYPE=SILK_SCREEN
   NAME=SST
   OLD_NAME =
   POLARITY=POSITIVE
   START_NAME=
   END_NAME=
}
...
```

The file contains two arrays: **STEP** and **LAYER**

Fields in the STEP array:

COL	The number of column in the matrix. Columns must be unique positive numbers (1 and above). Gaps are allowed between columns, causing vertical gaps to be created between steps in the displayed matrix.
NAME	The name of the step, according to the legal entity names described earlier. Each named step MUST have a step entity defined under the steps directory of the job, otherwise the job may be unreadable.

Fields in the **LAYER** array:

ROW	The number of row in the matrix. Rows must be unique positive numbers (1 and above). Gaps are allowed between rows, causing horizontal gaps to be created between layers in the displayed matrix.
CONTEXT	The layer context must be one of the two values: **BOARD** A layer which participates in the actual board production **MISC** Any other layer which is used for drawings, testing, etc.
TYPE	The layer type must be one of the following values: **SIGNAL** A layer used for regular signal transfer **POWER_GROUND** A plane layer, used for power or ground signals **MIXED** A combination of a signal and a plane layer **SOLDER_MASK** A layer used for solder mask application **SOLDER_PASTE** A layer used for depositing solder paste for assembly **SILK_SCREEN** A layer used for application of text legend **DRILL** A layer used to produce drill programs **ROUT** A layer used to produce rout program **DOCUMENT** A layer used for drawings, testing, auxiliary processes, etc. **COMPONENT** A layer containing components locations and outlines.

NAME	The name of the layer, according to the legal entity names described earlier. Each named layer MUST have a layer entity defined under the layers directory of each step in the job, otherwise the job may be unreadable.
OLD_NAME	The previous name of the layer. When this field has a value it means that a matrix layer has been renamed and this value is its old name. If the field is blank it means the layer has not been renamed.
POLARITY	This parameter describes the polarity of a whole layer. It is applied to the image when output (to a photoplotter for example). The layer polarity must be one of the two values: **POSITIVE** A copper layer in which features represent copper **NEGATIVE** A copper layer in which features represent laminate
START_NAME,END_NAME	These fields are only active for drill and rout layers. They specify the span of the drill or rout, in case it is partial (e.g. blind or buried via layers). Each field must be a valid board layer name. When the fields are empty, **START_NAME** is assumed to be the first board layer (which is not a drill or rout layer) and **END_NAME** is assumed to be the last board layer (which is not a drill or rout layer).

The layers should be ordered according to the stackup of the board, such as:

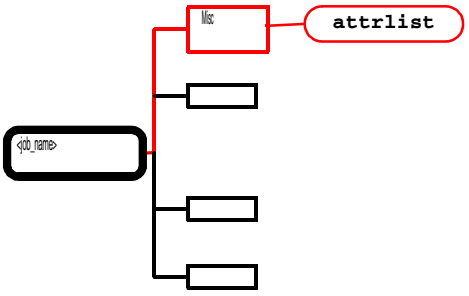
comp_+_top	comp_+_bot	drill_1 ... drill_5
sigt ... sigb	dril	...

misc (Miscellaneous)

attrlist (Attributes Used in Job)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/misc/attrlist**

This file contains the values for attributes (system and user) of a job. Only attributes (system and user) that have been defined are stored in the job.



Example

```
.customer = abc
connector = no
target = no
```

The file contains lines of the form:

```
<attribute> = <value>
```

System attributes for a job include:

```
.customer
.comment
.primary_side
```

last_save (Last Time Job Saved)

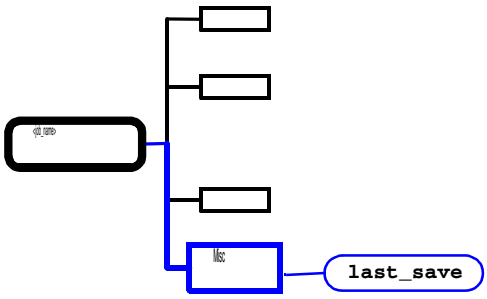
Type:	Line record text
Compression:	None
Sum file:	No
	<job_name>/misc/last_save

This file is written each time a "save" operation is done on a job. It records the time of the save operation.

Example

```
961224.183210
```

The file has one line of the format **yymmdd.hhmmss**.



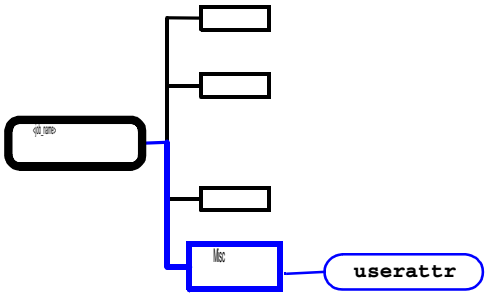
userattr (User Attributes)

Type:	Structured Text
Compression:	None
Sum file:	Yes
	<job_name>/misc/userattr

This file contains a list of the user attributes which were defined in the library at the time the job was created. It is read each time when the job is opened. All user attributes, for all entities are listed here.

Example

```
BOOLEAN {
   NAME=CONNECTOR
   PROMPT=CONNECTOR :
   ENTITY=ALL
   DEF=NO
}
```



Description

The file contains several arrays. Each array corresponds to one type of attribute:

```
BOOLEAN
TEXT
OPTION
```

INTEGER

FLOAT

Fields for a structure of type **BOOLEAN**:

NAME	The name of the attribute
PROMPT	The prompt used on the screen when this attribute is displayed
ENTITY	The entities for which this attribute is applicable. A semi colon separated list of entity types of: job, step, symbol, layer, stackup, wheel, feature, component
DEF	Default value (NO or YES)

Fields for a structure of type **TEXT**:

NAME	The name of the attribute
PROMPT	The prompt used on the screen when this attribute is displayed
MIN_LEN	Minimum length of the text attribute
MAX_LEN	Maximum length of the text attribute
ENTITY	See **ENTITY** for **BOOLEAN**
DEF	Default value

Fields for a structure of type **OPTION**:

NAME	The name of the attribute
PROMPT	The prompt used on the screen when this attribute is displayed
OPTIONS	A semi colon (;) separated list of options
DELETED	A semi colon (;) separated list of the values YES and NO.This corresponds to the list of options, possibly causing an option to be deleted (YES value)
ENTITY	See ENTITY for BOOLEAN
DEF	Default value

Fields for a structure of type **INTEGER**:

NAME	The name of the attribute
PROMPT	The prompt used on the screen when this attribute is displayed
MIN_VAL	Minimum value for the integer attribute
MAX_VAL	Maximum value for the integer attribute
ENTITY	See **ENTITY** for **BOOLEAN**
DEF	Default value

Fields for a structure of type **FLOAT**:

NAME	The name of the attribute
PROMPT	The prompt used on the screen when this attribute is displayed

MIN_VAL	Minimum value for the float attribute
MAX_VAL	Maximum value for the float attribute
ENTITY	See **ENTITY** for **BOOLEAN**
DEF	Default value
UNITS	**NO_UNITS, INCH_MM** or **MIL_MICRONS**. Affects the way the value is displayed (digits after the decimal point).

wheels (Gerber Wheel Templates)

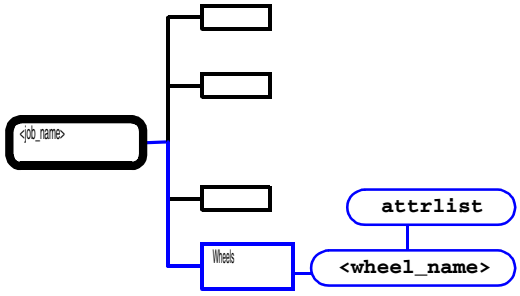
attrlist (Attributes Values)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/wheels/<wheel_name>/attrlist**

This file contains the values for attributes (system and user) of a wheel.

Example

```
comment=Mentor wheel
```



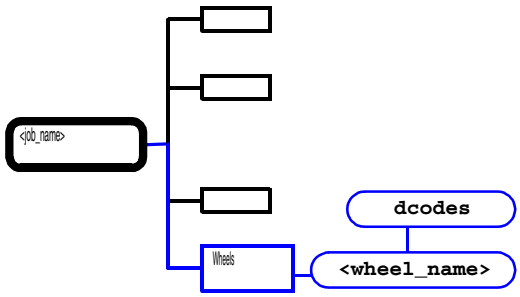
dcodes (Wheel Dcodes Definition)

Type:	Line Records Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/wheels/<wheel_name>/dcodes**

This file saves a wheel which is used during Gerber input.

Example

```
dcode10 r12
0 no_mirror
dcode11 r50
0 no_mirror
dcode12 r60
0 no_mirror
dcode13 r10
```



```
0 no_mirror
dcode14 r70                                    0 no_mirror
dcode15 r80                                    0 no_mirror
dcode17 r5                                    0 no_mirror
```

Each line in the file has the format:

```
dcode<n> <symbol_name> <angle> <mirror>
```

Where:

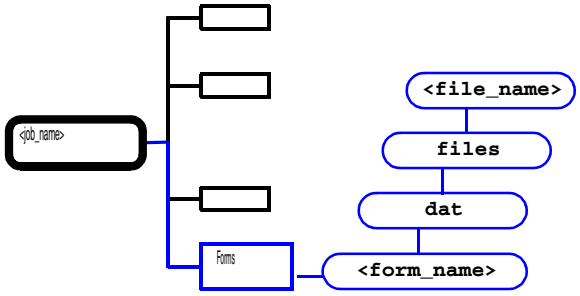
n	Dcode number
sym_name	Symbol name
angle	Always 0 (reserved for future use)
mirror	Always no_mirror (reserved for future use)

forms (Work Forms)

dat/files/<file_name> (Image File)

Type:	XPM or TIFF
Compression:	None
Sum file:	No
Path	**<job_name>/forms/<form_name>/dat/files/<file_name>**

This file corresponds to a form field of the type picture or drawing It contains a graphical image which is stored in this field. The name of the file must be the same as the name of the field.



Example

```
/* XPM */
static char * gns-
genesis186d4.1071 [] =
{
/* width height ncolors cpp [x_hot y_hot] */
"181 172 3 1 0 0",
/* colors */
"  c #EBEBF0F0CFCF",
"! c #FCFC00000000",
"# c #00000000FCFC",
/* pixels */
"                        !!!!!!!!!!!!!!!!!!!!!!!!!",
"                          !!!!!!!!      !!!!!!!!!!!",
"                          !!!!!!!!      !!!!!!!!!!!",
....
"                          !!!!!!!!!!!!!!!!!!!!!!!!!"};
```

Full Description

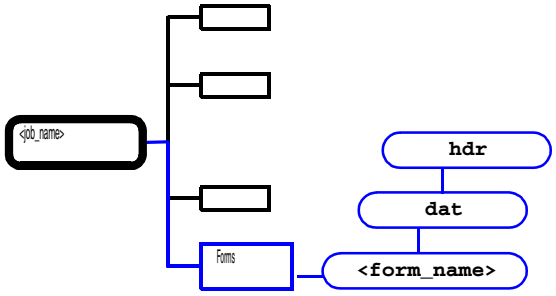
The system currently recognizes 2 standard formats for graphical images:

XPM	X11 pixmap, created by the HP Vueicon program
TIFF	Tagged Image File Format, created by various packages

dat/hdr (Data Header)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/forms/<form_name>/dat/hdr**

This file contains the textual contents for various fields in a Work Form. It is updated each time the form contents is changed.



Example

```
STEP=pcb
LAYER_1=L1
1_MIN_P2P=3
1_TYP_P2P=5
1_MIN_P2C=3
1_TYP_P2C=5
1_MIN_C2C=4
1_TYP_C2C=6
```

Each line of the file has the following structure:

```
<field> = <value>
```

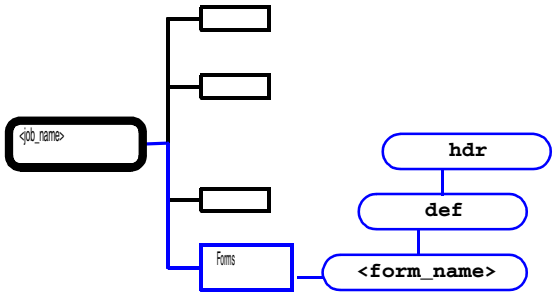
Where:

<field>	The internal (not displayed) name of the form field. This name must exist inside the definition portion of the form.
<value>	The string which represents the contents of the form field.

def/hdr (Definition Header)

Type:	Structured Text
Compression:	None
Sum file:	No
Path	**<job_name>/forms/<form_name>/def/hdr**

This file contains the definition of Work Form fields, including their types, geometry, action, etc.



Example

```
form {
  VER=0
  LABEL=analysis_results
  UNITS=I
  W=6500
  H=6500
  ACT=
  CLOSE_ACT=
  AUTO_UPDATE=YES
}
textf STEP {
 g {
      X=0
      Y=5736
      W=3501
      H=385
      BW=1
      BG=999980
      READABLE=YES
      EDITABLE=YES
      LTYPE=L
      FONT=tbr18
      LABEL=Step:
      PIXMAP=
      OR=H
 }
 cb {
      HELP=
      ACT=
 }
 te {
      FONT=tbr18
      NUMROWS=1
      NUMCOLS=0
 }
}
```

The file consists of multiple objects, each one representing one field in the form. The first object contains definitions about the form itself. Its structure is:

```
form {
 <field> = <value>
 ....
}
```

<field> can be one of the following types:

VER	Version number (Reserved for future use)
LABEL	Form name, to be displayed in the title bar
UNITS	Should be 0 (Reserved for future use)
W	Width of the form, in mils (0.001")
H	Height of the form, in mils (0.001")
ACT	Name of a call back to be activated each time the form is displayed
CLOSE_ACT	Name of a call back to be activated each time the form is closed.
AUTO_UPDATE	**YES** if the form definition is to be updated from the library each time the form is opened. **NO** if the form definition should not be affected by library changes.

Each following object has the following structure (note that not all substructure appear for each object):

```
<type> <name> {
  g {
     <field> = <value>
     ....
  }
  cb {
     <field> = <value>
     ....
  }
  te {
     <field> = <value>
     ....
  }
  ce {
     <field> = <value>
     ....
  }
  se {
     <field> = <value>
     ....
  }
}
```

<type> can be one of the following values:

sep	A separator object
label	A label object
textf	A text field object
choice	A radio, set or option menu object
picture	A graphical image
drawing	A graphical image
scale	A slider field
button	A push button object

The following table represents the mapping between object types and the substructures which appear in its definition:

	g	cb	te	ce	se
sep	x				
label	x				
textf	x	x	x		
choice	x	x		x	
picture	x	x			
drawing		x	x		
scale	x	x			x
button	x	x			

The fields which are available inside each structure are described below.

The g (geometry) fields:

X	X coordinate of the field lower left (in mils)
Y	Y coordinate of the field lower left (in mils)
W	Width of the field (in mils)
H	Height of the field (in mils)
BW	Border width (in screen pixels)
BG	Background color (A 6 digit number - rrggbb) **rr** - red value between 0 and 99 **gg** - green value between 0 and 99 **bb** - blue value between 0 and 99
READABLE	**YES** - if the field contents are to be displayed| **NO** - if the field contents have to be hidden
EDITABLE	**YES** - if the field can be edited on screen by the operator **NO** - if the field is for display only
LTYPE	**L** - for a textual label **P** - for a pixmap logo
FONT	The font used for the label. A string of the type xyznn where: **x** t(imes), h(elvetica) or c(ourier) **y** b(old) or m(edium) **z** r(egular) or i(talic) **nn** number of points (10,12,14,18 or 24). 1 point = 1/72".
LABEL	The text to be displayed in the label
JUSTIFY	The justification of the text in th field (**CENTER, LEFT** or **RIGHT**).
PIXMAP	The name of the pixmap used if **LTYPE=P**. The pixmap resides in the def/pixmaps directory of the form.
OR	orientation of the field (relevant for compound field such as text or choice). **H** for horizontal **V** for vertical.

The **cb** (callback) fields:

HELP	The name of the help file for the field (inside the **def/help** directory)
ACT	The name of the activation callback field (inside the **def/cb** directory)

The **te** (text extension) fields:

FONT	Font of the user entered text. See FONT in the **g** (geometry) section above.
NUMROWS	Number of rows in the field (1 for single line, 2 for multi line)
NUMCOLS	Should be 0 (Reserved for future use)
TYPE	Has one of the following values: **TEXT** Free text **INT** Integer values **FLOAT** Floating point (real) values **DATE** Legal date values **TIME** Legal time values
MIN	Minimal value for type INT or FLOAT
MIN_DATE_TIME	Minimal value for type DATE or TIME
MAX	Maximal value for type INT or FLOAT
MAX_DATE_TIME	Maximal value for type DATE or TIME
TEXT_FORMAT	**NONE, UPPER_CASE** or **LOWER_CASE**
DATE_FORMAT	**DD/MM/YY, MM/DD/YY** or **YY/MM/DD**
TIME_FORMAT	**HH:MM:SS**

The **ce** (choice extension) fields :

LTYPE	**L** for a textual options **P** for pixmap(logos) options
MODE	**R** for a radio choice field (one of many) **S** for a set choice field (some or many) **M** for an option menu choice field (one of many)
OR	Orientation of the options in the field. **H** for horizontal **V** for vertical
NUMCOLS	Number of columns for options

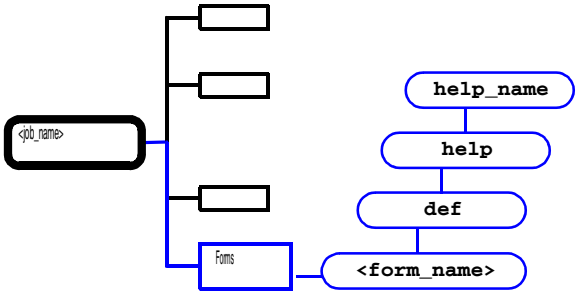
The se (scale extension) fields :

MIN	Minimal value of the slider
MAX	Maximal value of the slider
RADIX	Radix value (Currently must be 0)

S <help_name> (Help Text)

Type:	Free Text
Compression:	None
Sum file:	No
Path	**<job_name>/forms/<form_name>/def/help/<help_name>**

Contains help text to be used by the user who views the form. This file is only a part of the form definition in the library and is not copied into the form copy inside the job.



Example

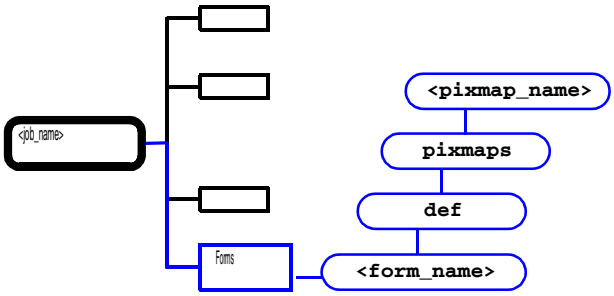
This field must be filled by the letters A,B,C or D

The file has a free text format.

S <pixmap_name> (Pixmap Field File)

Type:	XPM format.
Compression:	None
Sum file:	No
Path	**<job_name>/forms/<form_name>/def/pixmaps/<pixmap_name>**

This file is used for fields which are of LTYPE=P (pixmap). It contains the graphical image to be displayed.



Example:

```
/* XPM */
static char * logo []
= {
/* width height
ncolors cpp [x_hot
y_hot] */
"66 48 6 1 0 0",
/* colors */
"  s iconColor2m whitec white",
". c #000000000000",
"X c #FCFCC6C64D4D",
"o s iconColor5m blackc blue",
"O s iconColor3m blackc red",
"+ s iconColor8m blackc magenta",
/* pixels */
```

```
" .....................
",
" .XXX.XXXXXXXXXXXXX.
",
" .XXX.XXXXXXXXXXXXX.
",
" .XX...XXX...XXX...X.           o
",
" .X.....X.....X......     ooooooo    OOOO O  ++++    ++++
",
...
"
"};
```

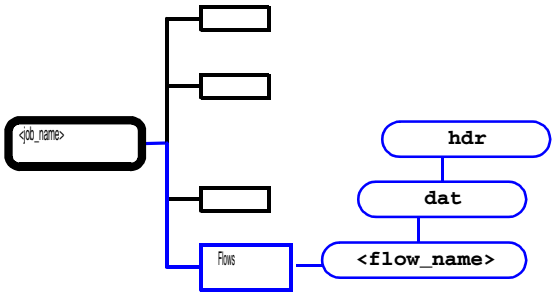
XPM - X11 pixmap, created by the HP Vueicon program.

flows (Job Process Charts)

dat/hdr (Data Header)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/flows/<flow_name>/dat/hdr**

This file represents the current state of a Work Flow. It contains a subset of the stages defined in the flow definition section and for each stage it holds information about status, date time, and operator.



Example

```
S0 = START-837427680-
john END-837427684-john
S1 = START-837427689-
john END-837427693-john
S2 = START-837427698-john END-837427703-john
S3 = Y
S9 = START-837427713-james END-837427717-james
S5 =
S6 = START-837427722-james END-837427727-james
S10 = START-837427731-mary END-837427736-mary
S4 = START-837427740-mary
S7 = Minor
S11 = 61

S12 =
S8 =
```

Each line of the file has the following structure:

```
<stage> = <value>
```

Where:

<stage>	The internal (not displayed) name of the flow stage. This name must exist inside the definition portion of the flow.
<value>	A string which differs according to the type of the stage (see below).

<value> can be:

For a stage of type Condition:

Y for yes
N for no

For a stage of type Switch:

One of the values allowed for the switch, according to the definition part of the flow.

For a stage of type Stage or Subflow:

START-time-operator

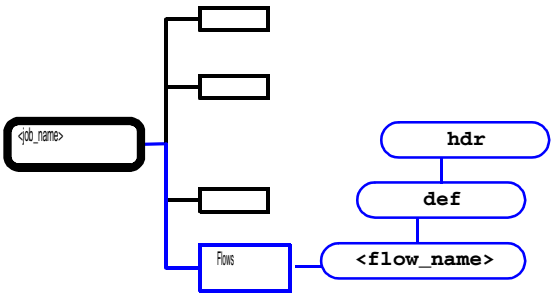
END-time-operator

Note 1) time is UNIX time (seconds starting January 1st, 1970)
2) the END string may not appear for stages in progress
3) both START and END may not appear for stages not started yet

def/hdr (Definition Header)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/flows/<flow_name>/def/hdr**

This file describes the structure of a Work Flow, including all the stages, the relation between them, and various additional information.



Example

```
flow 0 {
  VER=0
```

```
  LABEL=producibility_flow
  AUTO_UPDATE=NO
  MAX_SUBFLOW_LEVEL_ID=1
}
stage 2 {
  LEVEL_ID=1
  NAME=S0
  LABEL=
  TEXT=EDA Input
  FORM=
  GATE=OR
  HELP=
  PRE0=1
  STAGE_ACT=
  OPEN_ACT=
  CLOSE_ACT=
  NEW_LEVEL_ID=0
}
...
switch 11 {
  LEVEL_ID=1
  NAME=S7
  LABEL=
  TEXT=Determine Action
  FORM=
  GATE=OR
  HELP=
  PRE0=8
  PRE1=9
  PRE2=10
  PRE3=
  STAGE_ACT=
  OPEN_ACT=
  CLOSE_ACT=
  NEW_LEVEL_ID=0
}
```

The file consists of the following structures:

```
<type> <serial> {
  fields;
  ...
}
```

Where:

<type> is one of:

flow	Appears once as the first entity of the flow definition
stage	Appears as many times as needed to represent a stage in the process.

cond	Appears as many times as needed to represent a condition.
switch	Appears as many times as needed to represent a switch between multiple stages in the process
subflow	Appears as many times as needed to represent a composite flow which makes this stage.

<serial> is a unique number identifying the stage for the purpose of referencing from other stages.

The fields of a structure of type **flow**:

VER	Version number (Reserved for future use)
LABEL	Flow name to be displayed in the title row
AUTO_UPDATE	Reserved for future use. Should be set to NO
MAX_SUBFLOW_LEVEL_ID	Used internally. Should be set to 1.

The fields of all other structures:

LEVEL_ID	0 for the first structure (which represents the flow) 1 for all other stages.
NAME	Internal name of the stage. Used for references from the data section of the Work Flow.
LABEL	Only for subflow stages. The name of the lower level flow which represents this stage.
TEXT	The actual text which is displayed on the screen for this stage
FORM	An optional Work Form which is attached to the stage
GATE	One of the following values: **OR** - A stage can be started when at least one of its parents was finished. **AND** - A stage can be started when all its parents were finished.
HELP	A name of a file which provides information about the stage and can be displayed when the Work Flow is viewed. The file is a part of the Work Flow definition in the **flows/ <flow_name>/def** help code directory in the library.
PRE<n>	Up to 10 (n = 0 to 9) parents of this stage. The value for this field must be a valid serial number of the parent. If the parent is a condition or a switch, the serial number will be followed by the value for which this is the child.

Example

```
switch 11 {
 ...
 }

 stage 12 {
  ...
    PRE0 = 11 OK
```

```
    ...
}
```

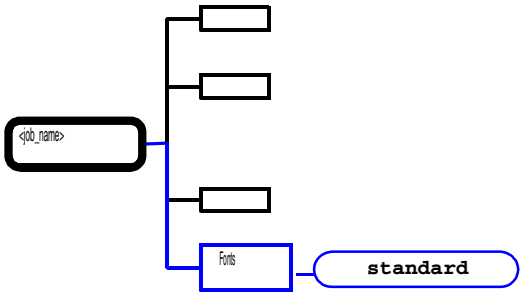
STAGE_ACT	The name of a callback to be executed when a stage changes its status. This field is only used in the main subflow of the flow, which is always the second structure. Callbacks are executed from the **def/cb** directory in the corresponding flow in the library and are not residing inside the job itself.
OPEN_ACT	The name of the callback to be executed when the flow is displayed. Same rules as **STAGE_ACT**.
CLOSE_ACT	The name of the callback to be executed when the flow is closed. Same rules as **STAGE_ACT**.
NEW_LEVEL_ID	For internal use. Must be 1 for the first stage structure and 0 for the rest.

fonts (Fonts used in Job)

standard (Standard Font)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/fonts/standard**

This file describes the vector representation of all the characters which can be a part of a text feature inside a layer. The system currently supports one font, named **standard**.



Example

```
XSIZE 0.302000
YSIZE 0.302000
OFFSET 0.000000
CHAR!
LINE 0.000000 0.000000 0.000000 0.200000 P R 0.012000
LINE 0.000000 -0.100000 0.000000 -0.100000 P R 0.012000
ECHAR
...
CHAR ~
LINE -0.100000 0.150000 -0.050000 0.200000 P R 0.012000
LINE -0.050000 0.200000 0.050000 0.100000 P R 0.012000
LINE 0.050000 0.100000 0.100000 0.150000 P R 0.012000
ECHAR
```

The file consists of a header containing global parameters followed by a collection of character blocks.

The header block consists of 3 lines:

XSIZE <size>	Horizontal size of a character, in inches
YSIZE <size>	Vertical size of a character, in inches
OFFSET <size>	Horizontal Distance between the end of one character block and the beginning of the next one

The character block consists of the following lines:

CHAR <char>	Defines the ASCII character which is defined by this block
LINE <xs> <ys> <xe> <ye> <pol> <shape> <width>	A definition of a line between (xs,ys) and (xe,ye). All coordinates are in inches. **<pol>** is the polarity of the line (P for positive, N for negative) **<shape>** is the shape of the ends of the line (R for rounded, S for square) **<width>** is the line width in inches
ECHAR	Ends the definition of a character

stackups (Stackups)

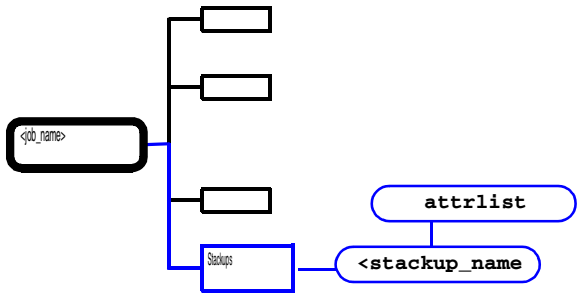
attrlist (Attribute List)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/stackups/<stackup_name>/attrlist**

This file contains the values for attributes (system and user) of a stackup.

Example

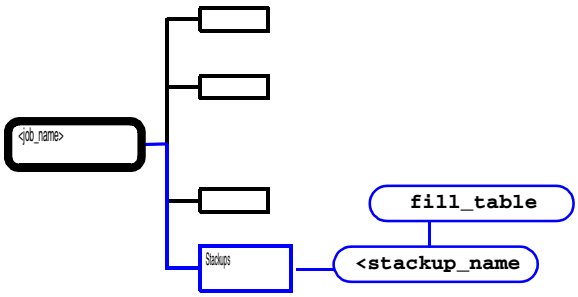
```
comment = Final Stackup
```



fill_table (Prepeg Combinations)

Type:	Line Record Text
Compression:	Yes
Sum file:	No
Path	**<job_name>/stackups/<stackup_name>/fill_table**

Contains the **fill_table** active at the time that the stackup was saved. The **fill_table** is the list of prepreg combinations which may be placed between foil layers.



Example

```
# A - ounce range for
first facing copper
# B - Copper area for
first facing copper (percentage)
# C - Position of first layer (Inner, Outer, Any)
# D - ounce range for second facing copper
# E - Copper area for second facing copper (percentage)
# F - Position of second layer (Inner, Outer, Any)
# G - Thickness of fill
# H - Positive tolerance on width of fill
# I - Negative tolerance on width of fill
# J - ordered list of names of sheets in fill
#
#      A      B     C       D       E     F       G  H   I
J
begin_fill
fill ((0-10oz 0-100 any)   (0-10oz  0-100 any)   (2  0.5 0.5)
(1080))
fill ((1-1oz  0-100 out)   (0.5-1oz 0-100 inner) (7  1   1)
(113 116))
fill ((1-1oz  0-100 inner) (1-1oz   0-100 inner) (7  1   1)
(113 113))
fill ((1-1oz  0-100 inner) (1-1oz   0-100 inner) (14 1   1)
(7628 7628))
fill ((1-1oz  0-100 out)   (1-1oz   0-100 inner) (8  1   1)
(116 116))
fill ((1-2oz  0-100 any)   (1-2oz   0-100 any)   (21 2   2)
(7628 7628 7628))
fill ((3-4oz  0-100 any)   (3-4oz   0-100 any)   (20 2   2)
(7628 7628 7628))
fill ((1-2oz  0-100 any)   (1-2oz   0-100 any)   (17 2   2)
(7629 7629 7629))
fill ((1-2oz  0-100 any)   (1-2oz   0-100 any)   (10 2   2)
(113 7628 113))
fill ((0.5-1oz 0-100 out)  (1-2oz   0-100 inner) (10 2   2)
```

```
(106 7628 106))
end_fill
```

The first record describes a 2 mil thick fill:

```
fill ((0-10oz 0-100 any)   (0-10oz  0-100 any)   (2 0.5 0.5)
(1080))
```

It consists of single sheet of 1080.

It provides a 2 mil spacer between two copper foils with a positive and negative tolerance of 0.5 mils.

Constructed Fill record

The second record describes a 7 mil thick fill:

```
fill ((1-1oz  0-100 out)   (0.5-1oz 0-100 inner) (7  1   1)
(113 116))
```

It consists of a sheet of 113 over a sheet of 116.

It provides a 7 mil spacer between two copper foils with a positive and negative tolerance of 1 mil.

It may only be used between 1oz copper foil outer layer and 0.5-1oz foil inner layer. It may not be used between other types of foils, or foils located in different layers than those specified (unless the same prepreg combination appears in another record).

The **fill** records are delimited by **begin_fill** at the beginning and **end_fill** at the end.

The fill records are of the form:

```
fill ((A B C)   (D  E F)   (G H I) (J))
```

Where the letters contain the following fields:

A	Ounce range for first facing copper
B	Copper area for first facing copper (percentage)
C	Position of first layer (Inner, Outer, Any)
D	Ounce range for second facing copper
E	Copper area for second facing copper (percentage)
F	Position of second layer (Inner, Outer, Any)
G	Thickness of fill (in mils)
H	Positive tolerance on width of fill (in mils)
I	Negative tolerance on width of fill (in mils)
J	Ordered list of names of sheets in fill

General rules

The prepregs which appear in column J must also appear in the 'material' file. There must also be a material of this kind in the bill file of the construct. The foils described in columns A, B, C may be placed either over or under the prepreg sheet combination. The ounce range and the copper area range should be complete as possible.

Actual copper area calculations are not made on the panels (this allows stackups to be designed prior to panelization).

Instead, the following copper area percentage values are automatically assigned to foils, according to the layer type:

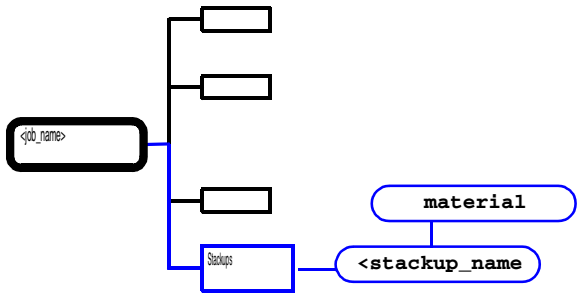
Layer type	Assigned copper area
signal	20%
mixed	50%
power and ground	80%

material (Material Specifications)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/stackups/<stackup_name>/material**

Contains the contents of the generic material file contents at the time that the stackup was saved.

Example




```
pile_begin
#
# Description of foil records
#
# A - Name of foil
# B - Thickness (in ounces) & +/- tolerances
# C - Color in Genesis display
# D - Ductility (HTE or STD)
# E - Resistance (in ohms)
#
#                    A     B          C      D    E
elem (Simple (Foil (1oz    (1    0 0) 856700 STD    0)))
elem (Simple (Foil (2oz    (2    0 0) 856700 STD    0)))
elem (Simple (Foil (1/2oz (0.5 0 0) 856700 STD    0)))
#
# Description of prepreg & laminate records
#
# A - Prepreg or Laminate
# B - Name
# C - Thickness (in mils) & +/- tolerances
# D - Color in Genesis display
# E - Dry permittivity
# F - Resin percentage (by weight)
```

```
# G - Dry weight
# H - Conductivity
#
#              A          B       C             D      E   F  G  H
elem (Simple (Prepreg  (7628   (7   0.5 0.8)   9900 4.2 20 10 0)))
elem (Simple (Prepreg  (108    (1.8 0.2 0.2)   9900 4.2 20 8  0)))
elem (Simple (Prepreg  (106    (1.6 0.2 0.2)   9900 4.3 20 6  0)))
elem (Simple (Laminate (Lam/40 (40  0   0  ) 505050 4.6 0  30 0)))
elem (Simple (Laminate (Lam/32 (32  0   0  ) 505050 4.6 0  26 0)))
#
# Description of core records
#
# A - Name
# B - Total Thickness (in mils) & +/- tolerances
# C - Type of top layer
# D - Name of top layer
# E - Type of middle layer
# F - Name of middle layer
# G - Type of bottom layer
# H - Name of bottom layer
# Note: the (G H) Expression is omitted for single sided clad cores.
#                     A           B          C    D     E   F        G    H
elem (Compound (Core fr4/40_1/1 (42.8 0 0) (Foil 1oz) (Lam Lam/40) (Foil 1oz))
elem (Compound (Core fr4/32_1/1 (34.8 0 0) (Foil 1oz) (Lam Lam/32) (Foil 1oz))
pile_end
```

Foil Record

The first record describes a conductive foil:

```
elem (Simple (Foil (1oz   (1   0 0) 856700 STD)))
```

Its name is **1oz**

Its weight is one ounce with positive and negative tolerances of zero

It will be displayed in a color whose Genesis number is **856700**

It has standard ductility (valid options: **STD, HTE**)

Prepreg Record

The fourth record describes a prepreg:

```
elem (Simple (Prepreg  (7628   (7   0.5 0.8)   9900 4.2 20 10
0)))
```

Its name is **7628**

It is 7 mils thick, with a positive tolerance of 0.5 mils and a negative tolerance of 0.8 mils

It will be displayed in a color whose Genesis number is **009900**

It has a dry permittivity value of **4.2**

It has 20% resin content (by weight)

It has a dry weight value of **10**

It has a conductivity value of **0** (zero)

Laminate Record

The seventh record describes a laminate.

```
elem (Simple (Laminate (Lam/40 (40  0   0  ) 505050 4.6 0  30
0)))
```

Its name is **Lam/40**

It is 40 mils thick with positive and negative tolerances of zero mils

It will be displayed in a color whose Genesis number is **505050**

It has a dry permittivity value of **4.6**

It has a 0% resin content (by weight)

It has a dry weight value of **30**

It has a conductivity value of **0** (zero)

Core Record

The ninth record describes a copper clad laminate core.

```
elem (Compound (Core fr4/40_1/1 (42.8 0 0) (Foil 1oz) (Lam
Lam/40) (Foil 1oz))
```

Its name is **fr4/40_1/1**

It is 42.8 mils thick with positive and negative tolerances of zero mils.

It is constructed of the following layers:

– A layer of a foil material as described in the first record
– A layer of a laminate as described in the third record
– A layer of a foil material as described in the first record.

Note Minimum / maximum core thickness defines the min/max thickness of the dielectric material in the core.

RCC Record

Below is a sample record describing the RCC material:

```
elem (Compound (RCC 1825 (1.4 0 0) (Foil (0.5 (0.5 0 0) 856700
STD 0)) (Resin (RES02 (0.7 0.1 0.1) 99 3.43 100 5 0)) (NULL)))
```

The name of the record is 1825. It is 1.4 mils thick with positive and negative tolerances of zero mils. It is constructed of the two following layers:

- A layer of a foil material
- A layer of a resin material

File Structure

There are three kinds of **elem** records in the file:

1. For foil records
2. For laminate and prepreg records
3. For compound records (Core, RCC).

Foil records are of the following structure:

```
elem (Simple (A (B   (C) D E)))
```

Where the letters refer to the following fields:

A	Name of foil
B	Thickness (in ounces) & +/- tolerances
C	Color in Genesis display
D	Ductility (HTE or STD)
E	Resistance (in ohms)

Laminate and prepreg records are of the following form:

```
elem (Simple (A  (B (C) D E F G H)))
```

A	Prepreg or Laminate
B	Name
C	Thickness (in mils) & +/- tolerances
D	Color in Genesis display
E	Dry permittivity
F	Resin percentage (by weight)
G	Dry weight
H	Conductivity

Compound records are of the following form:

```
elem (Compound (Core A (B) (C D) (E F) (G H))
```

or

```
elem (Compound (RCC A (B) (C D) (E F)())
```

A	Name
B	Total Thickness (in mils) & +/ tolerances
C	Type of top layer
D	Name of top layer
E	Type of middle layer
F	Name of middle layer
G	Type of bottom layer
H	Name of bottom layer

Note The **(G H)** expression is omitted for single sided clad cores and RCC.

General rules:

The first line of the file and the last line of the file open and close a block.

Each of the middle lines is a record describing a type of material.

Any material that appears in a core must be defined as a record by itself as well.

Material names are limited in length to 16 characters.

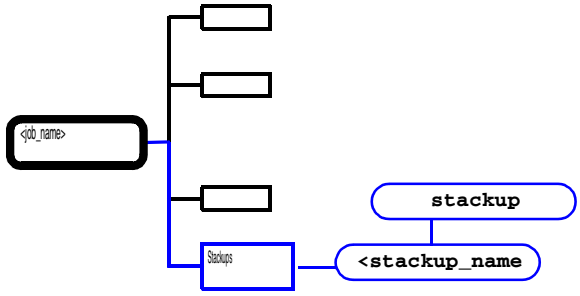
RCC Record

An RCC material is defined as a compound material, similar to Core. It consists of two layers - Foil and Resin.

stackup (Build)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/stackups/<stackup_name>/stackup**

Describes the build of a stackup.



Example

```
target (112 12 12)
# width height ffu plate_thick mask_thick thick_type vendor layer_match
stk_info 18 24 0 5 2 Laminate Any Yes (drill drill1 drill2) (drill drill1)
# The previous line indicates that:
# 18 24: the stackup is made of sheets which are 18 x 24.
# 5: the plating thickness of external layers is 5 mils
# 2: the solder mask thickness is 2 mils
# Laminate: the method for measuring thickness is laminate to laminate
# Any: materials from Any vendor may be used in the stackup
# Yes: one to one correspondance between foil board layers and stackup foil layers.
# (drill drill1 drill2): drill layers for sequential lamination.
# (drill drill1): microvia drill layers.
#     min sheets  max sheets
valid 2           3
# The previous line dictates that the minimum number of prepreg sheets used
# to separate layers is two and the maximum is three.
begin_pile
#         thickness       is_mirror construct cost   resin_Er copper_loss
pile_info (105.7 9.9 9.9) Yes       FR-4      11.983 0         0
begin_materials
# A - width
# B - height
# C - cost
# D - Reserved for future use
# E - material (Foil, Prepreg, Laminate, Core)
# F - weave (Vertical, Horizontal Null)
# G - thickness & +/- tolerances
# H - vendor
# I - generic name (from 'material' file)
```

```
# J - catalog number (Less than 16 characters preferred)
# K - construct
# L - Whether material is upsidedown in stackup
#
#    A  B  C     D E        F     G              H I       J      K    L
#############################################################
mat (18 24 0.091 1 Foil    None (1 0 0)         A 1oz     C90126 FR-4 No)
mat (18 24 1.58  1 Prepreg None (1.8 0.26 0.26) B 106     Z1261  FR-4 No)
mat (18 24 1.58  1 Prepreg None (1.8 0.26 0.26) B 106     Z1261  FR-4 No)
mat (18 24 0.139 1 Core    None (8.2 1.5 1.5)   B 8_1/1   H90120 FR-4 No)
mat (18 24 1.2   1 Prepreg None (9.8 0.4 0.4)   B 7628_10 Z0044  FR-4 No)
mat (18 24 0.139 1 Core    None (8.2 1.5 1.5)   B 8_1/1   H90120 FR-4 No)
mat (18 24 1.2   1 Prepreg None (9.8 0.4 0.4)   B 7628_10 Z0044  FR-4 No)
mat (18 24 0.125 1 Core    None (14.1 1.5 1.5)  B 14_1/1  H90119 FR-4 No)
mat (18 24 1.2   1 Prepreg None (9.8 0.4 0.4)   B 7628_10 Z0044  FR-4 Yes)
mat (18 24 0.139 1 Core    None (8.2 1.5 1.5)   B 8_1/1   H90120 FR-4 Yes)
mat (18 24 1.2   1 Prepreg None (9.8 0.4 0.4)   B 7628_10 Z0044  FR-4 Yes)
mat (18 24 0.139 1 Core    None (8.2 1.5 1.5)   B 8_1/1   H90120 FR-4 Yes)
mat (18 24 1.58  1 Prepreg None (1.8 0.26 0.26) B 106     Z1261  FR-4 Yes)
mat (18 24 1.58  1 Prepreg None (1.8 0.26 0.26) B 106     Z1261  FR-4 Yes)
mat (18 24 0.091 1 Foil    None (1 0 0)         A 1oz     C90126 FR-4 Yes)
end_materials
end_pile
```

The **stackup** file contains the following records:

A target record of the form "**target <target thickness> <positive tolerance> <negative tolerance>** where all units are in mils.

A **stk_info** record of the form:

stk_info <width> <height> <unused value> <plate_thick> <mask_thick> <thick_type> <vendor> <layer_match>

Where the width and height values are the width and eight of the sheets the stackup is made of.

The unused value is a numeric value reserved for future use.

The **thick_type** indicates the method used for measuring stackup thickness.

vendor is a vendor name if materials in the stackup are from a particular vendor, or **any** if materials may come from any vendor.

lyr match indicates whether there is a one to one correspondence between foil board layers and stackup foil layers.

Sequential lamination layers are drill and rout layers that generate sequential lamination requirements.

A **pile** record which is delimited by lines of **begin_pile** and **end_pile**.

Containing the subrecords:

pile_info which is a record of the form:

pile_info (<thickness>) <is_mirror> <construct> <cost> <resin Er> <copper loss>

Where:

thickness	is the calculated thickness of the stackup with tolerances
is_mirror	indicates whether the stackup is 100% symmetric

construct	indicates the make of the construct in the stackup
cost	the sum of the cost of all materials in the stackup
resin Er	the relative permittivity of the resin used in the resin system of the construct
copper_loss	the thickness of copper lost in internal layers due to processing. This value may also be positive, indicating that internal layers have been plated.

A **materials** sub-record delimited by **begin_materials** and **end_materials** containing **mat** records of the form:

```
mat (A B C D E F (G) H I J K L)
```

Where the letters contain the following fields:

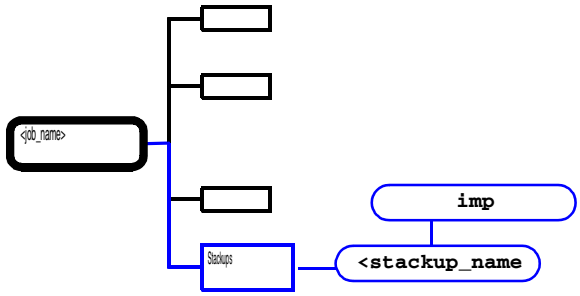
A	width
B	height
C	cost
D	Reserved for future use
E	material (Foil, Prepreg, Laminate, Core, RCC)
F	weave (Vertical, Horizontal Null)
G	thickness & +/- tolerances
H	vendor
I	generic name (from 'material' file)
J	catalog number (Less that 16 characters preferred)
K	construct
L	Whether material is upside-down in stackup
M	Which foils of a core are completely etched off (None, Top, Bottom, Both)

Note Material type (**prepeg, laminate, foil, core, RCC**) may be abbreviated to three letters (**pre, lam, foi, cor, RCC**).

imp (Impedance)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/stackups/<stackup_name>/imp**

Contains the impedance requirements and results of the stackup.



Example

```
# A - Allowed change to
line width
# B - Units of allowed
change (``Inch''
indicated English. --
i.e. mils)
# C - Width variation
of lines
# D - Units of width variation of lines (``Inch'' indicated
English. -- i.e. mils)
# E - Etch factor (Trapezoidal factor of lines due to etch
process) in mils
# F - Relative Permittivity of soldermask
# G - Impedance frequency (in MHZ)
#        A B      C D      E F G H I
imp_info 1 Inch   1 Inch   1 1 1 3 100
imp_begin
# A - Impedance model
# B - Reference layer
# C - Impedance layer
# D - Second Impedance layer (For broadside differential
models)
# E - Second Reference layer (For Microstrip models)
# F - Original line width (in mils)
# G - Current Line width  (in mils)
# H - Calculated impedance with tolerances (in ohms)
# I - Desired impedance with tolerances    (in ohms)
# J - Original spacing     (in mils) - for differential models
# K - Current spacing      (in mils)
# L - Tolerance of current width (in mils)
#    A                 B     C    D  E     F G  H
I       J KL
imp (Surface_Microstrip (12)  (11)  () ()   6 6 (98.3 4.9 4.9)
(90 0 0) 0 02)
imp (Dual_Stripline     (13)  (14)  () (16)  6 6 (138.8 13.8
13.8) (90 0 0) 0 01)
imp (Dual_Stripline     (13)  (15)  () (16)  6 6 (138.8 13.8
13.8) (0 0 0)  0 01)
imp (Dual_Stripline     (17)  (18)  () (110) 6 6 (138.8 13.8
13.8) (0 0 0)  0 01)
imp (Dual_Stripline     (17)  (19)  () (110) 6 6 (138.8 13.8
13.8) (0 0 0)  0 01)
imp (Surface_Microstrip (111) (112) () ()   6 6 (98.3 4.9 4.9)
(0 0 0)  0 01)
imp_end
```

The impedance records are of the following form:

```
imp (A (B) (C) (D) (E) F G (H) (I) J K L)
```

Where the letters contain the following fields:

A	Impedance model
B	Reference layer
C	Impedance layer
D	Second Impedance layer (For broadside differential models)
E	Second Reference layer (For Microstrip models)
F	Original line width (in mils)
G	Current Line width (in mils)
H	Calculated impedance with tolerances (in ohms)
I	Desired impedance with tolerances (in ohms)
J	Original spacing (in mils) for differential models
K	Current spacing (in mils)
L	Tolerance of current width (in mils)

input

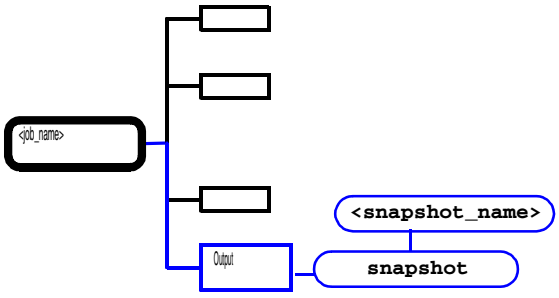
Location of input files when saved to a job.

output

snapshot (Measurement Information))

Path	**<job_name>/output/snapshots/<snapshot_name>**

A snapshot is a screen capture function available in the Graphic Station for the purpose of recording images, notes and measurement details of category violations in analysis.



Each snapshot, consisting of four files each (see list below), is stored in a directory under its own name **<snapshot_name>**. The snapshot name also becomes the name of the four files, each with the appropriate extension. Each snapshot directory consists of the following files:

<snapshot_name>.gif *(image in GIF format)*

<snapshot_name>.nte *(user notes typed in the Snapshot popup)*

<snapshot_name>.txt *(measurement information)*

<snapshot_name>.xpm.gz *(gzipped image in XWindows color bitmap format)*

user

Location to store user files.

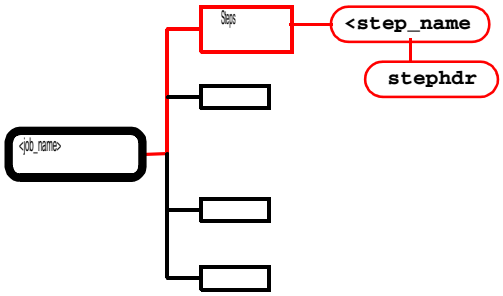
extension

Chapter 5 Job>steps Entity

stephdr (Step Header)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/stephdr**

This file contains data which is common to the whole step. This includes the step & repeat array for nest steps.



Example

```
X_DATUM=0
Y_DATUM=0

STEP-REPEAT {
   NAME=1UP
   X=1.5
   Y=1.6
   DX=1.2
   DY=1.2
   NX=6
   NY=6
   ANGLE=0
   MIRROR=NO
}

TOP_ACTIVE=1
BOTTOM_ACTIVE=1
RIGHT_ACTIVE=1
LEFT_ACTIVE=1
ONLINE_DRC_NAME=
ONLINE_DRC_MODE=DISABLED
ONLINE_DRC_STAT=RED
ONLINE_DRC_TIME=0
ONLINE_DRC_BEEP_VOL=2
ONLINE_DRC_BEEP_TONE=500
ONLINE_NET_MODE=DISABLED
ONLINE_NET_STAT=RED
ONLINE_NET_TIME=0
ONLINE_NET_BEEP_VOL=2
ONLINE_NET_BEEP_TONE=1000
```

The file consists of several fields and an array of **STEP-REPEAT** records.The fields are:

X_DATUM	x datum point (used for step & repeat)
Y_DATUM	y datum point (used for step & repeat)
X_ORIGIN	x origin point
Y_ORIGIN	y origin point
TOP_ACTIVE	active area for step & repeat (positive distance from the top edge)
BOTTOM_ACTIVE	active area for step & repeat (positive distance from the bottom edge)
RIGHT_ACTIVE	active area for step & repeat (positive distance from the right edge)
LEFT_ACTIVE	active area for step & repeat (positive distance from the left edge)
ONLINE_DRC_NAME	The name of the checklist (if any) used for on-line DRC
ONLINE_DRC_MODE	One of D**ISABLED, DEFERRED** or **IMMEDIATE**
ONLINE_DRC_STAT	One of **RED, YELLOW** or **GREEN**
ONLINE_DRC_TIME	The last time **check all** was done for on-line DRC
ONLINE_DRC_BEEP_VOL	Beep volume for immediate on-line DRC (0 to 3)
ONLINE_DRC_BEEP_TONE	Beep tone for immediate on-line DRC (200 to 1500)
ONLINE_NET_MODE	One of **DISABLED, DEFERRED** or **IMMEDIATE**
ONLINE_NET_STAT	One of **RED, YELLOW** or **GREEN**
ONLINE_NET_TIME	The last time **check all** was done for on-line netlist
ONLINE_NET_BEEP_VOL	Beep volume for immediate on-line netlist (0 to 3)
ONLINE_NET_BEEP_TONE	Beep tone for immediate on-line netlist (200 to 1500)
AFFECTING_BOM	Name of BOM last used in BOM_MERGE.
AFFECTING_BOM_CHANGED	Indicates whether **AFFECTING_BOM** was changed since last BOM_MERGE. (This requires that BOM_MERGE must be redone before retrieving information relating to BOM, such as Edit>Component>Set Chosen AVL.) If you attempt to retrieve info without performing BOM_MERGE, you will be required to confirm the action.

For the **STEP-REPEAT** array, the fields are:

NAME	Name of the step to be included in the current one (must be a valid step in the same job, without nesting)
X	Start X coordinate for placement of datum point of nested step
Y	Start Y coordinate for placement of datum point of nested step
DX	Horizontal distance between datum points (when angle = 0)
DY	Vertical distance between datum points (when angle = 0)

NX	Number of repetitions horizontally
NY	Number of repetitions vertically
ANGLE	Rotation angle of the steps (0, 90, 180 or 270 degrees)
MIRROR	**YES** for mirror (around X axis), **NO** for no mirror

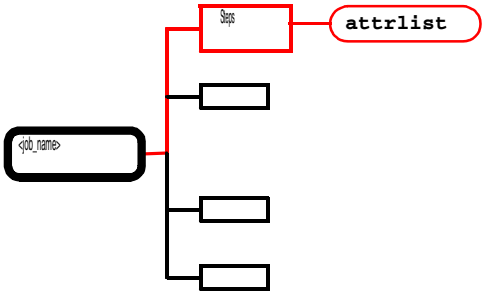
attrlist (Attribute List)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/attrlist**

This file contains the values for attributes (system and user) of a step.

Example

```
.out_drill_full = no
.out_drill_optional = no
.out_rout_optional = no
.fs_direction_top =
left2right
.fs_direction_bottom =
right2left
comment = Production Step
```



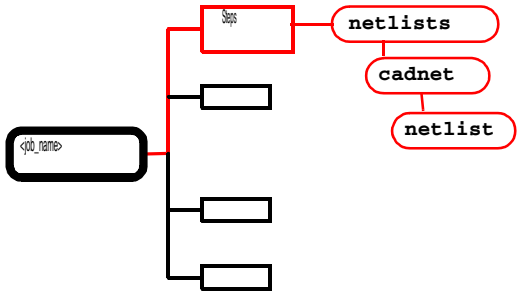
layers (See Chapter 6)

netlists (Netlists)

cadnet / netlist (CADnet)

Type:	Line record Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/netlists/cadnet/netlist**

This file represents a netlist as it was read from an external CAD system. A netlist is a collection of nets, each one referencing a group of points.



Definitions

Basic Netlist - contains only drill holes stored for drilled SMD pads.

Extended Netlist - contains both holes and drilled SMD.

Extended Netpoint - drill hole which has associated SMD pads.

Complex Netpoint - consists of both the hole and drilled SMD's, as stored in the netlist.

Netpoint Extension - drilled SMD pad stored in netlist together with the drill hole.

Test Side - of net-point is stored in netlist and determined by the Netlist Optimizer (cannot be changed except with the Electrical Testing Manager (ETM).

Example

```
H optimize n
$0 &1N1096
$1 &1N1526
$2 &1N289
$3 &1N312
$4 &1N338
$5 &1N340
$6 $NONE$
...
#
#Netlist points
#
0 0.002 4.96 -2.64 T e e staggered 0 0 0
0 0.002 4.94 -2.4775 T e e staggered 0 0 0
0 0.002 4.945 -2.575 B e e staggered 0 0 0
1 0.002 4.31 -4.045 T e e staggered 0 0 0
1 0.002 4.27 -3.893 T e e staggered 0 0 0
```

Note When a feature does not have a net defined it is assigned to **`NET $NONE$`**. All non-assigned features are defined as **`NET $NONE`** (see "NET - Electrical Net Record" on page 70).

The first line of the file has the form:

`H optimize <y|n>`

`y` - if netlist was optimized by the netlist optimizer

`n` - if netlist was not optimized

For CAD netlist, the net will always be non-optimized.

The next section of the file contains the nets, in the following format:

`$<serial_num> <net_name>`

Where:

<serial_num>	**is the net serial number, starting with 0**
<net_name>	**is the original net name as read from CAD**

The last section contains the net points. Each one has the following format:

```
<net_num> <radius> <x> <y> <side> [ <w> <h> ] <epoint> <exp> [ <c> ]
[staggerred <sx> <sy> <sr>] [v]  [f] [t] [m][<x>] [<e>] [<by>]
```

Where:

net_num	The number of the net (start from -1), corresponding to the previously defined netlist section (when a feature does not belong to a net it is defined as **$NONE$**). Net numbers start from -1 (-1 represents a tooling hole).
radius	Drill radius (inches) or 0.002 for SMD pads
x,y	point coordinates (inches)
side	**T** for top **D** for bottom **B** for both
w,h	Width and height of non-drilled pads (only when radius = 0)
epoint	**e** for net end point **m** for net mid point
exp	**e** for solder mask exposed point **c** for solder mask covered point **p** for solder mask covered primary point on top layer **s** for solder mask covered secondary point on bottom layer
c	Comment point
sx,sy	Coordinates of staggered point
sr	Radius of staggered point
v	**v** for a via point
f	Fiducial point
t	Test point
m	Appears when a netlist point is designated as a test point by assigning it the **.critical_tp** attribute. Normally this is applied to mid-points that need to be tested. The Netlist Optimizer determines mid-points to be not testable unless assigned this attribute. If both **.non_tp** and **.critical_tp** are assigned to the same point, **.critical_tp** takes precedence and the mid point is tested. In case of a drilled pad, the attribute must be added to the drill hole.
x	eXtended' appears if net point is extended

e	<Extension>' appears if net point is an extension
by	**{ c s b n }** c - test from component side s - test from solder side b - test from both sides a - test from any one side. n - side not defined (if **<by>** value not defined, **n** is assumed)

Example

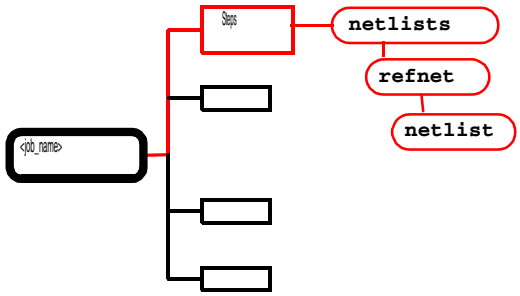
```
0 0.00675 0.8 3.3 B m e v x by=b
0 0 2.5 3.214393 T 0.04242 0.04242 e s staggered 0 0.01325 0 e by=c
```

A net point description for an extended point does not have to be grouped together in the netlist file.

refnet / netlist (Reference)

Type:	Line record Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/netlists/refnet/netlist**

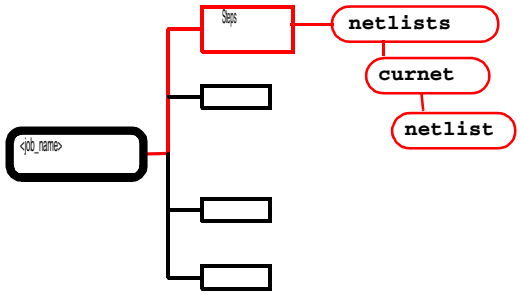
This file contains the reference netlist for the step. A reference netlist can be copied from the CAD netlist, the current netlist or the current-based-cad netlist.



curnet /netlist (Current)

Type:	Line record Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/netlists/curnet/netlist**

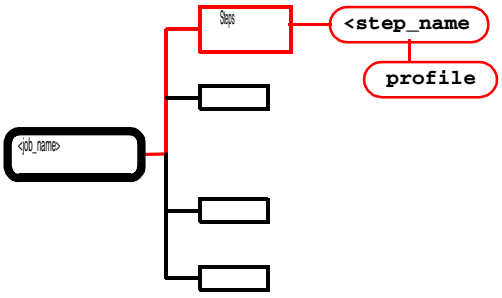
This file contains the Current netlist for the step. This is a temporary netlist that exists in the system memory only and is never saved with the job. It is extracted from the board layer in its current edited state, and always reflects any edits or modifications.



profile (Outline Shape of Step)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/profile**

The profile provides the outline shape of the step. It is required by many operations. A profile can be one closed polygon shape.



Example

```
#
#Layer features
#
S P 0
OB 0 0 I
OS 0 10
OS 10 10
OS 10 0
OS 0 0
OE
SE
```

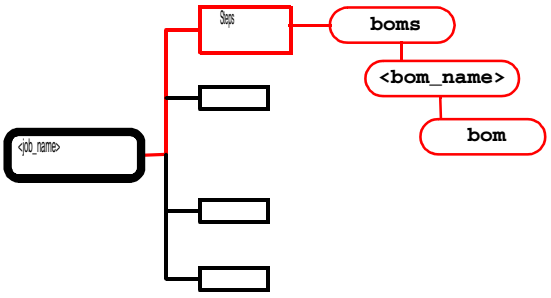
The profile consists of one positive surface feature. Please refer to the description of surface features inside the description of **<job_name>/steps/<step_name>/layers/<layer_name>/features**.

bom (Bill of Materials)

bom

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job_name>/steps/<step_name>/boms/<bom_name>/bom**

Example



 # Header Parameters

```
HEADER
BRD
REV
HEADER_END
```

 # Reference Descriptors and matching Customer Parts

```
RD_CPN
REF XTAL1
LNFILE 5 Rev14.v1
```
(where **5** *is the source line number and* **Rev14.v1** *is the source BOM)*

```
CPN 004-020-101
LNFILE 5 Rev14.v1

REF Y8
LNFILE 7 Rev14.v1
CPN 004-040-101
LNFILE 7 Rev14.v1


RD_CPN_END
```

 # Customer Parts and matching Manufacturer Parts

```
CPN_MPN
CPN 004-020-101
LNFILE 5 Rev14.v1
VPL_MPN
VPL_VND TOYOCOM
MPN TQC-216C-6R
LNFILE 5 Rev14.v1
VND TOYOCOM
LNFILE 5 Rev14.v1
QLF 0
CHS 1

