100 Elwood Davis Road ♦ North Syracuse, NY 13212 ♦ USA

Sonnet Project Format
Release 16

Specialists in High-Frequency Electromagnetic Software
(315) 453-3096 Fax: (315) 451-1694 http://www.sonnetsoftware.com

Rev 16.56.1

At Sonnet, we've been developing 3D planar EM software since 1983, and our software has earned a solid reputation as the world's most accurate commercial planar EM analysis package for single and multi-layer planar circuits and antennas.

Sonnet Software Inc., founded by Dr. Rautio, is a private company, dedicated completely to the development of commercial EM software. We take great pride in providing quality technical support for our products--which we believe to be very important for high-end technical software products.

Sonnet is based in Syracuse, NY, USA with representatives across the globe.

Table of Contents

Sonnet Project Format: Release 16

Introduction

This document details the Sonnet project file format. A list of new features for release 16 is followed by a detailed syntax for all possible entries in the file.

A Sonnet project file specifies a circuit geometry in the case of a geometry project and a circuit netlist in the case of a netlist project. This project file contains the specification of the circuit geometry or netlist, the analysis controls, and the analysis output data. What types of analysis data are contained in the project file depends on the types of analyses run on the project.

The structure of the project file consists of the main ".son" file which appears in the highest level directory. Residing in this directory is a folder "sondata" which retains all the response data for all the ".son" files in the parent directory. All of this data is now stored as part of the project. In the directory "sondata" is a directory for each ".son" file, with the basename of the project.

For example, you have a working directory `c:/myfilters`. You use the project editor to create three projects in this directory: steps, filter1 and filter2. These projects would produce the directory structure pictured below:

```
c:\myfilters
        |
        |-------- steps.son
        |-------- filter1.son
        |-------- filter2.son
        |-------- \sondata (Directory)
                       |
                       |-------- \steps  (Directory containing Sonnet data files)
                       |-------- \filter1  (Directory containing Sonnet data files)
                       |-------- \filter2  (Directory containing Sonnet data files)
```

A Sonnet geometry project specifies the circuit geometry to be analyzed by the electromagnetic analysis engine, ***em***. The geometry of the metalization is represented in terms of polygons. Any part of a polygon outside the box is ignored by ***em***. The coordinates of the polygon vertices are called out in terms of actual dimensions using floating point values. The polygons are automatically subsectioned by ***em*** for analysis. Polygons with dimensions smaller than the selected resolution can provide unreliable results.

A Sonnet netlist project specifies a circuit netlist composed of circuit elements defined in the project editor. The netlist is represented in terms of modeled elements, response data file elements, subproject elements and networks.

Both types of projects can be created using Sonnet's project editor or by direct editing in a text editor. The project must have a file name ending with ".son." Other information, including the box dimensions and substrate parameters, are also specified. The experienced user may wish to make minor modifications to a specific geometry or netlist by editing it directly. With this in mind, the file format has been set up to be as forgiving as possible; however, keep in mind that it is still possible to corrupt the file in a manner which will not be detected by the analysis software. For this reason, only experienced users should attempt any modification.

New Keywords in Release 16

New

Added Resistance per Via to via metal loss models: You may now specify your loss for the Volume and Array loss models for via metal by using resistance per via in addition to conductivity, resistivity and sheet resistance at DC. Please see the keyword **MET** on page 17.

Added Bar Mesh Fill for Vias: There is a new meshing fill, Bar, available for vias. Please see the keyword **NUM** on page 43.

Direction for Thick Metal or Rough Metal: You may now choose the direction of both the thick metal and rough metal loss models, either up from the metal level or down from the metal level. For more details, see the keyword **MET** on page 17.

Enhanced Resonance Detection: A new entry has been added to the Control block for the new Enhanced Resonance Detection advanced analysis option. For more details, see the keyword **DET_ABS_RES** on page 55.

Polygon Edge Checking: You may now define the range of edge checking between polygons using metal levels or Technology Layers. For more information, see the keyword **EDGECHECK** on page 52.

Changes

Updated the definition for FTYP and VER keywords.

Project File Syntax

Any line with a first non-space character of “!” unless immediately followed by a “<”, is ignored. Any blank line is also ignored. “&” is used as a continuation character. When it occurs, anything after it in that line is ignored and the next entry line is added at the point at which the “&” was placed. Comments following any complete line of data are allowed. Comments are not allowed in the polygon vertex lists.

In the keywords that follow, only the specified letters (3 or 4) are significant. Additional letters may be used but they do not alter the program’s execution. For example, “VER”, “VERSION” and “VERTRFGH” all have the same effect. There may be no more than 255 characters per line.

The .son file is comprised of the following blocks. All blocks start with a keyword and are ended with the statement END <Keyword>.

HEADER
DIM
GEO (Geometry projects only)
FREQ
CONTROL
OPT
VARSWP
FILEOUT
VAR (Netlist projects only)
CKT (Netlist projects only)
SUBDIV (Geometry projects only)

The blocks and corresponding keywords appear below in the order in which they should appear in the project file. It is especially important to note that index numbers for metal type, dielectric type, polygons and polygon vertices in geometry netlists are implicitly assigned by where the entries appear in a file. The keyword is followed by a brief description with the complete syntax following.

FTYP File type statement identifies the file type: Geometry or Netlist.

Syntax `FTYP [SONPROJ|SONNETPROJ] filever`

FTYP is followed by SONPROJ for a geometry project and followed by SONNETPROJ for a netlist project. The filever is an integer representing the file format version. Sonnet increments this number each time there is a change or new addition to the file format.

VER Project Editor Version Number. The VER line should immediately follow the FTYP line in the file.

Syntax VER version

The version is a character string identifying the version of Sonnet used to create the project; for example, 16.56. This entry is not required.

Header Block

The header block provides license and date information about the project file. **Special Note:** No Comments (!) or continuations (&) are allowed inside the header block since they will be read as standard characters. This is to allow any characters to be used in the ANN lines.

HEADER Beginning of Header Block

Syntax `HEADER`

Indicates the beginning of the header block. All statements following this entry are included in the Header block until you reach the END HEADER statement.

LIC License id number.

Syntax `LIC licenseID`

The License ID is a character string identifying the customer license ID.

DAT Date of last file change.

Syntax `DAT mm/dd/yyyy hh:mm:ss`
The DAT keyword is followed by a character string identifying the date and time the file was last saved.

BUILT_BY_CREATED Origin of project file.

Syntax `BUILT_BY_CREATED source version mm/dd/yyyy hh:ss`
The `BUILT_BY_CREATED` keyword is followed by the source program name, the version of the source program, then the date and time when the file was created. This entry is never updated. If the file source is unknown then “`source`” is set to “unknown” and `version` is set to “unknown version”. The following are possible file sources:

Entry	Program
xgeom	Sonnet Project Editor
sonntgbr	Gerber Translator
ebridge	Agilent ADS Interface
gds	GDSII Translator
dxfgeo	DXF Translator
sonntawr	AWR Microwave Office
sonntcds	Cadence Virtuoso Interface

BUILT_BY_SAVED Origin for the last time the file was saved.

Syntax `BUILT_BY_SAVED source version`

The `BUILT_BY_SAVED` keyword is followed by the source program which last executed a save on the project file. This entry is updated each time the project file is saved. See the `BUILT_BY_CREATED` syntax in the previous entry for a description of the fields.

MDATE Date of last file change with “Medium Importance” changes.

Syntax

```
MDATE mm/dd/yyyy hh:mm:ss
```

The MDATE keyword is followed by a character string identifying the date and time the file was last saved with “Medium Importance” changes. Not required.

HDATE Date of last file change with “High Importance” changes.

Syntax

```
HDATE mm/dd/yyyy hh:mm:ss
```

The HDATE keyword is followed by a character string identifying the date and time the file was last saved with “High Importance” changes. High importance changes are those which cause analysis data to be invalid. Not required.

ANN Comment Statements

Syntax

```
ANN text
```

This keyword is followed by any comments about the file. This is the only location in the header block that special comments are allowed. If you comment is longer than one line, you may use multiple ANN lines.

END End statement

Syntax

```
END HEADER
```

Indicates the end of the header block. Required.

Dimensions Block

The dimensions block provides all the units to be used in the project file. There is an entry line for each type of unit.

DIM Beginning of dimensions block

Syntax

```
DIM
```

Indicates the beginning of the dimensions block. All statements following this entry are included in the dimensions block until you reach the END DIM statement.

CDVY Conductivity Units

Syntax `CDVY <unit>`

The CDVY keyword is followed by a character string identifying the conductivity unit. Choices for <unit> include SM, SCM, MSCM, and USCM. If the default unit is used this entry is optional.

FREQ Frequency Units

Syntax `FREQ <unit>`

The FREQ keyword is followed by a character string identifying the frequency unit. Choices for <unit> include HZ, KHZ, MHZ, GHZ, THZ, and PHZ. The default is GHZ.