CPN 004-020-101
LNFILE 5 Rev14.v1
```

```
VPL_MPN
VPL_VND VF
MPN VM6S-20.0000-16PF
LNFILE 6 Rev14.v1
VND VALPEY-FISHER
LNFILE 6 Rev14.v1
QLF 0
CHS 0


CPN_MPN_END
```

 # Customer Parts and description

```
CP
CPN 004-020-101
LNFILE 5 Rev14.v1
IPN
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
PKG
LNFILE 0 Rev14.v1
QNT 1
ITEM 0

CPN 004-040-101
LNFILE 7 Rev14.v1
IPN
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
DSC
LNFILE 0 Rev14.v1
PKG
LNFILE 0 Rev14.v1
QNT 1
ITEM 0

CP_END
```

FILE_END

Description

The file is divided into four sections. Each section starts with a header (equivalent to section name) and ends with *name*_**END**.

HEADER section

Contains two parameters:

Parameter	Description
BRD	board number
REV	revision

Source information for those parameters indicated in each section are saved in the corresponding **files** sub-directory (for example, **LNFILE 5 Rev14.v1** (*where* **Rev14.v1** *is the source file and* **5** *is the source line number*).

RD_CPN section

Contains the Reference Descriptors and their matching Customer Parts:

Parameter	Description
REF	Reference designator name
CPN	Customer part number

LNFILE is saved for all parameters.

CPN_MPN section

Contains Customer Parts and their matching Manufacturer Parts:

Parameter	Description
CPN	Customer Part Number
VPL_MPN	MPN from the VPL database corresponding to original MPN (as determined in BOM Validation)
VPL_VND	Manufacturer from the VPL corresponding to original Vendor (as determined in BOM Validation)
MPN	Manufacturer Part Number
VND	Manufacturer (Vendor) name
QLF	Qualify - whether the part (**MPN+VENDOR**) is qualified for production: **-1** - Not qualified **0** - Unknown **1** - Qualified
CHS	Chosen - if this part is chosen from among the alternate parts for the CPN. Only one can be Chosen.

LNFILE is saved for **CPN MPN VND**

CP section

Contains Customer Parts and their description:

Parameter	Description
CPN	Customer Part Number
IPN	Internal Part Number
DSC	Up to 5 descriptions
PKG	Package name
QNT	Reference Designator quantity
ITEM	Item number

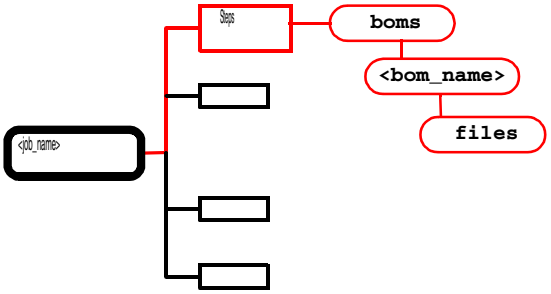
LNFILE is saved for **CPN IPN DSC PKG**

files (Source Files)

Type:	Directory
Compression:	None
Sum file:	No
Path	**<job_name>/steps/<step_name>/boms/<bom_name>/files**

The **files** sub-directory contains the source files that generated the BOM entity (BOM and AVL).

- BOM file containing description and contents of BOM entity, as described below:

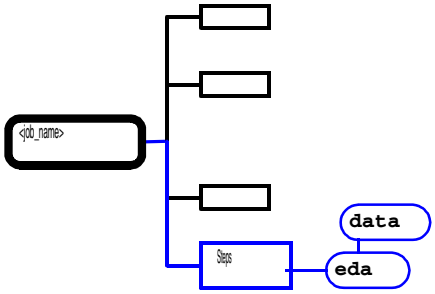


eda (Electronic Design Automation)

data

Type:	Line Record Text
Compression:	Yes
Sum file:	No
Path	**<job_name>/steps/<step_name>/eda/data**

This file contains information which is read from the EDA system database directly. It covers the library of packages, their outlines and properties, net connectivity information and more.



Example

```
#
HDR Mentor Boardstation database
LYR sst sst+1 sigt sig2 sig3
sig4 sig7 sig8 sig9 sigb smt smb
drill spt ssb spb ssb+1 pg6 pg5
#
PRP MILLING_ORIGIN 'MILLING 0 0.0 0' 0 0
PRP DRILL_ORIGIN '' 0 0
PRP FIXED_COMPONENT_LOCATION 'M3,m1' 1
....
#
#Net attribute names
#
#@0 .critical_net
#@1 .diff_pair
#@2 .net_type
#@3 .electrical_class
#@4 .dpair_gap
#@5 .eclass_rise_time
#@6 .eclass_max_stub_length
#
#
#Net attribute text strings
#
#&0 DEFAULT
#&1 clocks
#&2 EC_PUA
#&3 local
...

# NET 0
NET /D_CL_TX_CLK ;0,2=1,3=2,5=1.500000
SNT TOP T 16 0
FID C 2 33
FID C 10 33
FID C 13 30
...
# PKG 1
PKG *PDXC-L10/HX-L127W51T97 0.1 -0.25 -0.145 0.25 0.145
RC -0.25 -0.1 0.5 0.2
PIN 1 T -0.2 -0.1105 0 E S
RC -0.2125 -0.145 0.025 0.069
PIN 3 T -0.1 -0.1105 0 E S
RC -0.1125 -0.145 0.025 0.069
PIN 5 T 0 -0.1105 0 E S
RC -0.0125 -0.145 0.025 0.069
```

```
PIN 7 T 0.1 -0.1105 0 E S
RC 0.0875 -0.145 0.025 0.069
PIN 9 T 0.2 -0.1105 0 E S
RC 0.1875 -0.145 0.025 0.069
PIN 2 T -0.2 0.1105 0 E S
RC -0.2125 0.076 0.025 0.069
PIN 4 T -0.1 0.1105 0 E S
RC -0.1125 0.076 0.025 0.069
PIN 6 T 0 0.1105 0 E S
RC -0.0125 0.076 0.025 0.069
PIN 8 T 0.1 0.1105 0 E S
RC 0.0875 0.076 0.025 0.069
PIN 10 T 0.2 0.1105 0 E S
RC 0.1875 0.076 0.025 0.069
#
# PKG 2
PKG *MBCY-T2/XC-L80W80T115 0.1377953 -0.1574803 -0.1574803
0.1574803
0.1574803
CR 0 0 0.1574803
PIN P T 0.0688976 0 0 E T
CR 0.0688976 0 0.011811
PIN N T -0.0688976 0 0 E T
CR -0.0688976 0 0.011811

# FGR 1907
FGR TEXT
PRP string '030'
FID C 14 11018
FID C 14 11018
```

The file consists of records of the following types:
Main Records :

HDR	File Header
LYR	Layer Names
NET	Electrical Net Record
SNT	Subnet Record
PKG	Package Record
PIN	Pin Record
FGR	Feature Group Record
FID	Feature ID record
PRP	Property record

Net Attributes Header:

This header contains a table of net attribute names as well as a table of all net attribute values that are strings. The structure of the net attributes header is the same as that for features and components, except that each line begins with #.

```
#Net attribute names
#@<num1> <attribute_name>
```

```
#@<num2> <attribute_name>
#Net attribute text strings
#&<num3> <string>
```

Usage: <num1>,<num2>=<num3>

Outline Records:

CR	Circle record
SQ	Square Record
RC	Rectangle record
CT, OB, OS, OC, OE, CE	Contour record

Following is the format and description of each record.

HDR - File Header

This record contains the EDA system which was the source of the data.

Structure:

```
HDR <source>
```

Where:

<source> can be:

"Mentor Boardstation neutral file"

"Mentor Boardstation database"

"Cadence Allegro extract file"

"Zuken Redac CADIF file"

"PADS PowerPCB"

LYR - Layer Names

This record contains the names of the layers which are referenced in FID records later.

```
LYR  <name1> .... <namen>
```

Where:

<namex>	A legal name of a layer listed in the job matrix

NET - Electrical Net Record

This record contains a start record of an electrical net. Each net consists of one NET line and 0 or more SNT record

```
NET  <name>
```

Where:

<name>	The name of the net as defined in the EDA system*
<attributes>	This data is the same as for feature attributes (in the features file). It consists of comma separated list of values. Each can be: **n** indicating that (boolean) attribute **n** is set **n=m** indicating that option attribute **n** has value **m** **n=i** indicating that integer attribute **n** has value **i** **n=f** indicating that floating attribute **n** has value **f** **n=s** indicating that text attribute **n** has header value **s**
Note: **n** must match a @ record in the attribute header **s** must match a & record in the attribute header.	

When a feature does not have a net defined it is assigned to **NET $NONE$**. All unassigned outer layer pads are defined as **NET $NONE$**. With more than one **$NONE$** net, each is disconnected from the other. Any two points of a **$NONE$** net can be connected or disconnected, depending on the design.

In the Compare function of the Netlist Analyzer, disconnected **$NONE$** nets are not reported as opens. Shorts between **$NONE$** nets and other nets are reported. If no special treatment is done on the Xpert then if a **$NONE$** net points form more than one net typically they will all be reported as a large **$NONE$** net broken into subnets.

SNT - Subnet Record

This record contains a portion of a net. This portion can be:

toeprint	A connection of a component pin to the board
via	A connectivity padstack between layers
trace	A collection of lines/arcs leading from point to point
plane	A surface used for connectivity purposes

Each subnet record is followed by zero or more FID records mapped to the board features which are part of this subnet.

Structure for toeprint:

SNT TOP <side> <comp_num> <pin_num>

Where:

<side>	**T** for TOP, **B** for bottom
<comp_num>	Number of component in the components file (**comp_+_top/** components or **comp_+_bot/**components)
<pin_num>	Number of pin in the component

Structure for via

SNT VIA

Structure for trace

```
SNT TRC
```

Structure for plane

```
SNT PLN <fill_type> <cutout_type> <fill_size>
```

Where:

<fill_type>	**S** for solid **H** for hatched **O** for outline
<cutout_type>	**C** for circle **R** for rect **O** for octagon **E** for exact
<fill_size>	Size in inched of fill brush

Note The values for **SNT PLN** must appear with legal values, but the software does not consider them internally.

PKG - Package Record

This record contains a definition of a package, which is the generic shape of a component (e.g. each component refers to a package).

Each **PKG** line **must** be followed immediately by an outline record/s, 0 or more property (PRP) records and 0 or more PIN records.

Structure:

```
PKG <name> <pitch> <xmin> <ymin> <xmax> <ymax>
```

Where:

<name>	The name of the package as defined in the EDA system (geometry in Mentor terms, SYM_NAME in Cadence terms)
<pitch>	Distance between center of closest pins, in inches
<xmin>,<ymin>,<xmax>,<ymax>	Bounding box of package, relating to package datum

Note ODB++ requires closed geometries (polygons must be closed).

PIN - Pin Record

This record contains a definition of a pin, which belongs to a package.

Each pin is followed by (an) outline record(s).

Structure:

```
PIN <name> <type> <xc> <yc> <fhs> <etype> <mtype>
```

Where:

<name>	The name of the pin as defined in the EDA system
<type>	**T** for thru-hole (top--->bottom) **B** for blind (<top--->inner or inner--->bottom) **S** for surface (<top--->top or bottom-->bottom)
<xc> <yc>	Center of pin, relating to package datum
<fhs>	Finished hole size (Unused at the moment - should be 0)
<etype>	PIN Electrical Type: E - Electrical; M - Non-Electrical (Mechanical); U - Undefined
<mtype>	PIN Mount Type. S - SMT. D - Recommended SMT pad (where the pin size is the recommended pad size and not the pin size). T - Thru-hole. R - Thru-hole where the pin size is the recommended hole size and not the pin size. P - Pressfit. N - Non board, pins without contact area with the board. Used in components with lead forms of types: Solder Lug, High Cable, or Quick Connect. H - Hole, for physical holes that appear without the physical pin. U - Undefined.

By default, the last two parameters (**<pin_type>** and **<pin_mount_type>**) are defined as 'U' (Unknown) Only for packages that are imported from the VPL database are they defined otherwise.

FGR - Feature Group Record

This record contains the definition of a group of related features (e.g. the strokes of a text record).

Each **FGR** line is followed by zero or more **FID** records mapped to the board features which are part of this subnet.

```
FGR <type>
```

Where:

<type>	Only allowed value is TEXT

FID - Feature ID Record

This record contains a link to a feature in the board. The record is used to connect subnets and feature groups to the board features which are part of them.

```
FID <type> <lyr_num> <f_num>
```

Where:

<type>	**C** - copper **L** - laminate **H** - hole
<lyr_num>	A layer number (0 ... n-1) corresponding to the names of layers in the LYR record described earlier
<f_num>	A feature number (0 ... n-1) corresponding to the feature record sequence in the features file*.

* See below for example of **f_num** sequence in a feature file:

```
#
#Layer features
#
P 4.057087 4.5 6 P 0 0;1=6,3=0          #f_num = 0
P 4.057087 4.57874 6 P 0 0;1=6,3=0     #f_num = 1
P 4.057087 3.633858 6 P 0 0;1=6,3=0    #f_num = 2
P 4.057087 3.712598 6 P 0 0;1=6,3=0    #f_num = 3
```

PRP - Property Record

This record represents a property of the board, a net, a package or a feature group. A property consists of a name, a string value and 0 or more floating numbers.

```
PRP <name> '<value>' n1 n2 ...
```

Where:

<name>	The name of the property
<value>	The string of the property (between quotes)
n1,n2,...	The floating numbers to be kept in the property

Outline Records

Outline records **must** follow a **PKG** or **PIN** record. They describe the shape of the package/pin.

Note A PKG record must have an outline record as the immediate next entry (an outline record can be more than one line). A PIN record does require an outline record but not immediately after.

A shape can consist of a simple shape (circle, square, rectangle) or a complex contour.

CR - Circle record

```
CR <xc> <yc> <radius>
```

SQ - Square Record

```
SQ <xc> <yc> <half side>
```

RC - Rectangle record

```
RC <lower_left_x> <lower_left_y> <width> <height>
```

CT ... CE - Contour record

The structure of a contour record is the same as a surface feature in the features file and is restricted by the same limitations.

A contour consists of one or more polygons.

- Intersection is not allowed between edges of the same polygon
- Intersection is not allowed between edges of different polygons
- The polygons must form a closed shape
- Holes must be graphically contained inside island polygons
- The curves must be consistent (the start, end, and center point must construct a legal curve).

A polygon starts with **OB** command, contains **OS** (segment) or **OC** (curve) commands and ends with an **OE** command.

OB <start_x> <start_y> I/H (I=island, H=hole)

OS <end_x> <end_y>

OC <end_x> <end_y> <center_x> <center_y> <cw> (cw = Y or N)

OE

Net Attributes

The net attributes are found in the file "data", under the EDA directory of the job.

Each net can have attributes in the same way it is done for features and components. That is, each net name may be followed by a semi-colon followed by net attribute values, i.e., '**NET <net_name> ; <net attributes>**'

Also, in the EDA 'data' file, the net attributes header is found. This header contains a table of net attribute names as well as a table of all net attribute values that are strings. The structure of the net attributes header is exactly like the one used for features and components. The only difference is that for net attributes the header is commented (with #), in order to be read by Enterprise versions prior to v5.3. The header is located before the first net record.

For example:

NET /D_CL_TX_CLK ;0,2=1,3=2,5=1.500000

This should be interpreted as follows:

Net named '**/D_CL_TX_CLK**' has the following attributes:

- attribute #0,
- attribute #2, value: 1
- attribute #3, value: 2
- attribute #5, value: 1.5

A look at the attribute header reveals the following:

- attribute #0 is .critical_net, which is boolean, thus its appearance means: TRUE.
- attribute #2 is **.net_typ**e, which is of type string; its value is index 1, i.e., "clocks".

- attribute #3 is **.electrical_class**, which is of type string; its value is index 2, i.e., “EC_PUA”.
- attribute #5 is **.eclass_rise_time**, which is float and its value is 1.5.

Net type clearances.

In the EDA directory of the job, a **NEW** file, named ‘**net_prp**’ is found. This file contains the net type clearances. Versions prior to v5.3 are not aware of its existence, and thus cannot take advantage of it. The structure of this file is a list of net type clearances records, each of which has the following structure:

```
NET_TYPE_CLEARANCES {
    net_type1 = <net type 1>
    net_type2 = <net type 2>
    layers = <layer names>
    via2via = 0.005000
    trace2trace = 0.005000
    via2trace = 0.005000
    pin2pin = 0.005000
    via2pin = 0.005000
    trace2pin = 0.005000
    plane2plane = 0.005000
    via2plane = 0.005000
    trace2plane = 0.005000
    pin2plane = 0.005000
}
```

<net type 1> and **<net type 2>** are either net types that are supposed to be defined in the ‘data’ file, or an asterisk, i.e., ‘*’.

<layer names> is either a list of the job layer names, or an asterisk, i.e., ‘*’.

The 10 clearances that come after are optional. If exist, they are given in inches.

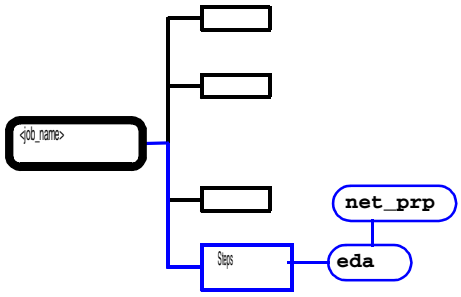
Examples:

```
NET_TYPE_CLEARANCES {
    net_type1 = local
    net_type2 = clock
    layers = sigt;sig4
    via2via = 0.005000
    trace2trace = 0.005000
    ...
}
NET_TYPE_CLEARANCES {
    net_type1 = local
    net_type2 = *
    layers = *
    via2via = 0.005000
    trace2trace = 0.005000
    ...
}
```

net_prp (Net Type Clearances Records)

Type:	
Compression:	
Sum file:	
Path	**<job_name>/steps/<step_name>/eda/net_prp**

In the EDA directory of the job, a **NEW** file, named '**net_prp**' is found. This file contains the net type clearances. Versions prior to v5.3 are not aware of its existence, and thus cannot take advantage of it. The structure of this file is a list of net type clearances records, each of which has the following structure:




```
NET_TYPE_CLEARANCES {
    net_type1 = <net type 1>
    net_type2 = <net type 2>
    layers = <layer names>
    via2via = <clearance>
    trace2trace = <clearance>
    via2trace = <clearance>
    pin2pin = <clearance>
    via2pin = <clearance>
    trace2pin = <clearance>
    plane2plane = <clearance>
    via2plane = <clearance>
    trace2plane = <clearance>
    pin2plane = <clearance>
}
```

<net type 1> and **<net type 2>** are either net types that are supposed to be defined in the 'data' file, or an asterisk '*'.

<layer names> is either a list of the job layer names (separated by semi-colons ';') or an asterisk '*'.

The 10 clearances that come after are optional. If exist, they are given in inches.

Examples:

```
NET_TYPE_CLEARANCES {
    net_type1 = local
    net_type2 = clock
    layers = sigt;sig4
    via2via = 0.005000
    trace2trace = 0.005000
    ...
}

NET_TYPE_CLEARANCES {
    net_type1 = local
    net_type2 = *
    layers = *
```

```
    via2via = 0.005000
    trace2trace = 0.005000
    ...
}
```

chk (Checklists) (See Chapter 7)

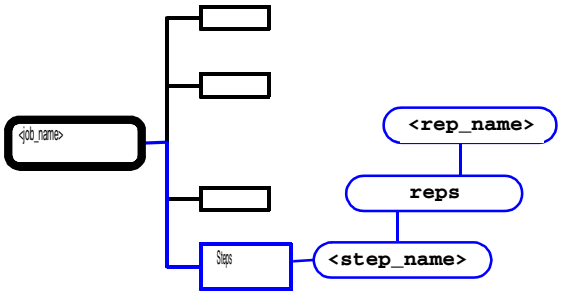
et (See Chapter 8)

reps (Reports)

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	`<job_name>/steps/<step_name>/reps/<rep_name>`

Example

```
TTL Library Merge Report
MSV 0
CAT No package in
library
CAT No part in library
CAT Ambiguous package
CAT No pin 1
CAT No BOM data
CAT No vendor name
CAT No vendor code
CAT Placement mismatch
(one)
CAT Placement mismatch (all)
CAT Inconsistent package rotation
CAT CAD/VPL pin count mismatch
CAT Package found
CAT Placement successful
_END_CAT
ITM 2 1
TXT VCODE: DALE, MPN: HAZ470MBABRAK
VAL S C388
LYR comp_+_top
AUX art.3
LIM 25222200 6146800 25831800 6553200
SHP S RC 2.4825 0.605 0.06 0.04
ITM 2 1
TXT VCODE: DALE, MPN: HAZ470MBABRAK
VAL S C389
```



```
LYR comp_+_top
AUX art.3
LIM 24968200 12623800 25577800 13030200
SHP S RC 2.4575 1.2425 0.06 0.04
```

Description

TTL - Report Title

```
TTL <title>
```

<title>	String serving as the report title (for display and for output).

MSV - Maximum Severity

```
MSV <sev>
```

<sev>	A number between 0 and 2 indicating the highest severity level of any item in the report, 0 being the highest possible severity (error) and 2 being the lowest (warning).

CAT - Category name

```
CAT <name> ...
```

_END_CAT - End of categories list

<name>	String serving as the printed/displayed name for this category. The list of categories ends with **_END_CAT**, and categories are later referenced by their index in this list, starting with 1.

ITM - Item entry

```
ITM <cat> <sev>
```

<cat>	Index of the category to which this item belongs, in the category listing.
<sev>	Severity of this item: 0 - error 1 - warning 2- informational An ITM record is followed by data pertaining to this item up to the next ITM record or the end of the file.

TXT - Item text description

```
TXT <string>
```

<string>	A text string describing this item. If omitted, the item's description will be the name of the category to which this item belongs.

VAL - Item value record

```
VAL S <str>
VAL I <intval>
```

VAL D <floatval>

<str>	String value
<intval>	Integer value
<floatval>	Floating-point value.

All, some, or none of these records can be present for any item. How the values are interpreted, depends on the viewing method of the report in the code and cannot be modified.

LYR - Item layer

LYR <lyrname>

<lyrname>	Name of a layer in the job's matrix. This layer is the primary layer associated with the item.

AUX - Auxiliary layers

AUX <lyrname1> <lyrname2> ...

<lyrname1>, <lyrname2> etc.	Names of layers in the job matrix. These layers serve as 'auxiliary layers' for this item, usually meaning that they will also be displayed when the item is displayed graphically.

LIM - Item limits record

LIM <xmin> <ymin> <xmax> <ymax>

<xmin>, <xmax>	Lower and upper limits for the X-axis of the graphical area relevant to this item.
<ymin>, <ymax>	Lower and upper limits for the Y-axis.

SHP - Item shape record

SHP S <shaperec>

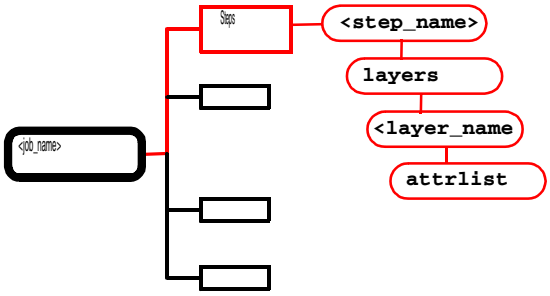
<shaperec>	One of the following: **CR** - Circle record **SQ** - Square Record **RC** - Rectangle record **CT...CE** - Contour record

Chapter 6 Job>steps>layers Entity

attrlist (Attribute List)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/attrlist**

This file contains the values for attributes (system and user) of a layer.



Example

```
.out_mirror = no
.inp_file =
.eda_layers =
"signal_2","signal","VIA"
.out_angle = 0.0
.out_polarity = positive
.out_x_scale = 1.000000
.out_y_scale = 1.000000
.out_comp = 0.000000
```

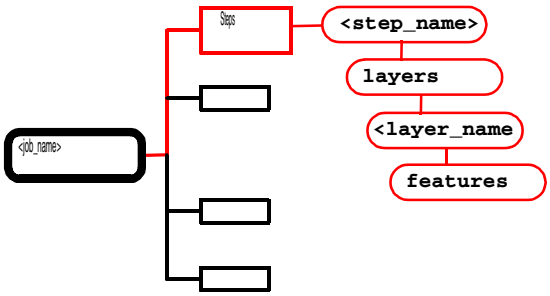
features

Type:	Line Record Text
Compression:	Yes
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/features**

The features file contains most of the graphical information of a layer (except for component layers which have the components file). Special symbols also have a feature file to describe their shape.

The feature files have 4 sections:

- Symbols table



Contains the names of all the symbols used by the features in the file and corresponding serial numbers for reference by the feature records.

- Attribute table

  Contains the names of attributes used by the features in the file, and the corresponding serial numbers for reference by the feature records

- Attribute texts

  Contains a list of text strings which are values for textual attributes.

- Features list

  Contains the actual features data

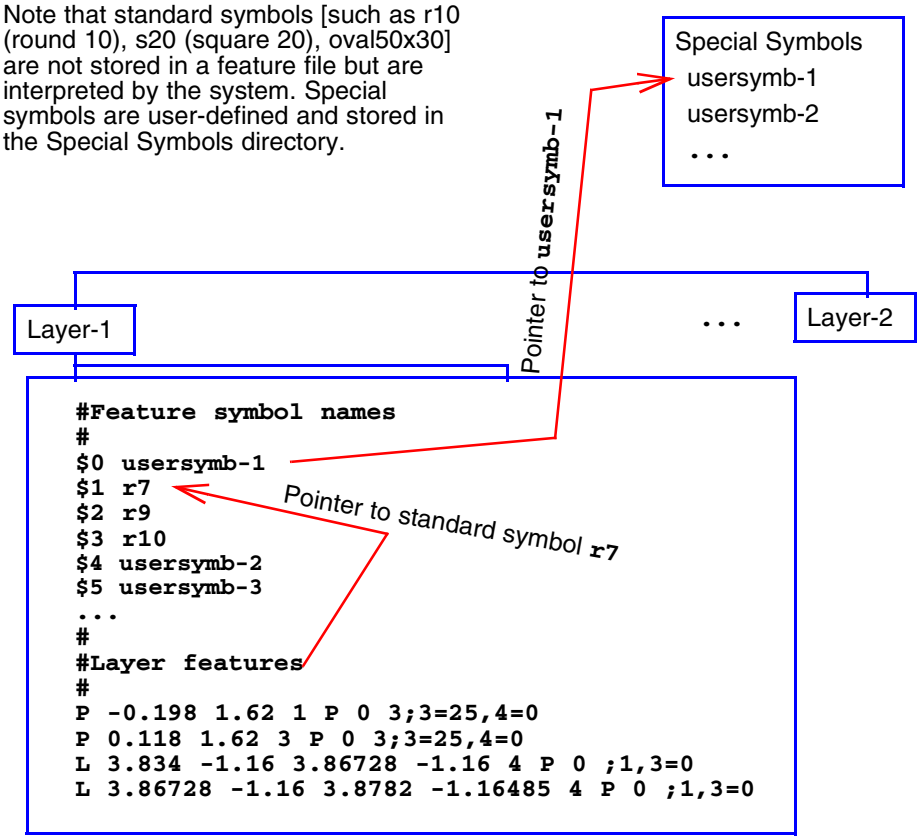
Example

```
#
#Feature symbol names
#
$0 r5
$1 r7
$2 r9
$3 r10
....
#
#Feature attribute names
#
@0 .smd
@1 .nomenclature
@2 .test_point
@3 .geometry
@4 .pad_usage
...
#
#Feature attribute text strings
#
&0 9796334
&1 fid_0_0_0
&2 moire
&3 p115_115_115_095
...
#
#Layer features
#
P -0.198 1.62 16 P 0 3;3=2,4=0
P 0.118 1.62 16 P 0 3;3=25,4=0
L 3.834 -1.16 3.86728 -1.16 2 P 0 ;1,3=0
L 3.86728 -1.16 3.8782 -1.16485 2 P 0 ;1,3=0
....
S P 0
OB -0.013 2.427 I
OS -0.013 2.218
OS -0.263 2.218
OS -0.263 2.427
OS -0.219 2.427
```

```
OS -0.219 2.262
OS -0.057 2.262
OS -0.057 2.427
OS -0.013 2.427
OE
SE
....
```

The following figure describes how symbols are treated. The sample feature file has been stripped of all but the relevant sections for this example:

Note that standard symbols [such as r10 (round 10), s20 (square 20), oval50x30] are not stored in a feature file but are interpreted by the system. Special symbols are user-defined and stored in the Special Symbols directory.



The feature symbol names section:

This section contains the symbols used by features in the file. The format of each line is:

```
$<serial_num> <symbol_name>
```

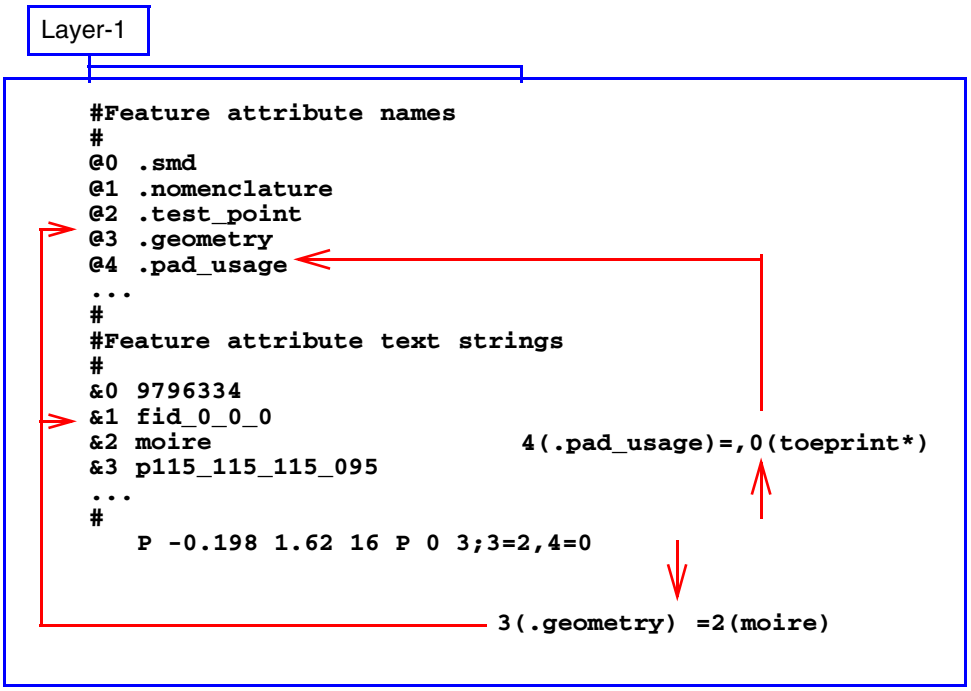
See "Symbol Definition" on page 135 for symbol naming conventions.

The feature attribute names section:

This section defines the names of attributes used by features in the file. The format of each line is:

```
@<serial_num> <attribute_name>
```

Attribute starting with the dot (.) character are system attributes (some system attributes must be defined for certain processes, such as analysis). Other attributes are user defined attributes.



* When the attribute is of type Option, the reference number points to the attribute options. In the case of **.pad_usage** these are:

0 - toeprint;
1 - via;
2 - g_fiducial;
3 - l_fiducial;
4 - tooling_hole.

Therefore, 4=0 in the feature record above, means **.pad_usage=toeprint**

The feature attribute text strings section:

This section contains texts which are values of textual feature attributes. Like its predecessors, the reason for this section is to save the repetition of long texts for each feature which uses it. The format of each line is:

&<serial_num> <text>

The features section:

This is the main section of the features file. It contains all the features in the file. Most features are represented by a single line in the file. Surface features may require multiple lines. The general format of a feature line is:

<type> <params> ; <atr>[=<value>],...

Where:

<type>	feature type which can be: **L** Line **P** Pad **A** Arc **T** Text **B** Barcode **S** Surface
<params>	A different set for each type. Please see below
<atr>	An attribute number, referencing an attribute from the feature attribute names section.
<value>	An attribute value which: - Is omitted for boolean attributes - Is a number for integer and float attributes - Is an option number for an option attribute - Is a number referencing the feature attribute text strings section for a textual attribute

The **<params>** field:

For line (L) records:

<xs> <ys> <xe> <ye> <sym_num> <polarity> <dcode>

xs, ys	start point
ye, ye	end point
sym_num	A serial number of the symbol in the feature symbol names section
polarity	P for positive, N for negative
dcode	gerber dcode number (0 if not defined)

For pad (P) records:

<x> <y> <sym_num> <polarity> <dcode> <orient>

x, y	center point
sym_num	A serial number of the symbol in the feature symbol names section
polarity	P for positive, N for negative
dcode	gerber dcode number (0 if not defined)
orient	pad orientation: **0 : 0** degrees, no mirror **1 : 90** degrees, no mirror **2 : 180** degrees, no mirror **3 : 270** degrees, no mirror **4 : 0** degrees, mirror in X axis **5 : 90** degrees, mirror in X axis **6 : 180** degrees, mirror in X axis **7 : 270** degrees, mirror in X axis

For arc (A) records:

<xs> <ys> <xe> <ye> <xc> <yc> <sym_num> <polarity> <dcode> <cw>

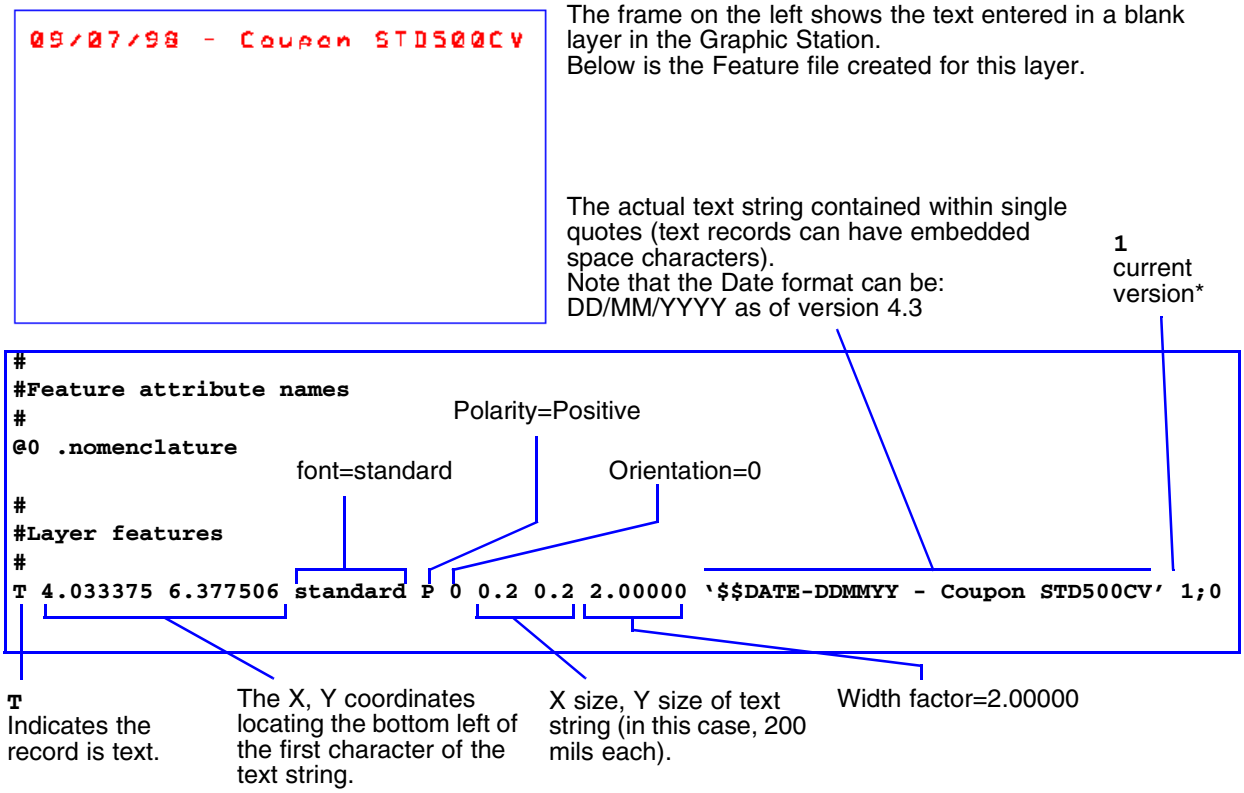
xs, ys	start point
ye, ye	end point
yc, yc	center point
sym_num	A serial number of the symbol in the feature symbol names section
polarity	**P** for positive, **N** for negative
dcode	gerber dcode number (0 if not defined)
cw	**Y** for clockwise, **N** for counter clockwise

For text (T) records:

<x> <y> <font> <polarity> <orient> <xsize> <ysize> <width factor> <text> <version>

x, y	text location (bottom left of first character for 0 orientation)
font	font name (Currently must be 'standard')
polarity	**P** for positive, **N** for negative
orient	text orientation: **0 : 0** degrees, no mirror **1 : 90** degrees, no mirror **2 : 180** degrees, no mirror **3 : 270** degrees, no mirror **4 : 0** degrees, mirror in X axis **5 : 90** degrees, mirror in X axis **6 : 180** degrees, mirror in X axis **7 : 270** degrees, mirror in X axis
xsize,ysize	Character size
width factor	width of character segment (in units of 12 mils) i.e. 1 = 12 mils, 0.5 = 6 mils
text	text string.
version	text field version values: **0** previous version **1** current version

Example of Text Records in Feature File



* This field is for future use. Currently, when version=0 it indicates the previous version. Old versions of feature files may not have this field at all, in this case it is assumed version=0. The version field does not affect the interpretation of the text data.

Dynamic Strings

The following strings are considered 'dynamic' and are assigned values on the fly each time the layer is displayed or being output:

$$date	Current date in mm/dd/yy format
$$date-ddmmyy	Current date in dd/mm/yy format
$$time	Current time in hh:mm format
$$job	Current job name
$$step	Current step name
$$layer	Current layer name
$$x	Current text x coordinate (inches)
$$y	Current text y coordinate (inches)
$$x_mm	Current text x coordinate (mm)
$$y_mm	Current text y coordinate (mm)
$$<attr_name>	Value of **<attr_name>** (job, step or layer attribute)

For barcode (B) records:

```
<x> <y> <barcode> <font> <polarity> <orient> E <w> <h> <fasc>
<cs> <bg> <astr> <astr_pos> <text>
```

See parameters of T (text) records for dynamic values.

x, y	text location (bottom left of first character for 0 orientation)
barcode	barcode name (currently must be UPC39)
font	font name (currently must be 'standard')
polarity	**P** for positive, **N** for negative
orient	text orientation: same as for T (text) records
E	A constant value (Reserved for future use)
w	element width
h	barcode height
fasc	**Y** for full ASCII, **N** for partial ASCII
cs	**Y** for checksum, **N** for no checksum
bg	**Y** for inverted background, **N** for no background
astr	**Y** for an addition of a text string
astr_pos	**T** for adding the string on top, **B** for bottom
Text	text string

For surface (S) records:

A surface is different from other features; it consists of multiple records:

```
S <params> ; <atr>=<value>...
<polygon 1>
<polygon n>
SE
```

The <params> section contains: **<polarity> <dcode>**

polarity - P for positive, N for negative

dcode - gerber dcode number (0 if not defined)

The first line is followed by a list of polygons. Each polygon is a collection of segments (lines without width) and curves (arcs without a width). Polygons must meet the following restrictions:

- Intersection is not allowed between edges of the same polygon.
- Intersection is not allowed between edges of different polygons.
- The polygons must form a closed shape (e.g, a polygon that contains only 2 segments is not valid).
- Holes must be graphically contained inside island polygons. The direction of island must be clockwise and of holes must be counter clockwise.
- The curves must be consistent (the start, end, and center point must construct a legal curve).

If any of the above mentioned violations occurs, the system will not be able to read the file, and will return an error.

The syntax of the polygons description for a surface feature is as follows:

```
OB  <xbs> <ybs> <poly_type>
OS  <x> <y>
OC  <xe> <ye> <xc> <yc> <cw>
OE
```

Where:

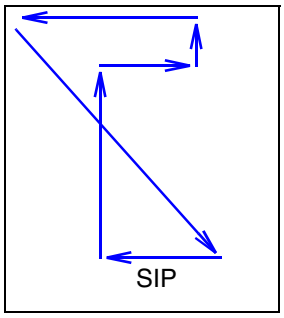
xbs,ybs	polygon start point
poly_type	**I** for island, **H** for hole
x, y	segment end point (previous polygon point is the start point)
xe, ye	curve end point (previous polygon point is the start point)
xc, yc	curve center point
cw	**Y** for clockwise, **N** for counter clockwise

Self intersecting Polygons

A self intersecting polygon (SIP) is a polygon with two non-consecutive edges (segments or curves) which touch each other. Genesis / Enterprise / Trilogy define legal polygons as those whose edges intersect only at endpoints of consecutive edges (see figure on the right).

SIPs are not a good base for mathematical representation. Problematic operations are:

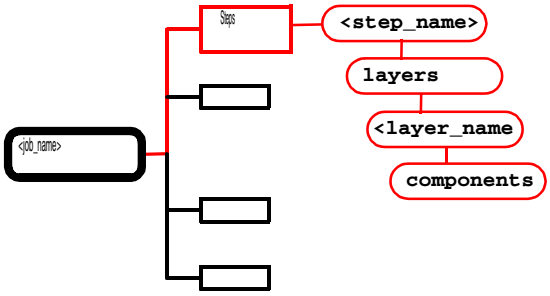
- Resize (enlarge, shrink, change shape)
- Calculation of copper areas (where unambiguous definition of the copper location is essential)



components

Type:	Line record Text
Compression:	Yes
Sum file:	No
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/components**

Each layer of type component has a unique name: **comp_+_top** or **comp_+_bot**. There are a maximum of two such layers in each job. Each layer has a components file which contains the information about actual components placed on the layer. The components file contains references to the **<step_name>/eda/data** file described earlier.



Example

```
#
#Component attribute names
#
@0 .comp_ign_spacing
@1 .no_tp_under
@2 .no_text_under
@3 .thvpad_required
@4 .comp_type
@5 .comp_height
@6 .comp_weight
# CMP 0
CMP 13 -0.04 1.22 270.0 N B70 2248827-0001
;1,2,4=11,5=0.100000,6=0.035273
PRP REFLOC 'IN,0.2,-0.225,270,CC,0.035,0.035,0.009,std,1'
TOP 0 -0.198 1.62 270.0 N 223 0 B70-1
TOP 1 0.118 1.62 270.0 N 223 1 B70-2
TOP 2 -0.04 1.22 270.0 N 466 0 B70-3
#
....
```

The components file may have a header, listing names of attributes used by the components in the file and possible textual values. This header is similar to the header of the features file (records starting with the @ and & character) and is described there.

Following the attribute header, components are listed in order, using 3 types of records:

CMP	Starts a component
PRP	Property of a component
TOP	Toeprint of a component

Following is the format and description of each record.

CMP - Component Record

This record contains a definition of a component.

Each **CMP** line is followed by 0 or more property (PRP) records and 0 or more TOP records.

CMP <pkg_ref> <x> <y> <rot> <mirror> <comp_name> <part_name> ; <attributes>

Where

<pkg_ref>	The number of the package in the eda/data file
<x>,<y>	The board location of the component in inches
<rot>	The rotation of the component, in degrees, clockwise.
<mirror>	N for not mirrored, M for mirrored
<comp_name>	component name (reference designator)
<part_name>	part identification
<attributes>	This data is the same as for feature attributes (in the features file). It consists of comma separated list of values. Each can be: **n** indicating that (boolean) attribute **n** is set **n=m** indicating that option attribute **n** has value **m** **n=i** indicating that integer attribute **n** has value **i** **n=f** indicating that floating attribute **n** has value **f** **n=s** indicating that text attribute **n** has header value **s**
Note: **n** must match a @ record in the attribute header **s** must match a & record in the attribute header.	

TOP - Toeprint Record

This record contains a definition of a toeprint of a component.

TOP <pin_num> <x> <y> <rot> <mirror> <net_num> <subnet_num> <toeprint_name>

Where:

<pin_num>	The pin number inside the package of the component
<x>,<y>	The board location of the pin in inches
<rot>	The rotation of the component, in degrees, clockwise.
<mirror>	N for not mirrored, M for mirrored
<net_num>	Number of net in the eda/data file*
<subnet_num>	Number of subnet within referenced net
<toeprint_name>	Name of the toeprint

* The **net_num** used in the TOP record corresponds to the sequence of the Net records in the **eda/data** file. The first Net record is **net_num 0**, the second is **net_num 1** and so on.

PRP - Property Record

This record represents a property of the component.
A property consists of a name, a string value and 0 or more floating numbers.

```
PRP <name> '<value>' n1 n2 ...
```

Where:

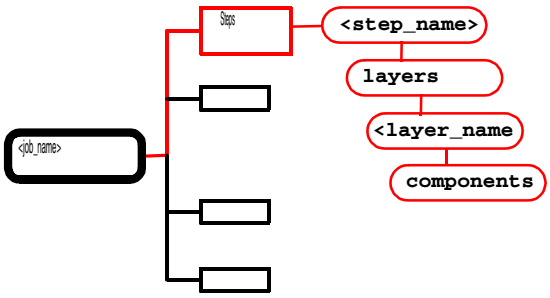
<name>	The name of the propery
<value>	The string of the property (between quotes)
n1,n2,...	The floating numbers to be kept in the property

components2

Type:	Line record Text
Compression:	yes
Sum file:	no
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/components2**

The **components** file describes the original EDA data for a component, while the **components2** file presents the data after processing with Assembly Merge (Bom Merge, Library Merge and Board Merge).

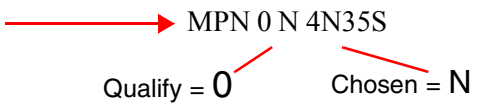
Note See preceding section for further information on the example below.



Example

```
#
#Component attribute names
#
@0 .comp_polarity
@1 .comp_height

# CMP 0
CMP 0 27.235992 6.19674 270.0 N W3_30B *
#
# BOM DATA
CPN 070-000-016
PKG
IPN
VPL_VND
VPL_MPN
VND CAL GREG
MPN 0 Y 4N35S
VPL_VND MOTOROLA
VPL_MPN
```

```
VND MOTOROLA
MPN 0 N 4N35S
VPL_VND SMNS-MTST
VPL_MPN
VND SIEMENS
MPN 0 N 4N35-X017
VPL_VND TI
VPL_MPN 4N35DCJ
VND TI
MPN 0 Y 4N35DCJ
# CMP 1
```



The component BOM DATA section contains BOM information on component

Parameter	Description
CPN	Customer part number
PKG	Package name
IPN	Internal part number
DSC	Up to 5 descriptions
VPL_VND	Manufacturer from the VPL corresponding to original vendor (as determined in BOM Validation)
VPL_MPN	MPN from the VPL database corresponding to original MPN (as determined in BOM Validation)
VND	Manufacturer (vendor) name
MPN	Manufacturer part number
Qualify	Whether the part (vendor+mpn) is qualified for production: **-1** - not qualified **0** - unknown **1** - qualified
Chosen	If this part is chosen from among the alternate parts for the CPN, only one can be chosen. **y** - yes, **n** - no

The MPN line contains the following parameters separated by spaces:

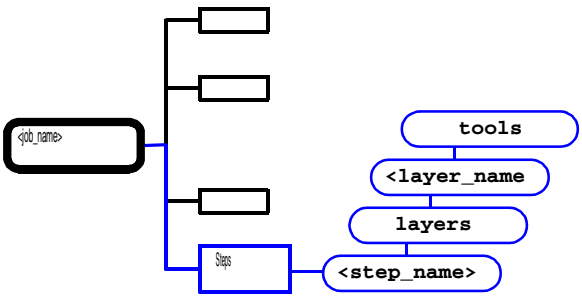
qualify chosen MPN

The section: **VPL_VND + VPL_MPN + VND + MPN** repeats for all the alternate parts of that CPN.

tools (Drill Tools)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/tools**

This file contains the tools table of a drill layer, initially created during input and further enhanced by the Drill Tools Manager.



Example

```
THICKNESS=0.0625
USER_PARAMS=method25

TOOLS {
   NUM=1
   TYPE=VIA
   MIN_TOL=0
   MAX_TOL=0
   BIT=
   FINISH_SIZE=11.5
   DRILL_SIZE=13.5
}
TOOLS {
   NUM=2
   TYPE=PLATED
   MIN_TOL=0
   MAX_TOL=0
   BIT=
   FINISH_SIZE=15
   DRILL_SIZE=19
}
```

The file contains 2 global parameters and a TOOLS array.
The global parameters are:

THICKNESS	board thickness (mils)
USER_PARAMS	free text that is used by the hook **drill_size** when converting finished hole sizes to drilled hole sizes

The fields of the **TOOLS** array structure are:

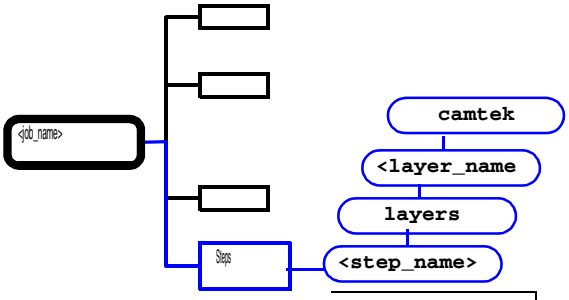
NUM	tool number
TYPE	one of **PLATED, NON_PLATED, VIA**
MIN_TOL, MAX_TOL	allowed tolerances (mils)

BIT	drill bit string
FINISH_SIZE	finished drill size (mils)
DRILL_SIZE	calculated drill size (mils)

camtek

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job name>/steps/<step name>/layers/<layer name>/camtek**

The file contains parameters of a Camtek set, describing parameters to be used when testing the layer for this entity.



ANGLE	Alignment of panel: rotation value
MIRROR	Alignment of panel: mirroring value
X_SCALE	Alignment of panel: X axis scale value
Y_SCALEL	Alignment of panel: Y axis scale value
POLARITY	Polarity of panel
DRILLS	Whether drill holes must be considered?
ETCH	Value of etch factor
RESOLUTION	Mania pixel size
MIN_LINE	Minimal line value
MIN_SPACE	Minimal space value
REG_DEFINED	True if registration pins were defined
REG_X1	X coord of first registration pin
REG_Y1	Y coord of first registration pin
REG_X2	X coord of second registration pin
REG_Y2	Y coord of second registration pin
CALIB	Calibration value
THICKNESS	Board thickness
TOLERANCE	Tolerance
LAMINATION	Lamination type
MACHINE	Machine type

REG_METHOD	Type of registration
SCAN_AREA	Dimension of area-to-test
EXCLUSION	Dimension of area not to test (rectangle, circle or polygon)

Example

```
ANGLE=0
MIRROR=NO
X_SCALE=1
Y_SCALE=1
POLARITY=POSITIVE
DRILLS=NO
ETCH=0
RESOLUTION=1.25
MIN_LINE=0
MIN_SPACE=0
REG_DEFINED=YES
REG_X1=1.552809350393701
REG_Y1=0.2045889763779528
REG_X2=1.475844488188976
REG_Y2=-0.1904967519685039
CALIB=C1
THICKNESS=0
TOLERANCE=0
LAMINATION=SHEET
MACHINE=PANEL
REG_METHOD=1

SCAN_AREA {
     X1=-0.9562419291338583
     Y1=-0.9396204724409449
     X2=1.753758070866142
     Y2=1.065379527559055
 }
EXCLUSION {
     X1=-0.4072266732283464
     Y1=0.2558988188976378
     X2=-0.114760531496063
     Y2=0.5381029527559055
 }
EXCLUSION {
     X1=0.5471363188976378
     Y1=0.3636495078740157
     X2=0.7626376968503937
     Y2=0.4970551181102362
 }

 EXCLUSION_C {
     X=0.8190785433070866
     Y=0.3482564960629921
     R=0.3574053149606299
 }
```

```
EXCLUSION_P {

    POINT {
        X=0.7010658464566929
        Y=-0.2725925196850394
    }
    POINT {
        X=0.7677687007874016
        Y=-0.3700812007874016
    }
    POINT {
        X=0.3624209645669291
        Y=-0.3752122047244095
    }
    POINT {
        X=0.3418969488188977
        Y=-0.2366755905511811
    }
    POINT {
        X=0.3675518700787402
        Y=-0.1084009842519685
    }
}
```

Translation of AOI-SET Fields into Camtek Output

```
ANGLE=90              -> to calculate transformation from panel coords to AOI table coords
MIRROR=YES            -> to calculate transformation from panel coords to AOI table coords
X_SCALE=1.01          -> inf.dat, Layer-info, Xstretch
Y_SCALE=1.02          -> inf.dat, Layer-info, Ystretch
POLARITY=NEGATIVE     -> to allow AOI machine to identify copper
DRILLS=NO             -> whether to create drill data (drill01.dat)
ETCH=1.1              -> inf.dat, Layer-info, Etch
RESOLUTION=0.5        -> inf.dat, Layer-info, PixSize
MIN_LINE=0            -> none
MIN_SPACE=0           -> none
REG_DEFINED=YES       -> inf.dat, Layer-info, Ref_pins
REG_X1=1              -> inf.dat, Layer-info, Ref_pins
REG_Y1=9              -> inf.dat, Layer-info, Ref_pins
REG_X2=9              -> inf.dat, Layer-info, Ref_pins
REG_Y2=9              -> inf.dat, Layer-info, Ref_pins
CALIB=A0              -> inf.dat, Layer-info, Calib
THICKNESS=2.2         -> inf.dat, Layer-info, Thick
TOLERANCE=0           -> none
LAMINATION=FOIL       -> out of lamination, layer number and number of layers:
                         inf.dat, Layer-info, Layer_view
                         inf.dat, Layer-info, Layer_pair
MACHINE=ARTWORK       -> none
REG_METHOD=1          -> inf.dat, Layer-info, Align_method

SCAN_AREA {           -> to calculate number of frames and overlap values
    X1=0.4914508858267717
    Y1=0.5613854330708662
    X2=9.581646948818898
    Y2=8.516830905511812
}
```

```
EXCLUSION {              -> don't-inspect areas, as listed in mask01.dat
    X1=0
    Y1=0
    X2=10
    Y2=0.5648308070866142
}

EXCLUSION {              -> yes, the hdr file may contain multiple EXCLUSION arrays
    X1=0
    Y1=0.5648308070866142
    X2=0.4946468503937008
    Y2=1.498834744094488
}
```

attrlist (Attribute List)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/camtek/attrlist**

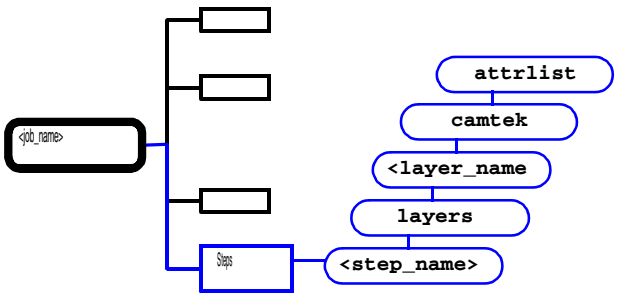
The user attributes are defined by Camtek, and a user attributes ASCII file is normally supplied with the Camtek AOI system. In the inf.dat output file:

<output path>/<job name>/<layer number>/<AOI set name>/inf.dat

the user attribute values appear (as set) in the [Learn Type Definitions] section.

Example

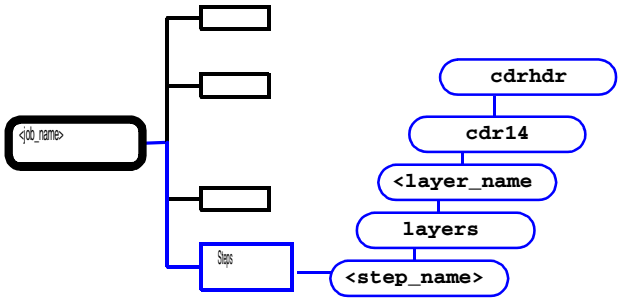
```
.drc_min_space = 5
.drc_min_width = 7
.drc_add_rad = 2
```



S cdrhdr (CDR14 Header)

Type:	Structured Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/cdr14/cdrhdr**

This file contains the values for CDR14 parameters of the layer.



Example

```
NOM_SPACE=0.02
MIN_SPACE=0.015
NOM_LINE=0.02
MIN_LINE=0.015
MULTI_LINE=YES
MULTI_LINES {
   MIN_WIDTH=0.15
   MAX_WIDTH=0.17
}
MULTI_LINES {
   MIN_WIDTH=0.18
   MAX_WIDTH=0.20
}
...

WORKING_TABLE=24X24
PADS {
    SYMBOL_NAME=r100
}
PADS {
    SYMBOL_NAME=rect150x120
}
...
STAGES {
    STAGE_NAME=PHOTO
    ETCH_SET=YES
    ETCH=0.005
CLASSES {
        CLASS_NAME=photo1
    }
    ...

    TOOLSET_NUMBER=-1
MANUAL_ALIGNMENT {
       IS_USED=YES
OFFSET {
            X=20
            Y=10
        }
        MIRROR=X
```

```
        ANGLE=270
        POLARITY=POSITIVE
    }
}
...

DRILLED_STAGE=-1
SCALE {
    X=1
    Y=1
}
SCALE_ORIGIN {
    X=0
    Y=0
}
```

Description

NOM_SPACE	Nominal Spacing. Valid range is 0.0005..0.128 (inch). Null/default value is 0.0.
MIN_SPACE	Minimal Spacing where **MIN_SPACE <= NOM_SPACE**. Valid range is 0.0005..0.128 (inch) Null/default value is 0.0.
NOM_LINE	Nominal Line Width. Valid range is 0.0005..0.128 (inch) Null/default value is 0.0.
MIN_LINE	Minimal Line Width, where **MIN_LINE <= NOM_LINE**. Valid range is 0.0005..0.128 (inch) Null/default value is 0.0.
MULTI_LINE	Yes/No. Yes - use **MULTI_LINES** array instead of **MIN_LINE**
MULTI_LINES	Array of a maximum of 4 elements defining multiple line width ranges.
WORKING_TABLE	Name of the working table matching the template **<w>X<h>** where **w** is table width **h** is table height. Should be defined in the **cdr14.ini** file.
PADS	Array of a maximum 8 elements defining pads symbol names
HOLES	Array of a maximum of 8 elements defining holes symbol names.
CLEARANCES	Array of a maximum of 8 elements defining clearance symbol names.
STAGES	Array of a maximum of 10 elements defining working stages parameters.
DRILLED_STAGE	Index of the stage in STAGES array which is defined as a drill stage. Null/default value is **-1**

SCALE	X and Y scale factors applied on output. Valid range is 0.001..9.99 (0.1..999%). Null/default value is 1.0.
SCALE_ORIGIN	X and Y scale origin coordinates. Valid range is unlimited. Null/default value is 0.0.
PANELIZATION	**PANEL_DEFINED**: use the genesis automatic panelization. **USER_DEFINED**: Use the panelization supplied by user.
GENESIS_VERSION	Version of the Genesis software which created the **cdr14** set of the form <**major**>.<**minor**><**patch**>.
MARGINS_SET	0. Field not in use.
X_MARGIN	0. Field not in use.
Y-MARGIN	0. Field not in use.

MULTI_LINES Array Structure:

MIN_LINE	Minimal Line Width
MAX_LINE	Maximal Line Width where **MIN_LINE <= MAX_LINE** and **MIN_LINE <= NOM_LINE**. Valid range is 0.0005..0.128 (inch). Null/default value is 0.0

* **MULTI_LINES** pairs should be defined in the order of **MIN_WIDTH** increasing.

PADS/HOLES/CLEARANCES Arrays Structure

SYMBOL_NAME	Feature symbol name.

STAGES Array Structure

STAGE_NAME	Working stage name. Should be defined in the **cdr14.ini** file.
ETCH_SET	Yes/No. Yes - use the **ETCH** value. No - use the default **ETCH** value defined for the stage in the **cdr14.ini** file instead.
ETCH	Etch value. Valid range is -255.0..255.0 (inch). Null/default value is 0.0.
CLASSES	Class names for the stage. Array for a maximum of 5 elements.
DRILL_LAYER	Name of the drill layer. Relevant only for stage defined as drill stage.
TOOLSET_NUMBER	Toolset number used for the stage alignment, should be defined in the **cdr14.ini** file. Null/default value is -1.

MANUAL_ALIGNMENT	Stage alignment used if no toolset defined.
DRILL_LAYERS	Names of drill layers. In case of multiple drill layers, names are separated by semi-colons (**;**)

* Either **TOOLSET_NUMBER** or **MANUAL_ALIGNMENT** should be defined for each stage

MANUAL_ALIGNMENT Structure

IS_USED	Yes/No. Yes - use manual alignment instead of toolset.
OFFSET	X and Y alignment offsets.
MIRROR	None/X/Y/Both.
ANGLE	0/90/180/270 measured in degrees CW (clockwise).
POLARITY	Positive/Negative.

Translation of CDR-SET Fields into AOIProg Commands

```
CDR-SET Field                          AOIProg Translation
-------------                          -------------------
NOM_SPACE=0.008                      | SPACE = 8.000
MIN_SPACE=0.007                      | MSPACE = 0.875 (= MIN_SPACE/NOM_SPACE.
                                     |                  if MIN_SPACE not set,
                                     |                  cdr14_min_spacing_factor cfg value
                                     |                  is taken)

NOM_LINE=0.008                       | LINE = 8.000
MIN_LINE=0.007                       | MLINE = 0.875 (= MIN_LINE/NOM_LINE.
                                     |                  if MIN_LINE not set,
                                     |                  cdr14_min_line_factor cfg value
                                     |                  is taken)

MULTI_LINE=NO                        | No direct translation to AOIProg command.

WORKING_TABLE=24x24                  | Appears as a comment in AOIProg file.

PADS {                               | PAD = 1:100.000, 1:200.000
    SYMBOL_NAME=r100
}
PADS {
    SYMBOL_NAME=r200
}


HOLES {                              | HOLE = 1:50.000, 1:75.000
    SYMBOL_NAME=r50
}
HOLES {
    SYMBOL_NAME=r75
}



STAGES {
    STAGE_NAME=COPPER
    ETCH_SET=YES
    ETCH=0.0005                      | ETCH \COPPER = 0.500
```

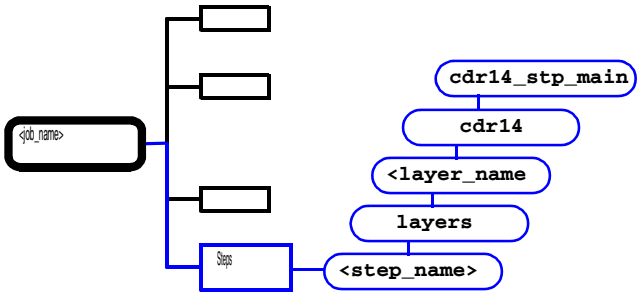
```
    CLASSES {
        CLASS_NAME=c_sig_cop            | CLASS \COPPER = c_sig_cop:drl
    }
    DRILL_LAYER=drl

    TOOLSET_NUMBER=99                   | If toolset alignment, translated to
                                        | TOOL \COPPER = 99, 10000.00,10000.0, H, RCCW270
                                        | (where 10000.00,10000.0, H, RCCW270 are the
                                        | Toolset parameters)


    MANUAL_ALIGNMENT {
        IS_USED=YES                     | If toolset alignment, IS_USED set to NO

        OFFSET {
            X=10
            Y=10
        }
        MIRROR=Y                        | Mirror around Y axis - Horizontal
        ANGLE=90                        | Rotation 90 deg CW (= 270 deg CCW)
        POLARITY=POSITIVE               | If Negative, NEG is added to the TT command.
    }                                   | The MANUAL_ALIGNMENT structure,
                                        | together with the layer's alignment targets,
                                        | determine the AOIProg's CT and TT commands:
                                        | CT \COPPER = 9000.000:9000.000:1:133.000,
1000.000:9000.000:1:133.000
                                        | TT \COPPER = 1000.000:1000.000:1,
1000.000:9000.000:1, H, RCCW270

    DRILL_LAYERS=drl                    | drill layer name is added to the CLASS command:
                                        | CLASS \COPPER = c_sig_cop:drl
                                        | In case of more than one drill layer, all defined
                                        | drill layers are merged into a single temporary
layer
                                        | named mdxxxxx
}

DRILLED_STAGE=0                         | No direct translation to AOIProg command

SCALE {                                 | No direct translation to AOIProg command
    X=1
    Y=1
}


SCALE_ORIGIN {                          | No direct translation to AOIProg command
    X=0
    Y=0
}

PANELIZATION=PANEL_DEFINED
GENESIS_VERSION=08.01DV
MARGINS_SET=0
X_MARGIN=0
Y_MARGIN=0
INSPECTED_STEPS=pcb                     | The steps that are translated into
                                        | PCB/RPCB AOIProg commands.
```

cdr14_stp_main (CDR14 Main Step)

`<job_name>/steps/<step_name>/layers/<layer_name>/cdr14/steps/cdr14_stp_main`

This step holds the cdr14 graphic data. Inspection areas and exclusion zones are placed in the **`lyr_area`** layer and alignment targets are placed in the **`lyr_targ`** layer.

See **`<job_name>/steps`** for step structure and **`<job_name>/steps/<step_name>/layers/`** for layer structure.

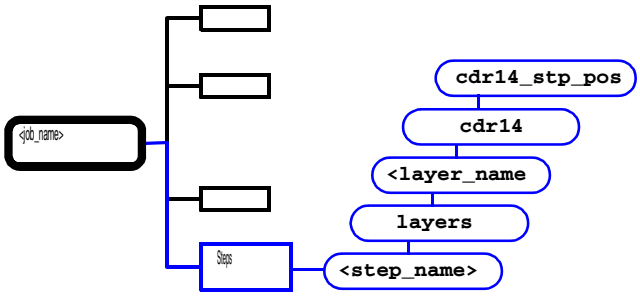


This step contains the steps described below, optionally step&repeated.

crd14_stp_pos (CDR14 Positive Step)

`<job_name>/steps/<step_name>/layers/<layer_name>/cdr14/steps/cdr14_stp_pos`

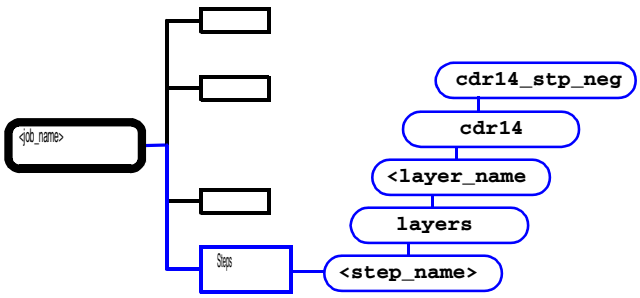
This step contains the inspection areas represented as positive features (a rectangular surface) in its **`lyr_area`** layer. This step can be placed by a maximum of one step&repeat command.



crd14_stp_neg (CDR14 Negative Step)

`<job_name>/steps/<step_name>/layers/<layer_name>/cdr14/steps/cdr14_stp_neg`

This step contains the non-step&repeated exclusion zones as a negative feature (a rectangular / polygonal surface or round pad) in its **`lyr_area`** layer.

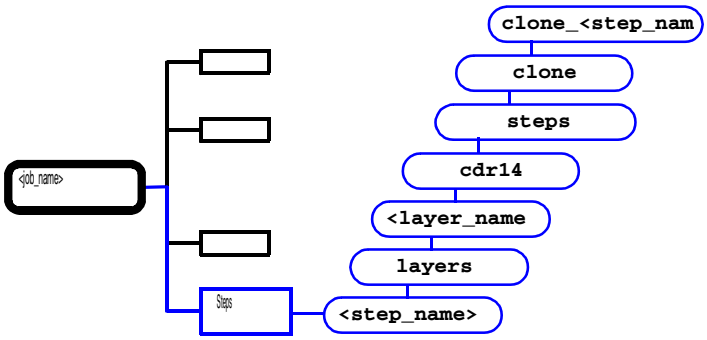


clone_<step_name> (S&R Exclusion Zones)

```
<job_name>/steps/<step_name>/layers/<layer_name>/cdr14/steps/clone_<step_name>
```

These steps contain the step&repeated exclusion zones as negative features (a rectangular/polygonal surface or round pad) in its **lyr_area** layer.

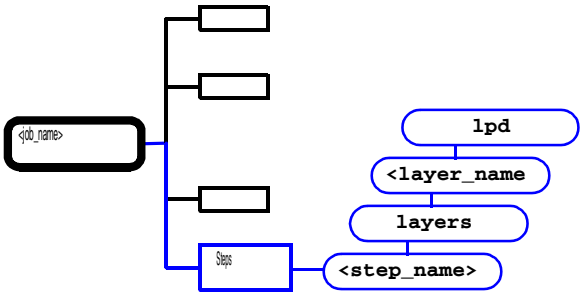
These steps are step&repeated with respect to the corresponding step **<step_name>**.



lpd (Layer Production Data)

Type:	Line Record Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/lpd**

This file sets Layer Production Data (LPD) values for IMG output.



Example

```
WAS_INPUT=YES
IS_DEFINED=YES
POLARITY=NEGATIVE
SPEED=0
XSTRETCH=100
YSTRETCH=100
XSHIFT=1.2
YSHIFT=1.2
XMIRROR=0
YMIRROR=0
COPPER_AREA=0
XCENTER=0
YCENTER=0
PLOT_KIND1=0
PLOT_KIND2=0
MINVEC=0
ADVEC=0
MINFLASH=0
ADFLASH=0
CONDUCTOR1=0
```

```
CONDUCTOR2=0
CONDUCTOR3=0
CONDUCTOR4=0
CONDUCTOR5=0
MEDIA=SECOND
RESOLUTION=QUARTER_MIL
SMOOTHING=SMOOTH
SWAP_AXES=SWAP
```

This file is only for the input and output of Optrotech Image files. When used for output the file is the **/<panel>/<layer>/layerhdr** file. Also refer to line mode command **image_set_lpd**.

Where:

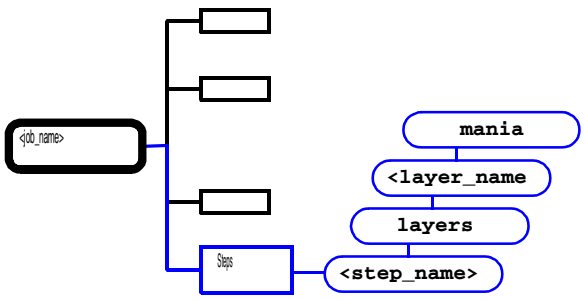
WAS_INPUT	YES, NO
IS_DEFINED	YES, NO
POLARITY	NEGATIVE, POSITIVE
MEDIA	FIRST, SECOND, THIRD
RESOLUTION	HALF_MIL, QUARTER_MIL
SMOOTHING	SMOOTH, ROUGH
SWAP_AXES	NO_SWAP, SWAP
COPPER AREA	in square inches
XSTRETCH, YSTRETCH	in percent
PLOT_KIND1, PLOT_KIND2	minimum = 0, maximum = 255
SPEED	minimum = 0, maximum = 255

Remaining values are in pixels as is standard for ODB++.

S *mania (MANIA Automatic Optical Inspection)*

Type:	Structured text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/mania**

The file contains parameters of a Mania set describing parameters to be used when testing the layer for this entity.



angle	Alignment of panel: Rotation value.
mirror	Alignment of panel: Mirroring value
offset	Alignment of panel: Offset value
<mirror>	N for not mirrored, M for mirrored
resolution	Mania pixel size
scan_area	Size of scan area to be used for testing
outdir	Location to place output files for the Mania Sapphire AOI machine
size_table	Set of legal space and track values

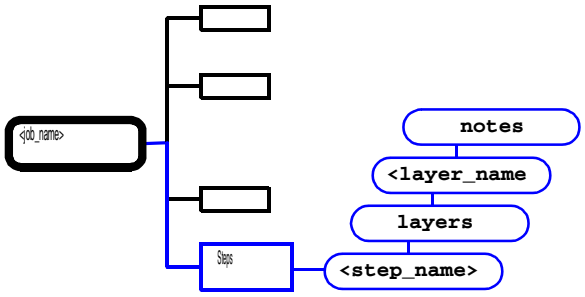
Example

```
ANGLE=0
MIRROR=NO
OFFSET {
     X=0
     Y=0
 }
RESOLUTION=0.8267716535433072
SCAN_AREA {
     X1=-1.613008070866142
     Y1=-1.437398326771653
     X2=2.081300787401575
     Y2=1.52845531496063
 }
OUTDIR=/tmp
SIZE_TABLE {
     SPACE=5
     TRACK=20
 }
SIZE_TABLE {
     SPACE=4
     TRACK=34
 }
```

notes (Electronic Job Notes)

Type:	Line Record Text
Compression:	None
Sum file:	Yes
Path	**<job_name>/steps/<step_name>/layers/<layer_name>/notes**

This file contains all the notes added by the user to the graphical layer.



Example

```
866467418,moshik,2.03807,
-1.22818,,,,,,First
line\nSecond line
```

Each line in the notes file has the following format:

```
<time>,<user>,<x>,<y>,
,,,,,<note>
```

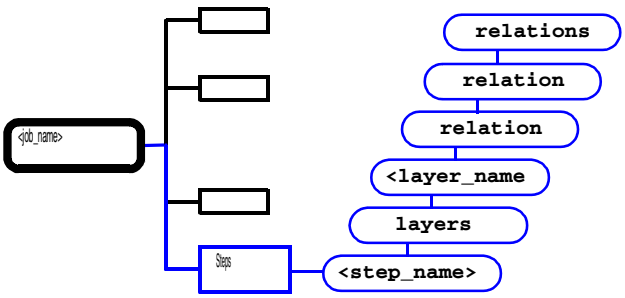
Where:

<time>	Last update date .UNIX time (seconds starting January 1st, 1970)
<user>	Last user updating the note
<x>,<y>	Graphic location, in inches
<note>	Up to 4 lines of text when the \n character describes the line break

relations (Connections between Features)

Type:	Structured Text
Compression:	None
Sum file:	Yes
<job_name>/steps/<step_name>/layers/<layer_name>/relation/relation/relations	

This optional file contains all dimension and connections between features in a layer.



Examples

```
"relations" {
    "version"=1
    "rel_type"=DIM
    "dimension" {
        "dmode"=DXDY
```

```
            "dx"=0
            "dy"=0.74
            "angle"=0
            "linetype"=HORZ
            "is_special"=0
            "source_f" {
                "type"=FEAT
                "feature" {
                    "index"=0
                    "mode"=ALL
                }
            }
            "dest_f" {
                "type"=FEAT
                "feature" {
                    "index"=1
                    "mode"=ALL
                }
            }
        }
        "graphic" {
            "grp-params" {
                "ang_arrmode"=EDGE
                "dim_arrmode"=EDGE
                "ang_boxmode"=SQR
                "dim_boxmode"=SQR
                "inline_mode"=ALL
                "outline_mode"=ALL
                "font_spec"=TMR10
                "text_sufx"=
            }
            "dim_x"=0
            "dim_y"=0.374
            "ang_x"=0
            "ang_y"=0
        }
    }

    "relations" {
        "version"=1
        "rel_type"=CON
        "connection" {
            "feature-1" {
                "type"=FEAT
                "feature" {
                    "index"=3
                    "mode"=PS
                }
            }
            "feature-2" {
                "type"=FEAT
                "feature" {
                    "index"=4
```

```
                "mode"=PE
            }
        }
        "feature-c" {
            "type"=FEAT
            "feature" {
                "index"=-1
                "mode"=ALL
            }
        }
        "mode"=CORNER
        "size1"=0
        "size2"=0
        "type_x"=DIST
        "type_y"=DIST
        "point_rel_2_f1"=ALL
        "func"=LINE2ARC
        "intersect"=0
        "radius"=0
    }
    "graphic" {
        "grp-params" {
            "ang_arrmode"=EDGE
            "dim_arrmode"=EDGE
            "ang_boxmode"=SQR
            "dim_boxmode"=SQR
            "inline_mode"=ALL
            "outline_mode"=ALL
            "font_spec"=
            "text_sufx"=
           }
        "dim_x"=0
        "dim_y"=0
        "ang_x"=0
        "ang_y"=0
    }
}
```

The file contains relations in the following structure:

`version` = always one (for future use)

`rel_type` = **`CON`** for connection or **`DIM`** for dimension.

connection or dimension structure = according to type

graphic structure:

grp-params structure:

ang_arrmode	graphic description of dimension arrow in angle dimension	**EDGE** - arrow head outline **FULL** - not supported **NONE** - no arrow
dim_arrmode	graphic description of dimension arrow in regular dimension	**EDGE/FULL/NONE**
ang_boxmode	graphic description of box displayed around dimension value in angle dimension:	**SQR** -square box **RND** -round box **NONE** -values only **EMPTY**-does not display any thing
dim_boxmode	graphic description of box displayed around dimension value in regular dimension	(options the same as above)
inline_mode	to display or not display the inner line of the dimension	**ALL** - display **NONE** - don't display
outline_mode	graphic description of the extension to the inner line of dimension:	**ALL** - show extended line as an extension to the inner line **HORZ** - shows extended line horizontally **VERT**- shows extended line vertically
font_spec	type and size of font used to display dimension values **<font type> <medium(M) or bold(B)> <regular(R) or italic(I)> <font size>**	where font types can be Times(**T**)/Helvetica(**H**)/ Courier(**C**)
text_sufx	if there is a suffix to attach to dimension text	
dim_x	place of dimension box in x (inches)	
dim_y	place of dimension box in y (inches)	
ang_x	place of angle box in x (inches)	
ang_y	place of angle box in y (inches)	

Dimension type structure:

dmode	only DXDY available	
dx	delta x in inches	
dy	delta y in inches	
angle	angle if exist	
linetype	for line destinations	**HORZ** - horizontal **VERT** - vertical **DIAG** - diagonal

is_special	if this dimension belongs to a symbol predefined **source_f** or **dest_f**	
source_f		**type** is always **FEAT** **feature** **index** - feature index in database **mode** - dimension mode **ALL** - all features **PS** - start of feature **PE** - end of feature
dest_f		**type** is always **FEAT** **feature** **index** - feature index in database **mode** - dimension mode **ALL** - all features **PS** - start of feature **PE** - end of feature

Connection type structure:

feature-1, feature-2	the features that are connected in the above feature type	
feature-c	the connecting feature in the above feature type(-1 if none)	
mode	connection mode	**ROUND/CORNER/CHAMFER**
size1	for chamfer connections	if type_x is **DIST**: distance in x to cut from feature in inches if type_x is **ANGLE**: angle between chamfer line and feature
size2	the same as above for y	
type_x	for chamfer connection if the above size in **DIST** or **ANGLE**	
type_y	the same as above for y	
point_rel_ 2_f1	in case of more than one intersection point between features ALL/PS/PE	
func	connection function:	**LINE2ARC** - intersection **LINE2CIRCLE** - line tangent to 2 circles **CIRCLE2LINE** - circle tangent 2 two lines **CIRCLE2CIRCLE** - arc tangent 2 two circles

intersect	for tangent features indication if arcs should be fixed.
radius	radius of round intersection and of circle

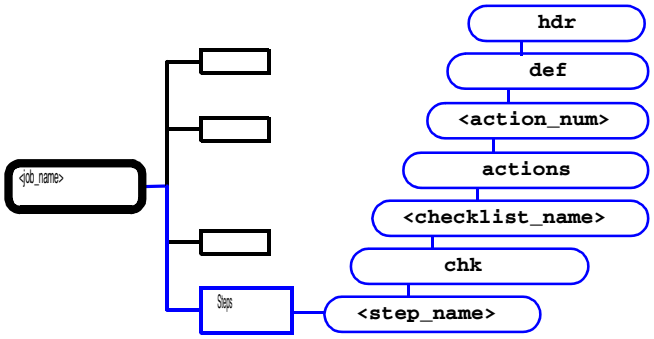
Note Some of the relations, such as Dimension Types and Connection Types, relate mostly for use in a rout layer, even though they can be used in any other layer. A rout layer should be created exclusively for the definition of a rout.

Chapter 7 Job>steps>chk (Checklists)

def/hdr (Definition Header)

Type:	Structured Text
Compression:	None
Sum file:	Yes
<job_name>/steps/<step_name>/chk/<checklist_name>/actions/<action_num>/def/hdr	

This file holds all the default values of the action parameters. It also determines the location of the action in the checklist.



Example

```
ROW = 1
NAME =
valor_analysis_signal
        ERF = STD 6
        UNITS = mm
        PARAMS {
                pp_layer = .type=signal | mixed & context=board
                pp_spacing = 8.0000
                pp_r2c = 24.0000
                pp_d2c = 24.0000
                pp_sliver = 4.0000
                pp_tests =
Spacing;Drill;Rout;Size;Sliver;Stubs;Center;SMD
        }
        LAST_PARAMS {
                ...
        }
        ONLINE_PARAMS {
                ...
        }
        ONLINE_VALUES {
                p2p = 6.0000
                p2c = 6.0000
                c2c = 6.0000
                ...
        }
        RANGES {
                p2p = 6.0000;8.0000;10.0000
                p2c = 6.0000;8.0000;10.0000
```

```
        c2c = 6.0000;8.0000;10.0000
        ...
}
VARIABLES {
        rm_d2c = 4
        rm_ar = 4
        ...
}
```

The hdr file consists of general parameters and several blocks of parameters.

The general parameters are:

ROW - the location of the action in the checklist (0 = first row)

NAME - the internal name of the action (matches the name inside the action code)

ERF - the current ERF model used

UNITS - the units of measurement used (**mm** or **inch**) in the ERF model.

PARAMS block:

This block contains the current values which were entered for the screen parameters. Each line has the structure:

<param> = <value>

The fields in this block depend on the particular action and its parameter definition in its ERF (External Rules File).

LAST_PARAMS block:

This block contains the parameters which were used the last time the action was run successfully.

ONLINE_PARAMS block:

This block contains the parameters which are to be used if the action is a part of an On-line DRC checklist.

ONLINE_VALUES Block

This block contains the pass/fail values which are used when the action is a part of an On-line DRC checklist. Each value corresponds to a category of the action and contains one floating point value (in mils).

RANGES block:

This block contains the ERF ranges which were used the last time the action's results were viewed. Each line has the structure:

<range> = <value1>;<value2>;<value3>

Note There can be between 1 and 5 values for each range.

VARIABLES block:

This block contains the ERF variables which were used the last time the action was run successfully. Each line has the structure:

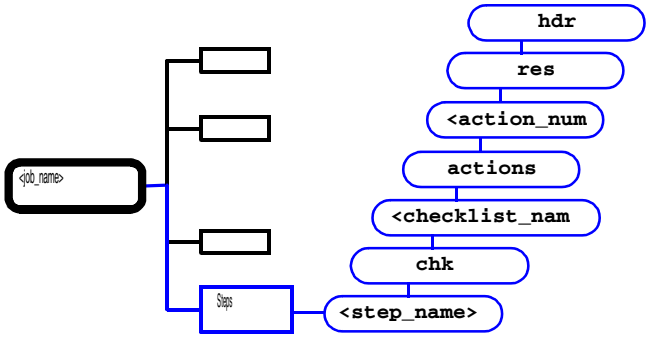
<variable> = <value>

These are the values of the variables in the ERF model.

S res/hdr (Results Header)

Type:	Structured Text
Compression:	None
Sum file:	Yes
<job_name>/steps/<step_name>/chk/<checklist_name>/actions/<action_num>/res/hdr	

This file contains some of the results of a successful run of an action. It includes the status, date, time and duration of run, a list of generated categories with their characteristics and a list of result attributes generated by the action run.



Example:

```
STAT = D
SEVERITY = 0
LAST_TIME = 848580240
DURATION = 755 SECONDS
CATEGORY {
   NAME = p2p
   TITLE = Pad to pad
   DESCRIPTION = Spacing;Pad 1;Pad 2
   SORT_ORDER = 0
   RANGE = 0.0040 0.0060 0.0080
}
...
ATTRIBUTE {
   sigt_par_space_5.0 = 5.134
   sigt_par_space_5.4 = 0.050
   sigt_par_space_5.6 = 0.074
   ...
}
```

The hdr file consists of some general parameters and several blocks of parameters. The general parameters are:

STAT	The status of the action. One of the values: **U** - undone, Action was not run yet **O** - outdate, Action parameters were changed since last run **D** - done, Action completed successfully **E** - error, Action terminated with an error
SEVERITY	0 (red), 1(yellow), 2 (green), 3 or 4
LAST_TIME	The time the action finished running the last time (in seconds, starting from January 1st, 1970.
DURATION	Number of seconds the action actually took to complete.

The **CATEGORY** block contains the following fields:

NAME	The formal name of the category, as appears in the ERF
TITLE	The title text of the category, to be displayed on screen
DESCRIPTION	Up to 3 indicators, separated by semi colons (;), describing the meaning of the measurement shape and feature(s) for this category.
SORT_ORDER	A number which defines the relative order of the category in the list of displayed categories in the result viewer.
RANGE	The range values which were defined in the ERF for this category the last time the action was run.

The **ATTRIBUTE** block contains the following fields:

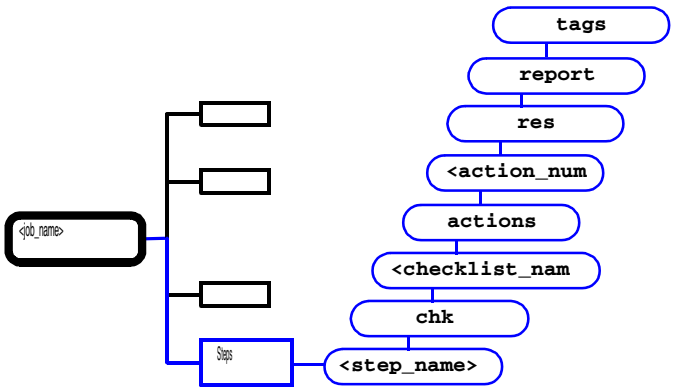
<attribute_name> = value

The result attribute names are different for each action.

S report/tags (Report Tags)

Type:	Line record text
Compression:	None
Sum file:	Yes
<job_name>/steps/<step_name>/chk/<checklist_name>/actions/<action_num>/res/report/tags	

The tags file is an optional file which links lines in the textual report with graphical locations in the associated step.



Example

```
2807 2851 0 -4 0
1237
2851 2900 -0.4 -
3.6 0 1238
2900 2949 -0.4 -
1.4 0 1239
2949 2997 4.1 -1.4
0 1240
2997 3043 3.7 -1 0 1241
3043 3091 4.1 -3.6 0 1242
...
```

Each line in the tags file has the following format:

<start_pos> <end_pos> <x> <y> <sres_num> <meas_num>

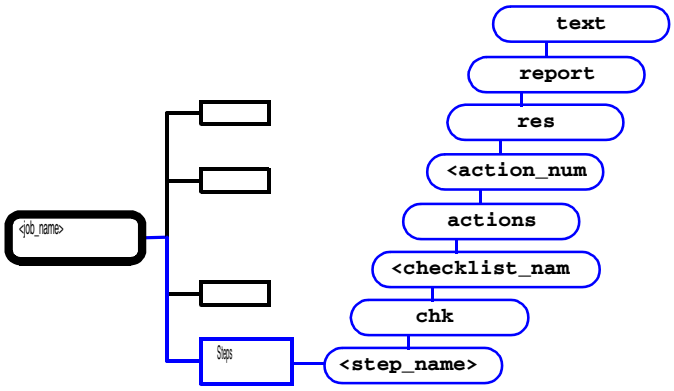
start_pos, end_pos	Start and end character position of the tag in the report
x,y	Graphic location, in inches

sres_num	The number of the sub-result within the action result (Each result directory contains one or more sub-result directories)
meas_num	The number of the measurement within the measurement of the sub result.

report/text (Text Report)

Type:	Free Text
Compression:	Yes
Sum file:	Yes
<job_name>/steps/<step_name>/chk/<checklist_name>/actions/<action_num>/res/report/text	

This file is a free text report created by the analysis or DFM action.



Example

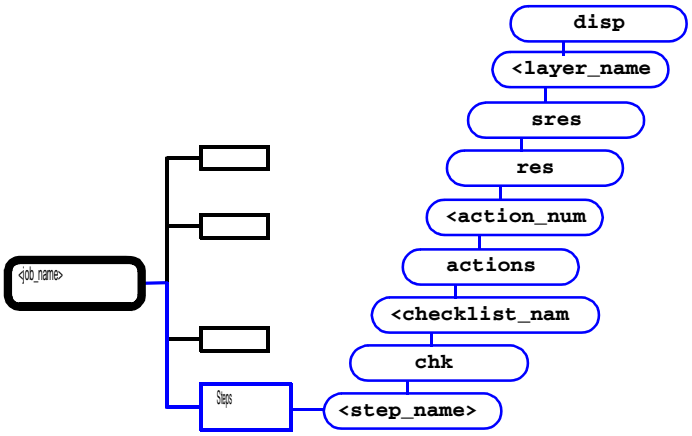
```
JOB   :  e.demo.001
STEP  :  pcb
DATE  :  28 Apr 97
TIME  :  18:57:14
Created on GENESIS
******************
***************
Action : Signal
layer checks
Layer  : sigt

Spacing Checks
**************

  Pad to pad
  ==========
  No results

  Pad to circuit
  ==============
  Size(Spacing) / Count
  --------------------
  3- 4                    4
  4- 5                    4
  ....
```

This file has a free format. Please refer to the specific action for special details.

S disp (Display Records for Measurements)

Type:	Structured Text
Compression:	None
Sum file:	Yes
<job_name>/steps/<step_name>/chk/<checklist_name>/actions/<action_num>/res/sres/<layer_name>/disp	

This file contains display records. Display records are groups of layers which will be displayed when a certain measurement is selected. Each measurement (see below) may refer to a display record from this file.



Example

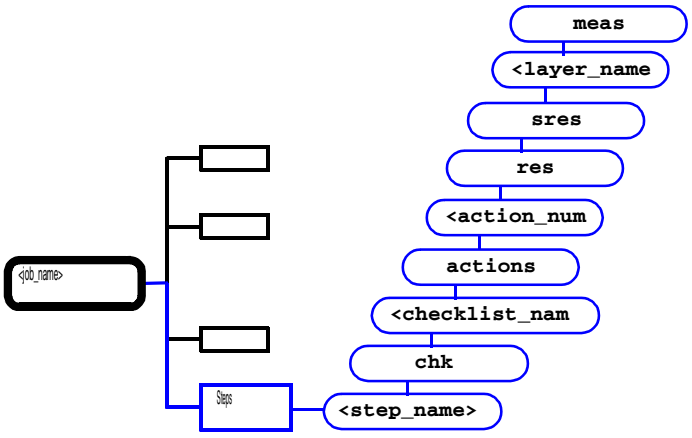
```
DISPLAY {
   L0 = sigt
}
DISPLAY {
   L0 = sigt
   L1 = drill
}
DISPLAY {
   L0 = sigt
   L1 = rout
}
```

The file includes an array of **DISPLAY** records. Each record contains a variable (up to 4) number of layers, tagged L0 to L3. Each layer should reference an existing layer in the job matrix. Nonexisting layers are ignored.

S meas (Measurements)

Type:	Line Record Text
Compression:	Yes
Sum file:	Yes
<job_name>/steps/<step_name>/chk/<checklist_name>/actions/<action_num>/res/sres/<layer_name>/meas	

This file contains all the measurements which were created for an Action in one sub-result (one layer), for all categories.



Example