RSVY Resistivity Units

Syntax `RSVY <unit>`

The RSVY keyword is followed by a character string identifying the resistivity unit. Choices for <unit> include OHMM and OHCM. The default is OHMCM. If the default unit is used this entry is optional.

SRES Sheet Resistance at DC Units

Syntax `SRES <unit>`

The SRES keyword is followed by a character string identifying the sheet resistance at DC unit. Choices for <unit> include MOSQ and OHSQ. The default is OHSQ. If the default unit is used this entry is optional.

IND Inductance Units

Syntax `IND <unit>`

The IND keyword is followed by a character string identifying the inductance unit. Choices for <unit> include H, MH, UH, NH, PH, and FH. The default is NH.

LEN Length Units.

Syntax `LEN <unit>`

The LEN keyword is followed by a character string identifying the length unit. Choices for <unit> include MIL, UM, MM, CM, IN, M and FT. The default is MIL.

ANG Angle Units.

Syntax `ANG <unit>`

The ANG keyword is followed by a character string identifying the angle unit. Choices for <unit> include only DEG (degrees) at the present time.

CON Conductance Units.

Syntax `CON <unit>`

The CON keyword is followed by a character string identifying the conductance unit. The conductivity units are always set to "/OH."

RES Resistance Units.

Syntax `RES <unit>`

The RES keyword is followed by a character string identifying the resistance unit. Choices for <unit> include WOH, OH, KOH, MOH, GOH and TOH. The default is OH.

END End statement

Syntax `END DIM`

Indicates the end of the dimensions block. Required.

Geometry Block for Geometry Project

The geometry block specifies the circuit geometry in a geometry project.

GEO Beginning of geometry block

Syntax `GEO`

Indicates the beginning of the geometry block. All statements following this entry are included in the geometry block until you reach the END GEO statement.

SYM Enables symmetry for the circuit.

Syntax `SYM`

If the circuit geometry AND excitation are symmetric about the center line parallel to the X axis, a significant reduction in computation time can be realized. To indicate this, include a line with “SYM” as the first non-space characters. In this case, a mirror image of the geometry of the half substrate above the symmetry line is used on both halves of the substrate.

VGMODE Enables auto height vias for the circuit.

Syntax `VGMODE STOP`

This entry only appears if the Auto Height Vias option is on. This option may be selected in the Import Options dialog box during an import or in the Advanced Subsectioning dialog box when editing in the project editor. This setting causes a via to terminate if it encounters planar metal in its path. If the option is not selected, this entry does not appear.

SNPANG Angle for ortho mode.

Syntax `SNPANG angle`

This entry defines the angle used in orthogonal mode. <angle> is the angle in degrees selected which is used to control the creation and movements of objects in your project. Values for <angle> are 90, 45, 30, 22.5 and 5. The angle may be selected in the Snap Setup dialog box (Tools Snap/Ortho Setup) in the project editor. This entry is optional. If it does not appear then the default value of 45 degrees is used.

PSB1 Parallel Subsections

Syntax `PSB1 boxside distance`

This statement defines the parallel subsections in the geometry project. <boxside> is the side of the box to which the parallel subsections are attached. The value is a character string; possible values are “LEFT” “RIGHT” “TOP” “BOTTOM”. <distance> is a floating point number which provides the distance from the box wall to which the parallel subsections extend. There may be up too four PSB1 statements, one for each box wall, in the GEO block.

DRP1 Reference Planes

Syntax

```
DRP1 Position type [LEN|
POLY iPolygon #Points
iVertex]
```

Each side of the Box may have a reference plane offset associated with it. Each DRP1 statement defines one reference plane, so that there may be up to four DRP1 statements in a project file. All orthogonal, shared ports on a given side have the same offset. The Position defines the box wall from which the reference plane extends. Values for Position are LEFT, RIGHT, TOP, and BOTTOM.

If <type> is FIX, then the LEN field is a floating point number specifying the length of the reference plane in the units presently specified in the DIM block. The actual reference plane is moved to the nearest cell location (see BOX).

If <type> is LINK, then the DRP1 entry is following by additional entry lines specifying the polygon vertex to which the reference plane is linked. See the POLY statement syntax below.

Syntax

```
POLY iPolygon #Points
iVertex
```

This statement is used to specify the vertex to which the reference plane is linked. The keyword POLY is followed by an integer which identifies the polygon. This is the file ID, not the position in the GEO block. This is followed by the number of points to be specified. In the case of a linked reference plane this value is always 1.