```
0 1 0 N P r60 L r12
S SG 0.3 -3.07 0.3 -
3.06496
1 1 0 N P r60 L r12
S SG 2.4 -3.33 2.4 -
3.33491
2 1 0 N P r60 L r12
S SG 3.3 -3.33 3.3 -
3.33491
3 1 0 N P r60 L r12
...
11 24 0 A L r12 P r60
S PT 3.08286 -3.42462
12 26 1 N P r33 P r60
S SG -0.2165 -2 -0.23 -2
13 26 1 N P r33 P r60
S SG 0.6165 -2 0.63 -2
...
272 10 0 N P r60
S RC 0.87 -1.73 0.06 0.06
273 10 0 N P r60
S RC 0.87 -1.63 0.06 0.06
...
```

Each measurement in the file consists of 2 or 3 lines, depending on existence of Scalar Value:

1. Measurement identification, always the first line.
2. Scalar Value (optional) - when available, second line.
3. Measurement shape description. If a scalar value is defined, this is the third line; if not, it is the second line.

1. Measurement Identification Structure:

<meas_num> <cat_num> <disp_num> <alarm> <ftype1> <fsym1> <ftype2> <fsym2>

Where:

<meas_num>	Serial number of measurement (0 and up). A dash (-) prefix signifies a reference measurement.
<cat_num>	Category number (0 and up) which must refer to a valid category in the **res/hdr f**ile.
<disp_num>	The display record number (0 and up) which must refer to a valid category in the **res/sres/<layer_name>/disp** file.
<alarm>	**N** (no alarm) or **Y** (alarm). Action may generate alarm measurements which can be listed together with tags, in the textual report.
<ftype>	Type of feature which contributed to the measurement: **L** Line **P** Pad **S** Surface **A** Arc **T** Text feature **C** Component (top) **c** Component (bottom) **N** Net **D** Diff. pair **X** Free text **V** Scalar value **U** Unit-sensitive scalar value **Q** Square area scalar value
<fsym>	Symbol of feature/component which contributed to the measurement: For **L,P,S,A** and **T** - name of a valid symbol For **C** or **c** - reference designator of the component For **N** - name of the net For **D** - name of the differential pair net For **X** - a text string (without spaces) For **V**, **U** and **Q** - a scalar value

For **V**, **U** and **Q** the **<fsym>** scalar value is the value given to the **ODB_RES_SET_MEAS_ID_SCALAR** function. For example, if the Measurement ID was added using the call **ODB_RES_SET_MEAS_ID_SCALAR (1, VAL2NONE(val), 0,0)**; then '**VAL2NONE (val)**' is the **<fsym>** value.

Note There may be 0, 1 or 2 pairs of **<ftype> <fsym>**

2. Scalar Value Structure (optional)

V <scalar value> <unit sensitive>

<scalar value>	Numeric value
<unit sensitive>	**U** - unit-sensitive scalar value. To display, convert according to required system units (imperial/metric). **N** - non unit-sensitive scalar value. Displayed value is not affected by the system units (imperial/metric). **S** - square units scalar value representing area (mil^2). To display, convert according to required system units (imperial/metric).

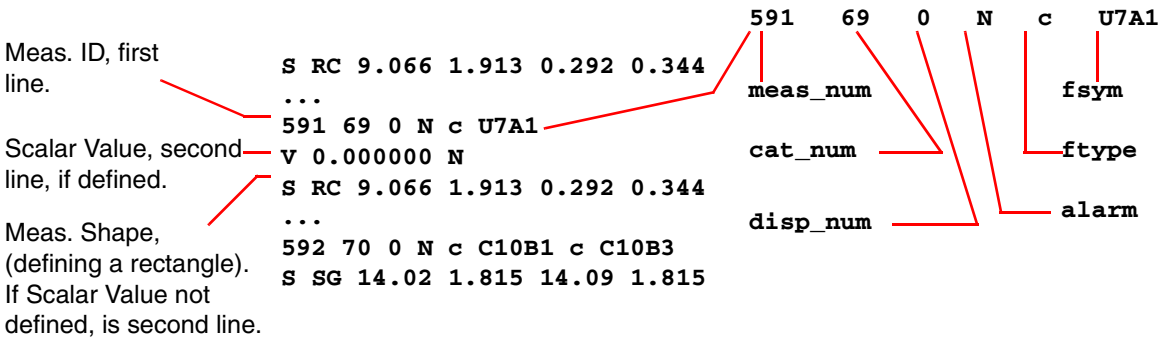
3. Measurement Shape Structure:

S PT <x> <y>	Point
S SG <xs> <ys> <xe> <ye>	Segment
S CU <xs> <ys> <xe> <ye> <xc> <yc> <cw>	Curve
S LN <xs> <ys> <xe> <ye> <size> <cap>	Line
S CR <x> <y> <size>	Circle
S SQ <x> <y> <size>	Square
S RC <xll> <yll> <xur> <yur>	Rectangle
S AR <xs> <ys> <xe> <ye> <xc> <yc> <size> <cap> <cw>	Arc

Where:

<x>,<y>	Center coordinates (inches) of a point, circle, square
<xs>,<ys>	Start point of segment, curve, line, arc
<xe>,<ye>	End point of segment, curve, line, arc
<xc>,<yc>	Center point of curve and arc
<cw>	**Y** - clockwise, **N** - counter clockwise (for curve, arc)
<size>	Diameter of the brush of a line, circle, square or arc
<xll>,<yll>	Lower left point of a rectangle
<xur>,<yur>	<**xur**> - width of rectangle, <**yur**> - height of rectangle (relative to the lower left corner at <**xll**> and <**yll**>)
<cap>	**r** - round, **s** - square

The following example clarifies:



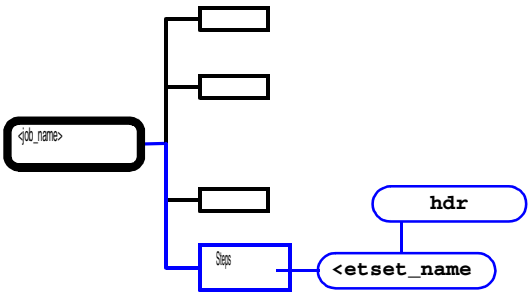
Chapter 8 Job>Steps>et (Electrical Test)

<etset_name>/hdr

Type:	Structured Text
Compression:	None
Sum file:	No
Path	**<job Name>/steps/<step_name>/et/<etset_name>/hdr**

All the coordinates in this section are taken to be board coordinates. Within ETM (Electrical Test Manager) we use two coordinate systems: board and adapter coordinates. Board coordinates are the coordinate system found throughout Enterprise / Genesis, whereas Adapter coordinates refer to the coordinate system as they should appear within the adapter.

This file provides general information at the **etset** level.



Example

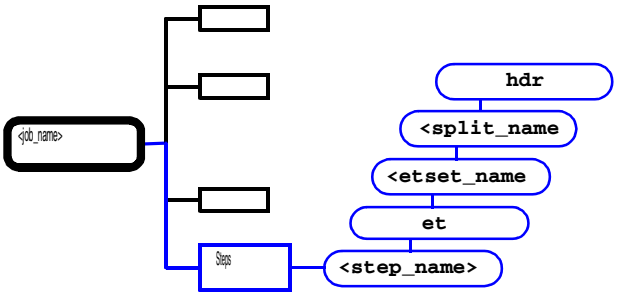
```
X_DATUM=16.33848897637795
Y_DATUM=12.99838996062992
WIDTH=7.322835039370079
HEIGHT=5.960630019685039
STATUS=new
NET_STATUS=undefined
```

X_DATUM	X-coordinate of the datum (inches)
Y_DATUM	Y- coordinate of the datum (inches)
WIDTH	Total width of the data (inches)
HEIGHT	Total height of the data (inches)
STATUS	New/ready/not_ready
NET_STATUS	Undefined/ok/disrupted

S <split_name> / hdr

Type:	Structured Text
Compression:	None
Sum file:	No
Path	**<job Name>/steps/<step_name>/et/<etset_name>/<split_name>/hdr**

This file provides general information at the split level.



Example

```
ACCESS=grid_grid
ADAPTER=mct_tws
TESTER=mania
STYLE=mania
TRANS_NET2A {
    XC=0
    YC=0
    XOFF=-4.905
    YOFF=-9.633749999999999
    ROTATE=0
    MIRROR=no
}

TRANS_A2NET {
    XC=0
    YC=0
    XOFF=-4.905
    YOFF=-9.633749999999999
    ROTATE=0
    MIRROR=no
}

READY_STATE {
    SPLIT_READY=yes
    NETS_NOT_CHANGED=yes
    ALL_PINS_ASSIGNED=no
    P2G_NOT_CHANGED=yes
    PLATES_READY=no
    FA_DONE=no
    MAP_DONE=no
    DRILL_FILES_READY=no
    OUTPUT_DONE=yes
}

READY=no
COMPONENT_SIDE_UP=no
CHANGED=no

NC_SET_PARAMETERS {
    TYPE=1
```

```
    NAME=format
    VALUE=excellon
}

NC_SET_PARAMETERS {
    TYPE=3
    NAME=test_ar_above
    VALUE=135.0
}

NC_SET_PARAMETERS {
    TYPE=4
    NAME=stagger
    VALUE=yes
}

NC_SET_PARAMETERS {
    TYPE=4
    NAME=finished_drills
    VALUE=yes
}

OUT_FORMAT=mania_b640
CAR_FORMAT=tws2000

ADAPTER_POS {
    REFFERS=profile
    ALIGN=5
}
```

This file contains nine global parameters and four arrays.
The global parameters are:

ACCESS	At present only flying probe and universal (grid) testers are supported. Hence the options are: NO_TEST/FP/ TOP_GRID/BOT_GRID/GRID_GRID
ADAPTER	User name for adapter
TESTER	mania/ everett charles/ circuitline/ luther maelzer/ probot/ bsl/ integritest/ microcraft/ atg
STYLE	regular/ mania (meaning that the pin can bend)
READY	No/Yes - for internal use
COMPONENT_SIDE_UP	No/Yes
CHANGED	No/Yes - for internal use
OUT_FORMAT	mania_b640/evc/circuit_line/lm-udl/probot/bsl/ integritest/ microcraft/atf/tif/tti/anf/ ipc356/ipcd-356a
CAR_FORMAT	None/epc/tws2000

The arrays **TRANS_NET2A** and **TRANS_A2NET** describe the transformation when converting from Board coordinate system to Adapter coordinate system and vice-versa. Their fields are the standard transformation fields, which are:

XC	X-coordinate of the centre of the transformation
YC	Y-coordinate of the centre of the transformation
XOFF	X-offset of the transformation
YOFF	Y-offset of the transformation
ROTATE	(0/1/2/3) x 90°
MIRROR	No/Yes

The fields of the array **READY_STATE** are (for internal use only):

SPLIT_READY	No/Yes
NETS_NOT_CHANGED	No/Yes
ALL_PINS_ASSIGNED	No/Yes
P2G_NOT_CHANGED	No/Yes
PLATES_READY	No/Yes
FA_DONE	No/Yes
MAP_DONE	No/Yes
DRILL_FILES_READY	No/Yes
OUTPUT_DONE	No/Yes

The fields of the array **OUTPUT_PARAMETRS** and **NC_SET_PARAMETRS** are:

TYPE	0-empty/ 1-text/ 2-integer/ 3-double/ 4-boolean
NAME	Parameter name
VALUE	Parameter Value

The fields of the array ADAPTER_POS are:

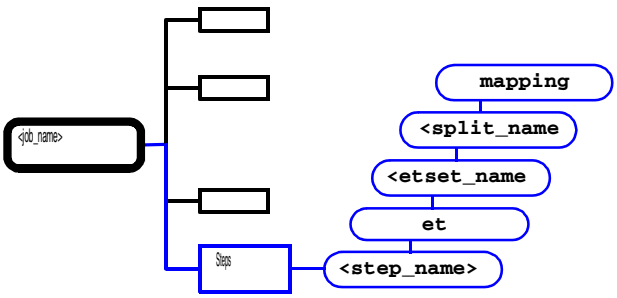
REFFERS	Profile/ net_limit
ALIGN	0-empty, 1-upper-left, 2-upper-mid, 3-upper-right, 4-mid-left, 5-mid-mid, 6-mid-right, 7-lower-left, 8-lower-mid, 9-lower-right

S *<split_name> / mapping*

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job Name>/steps/<step_name>/et/<etset_name>/<split_name>/mapping**

This file gives general information at the split level.

Example




```
   #
# Mapping File
#
#                                  GNET GRID      GRID      ASSIGN TEST    TEST
#  ID     X         Y       PIN     NUM IND X     IND Y     TO     ALLOWED FROM
#
    0  6.100000  5.449000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 0
    1  6.100000  5.349000 39003d    0  6.095000  5.366250 TOP     TOP       TOP       ON_AR N NET# 0
    2  6.000000  5.324000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 0
    3  6.050000  5.349000 39003d    0  5.995000  5.366250 TOP     TOP       TOP       ON_AR N NET# 0
    4  7.050000  5.449000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 1
    5  7.050000  5.349000 39003d    0  7.095000  5.266250 TOP     TOP       TOP       ON_AR N NET# 1
    6  6.938000  5.424000 39003d    0  6.995000  5.466250 BOT     BOT       BOT       ON_AR N NET# 1
    7  6.450000  6.324000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 2
    8  6.350000  6.324000 39003d    0  6.295000  6.366250 TOP     TOP       TOP       ON_AR N NET# 2
    9  7.375000  6.324000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 3
   10  7.275000  6.324000 39003d    0  7.195000  6.366250 TOP     TOP       TOP       ON_AR N NET# 3
   11  7.050000  6.874000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 4
   12  7.050000  6.949000 39003d    0  7.095000  6.866250 TOP     TOP       TOP       ON_AR N NET# 4
   13  6.600000  6.899000 39003     0  6.595000  6.866250 BOT     BOT       BOT       ON_AR N NET# 4
   14  6.100000  6.874000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   15  6.100000  6.949000 39003d    0  6.095000  6.966250 TOP     TOP       TOP       ON_AR N NET# 5
   16  8.950000  5.824000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   17  9.050000  5.424000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   18  8.750000  5.824000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   19  8.450000  5.924000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   20  8.300000  5.674000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   21  9.150000  5.124000 NO_PIN    0  0.000000  0.000000 NONE    BOTH      NO        ON_AR N NET# 5
   22  9.675000  5.674000 39003     0  9.695000  5.666250 TOP     TOP       TOP       ON_AR N NET# 5
```

Each row has the following structure

<id> <board x> <board y> <pin> <grid num> <grid x> <grid y> <assign> <testable side> <side tested> <on annular ring> <net number>

Id	ID number of the point
Board x,board y	Location of the test point on the board
Pin	Pin name assigned to this point
Grid num	The number of the grid structure used
Grid x,grid y	The grid position assigned to this point
Assign	The grid side to whom the point is assigned
testable side	The side from where the point can be tested
side tested	The side from where the point should be tested

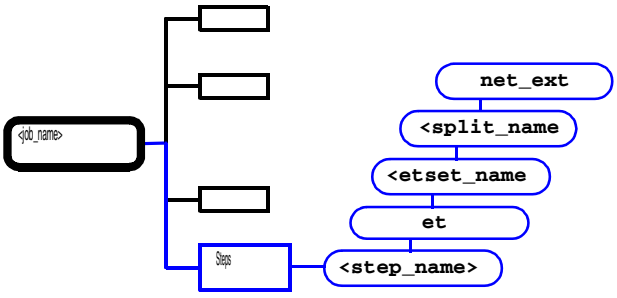
on annular ring	Whether the test point is on the annular ring
net number	Corresponds to the number in the netlist

All the sides here refer to the board side (ie. Top is the component side etc.)

S <split_name> / net_ext

Type:	Line Record Text
Compression:	None
Sum file:	No
Path	**<job Name>/steps/<step_name>/et/<etset_name>/<split_name>/net_ext**

This file gives general information regarding the test status of each net in this specific split.



Example

```
ET_NET {
    NETNUM=0
    TYPE=all
}

ET_NET {
    NETNUM=1
    TYPE=all
}

ET_NET {
    NETNUM=2
    TYPE=shorts
}
...
```

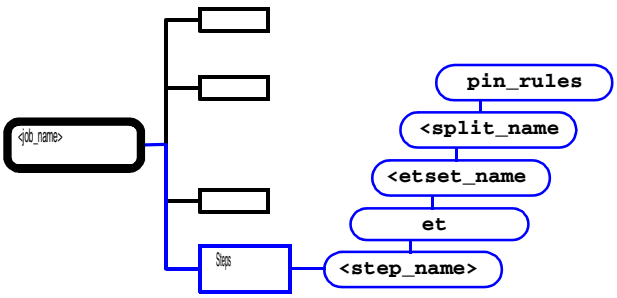
Each array **ET_NET** has the following structure:

<NETNUM>	Number of the net as in the netlist
<TYPE>	NO_TEST/SHORTS/CONNECT/ALL

S <split_name> / pin_rules

Type:	Structured Text
Compression:	None
Sum file:	No
Path	**<job Name>/steps/<step_name>/et/<etset_name>/<split_name>/pin_rules**

This file lists the rules to be used in assigning pins. The pins themselves are defined in the **<pin_name>** section.



Example

```
RULE {
    NAME=39003
    TYPE=def_smd
    SIDE=both
    PITCH_MIN=0
    PITCH_MAX=0
    SIZE_MIN=0
    SIZE_MAX=0
    DIST_X=0
    DIST_Y=0
    NUM_NEEDED=0
    EQUALS_TO=1
    CONTACT_MIN=0
    ALIGN_MIN=0
    ALIGN_MAX=0
```

This file is made up of repetitions of the **RULE** array;

NAME	Name used to refer to the pin
TYPE	def_smd/ def_pin/ def_over/ smd/ hole/ npth/ comp/ air/ tool/ fp_info/ alignment
SIDE	Side as determined by the placement in the adapter (both/top/ bot)
PITCH_MIN	Minimum pitch required for this pin
PITCH_MAX	Maximum pitch required for this pin
SIZE_MIN	Minimum contact size for the pin head
SIZE_MAX	Maximum contact size for the pin head
DIST_X ,DIST_Y	Distance between air-holes
NUM_NEEDED	Minimum number required
EQUALS_TO	Number of pins equivalent to a compensation post
CONTACT_MIN	Contact size for flying probe
ALIGN_MIN	Minimum number of alignment points for flying probe output
ALIGN_MAX	Maximum number of alignment points for flying probe output

This table shows which field is relevant to each type

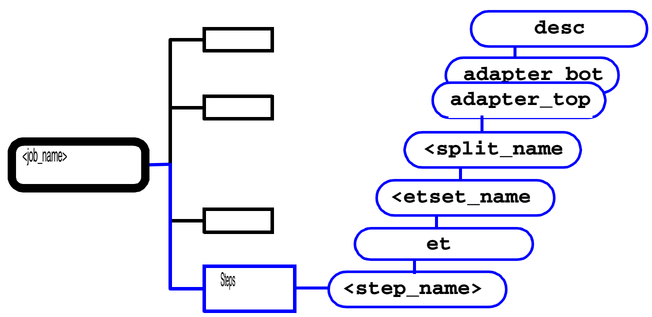
TYPE	NAME	SIDE	PITCH	SIZE	DIST	NUM_NEEDED	EQUALS_TO	CONTACT	ALIGN
def_smd	x	x							
def_pin	x	x							
def_over	x	x							
smd	x	x	x	x					

hole	X	X		X					
npth	X	X		X		X			
comp	X	X					X		
air	X	X			X				
tool	X	X				X			
fp_info			X	X				X	
alignment									X

S *adapter_top(bot) / desc*

Type:	Structured Text
Compression:	None
Sum file:	No
<job Name>/steps/<step_name>/et/<etset_name>/<split_name>/adapter_top(bot)/desc	

This file provides general information describing the build-up of the adapter.



Example

```
X_MIN=-2.605
Y_MIN=-0.93375
X_MAX=18.295
Y_MAX=13.56625

GRID {
    STEP_X=0.1
    STEP_Y=0.1
    X_MIN=-1.705
    Y_MIN=-0.03375
    X_MAX=17.395
    Y_MAX=12.66625
}
HEIGHT=3.75

PLATES {
    ELEVATION=0.1125
    THICKNESS=0.12
    COUNTER_SINK_TOP_H=0
    COUNTER_SINK_TOP_R=0
    COUNTER_SINK_BOT_H=0
    COUNTER_SINK_BOT_R=0
    PLATE_NAME=tus
    CONST_DRILL=tus.3
    SPEC_PROCESS=none
    MASK_SIZE=0
    GUIDING_MAX_PITCH=0
    GUIDING_DEFL_LIMITS=0
    GUIDING_SMALL_SIZE=0
    GUIDING_BIG_SIZE=0
```

```
    GUIDING_MARGIN=0

    TRANS_PLATE {
        XC=0
        YC=0
        XOFF=0
        YOFF=0
        ROTATE=0
        MIRROR=no
    }
}

TOOLS {
    NAME=man_tool2
    X=16.9298279527559
    Y=13.91177598425197
}
```

This file contains five global parameters and three arrays. The global parameters are:

X_MIN, Y_MIN	Bottom left corner of the adapter
X_MAX, Y_MAX	Top right corner of the adapter
HEIGHT	Vertical distance from the grid to the board

Numerous grids can be defined. For example double density grids require two definitions of grids each with a step of 100 mil with a 50 mil step between them.

The fields of the **GRID** array are:

STEP_x, STEP_Y	Grid step in inches
X_MIN, Y_MIN	Bottom left corner of the grid
X_MAX, Y_MAX	Top right corner of the grid

The **PLATES** array has a sub array, **TRANS_PLATE**. The fields of the **PLATES** array are:

ELEVATION	Distance from the board to the plate
THICKNESS	Thickness of the plate
COUNTER_SINK_TOP_H	Counter sink depth for the upper side
COUNTER_SINK_TOP_R	Counter sink drill holes on top layer with a radius less than this value
COUNTER_SINK_BOT_H	Counter sink depth for the lower side
COUNTER_SINK_BOT_R	Counter sink drill holes on bottom layer with a radius less than this value
PLATE_NAME	Name of the plate
CONST_DRILL	Name of the layer from which the constant drills are taken
SPEC_PROCESS	NONE/ SHOULDER/ MASK/ GUIDING
MASK_SIZE	Standard drill size used for the mask layer

GUIDING_MAX_PITCH	Maximum pitch required for detecting rows for guiding plate
GUIDING_DEFL_LIMITS	Below this value, **GUIDING_SMALL_SIZE**. is used
GUIDING_SMALL_SIZE	Standard drill size for pins with a small deflection
GUIDING_BIG_SIZE	Standard drill size for pins with a large deflection
GUIDING_LINE_SIZE	Size of the routing line
GUIDING_MARGIN	Additional border width around the component area

The fields of the sub array **TRANS_PLATE** (the transformation parameters for each plate to be used in the output of the drill file) are:

XC,YC
XOFF,YOFF
ROTATE
MIRROR

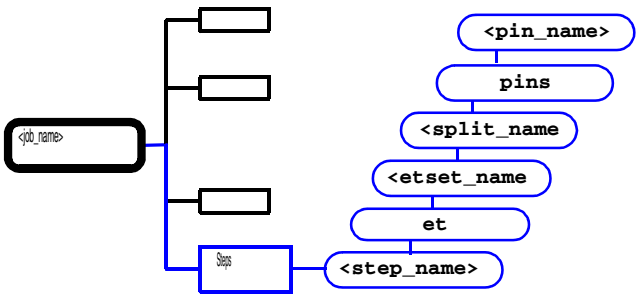
The fields of the **TOOLS** array are:

NAME	Name of the tooling pin
X,Y	Its position on the board

S *pins / <pin_name>*

Type:	Structured Text
Compression:	None
Sum file:	No
<job Name>/steps/<step_name>/et/<etset_name>/<split_name>/pins/<pin_name>	

For each pin there should be the following structure to describe its setup.



Example

```
NAME=39003d
TYPE=probe
SYMBOL=oval24.0x24.
0
HEIGHT=3.75
DEFLECTION=0.3
MAN_DEFLECTION=0.3
BOARD_SNAP=pad
GRID_SNAP=grid
CRIMP=0

MEASURES {
   OFFSET=0
```

```
    DIAMETER=0.024
    SPACING=0
    DRILL_SIZE=0.029
}

MEASURES {
    OFFSET=1
    DIAMETER=0.024
    SPACING=0
    DRILL_SIZE=0.029
}

MEASURES {
    OFFSET=1
    DIAMETER=0.024
    SPACING=0
    DRILL_SIZE=0.045
}

MEASURES {
    OFFSET=3.75
    DIAMETER=0.024
    SPACING=0
    DRILL_SIZE=0.045
}
```

This file contains nine global parameters and one array. The global parameters are:

NAME	Name of the pin
TYPE	probe/ tooling/align_pt
SYMBOL	Name of the symbol used to represent the pin in the display
HEIGHT	Total pin height
DEFLECTION	Normal maximum deflection allowed
MAN_DEFLECTION	Maximum deflection possible
BOARD_SNAP	pad/ npth/ empty/ none
GRID_SNAP	grid/ gnone
CRIMP	Not in use

The fields of the **MEASURES** array are:

OFFSET	Distance from the pin head
DIAMETER	Cross-sectional diameter of the pin at this offset
SPACING	Required spacing for the pin at this offset
DRILL_SIZE	Recommended drill size (if zero then the drill size is calculated)

There is also another tree, which has some of the above files.

Under **/genesis/sys/hooks/** there is the "**et**" directory. Here are stored a library of pins and adapters that are generally available.

et *(scripts specifically used in etm)*
pins *(parent pin library)*
<pin_name> *(explained above)*
adapters
<adapter_name>
hdr *(explained above)*
pin_rules *(explained above)*
adapter_top
desc *(explained above)*
const_drill
<layer_name> **(***according to standard definitions)*
adapter_bot
desc *(explained above***)**
const_drill
<layer_name>*(according to standard definitions)*
pins *(pins defined with rules for each adapter)*
<pin_name> *(explained above)*

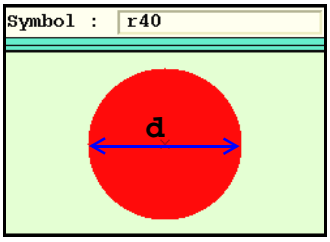
Chapter 9 Symbol Definition

Standard Symbols

The system supports the following standard (system) symbols:

Round

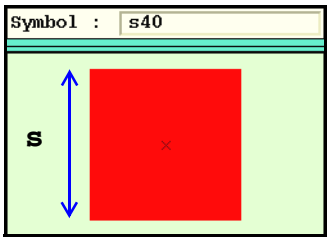
r<d>



d - circle diameter

Square

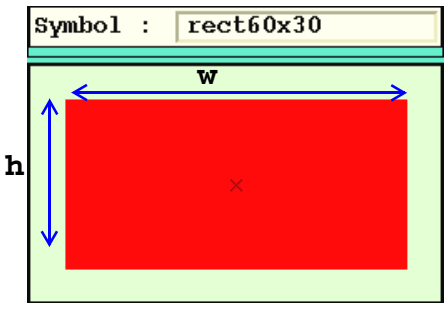
s<s>



s - square side

Rectangle

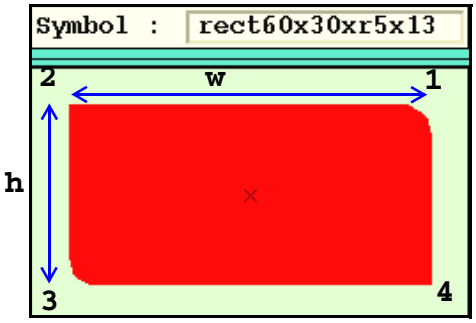
rect<w>x<h>



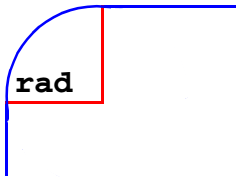
w - rectangle width
h - rectangle height

Rounded Rectangle

rect<w>x<h>xr<rad>x<corners>

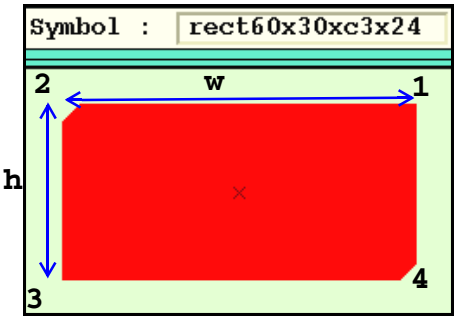


w - rectangle width
h - rectangle height
rad - corner radius
corners - a combination of **1,2,3,4** to specify which corners are rounded. **x<corners>** can be omitted if all corners are rounded)
1 - upper right
2 - upper left
3 - lower left
4 - lower right

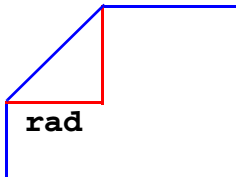


Chamfered Rectangle

rect<w>x<h>xc<rad>x<corners>

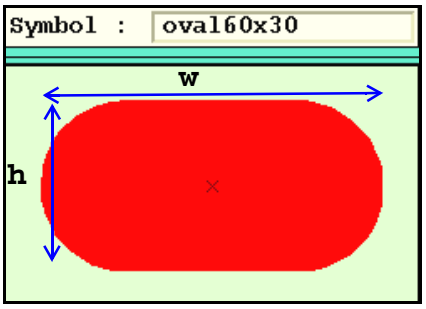


w - rectangle width
h - rectangle height
rad - corner radius
corners - a combination of **1,2,3,4** to specify which corners are rounded (**x<corners>** can be omitted if all corners are rounded)
1 - upper right
2 - upper left
3 - lower left
4 - lower right



Oval

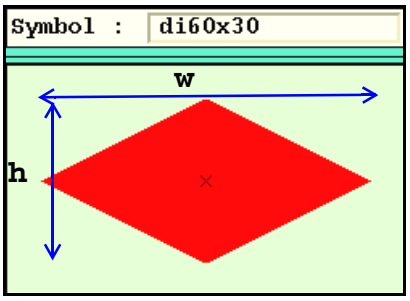
oval<w>x<h>



w - oval width
h - oval height

Diamond

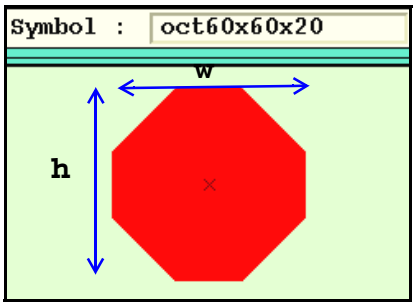
di<w>x<h>



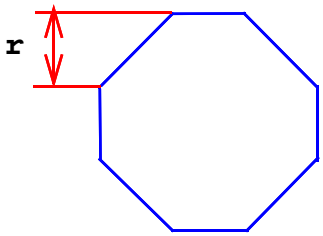
w - diamond width
h - diamond height

Octagon

oct<w>x<h>x<r>

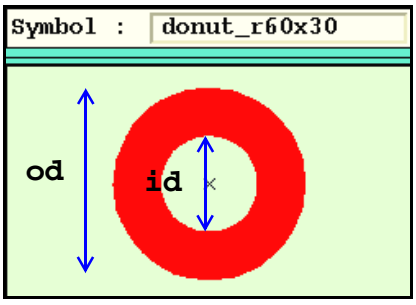


w - octagon width
h - octagon height
r - corner size



Round Donut

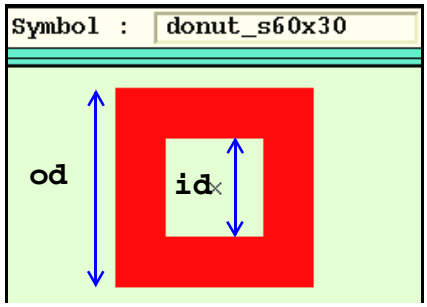
donut_r<od>x<id>



od - outer diameter
id - inner diameter

Square Donut

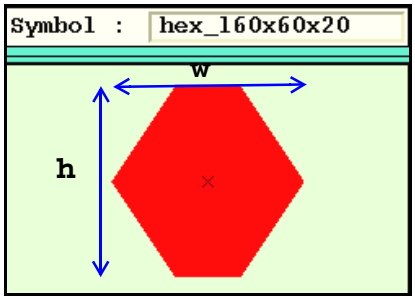
donut_s<od>x<id>



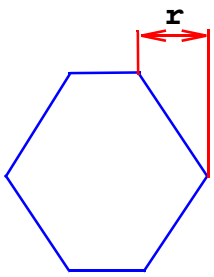
od - outer diameter
id - inner diameter

Horizontal Hexagon

hex_l<w>x<h>x<r>

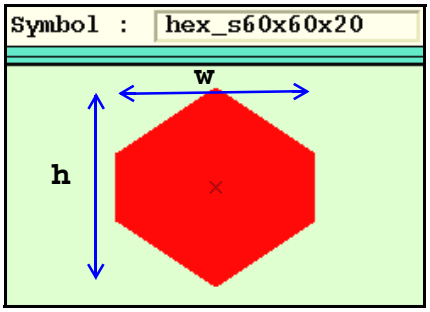


w - hexagon width
h - hexagon height
r - corner size

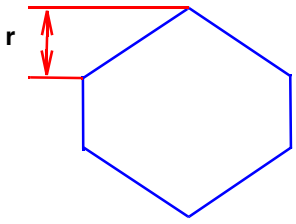


Vertical Hexagon

hex_s<w>x<h>x<r>

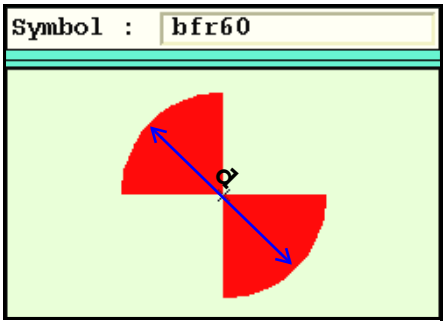


w - hexagon width
h - hexagon height
r - corner size



Butterfly

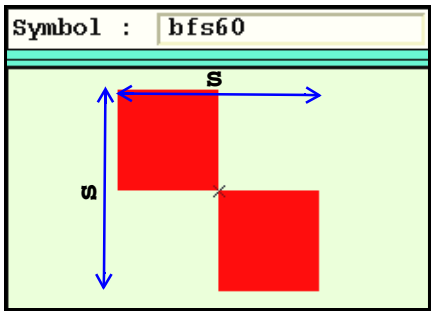
bfr<d>



d - diameter

Square Butterfly

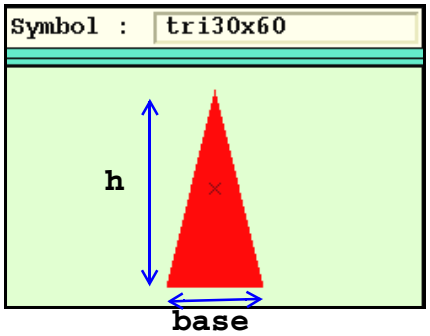
bfs<s>



s - size

Triangle

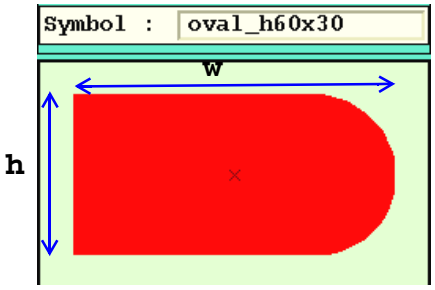
tri<base>x<h>



base - triangle base
h - triangle height

Half Oval

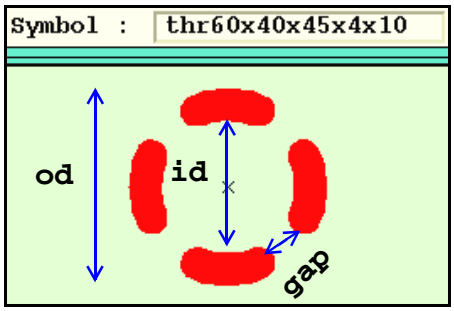
oval_h<w>x<h>



w - width
h - height

Round Thermal (Rounded)

thr<od>x<id>x<angle>x<num_spokes>x<gap>

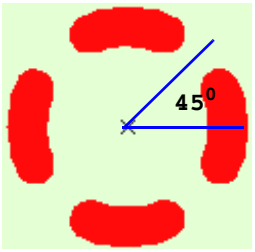


num_spokes = 4

od - outer diameter
id - inner diameter
angle - spoke angle
num_spokes - number of spokes
gap - size of spoke gap

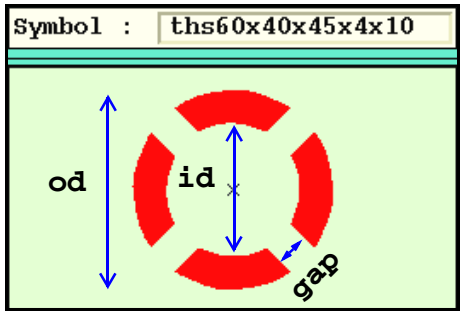
Specification of **od** and **id** determine the air gap (size of laminate separation)

angle



Round Thermal (Square)

ths<od>x<id>x<angle>x<num_spokes>x<gap>

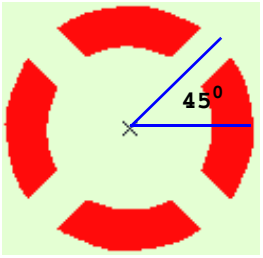


num_spokes = 4

od - outer diameter
id - inner diameter
angle - gap angle from 0^0
num_spokes - number of spokes
gap - size of spoke gap

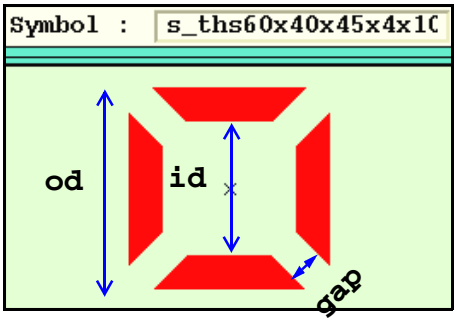
Specification of **od** and **id** determine the air gap (size of laminate separation)

angle



Square Thermal

s_ths<od>x<id>x<angle>x<num_spokes>x<gap>

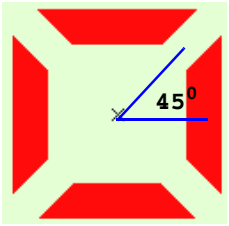


num_spokes = 4

od - outer diameter
id - inner diameter
angle - gap angle from 0^0
num_spokes - number of spokes
gap - size of spoke gap

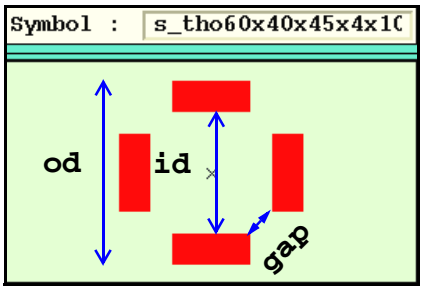
Specification of **od** and **id** determine the air gap (size of laminate separation)

angle



Square Thermal (Open Corners)

s_tho<od>x<id>x<angle>x<num_spokes>x<gap>

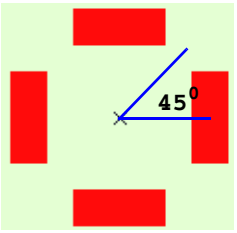


num_spokes = 4

od - outer diameter
id - inner diameter
angle - gap angle from 0^0
num_spokes - number of spokes
gap - size of spoke gap

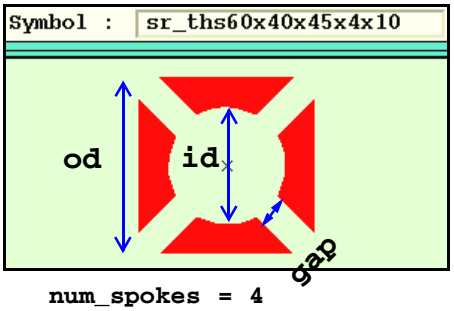
Specification of **od** and **id** determine the air gap (size of laminate separation)

angle



Square-Round Thermal

sr_ths<od>x<id>x<angle>x<num_spokes>x<gap>

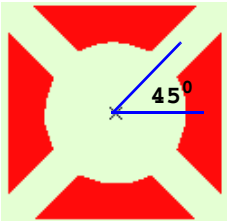


num_spokes = 4

od - outer diameter
id - inner diameter
angle - gap angle from 0^0
num_spokes - number of spokes
gap - size of spoke gap

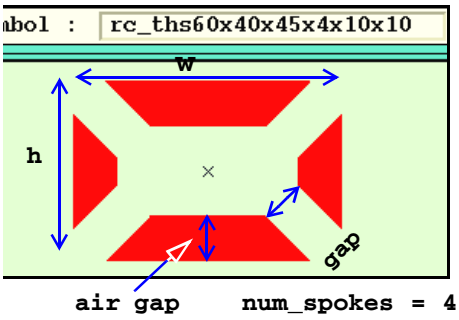
Specification of **od** and **id** determine the air gap (size of laminate separation)

angle



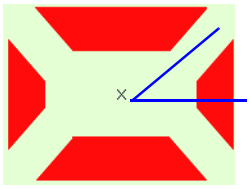
Rectangular Thermal

rc_ths<w>x<h>x<angle>x<num_spokes>x<gap>x<air_gap>



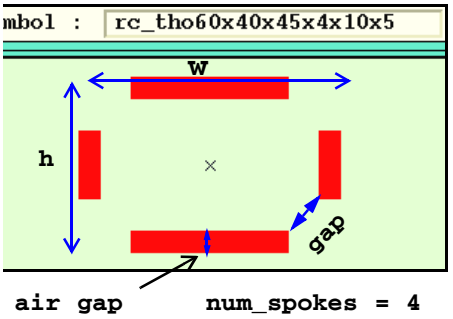
w - rectangle width
h - rectangle height
angle - gap angle from 0^0
num_spokes - number of spokes
gap - size of spoke gap
air_gap - size of laminate separation

angle



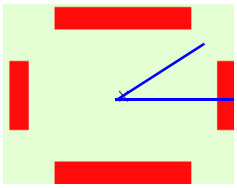
Rectangular Thermal (Open Corners)

rc_tho<w>x<h>x<angle>x<num_spokes>x<gap>x<air_gap>



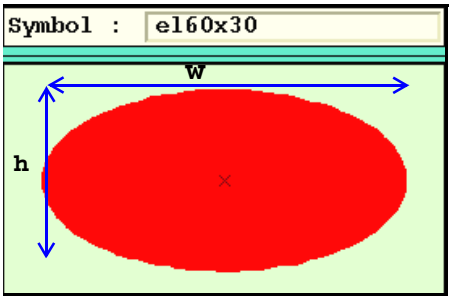
w - rectangle width
h - rectangle height
angle - gap angle from 0^0
num_spokes - number of spokes
gap - size of spoke gap
air gap - size of laminate separation

angle



Ellipse

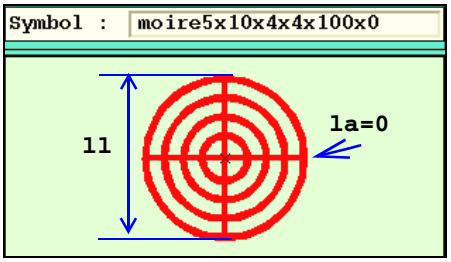
el<w>x<h>



w - width
h - height

Moire

moire<rw>x<rg>x<nr>x<lw>x<ll>x<la>

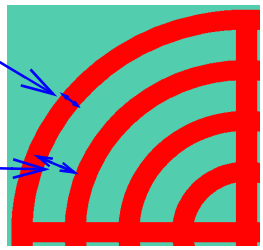


number of rings (nr)=4

rw - ring width
rg - ring gap
nr - number of rings
lw - line width
ll - line length
la - line angle

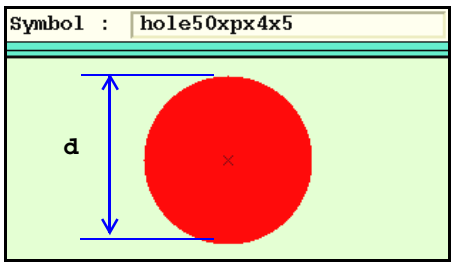
ring width (rw)

ring gap (rg) from center of ring to edge of adjacent ring



Hole

hole<d>x<p>x<tp>x<tm>



d - hole diameter
p - plating status (p(lated), n(on-plated) or v(ia))
tp - + tolerance
tm - - tolerance
This symbol is specifically intended for wheels created by the Wheel Template Editor for drill files.

Null

null<ext>

ext - extension number
This symbol is empty and used as a place holder for non-graphic features.

Rotated Standard Symbols

To create symbols that are rotated at angles that are not in the standard 90 degree increments, a symbol is required to be created for each angle. For example, **rect25x50_315** is a standard rectangle rotated around its center to 315 degrees. These symbols are created automatically when specified by name when adding a pad, for example. They can also be resized as needed.

A corresponding feature file containing a single contour representing the symbol must be placed in:

<job_name>/symbols/<symbol_name>/features

Note Rotation is clockwise.

Chapter 10 *System Attributes*

The following is a combined list of the system attributes currently used by the Genesis, Enterprise and Trilogy programs.

Attribute	Type	Description
`.ar_pad_drill_bottom_max`	Float	(-10 to 1000). Assigned to a drill to define the maximum annular ring size between the drill and the copper of the bottom layer of the drill span.
`.ar_pad_drill_bottom_min`	Float	(-10 to 1000). Assigned to a drill to define the minimum annular ring size between the drill and the copper of the bottom layer of the drill span.
`.ar_pad_drill_inner_max`	Float	(-10 to 1000). Assigned to a drill to define the maximum annular ring size between the drill and the copper of an inner layer in the drill span.
`.ar_pad_drill_inner_min`	Float	(-10 to 1000). Assigned to a drill to define the minimum annular ring size between the drill and the copper of an inner layer in the drill span.
`.ar_pad_drill_top_max`	Float	(-10 to 1000). Assigned to a drill to define the maximum annular ring size between the drill and the copper of the top layer of the drill span.
`.ar_pad_drill_top_min`	Float	(-10 to 1000). Assigned to a drill to define the minimum annular ring size between the drill and the copper of the top layer of the drill span.
`.ar_sm_drill_bottom_max`	Float	(-10 to 1000). Assigned to a drill piercing the bottom layer, to define the maximum annular ring size between the drill and the soldermask on the bottom layer.
`.ar_sm_drill_bottom_min`	Float	(-10 to 1000) Assigned to a drill piercing the bottom layer, to define the minimum annular ring size between the drill and the soldermask on the bottom layer.
`.ar_sm_drill_top_max`	Float	(-10 to 1000). Assigned to a drill piercing the top layer, to define the maximum annular ring size between the drill and the soldermask on the top layer.
`.ar_sm_drill_top_min`	Float	(-10 to 1000). Assigned to a drill piercing the top layer, to define the minimum annular ring size between the drill and the soldermask on the top layer.
`.ar_sm_pad_bottom_max`	Float	(-10 to 1000). Assigned to a drill piercing the bottom layer, to define the maximum annular ring size between the drilled pad of the bottom layer and the soldermask above.

Attribute	Type	Description
.ar_sm_pad_bottom_min	Float	(-10 to 1000). Assigned to a drill piercing the bottom layer, to define the minimum annular ring size between the drilled pad of the bottom layer and the soldermask above.
.ar_sm_pad_top_max	Float	(-10 to 1000). Assigned to a drill piercing the top layer, to define the maximum annular ring size between the drilled pad of the top layer and the soldermask above.
.ar_sm_pad_top_min	Float	(-10 to 1000). Assigned to a drill piercing the top layer, to define the minimum annular ring size between the drilled pad of the top layer and the solder mask above.
.area_name	Text	[0 to 64). Assigned to surface features which are drawn in a process map layer. A process map layer is used in assembly analysis for determining the process type used in the location a measurement is found.
.array_with_rotation	Boolean	If TRUE, this step is a multi-panel array, with the same panel possibly appearing in 180-degree rotation to itself
.assembly_proc_bottom	Text	[0 to 20]. Default assembly process for the bottom side, to be used when there is no specific area defined in the process map layer (or no process map layer at all)
.assembly_proc_top	Text	[0 to 20]. Default assembly process for the top side, to be used when there is no specific area defined in the process map layer (or no process map layer at all).
.bit	Text	(0 - 64). Contains the drill designator which is set to each tool in the Drill Tools Manager.
.board_thickness	Float	[0.0 to 10.0). Total thickness of the board
.break_away	Boolean	Assigned to a symbol representing a break-away to be inserted into any line or arc of the rout path. When adding a **break_away** symbol thru dimensions, it automatically adjusts to the line or arc angle, breaks that feature (in the breaking points defined in that symbol with the **.brk_pnt** attribute), and adds all the necessary connections and dimensions.
.brk_point	Boolean	Assigned to a pad or a dpoint in a break-away symbol (that was given the attribute **.break_away**). When adding the break-away to the line/arc in the layer, thru dimensions, the line/arc is broken at the connection point with the dpoint that has the **.brk_point** attribute. In each break-away symbol there should be two points with this attribute.
.cad_part_override	Boolean	Assigns component properties in accordance with data received from the **ASSY_PN_OVERRIDE** property.
.canned_text	Boolean	Indicates that a text is drilled (applies to features).
.cdr14_stages	Text	(0 - 400). Assigned to alignment target features, and describes the work stage(s) for which the target was set.
.cdr14_zone_type	Text	(0 - 30). Assigned to features representing exclusion zones, describing zone type as set by the operator.
.cdr_val	Integer	(-1 to 100000)
.center_fiducial	Boolean	Specifies component is expected to have a fiducial at its center.

Attribute	Type	Description
.color	Text	(00 to 99 for each color). Can be attached to any feature or component to define the color to be used in plotting a layer in HPGL-1 or 2. The format is **rrggbb** (where **r**=red, **g**=green, **b**=blue). **White - .color = "999999"** **Black - .color = "000000"** **Red - .color = "990000"** **Green - .color = "009900"** **Yellow - .color = "009999"** **Blue - .color = "000099"** **Magenta - .color = "990099"** **Cyan - .color = "999900"**
.comment	Text	(0 to 500). Used for general textual comments.
.comp	Option	(none; right; left). For a chained feature, this attribute sets the offset of the cutting tool from the rout path. Three options: - None - in center of the rout path - Left - to the left of the rout path in the direction of cutting - Right - to the right of the path
.comp_height	Float	(0.0 to 10.0). Stores the height of the component above the board surface (in inches).
.comp_htol_minus	Float	(0.0 - 10.0). Contains the minus tolerance (in inches) for component height, used for calculation of plug-in boards.
.comp_htol_plus	Float	(0.0 - 10.0). Contains the plus tolerance (in inches) for component height, used for calculation of plug-in boards.
.comp_ign_spacing	Boolean	This attribute, when set, disables spacing checks on a component during assembly analysis. It is used for printed components which have no actual body
.comp_ignore		Determines whether the component is to be ignored when calculating statistics, or during certain operations, such as Analysis.
.comp_mount_type	Option	(Other; SMT; THMT). Indicates whether the component is a surface mount, through-hole mount, press-fit or other. (SMT;THMT;PRESSFIT)
.comp_polarity	Boolean	Assigned to components when packages are imported from the VPL (Valor Parts Library) with the value of: - POLARIZED, has a specific pin designated as pin #1. - NON_POLARIZED has no specific pin #1. A component without this attribute means that its package was not imported from the VPL.

Attribute	Type	Description
.comp_type	Option	This attribute is very important for determining dynamic categories during assembly analysis. It represents the type of the component. When the user defines a user attribute: **_comp_type**, it shadows this system attribute. **Important**: Do not use the underscore "_" character in the Type values of this attribute. Options are listed on the right: - axial - bga - cbga - cob - dip - discrete - discrete402 - abel - pga - pihconn - pihmisc - plcc - pqfp - printed - qfp - radial - sip - smtconn - smtmisc - socket - soic - soj - sop - sot - tab - tqfp - tsoic - tsop
.comp_weight	Float	(0.0 - 1000.0). Stores the weight of the component (in ounces) for the purpose of the total weight calculation.
.critical_net	Boolean	Specifies critical nets.
.critical_tp	Boolean	Assigned to the mid-point of a netlist to force it to become a testpoint (it will not be removed by the Netlist Optimizer). If both **.non_tp** and **.critical_tp** are assigned to the same point, **.critical_tp** takes precedence and the mid point is tested. In case of a drilled feature the attribute must be added to the drill hole.
.cu_base	Boolean	This attribute indicates to an analysis action (Signal Layer Checks or Power & Ground Checks) that the specific via layer is built in such a way that it necessitates a copper pad on each layer of the stackup, since the vias are drilled and filled (rather than plated), and the pads are an essential element in ensuring connectivity.
.customer	Text	(0 to 100). This attribute is used for information purposes. It is used specifically in the input process when processing the **lyr_rule** file.
.cut_line	Integer	Assigned to lines added in the creation of film layers by the film optimization algorithm. The attribute is given to three kinds of lines: - frame of the film - cutting lines inside the film - frame of each layer inside the film.
.depth	Float	(1.0 - 1000.0). Depth of drill layer in mils (applies to layers)
.diff_pair	Text	(0 - 64). Differential pair name associating two nets that must be routed together.
.dpair_gap	Float	(0.0 - 10.0). Spacing gap value specifying the spacing between differential pair nets.
.drc_assembly_lyrs	Option	(Top; Bottom; Both). In Component Analysis, specifies whether the keepout/keepin area applies to Top, Bottom, or Both component layers. In Testpoint Analysis, as above to outer layers.
.drc_board	Boolean	Assigned to a DRC area defined for the whole board.
.drc_comp_height	Boolean	Assigns component height restriction to a keepin/keepout area.

Attribute	Type	Description
.drc_comp_height_lyr	Text	(0 to 80). Stores name of document layer in which all component height restriction keepin/keepout areas are stored.
.drc_comp_keepin	Boolean	Defines an area as the board's component placement keepin boundary.
.drc_comp_keepin_lyr	Text	(0 to 80). Stores name of document layer in which all component keepin areas are stored.
.drc_comp_keepout	Boolean	Defines an area as the board's component placement keepout boundary.
.drc_comp_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all component keepout areas are stored.
.drc_etch_lyrs	Text	(0 to 1000). Value = layer names separated by semi-colons (;). User-defined attribute for user to specify name of layers in which to activate keepin/keepout areas. For example, when **.drc_etch_lyrs = pg1;pg2**, this enables you to select/ highlight (on the relevant document layer) keepin/keepout areas that are active in layers **pg1**, **pg2**. The attributes: **.drc_etch_lyrs_bit** and **.drc_etch_lyrs** must both specify the same layers. If there is a discrepancy between the two, then **.drc_etch_lyrs_bit** is the determining attribute.
.drc_etch_lyrs_all	Boolean	Defines a keepin/keepout area as effective on all layers.
.drc_etch_lyrs_bit	Text	(0 to 64). Values = string consisting of '0' and '1' characters. Allows the keepin/keepout area to apply only to specified board layers. The attribute's length is equal to the number of board layers. 0 = ignore layer, 1 = activate areas in that layer
.drc_max_height	Float	(0.0 to 10.0). Stores the maximum height of components to be allowed in a height restriction area (area with .**drc_comp_height** attribute).
.drc_min_height	Float	(0.0 to 10.0). Stores the minimum height of components to be allowed in a height restriction area (area with .**drc_comp_height** attribute).
.drc_pad_keepout	Boolean	Specifies area to be used as pads keepout boundary.
.drc_pad_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all pad keepout areas are stored. Default as defined in the **drc_pad_keepout** configuration parameter.
.drc_plane_keepout	Boolean	Specifies area to be used as planes keepout boundary
.drc_plane_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all plane keepout areas are stored.
.drc_ref_des	Text	(0 to 100). Assigned to DRC areas defined for components.
.drc_route_keepin	Boolean	Specifies areas to be used as the rout keepin boundary(rout = lines, arcs, vias, pads and surfaces on signal and/or power&ground layers).
.drc_route_keepin_lyr	Text	(0 to 80). Stores name of document layer in which all rout keepin areas are stored.

Attribute	Type	Description
.drc_route_keepout	Boolean	Specifies areas to be used as the rout keepout boundary (rout = lines, arcs, vias, pads and surfaces on signal and/or power&ground layers).
.drc_route_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all rout keepout areas are stored.
.drc_tp_keepin	Boolean	Defines areas to be used as testpoint keepin area boundaries.
.drc_tp_keepin_lyr	Text	(0 to 80). Stores name of document layer in which all testpoint keepin areas are stored.
.drc_tp_keepout	Boolean	Specifies areas to be used as the testpoint keepout boundary.
.drc_tp_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all testpoint keepout areas are stored.
.drc_trace_keepout	Boolean	Defines areas to be used as trace keepout boundaries (traces = lines and arcs on signal and/or power&ground layers).
.drc_trace_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all traces keepout areas are stored.
.drc_via_keepout	Boolean	Defines areas to be used as vias keepout boundaries.
.drc_via_keepout_lyr	Text	(0 to 80). Stores name of document layer in which all vias keepout areas are stored.
.drill	Option	(plated; non_plated; via). Assigned to hole features in drill layers. It defines the type of the drill and is used extensively during fabrication analysis.
.drill_flag	Integer	(0 to 100000). Used by the Auto Drill Manager. It is an integer feature attribute that should be used on the drill layer. When the Auto Drill Manager package creates the NC Drills table it separates the different drills based on several values: size, drill type and also the value of this attribute. This is useful in cases where specific drills need to be treated in a specific way.
.drill_noopt	Boolean	Used by the 'Auto Drill Manager'. Feature attribute that is used on the drill layers. Setting a group of drills with this value will force the drill optimizer to keep the order within that group. This is important for preventing the drill path to pass through mechanical pins.
.drill_sr_zero	Option	(1; 2; 3). Used in the Auto Drill Manager to be assigned to a single drill feature in the PCB step. If a single feature in a step is assigned, it is used for setting the 'step & repeat zero offset' of that step. That is, that feature will receive the coordinates - (0,0) in the step & repeat block, and all other coordinates will be relative to it. In order for this attribute to be used, other configuration parameters of the package should be set.
.drill_stage	Option	(1;2;3). Used in the Auto Drill Manager on the drill layer. This attribute receives three values - '1', '2', and '3', specifying the drill stage of that specific drill hole/slot.
.dxf_dimension	Boolean	Assigned during DXF file input to mark its features as part of a DXF dimension entity.
.eclass_impedance	Float	(0.0 to 100.0). Electrical class rule
.eclass_max_stub_length	Float	(0.0 to 100.0). Electrical class rule - high limit of the stub length.

Attribute	Type	Description
.eclass_max_via_count	Integer	(0 to 1000). Maximal number of vias on the nets.
.eclass_min_stub_length	Float	(0.0 to 100.0). Electrical class rule - low limit of the stub length.
.eclass_parallel_dist	Float	(0.0 to 100.0). Electrical class rule - defines the separation distance within which two traces are considered parallel.
.eclass_parallel_max_length	Float	(0.0 to 100.0). Electrical class rule.
.eclass_parallel_min_jog	Float	(0.0 to 100.0). Electrical class rule - defines the separation that between parallel traces that must be maintained before the separation is acknowledged as a break in parallelism.
.eclass_rise_time	Float	(0.0 to 100.0). Electrical class rule specifying the interval of a rising signal transition (low to high)
.eclass_voltage_swing	Float	(0.0 to 100.0). Electrical class rule
.eda_dimension_id	Integer	Assigns system-generated ID to dimensions
.eda_layers	Text	(0 to 1000). Contains the EDA system layer names which compose a physical layer. It is loaded during the direct EDA translation and is used for graphic synchronization with the EDA system.
.electrical_class	Text	(0 to 64). Electrical class name associating a net with a set of electrical call rules. Electrical class rules include physical and electrical limitations required to assure and analyze the signal integrity of a high speed net.
.entity_version	Integer	(0 to 2147418112). Counts the number of changes made in an entity (applies to steps and symbols). **Note** - Do not modify!
.et_adjacency	Float	(1.0 to 1000.0). A distance value (per layer) to use in netlist adjacency calculation for moving probe testers (currently BSL and PROBOT).
.et_align	Boolean	Determines that a feature will be used as an alignment target for PROBOT output
.etch_comp_addition	Boolean	
.etm_adapter_h	Integer	(0.000001-5000). Adapter Height in Mils.
.etm_constant_drill_usage	Option	(plate; cs_board; cs_grid; test).
.etm_height	Float	For the Job to Adapter option. Defines the height of the plate in the adapter represented by the given layer.
.etm_mirror	Boolean	For the Job to Adapter option. Updates the mirror of the drill output transformation for the required plate.
.etm_pin_name	Text	(0-64). ETM pin name.
.etm_pin_style	Option	(Regular, Mania). ETM Pin Guiding Style.
.etm_prim_sink_h	Float	(0.0 to 1000.0). For the Job to Adapter option. Defines the depth of the countersink from the board side of the plate for the required plate.
.etm_repair_fmt	Option	ETM Repair file format. Options: None, EPC
.etm_rotate	Option	(0; 90; 180; 270). For the Job to Adapter option. Defines the rotation of the drill output transformation for the given plate definition.

Attribute	Type	Description
.etm_shift_x	Float	(-100000.0 to 100000.0). For the Job to Adapter option in the ETM. Defines the **x** offset of the drill output trnasformation for the given plate (represented by the layer to which it is assigned).
.etm_shift_y	Float	(-100000.0 to 100000.0). For the Job to Adapter option in the ETM. Defines the **y** offset of the drill output trnasformation for the given plate (represented by the layer to which it is assigned).
.etm_step_x	Float	(0.0 to 1000.0). For the Job to Adapter option in the ETM. Defines the step of the grid being defined on the x axis.
.etm_step_y	Float	(0.0 to 1000.0). For the Job to Adapter option in the ETM. Defines the step of the grid being defined on the y axis.
.etm_tester	Text	(0-64). Options Mania; Everett Charles; Circuitline; Luther; Maelzer; Probot; BSL; IntegriTest; MicroCraft; ATG. ETM tester name.
.etm_thickness	Float	(0.0 to 1000.0). For the Job to Adapter option in the ETM. Specifies the thickness of the plate being defined.
.extended	Integer	(-1 to 100000). Assigned to construction features (lines and pads) added to assist in the generation of a rout path. These features have zero width and are not output to the rout machine as regular features. They are used, for example, as source elements from which to create actual features by dimensions. If the attribute value is not zero then the feature is an extended feature and the decimal value is its serial value in the layer (to be referenced in dimension creation).
.fab_drc	Text	(0 to 20). Stores the default DRC area name. This name is applied when no specific area is defined in the DRC map layer, or no such map layer at all. If the value of this attribute is not defined, then the default DRC name is applied from the configuration variable **default_fab_drc.**
.feed	Integer	(0 to 100000). For a chained feature, this attribute sets the table feed rate when routing.
.fiducial_name	Text	(0 to 64). This attribute is used for **etec** output format. A pad that was given a fiducial name is used for registration between layers.
.fill_dx	Float	(0.000001 to 50.0). This attribute is used as the default horizontal distance between symbols when the symbol is used for pattern filling.
.fill_dy	Float	(0.000001 to 50.0). This attribute is used as the default vertical distance between symbols when the symbol is used for pattern filling
.flipped_of	Text	This attribute defines a STEP as a flipped step. When attached to a LAYER, it indicates that the layer was created as a result of (layer) flipping. The attribute value is the name of the original (unflipped) layer. This is done in order to keep the elements of the original layer.

Attribute	Type	Description
.flipped_out_of_date	Boolean	**No** (default) = indicates that the flipped step is an accurate copy of the original step. **Yes** = indicates that the flipped step is no longer an accurate copy of the original step. One or the other has changed since the first flipping operation that created the step.
.foot_down	Text	Attached to feature it causes a **foot_down_cmd** to be generated by the Auto Rout Manager in the rout file just before the feature. Used only for Excellon files (ignored for other formats).
.fs_direction_bottom	Option	(Left2Right; Right2Left; Top2Bottom; Bottom2Top). This attribute is used for the thieving pad check in assembly analysis. It determines the flow direction for the bottom layer. Thieving pad check is required for some components during the flow solder process
.fs_direction_top	Option	(Left2Right; Right2Left; Top2Bottom; Bottom2Top) This attribute is used for the thieving pad check in assembly analysis. It determines the flow direction for the top layer.Thieving pad check is required for some components during the flow solder process.
.gencad_device_ntol	Float	This is a real value expressing the percent of the value to use as a tolerance (negative tolerance). This is used for all devices: Range of characters: all floating point numbers.
.gencad_device_ptol	Float	This is a real value expressing the percent of the value to use as a tolerance (positive tolerance). This is used for all devices:
.gencad_device_style	Text	(1-24). This attribute is an enhancement of **.gencad_device_type** and is used to store the style of the component as defined in GenCAD (such as, NPN, PNP, NFET, PFET, NJFET, PJFET, TTL, CMOS and ECL)
.gencad_device_type	Text	(0-24). Stores the type of the component as defined in GenCAD (such as, RES, VRES, DIODE, ZENER, LOGIC, SWITCH, CONN, etc.).
.gencad_device_value	Text	(0-24). Stores the electrical value of a component.
.geometry	Text	(0 to 100). Contains the name of the padstack which created this feature. It is loaded during direct EDA translation. For layers which are created from component layers during the 'Draw to Layer' operation, the attribute will contain (for centroid pads) useful information on the component, package and part name.
.global_camtek_aoiset	Text	(0-80). Contains the name of the AOIset to be assigned to each layer upon layer selection in the CAMTEK AOI Interface. Once a name is defined, the AOIset field in the CAMTEK popup will be filled with this name and a new AOIset created in the layer (if already exists, the AOIset will become the current set). The value in this attribute overrides the value defined in the configuration parameter **camtek_def_aoiset**, but if no value is specified in this attribute, the **camtek_def_aoiset** value will apply.
.gold_plating	Boolean	This attribute should be attached (manually) to features which are a part of a gold plated connector. It is used during auto-panelization to orient the gold plated area toward the extreme side of the panel.

Attribute	Type	Description
.guard_comp	Boolean	Assigned to a component that “guards” other components. If TRUE, this component is considered a "guard component" (that is, not likely to be knocked off the board accidentally. To be used in future actions.)
.hatch	Boolean	Assigned to hatched planes [filled with lines (hatches) or cross lines (cross-hatch) instead of solid copper]. The lines which make up the border and fill the surface are hatches.
.hatch_border	Boolean	The lines making up the border of a surface.
.hp3070_comment	Text	Allows the contents of the attribute field to be appended to a component record (preceded by a "**!**"). For example: **C1 PN"11_215705" "11_215705 POLCAP_10UF,20%,10V TOP"; !** *comment*
.hp3070_common_pin	Text	(0-16). For the device SWITCH this is used to designate the COMMON pin.
.hp3070_contact_pin	Text	(0-16). For the device SWITCH this is used to designate the CONTACT pin.
.hp3070_device	Text	(0 -16). The device of the component, one of the following: All other components will be categorized as Undefined. - CAPACITOR - CONNECTOR - DIODE - FET - FUSE - INDUCTOR - JUMPER PIN - LIBRARY - POTENTIOMETER - RESISTOR - SWITCH - TRANSISTOR - ZENER
.hp3070_fail_msg	Text	(0-64). Specifies the failure message associated with the component. This applies to all device types. In output of HP3070 formats, the text field (within quotes) consisting of the Part number and this error message will be truncated to 40 characters.
.hp3070_hi_value	Float	(0-100000.0). Specifies the upper test limit of the device. Its specific meaning is dependent on the device type. - For DIODE: Upper test limit, in volts for the diode's forward bias voltage. - For FET: The high resistance limit in ohms. - For TRANSISTOR: The high limit for the transistor beta.
.hp3070_lo_value	Float	(0-100000.0). Specifies the lower test limit of the device. Its specific meaning is dependent on the device type. - For DIODE: Lower test limit, in volts, for the diode's forward bias voltage. - For FET: The low resistance limit in ohms. - For TRANSISTOR: The low limit for the transistor beta.
.hp3070_probe_access	Text	(0-16). Specifies the probe access for the component and toeprint. This value will be applied to ALL the pins of the component. Possible values are: PREFERRED, NO_PROBE, or TOP. If toeprints are assigned this attribute, their settings override the component setting.

Attribute	Type	Description
.hp3070_seriesr	Float	(0-100000.06). For INDUCTOR devices this is used to specify the series resistance (in Ohms).
.hp3070_test	Text	Determines that a component be tested. This attribute applies to all device types. Devices of type CONNECTOR must be NT (Not Tested).
.hp3070_tol_neg	**Float**	**(**0-100**).** This is a real value expressing the percent of the value to use as a tolerance (negative tolerance). This is used for devices: - CAPACITOR - INDUCTOR - POTENTIOMETER - RESISTOR - ZENER
.hp3070_tol_pos	**Float**	**(**0-100**).** This is a real value expressing the percent of the value to use as a tolerance (positive tolerance). This is used for the devices: - CAPACITOR - INDUCTOR - POTENTIOMETER - RESISTOR - ZENER
.hp3070_type	Text	The type of device:. For CAPACITOR: - F = Capacitor Value is Fixed. - V = Capacitor Value is Variable. For FET: - N = N-Channel Field Effect Transistor - P = P-Channel Field Effect Transistor For INDUCTOR: - F = Inductor value is Fixed - V = Inductor value is Variable For JUMPER: - O or OPEN = Jumper is Open - C or CLOSED = Jumper is Closed For RESISTOR: - F = Resistor value is Fixed - V = Resistor value is Variable For TRANSISTOR: - N = Transistor is an NPN - P = Transistor is a PNP Range of characters: 0-8
.hp3070_value	Text	(0-16). The value of the component. The meaning varies depending on the component device. For CAPACITOR it is used for capacitance (in Farads). For INDUCTOR it is the inductance (in Henries). For PIN LIBRARY it is used for the PN (Part Name). For the devices POTENTIOMETER and RESISTOR, it is used for the device's resistance. For the ZENER device it specifies the breakdown voltage (in Volts).
.image_dx **.image_dy**	Float	(-1.0 to 50.0). These values are set when inputting Image files into the system. They contain the datum point of an Image special symbol entity used to set the datum when performing output back into Image format. These values should not be changed by the user as this can cause data corruption.

Attribute	Type	Description
.imp_line	Boolean	Assigned to lines which are impedance-controlled. When set, it prevents the lines from being rerouted or thinned during signal layer optimization.
.ind_orient_req	Boolean	Indicates that the component requires silkscreen orientation indication. (To be used in future actions.)
.inp_file	Text	(0 to 480). Contains the name of the file (Gerber, Drill) from which the data was input into the layer.
.is_burried	Boolean	Assigned to buried components specifically input from CADIF files in order to mark them as buried. This attribute, although specifically designed for CADIF files, can be used in any other function or script. Note that the attribute name is misspelled, but that is its name.
.is_capped	Boolean	Used on via pads on top & bottom signal layers to indicate that the via is capped on this side.
.is_shadowed	Boolean	Components with this attribute are considered for the Shadowing categories, as the shadowed component.
.label_clearance	Boolean	Assigned to components which are not allowed to be too close to a glued label (e.g. fine pitch SOIC components). During the component analysis, these components are checked vs. the label components.
.layer_dielectric	Float	(0.0001 to 0.1 inch). Specifies the dielectric thickness below a layer.
.layer_hdi_type	Option	(Buildup; Core). Distinguishes buildup layers from core layers in HDI jobs. Some HDI categories are relevant to buildup or core layers but not to both. Therefore, it is important to set this value appropriately.
.local_fiducial_dist	Float	(0.0 to 100.0). Defines the allowed distance (in inches) of fiducials from the outline of the components which require local fiducials (See **.num_local_fiducial**). If set to 0, the fiducials must be included INSIDE the outline.
.lpol_done	Boolean	Indicates to the output that polarity sort according to a format has already been done during film optimization.
.lpol_surf	Boolean	Indicates surface modified by layer polarity reduction algorithm.
.machine_pkg	Text	(0-100). Assigned to a component to indicate the name of a corresponding package in the assembly machine libraries.
.mount_hole	Boolean	Used on drill features to indicate that they are mounting holes.
.n_electric	Boolean	Assigned to a feature, defines it as non-electric (it is not considered for the current netlist for the step).
.needs_guarding	Boolean	**TRUE** - this component needs to be protected by guard components (see **.guard_comp**) else it is likely to be knocked off the board accidentally.
.net_length_max	Float	(0.0 to 100.0). High limit of net length
.net_length_min	Float	(0.0 to 100.0). Low limit of net length

Attribute	Type	Description
.net_point	Boolean	When assigned to a pad in an inner layer, defines the pad as an internal test point.
.net_type	Text	(0 to 64). A name for the type of net. The **.net_type** attribute can reference the set of routing rules for a net.
.nfp	Boolean	Indicates that a pad is not functional (applies to features).
.no_fiducial_check	Boolean	Components with this attribute are not checked for the "Component Covers Fiducial" category, or for any of the categories under the Coverage test.
.no_hole_under	Boolean	If TRUE, no drill holes are allowed under this component.
.no_pop	Boolean	A RefDes with the attribute **.no_pop** (non populated) declares a component as being not populated for the current version of the BOM. When attributed as **.no_pop** (Yes), even though the component is defined in the CAD data it will not be placed during the assembly process.
.no_text_under	Boolean	Assigned to a component, does not allow silk screen text to be placed under the component outline. Printed components (e.g. edge connectors) may not have this attribute.
.no_tp_under	Boolean	Assigned to a component, does not allow testpoints to be placed under the component outline. Printed components (e.g. edge connectors) may not have this attribute.
.no_trace_under	Boolean	TRUE - traces are NOT allowed under the component except for those that touch the component's toeprint pads and exit the component on that toeprint's side.
.no_uncap_via_under	Boolean	TRUE - uncapped vias are NOT allowed under this component.
.nomenclature	Boolean	Defines a feature as a nomenclature (legend) feature. This attribute affects the fabrication analysis by directing spacing checks between such features into a new category (Text to text).
.non_tp	Boolean	Assigned to a feature causes it NOT to be considered as a net testpoint. It is used for connectivity calculation but is not used as a test point (bare board testing).
.notest_req	Boolean	Any pad assigned with this attribute will not be tested. If it is tested by other means, drop back will be performed.
.num_local_fiducials	Integer	(0 to 20). Defines how many local fiducials are expected to be inside or near a component. This is checked during Fiducial Analysis.
.orig_surf	Integer	(0-1000000). Identifies original surface which will be rebuilt.
.otherside_keepout	Option	(full_area; pins_only). Defines for components whether the other side of the board may also contain components in the same area.
.out_angle	Option	(0.0; 90.0; 180.0; 270.0). Layer entity attributes with default values that are used by the output translator. These values populate the output screen when selecting the 'step' to be translated.

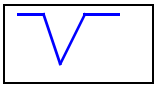
Attribute	Type	Description
.out_break	Boolean	Feature and symbol attribute. When assigned to a specific feature using a special symbol, the feature will be broken into it's primitives in the output translation stage, regardless of the settings of other output parameters. If the attribute is set for a special symbol (entity attribute) then all features that use these symbols will always be broken into primitive features in the output translation stage, regardless of the settings of any other output parameters
.out_comp	Float	(-100.0 to 100.0). Layer entity attributes with default values that are used by the output translator. These values populate the output screen when selecting the 'step' to be translated.
.out_drill_full	Boolean	The STEP entity attribute used by the Auto Drill Manager. This attribute can be used for drilling coupon STEPs that need to be fully drilled before continuing to the next step & repeat entity.
.out_drill_optional	Boolean	Used by the 'Auto Drill Manager'. Both a STEP entity and feature attribute. If the drill feature is set with this attribute it will have the '**/**' command prefix in the final output file, indicating that the drill is optional. If a step entity attribute is set, then all the commands that are part of that step will have the '**/**' command prefixed. Thus, the whole step is optional.
.out_drill_order	Integer	(-10000 to 10000). The STEP entity attribute used by the Auto Drill Manager. The attribute controls the order in which the steps will be drilled. Thus, who is first, second,,,etc. The attribute has the following valid values: 0 - no special order for that step 1 - first 2 - second 3 - and above - order from the beginning -1 - last -2 - one before last -3 - and on (drill order from the end)
.out_mirror	Boolean	Layer entity attributes with default values that are used by the output translator. These values populate the output screen when selecting the 'step' to be translated.
.out_name	Text	(0 to 64). Entity attribute that is used by the Image output translator. If this attribute is not an empty string it will serve as the entity name on the Image system. If it is an empty string the original system entity name will be used. This attribute is important in cases where the Genesis name does not form a legal Image name. If this attribute is not set the Genesis output translator decides about the new name with its own internal algorithm.
.out_polarity	Option	(Positive; Negative). Layer entity attributes with default values that are used by the output translator. These values populate the output screen when selecting the step to be translated.

Attribute	Type	Description
.out_rout_optional	Boolean	Used by the Auto Drill Manager. Both a STEP entity and feature attribute. If drill feature is set with this attribute it will have the '/' command in front of it in the final output file. This means that the drill is optional. If a step entity attribute is set then all the commands that are part of that step will have the '/' command at the beginning. Thus, the whole step is optional.
.out_rout_order	Integer	(-10000 to 10000). STEP entity attribute used by the Auto Drill Manager. The attribute controls the order in which the steps will be drilled. Thus, who is first, second,,,etc. The attribute has the following valid values: 0 — no special order for that step 1 — first 2 — second 3 and above — order from the beginning -1 — last -3 and on — drill order from the end
.out_scale	Boolean	Feature and symbol attribute. In the output translation package there is a special parameter that controls the way features will be scaled. In two of the options the user can specify whether certain features can be scaled or not. This is important in cases where special registration targets would not be scaled together with all the other features. This special output option applies only to features that have this attribute set. In case of a special symbol, the customer can set the attribute, and by this control the scaling of all features that use this symbol.
.out_x_scale	Float	(0.000001 to 5.0). Layer entity attributes with default values that are used by the output translator. These values populate the output screen when selecting the step to be translated.
.out_y_scale	Float	(0.000001 to 5.0). Layer entity attributes with default values that are used by the output translator. These values populate the output screen when selecting the step to be translated.
.output_dcode	Integer	(0-1,000,000). Assigned to features to provide action codes for an assembly machine, such as the GSI Lumonics laser cutter.
.pad_usage	Option	(toeprint;via;g_fiducial;l_fiducial;tooling_hole) This attribute defines the specific usage of a pad. It is loaded during the direct EDA translation and by the attribute derivation script.
.patch	Boolean	Assigned to patches added by the pinhole elimination DFM action.
.pattern_fill	Boolean	Assigned to features which are added during a pattern fill operation, either manually or through the Copper Balance DFM action.
.pilot_hole	Integer	(0 to 100000). Attribute assigned to pads that are pilot holes in a chain (holes that are drilled in each tool down in the chain rout path before routing the chain). Pilot holes are set from the chaining popup. The value of the attribute is the serial chain number to which the pilot hole belongs. When merging or inserting chains, the pilot holes are updated automatically.

Attribute	Type	Description
.pnl_class	Text	(0-64). The value of the attribute is the name of the panel class whose parameters were used by the Automatic Panelization algorithm. Used only when the step is created by the Automatic Panelization Package.
.pnl_pcb	Text	(0-64). The value of the attribute is the name of the panelized PCB whose parameters were used by the Automatic Panelization algorithm. Used only when the step is created by the Automatic Panelization Package.
.pnl_place	Text	(0-64). Applies to STEP and FEATURE. The value of the attribute is the name of the placement rule used when an element was added to the panel overlay. Used only when an element is added to the panel overlay by the Automatic Panelization Package.
.pnl_scheme	Text	(0-64). The value of the attribute is the name of the panelization scheme whose rules were used in creating the panel overlay. Used only when the panel step was created by the Auto Panelization Package.
.primary_side	Option	(Top; Bottom). Indicates the primary side for this job.
.rot_correction	Integer	(0 to 359). Component machine rotation correction to apply.
.rotated_of	Text	(0 to 64). Source step of a rotated step
.rotation_angle	Float	(-360.0 to 360.0). Angle of rotation (in degrees) that this step was rotated (applies to steps)
.rout_chain	Integer	(0 to 100000). Contains the serial number of the chain to which the feature belongs. Features belonging to that chain are rearranged in the features database according to their order inside the chain. Additional attributes that are added to a chained feature: .feed, .speed, .rout_flag, .comp
.rout_flag	Integer	(0 to 100000). For each chained feature this attribute represents a numeric value supplied to a chain to provide data for the automatic process of the Auto Rout Manager.
.shave	Boolean	Assigned to all the shaves (negative merges) that the silk screen optimization adds in merge mode.
.sip	Text	(Detected / Repaired). Indicates whether the SIP (self-intersecting polygon) has been detected or repaired.
.sliver_fill	Boolean	Assigned to all the fills added by the sliver fill DFM actions.
.smd	Boolean	Assigned to outer layer pads which are lands for SMD components. It is set by the 'Set SMD Attribute' Cleanup Action.
.smt_direction_bottom	Option	(Left2Right; Top2Bottom; Right2Left; Bottom2Top). Defines the direction of the SMT process flow on the bottom side.
.smt_direction_top	Option	(Left2Right; Top2Bottom; Right2Left; Bottom2Top). Defines the direction of the SMT process flow on the top side.
.source_llayer	Text	(0-64). This attribute is used by the Enterprise Mentor EDA translator to identify the "Source Logical Layer" of features (traces) appearing on signal or mixed layers. The translator uses this attribute in a filtering stage that addresses pad/signal mapping.

Attribute	Type	Description
.source_name	Text	(0-64). The name of the source step (or symbol) of a flipped step (or symbol).
.speed	Integer	(0 to 100000). For a chained feature this attribute sets the spindle speed (in revolutions per minute) when routing.
.spo_h_fact	Integer	(0.3 <-> 2.0). When **.spo_h_mode** = Factor, **.spo_h_fact** specifies the factor by which paste pad heights are sized relative to their SMD pads. For example, 0.9 means height is 90% of SMD pad.
.spo_h_mode	Integer	(values = Distance, Factor, Value). Defines how heights of paste pads are sized: by distance, factor or value.
.spo_h_val	Integer	(-500 to +500). When **.spo_h_mode** = Distance, **.spo_h_val** is the reduction/expansion of the paste pad width relative to the SMD pad width. For example, **.sp_h_val** = 5.0 mils shrinks paste pad by 5.0 mils (2.5 mils on each side) relative to SMD pad width. Positive number results in smaller paste pad, negative number in larger paste pad. When **.spo_h_mode** = Value, **.spo_h_val** becomes the absolute width of the paste pad (for example, 5.0 mils becomes the actual width of the paste pad).
.spo_move_center	Integer	(-100 to 100). To move the paste pad from the SMD pad center. A positive value will move the paste from the component center out. A negative value will move the paste towards the component center.
.spo_p_fact	Integer	(0.3 <-> 2.0). When **.spo_p_mode** = Factor, **.spo_p_fact** specifies the factor by which paste pad heights are sized relative to their SMD pads. For example, 0.9 means area is 90% of SMD pad.
.spo_p_mode	Integer	(values = Distance, Area). Defines how paste pads for non-standard symbol SMD pads are sized: by distance, or area.
.spo_p_val	Integer	(-500 to +500). When **.spo_p_mode** = Distance, **.spo_p_val** is the reduction/expansion of the paste pad width relative to the SMD pad width. For example, **.sp_p_val** = 5.0 mils shrinks paste pad by 5.0 mils (2.5 mils on each side) relative to SMD pad width. Positive number results in smaller paste pad, negative number in larger paste pad.
.spo_s_fact	Integer	(0.3 <-> 2.0). When **.spo_s_mode** = Factor, **.spo_s_fact** specifies the factor by which paste pad heights are sized relative to their non-standard SMD pads. For example, 0.9 means height is 90% of SMD pad.
.spo_s_mode	Integer	(values = Distance, Factor, Value, Area). Defines how heights of paste pads for symmetric SMD pads are sized: by distance, factor, value, area.
.spo_s_val	Integer	(-500 to +500). When **.spo_s_mode** = Distance, **.spo_s_val** is the reduction/expansion of the paste pad width relative to their non-standard SMD pad width. When **.spo_h_mode** = Value, **.spo_h_val** becomes the absolute size of the paste pad.

Attribute	Type	Description
.spo_shape	Text	Specifies the symbol to be used as the solder paste pad applied to a feature (smd pad) or to the toeprints of a component. The initial orientation of the symbol is also affected by the **.spo_shape_rotate** attribute when defined
.spo_shape_rotate	Float	Specifies the initial rotation of the symbol defined in the **.spo_shape** attribute. Both **.spo_shape_rotate** and **.spo_shape** should appear as a pair at the level at which they are activated (component or feature). This means, for example, that an **.spo_shape_rotate** defined without a corresponding **.spo_shape** in the feature level is ignored.
.spo_shape_stretch	Boolean	Specifies that the symbol defined in **.spo_shape** is to be stretched to fit the copper pad dimensions. The stretch limits are determined by applying the SPO width and height parameters (**pp_w_***, **pp_h_***) or attributes (**.spo_w_***, **.spo_h_***) on the copper pad bounding box. Both **.spo_shape_stretch** and **.spo_shape** should appear as a pair at the level at which they are activated (component or feature). This means, for example, that an **.spo_shape_stretch** defined without a corresponding **.spo_shape** in the feature level is ignored.
.spo_w_fact	Integer	(0.3 <-> 2.0). When **.spo_w_mode** = Factor, **.spo_w_fact** specifies the factor by which paste pad widths are sized relative to their SMD pads. For example, **.spo_w_fact** = 0.9 width of paste pad is 90% of width of SMD pad.
.spo_w_mode	Integer	(Distance; Factor; Value; Area). Defines how widths of paste pads are sized: by distance, factor or value.
.spo_w_val	Integer	(-500 to +500). When **.spo_w_mode** = Distance, **.spo_w_val** is the reduction/expansion of the paste pad width relative to the SMD pad width. For example, **.sp_w_val** = 5.0 mils shrinks paste pad by 5.0 mils (2.5 mils on each side) relative to SMD pad width. Positive number results in smaller paste pad, negative number in larger paste pad. When **.spo_w_mode** = Value, **.spo_w_val** becomes the absolute width of the paste pad (for example, 5.0 mils becomes the actual width of the paste pad).
.sr_pcb	Boolean	(0.0 to 1000.0). Indicates the name of the pcb step placed in the panel by automatic panelization.
.src_orientation	Integer	(-1 to 3). Defines the zero orientation of this component relative to its orientation in the packages database. That is, the orientation of the component on the automated assembly tape, or (for manually inserted components) the orientation in which pin #1 is "in the same position" for all similar components
.string	Text	(0 to 1000). For nomenclature features, the value of this attribute is the original text string which the feature is part of. During EDA input, all occurrences of the asterisk character '*' are replaced by the hyphen character '-' (the reason: when filtering, the asterisk character '*' is used to denote any substring match).
.tear_drop	Boolean	Assigned to features which are added during a tear drop operation, either manually or through the Teardrop Creation DFM action.

Attribute	Type	Description
.test_point	Boolean	Assigned to features which are used for In-Circuit Testing operations. It is loaded during the direct EDA translation and is used during the Testpoint Analysis action.
.testpoint_count	Integer	(0 to 1000). Specifies the number of testpoints expected on this net. If this variable is not defined, or its value is **-1**, the number of expected testpoints in unlimited. A value of **-1** is given to a net that does not require a specific number of test points. When the Num-verify test is run, it ignores such nets (even when the ERF variable **v_testpoint_count_default** is defined).
.thvpad_required	Boolean	Assigned to components which require a thieving pad check during the Padstack Analysis action (e.g. fine pitch SOIC).
.toep_nochk_o_side	Boolean	Assigned to components so that their toeprints on the opposite side will be excluded from the Toeprint to Toeprint category measurement results in Padstack Analysis, and from the Component to Toeprint category in Component Analysis. **Important Note**: This attribute is applied only when the following ERF variables are set: **c2toep_by_comp=1** (in **component.erf**) **toep2toep_by_comp=1** (in **padstack.erf**)
.toep_spacing_req	Float	(1 to 500). Assigned to components for reporting in the Toeprint to Toeprint category in the Signal Layers Check in Analysis. It defines the maximum spacing within which to report pad to pad spacing measurements. Units: inch/mm
.tooling_hole	Boolean	Used on drill features to indicate that they are tooling holes.
.vcut		Assigned to rout features that are cut in a V-shape (such as in the figure). Another machine performs the V-shape cutting. 
.viacap_layer	Option	(Top; Bottom; Both; None). Defines on which layer via capping can occur, if any.
.via_type	Option	(Drilled; Laser; Photo). Assigned to via drills for the classification of various via pad and via drill categories in the HDI analysis.
.wheel_type	Option	(Gerber; Tools). Specifies for a wheel whether it is used for Gerber files translation or for drill file translation.

Appendix A *Frequently Asked Questions*

A.1. Why is the database in ASCII ?

An ASCII database provides the user with numerous advantages:

- It is easy to read and understand
- Translators to and from the database formats are easier to write
- The data is portable between different architectures, independent of byte order, floating point formats, etc.

By compressing the ASCII files using standard compress commands, the size of the data is even smaller than the binary equivalent! This is due to the fact that the compression algorithm is adaptive and work very well when certain strings are repeated.

A.2. When I wish to rotate a feature pad by 90 degrees is the aperture rotated left or right?

Clockwise.

A.3. Regarding donuts, butterflies, thermals, and moires, do any of these symbols have negative components?

Standard symbols are all positive. All holes in symbols are see-thru by definition.

A.4. When the start and end-points of a feature coincide, is this considered a 360-degree arc or a single point? Can I draw an arc with a square symbol?

A 360-degree arc; there are no single point arcs in the ODB++ database. Arcs can be drawn only with a round symbol.

A.5. When I specify an x,y location for text where will the text string be located?

The x,y coordinates will determine the bottom left corner position of the first character of the text string.

A.6. What is the meaning of the optimize field in a netlist file?

It indicates that the net has been optimized by the Netlist Optimizer function and the end-point markers have been removed from mid-points.

A.7. In a netlist file, how is the radius field supposed to be set for drills of 0.002 inches thru non-SMDs?

The radius field will be 0.001 mils in this case.

A.8. In a netlist file, what does the term staggered points mean?

These are points that have been staggered by the staggering algorithm to make them accessible to test probes.

A.9. For rectangular thermals can I define spoke angles at other than multiples of 45 degrees?

Rectangular thermals cannot have spoke angles of 45 degrees, only square/round thermals can have angles that are non-multiples of 45 degrees.

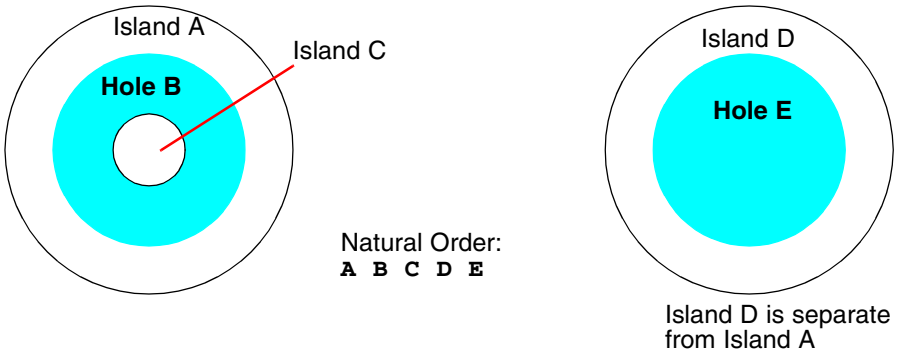
A.10. Can feature files of user-defined symbols contain references to other user-defined symbols?

Yes, they can. But recursion, direct or indirect is not allowed.

A.11. Regarding surfaces, is there a particular order in specifying holes and islands?

The order of containment must be preserved. Islands precede holes that are contained in them. Holes precede islands that are contained in them.

Take, for example, the following containment order:



A.12. Regarding surfaces, does the outermost island come first?

Yes.

A.13. Regarding rounded or chamfered rectangles, how do I specify corners?

Corners must be specified in ascending order, starting from #1 the top-right corner going counter-clockwise (that is, top-left corner=2, bottom-left=3, etc.).

A.14. If I want to offset e a rectangular pad in X or Y, should I a symbol and use the standard valor definitions to create the offset. As an example, suppose we have a rect pad 70x50 with an X offset of 5. The feature file for the symbol I create would contain:

```
#
# Symbol name
#
$0 rect70x50
#
# Pad definition
#
P .005 .0 0 P 0 0
```

Is this correct?

Instead of defining your own symbols with offsets, you should use the standard rect symbol and offset the coordinate that references it (in the layer features file) .

A.15. Why are user-defined symbols not scalable? This means that for every pad size which does not fit in the standard I will have to create a new symbol.

Yes, that is right. User-defined special symbols cannot be scaled as standard symbols can. You need to create a new symbol for each set of parameters (make the name signify the dimensions of the symbol, such as: rect70x50, rect50x30, etc.). See "Symbols" on page 13 for further details.

A.16. Regarding properties (PRP) on components, is there a list of properties that are recognized by the system (such as with system attributes). Where can I find it? The same goes for PRP in the `eda/data` file.

There is no list of predefined properties in the ODB++ database. These are EDA-specific.When we input Mentor data we read all the properties of the components in the data. These properties are shown when displaying a component in the Graphic Station. They can also be used to automatically set an attribute by calling a function that maps properties to attributes.

A.17. When defining a PKG record (using `/steps/step_name/eda/data`), it seems that ODB++ expects closed geometries.Is it critical to have only closed elements?

Yes, you should close all polygons.

A.18. In what order should the matrix layers be in?

The layers should be ordered according to the stackup of the board: i.e. comp_+_top, sigt .. sigb, comp_+_bot, drill, drill_1to5, etc .

A.19. I have problems with the `/steps/step_name/eda/data` in the PKG section?

A PKG record must be followed (the next line) by an outline record. check for some PKG records that have PRP's before the outline and make sure you have an outline defined.

A.20. How are the net_num records numbered?

The **`net_num`** used in the TOP record corresponds to the sequence of the Net records in the **`eda/data`** file. The first Net record is **`net_num 0`**, the second is **`net_num 1`** and so on.

A.21. We have an elaborate tool set to define routing slots and milling contours. How is this data written to ODB++?

Milling (referred to as routing in Fabrication, and not to be confused with routing of traces in design) is handled by defining a 'rout' layer (similar to a drill layer). The features in this layer correspond to the outline of the shapes that need to be cut out. A rout layer is like any other layer, but in order for it to be used during fabrication, should contain only lines, arcs and circular pads (rout machines can also drill).

A.22. What net do I assign to points that have no net defined?

All features which do not have a net defined should be assigned to net **`$NONE$`**which should be defined in the eda/data file. You must add the net and all it's points to the CADnet.

A.23. Is there any restriction on the maximum line length in an ODB++ file? Can comment lines be more than 500 characters?

The restrictions are different for different files. In general 500 characters are the limit, but there are exceptions. Any line over the defined limit will be read with the remainder of the line ignored, so comments can be longer than the limit.