The next line consists of an integer number which identifies the vertex of the polygon specified in the POLY statement above. The first polygon vertex is number zero.

TMET Top Cover Metal- only one allowed per file.

Syntax `TMET name patternid type value1 value2 ...`

`name` This is the metal name. This name must be in quotes if it contains spaces. Quotes may be used when they are not required.

`patternid` The index of the pattern to be displayed. Starts at 0.

`type` This identifies the type of metal definition. This is one of the following character strings: WGLOAD, FREESPACE, NOR, RES, NAT, SUP, SEN. These keywords and the values associated with them are defined below.

`WGLOAD`
Wave Guide Load. There are no values associated with this type.

`FREESPACE 376.7303136 0 0 0`
Free Space. The values model no top being present on the box.

`NOR conductivity currentratio thickness`
Normal metal. The conductivity is a floating point number in S/m. The `currentratio` is a floating point number which is the ratio of current on the top surface to current on the bottom surface. Thickness is a floating point number for the thickness of the metal.

`RESISTOR rdc`
Resistor. R_{dc}, the DC resistance, is a floating point number in Ohm/sq.

`NAT rdc rrf`
Native. R_{dc}, the DC resistance, is a floating point number in ohms/sq. R_{rf}, the skin effect coefficient, is a floating point number.

`SUP rdc rrf xdc ls`
General. R_{dc}, the DC resistance, is a floating point number in ohms/sq. R_{rf}, the skin effect coefficient, is a floating point number. X_{dc}, the DC Reactance, is a floating point number in ohms/sq. Ls, the kinetic inductance, is a floating point number in pH/sq.

`SEN xdc`
Sense metal. Xdc, the DC reactance, is a floating point number in ohms/sq.

Example 1 `TMET “Freespace” 0 FREESPACE 376.7303136 0 0 0`

The four floating point numbers define the metal loss for the top of the circuit enclosure. This is a special case. When FREE SPACE is selected as the top metal in ***xgeom***, the circuit editor, then these loss values are used.

Example 2 `TMET “WG Load” 0 WGLOAD`

When WGLOAD is selected as the top metal in ***xgeom***, the circuit editor, this syntax is used. When WG Load is chosen, ***em*** models a perfect matched waveguide load whose values are used as the metal’s parameters.

BMET Box Bottom Metal - Only one allowed per file.

Syntax `TMET name patternid type value1 value2 ...`
The syntax for the BMET keyword is the same as that of the TMET command detailed above.

MET Metal Type - Unlimited number allowed per file.

Syntax `MET name patternid type value1 value2 ...`

`name` This is the metal name. This name must be in quotes if it contains spaces. Quotes may be used when they are not required.

`patternid` The index of the pattern to be displayed. Starts at 0.

`type` This identifies the type of metal definition. This is one of the following character strings: NOR, TMM, RES, NAT, SUP, RUF, and SEN for planar metals. VOL, SFC and ARR for via metals. These keywords and the values associated with them are defined below.

`NOR lossfactor currentratio thickness spec_using`
Normal loss model for planar metal. The `lossfactor` is a floating point number which is either the conductivity, resistivity or sheet resistance at DC. See the `spec_using` field to determine which value is being used. The `currentratio` is a floating point number which is the ratio of current on the top surface to current on the bottom surface. Thickness is a floating point

number for the thickness of the metal. If nothing or CDVY is entered in the `spec_using` field, then the `lossfactor` is defined as conductivity. If `spec_using` is the string "RSVY" then the `lossfactor` is resistivity. If the string "SRVY" appears in the `lossfactor` field, then the `lossfactor` is sheet resistance at DC. The units are the default units in the project for each type of value.

```
TMM lossfactor currentratio thickness numsheets
spec_using direction.
```

Thick Metal loss model for planar metal. See the table below for an explanation of the fields.

Field	Values
lossfactor	The lossfactor is a floating point number which is either the conductivity, resistivity or sheet resistance at DC. See the spec_using field to determine which value is being used.
currentratio	The currentratio is a floating point number which is the ratio of current on the top surface to current on the bottom surface. Note that the currentratio is unused for this metal type.
thickness	Thickness is a floating point number for the thickness of the metal.
numsheets	The number of sheets is an integer value indicating the number of sheets to be used to model the thick metal. You may also use a variable for the number of sheets, in which case the variable name would appear in quote marks.
spec_using	If nothing is entered in the spec_using field, then the lossfactor is defined as conductivity. You must enter "CDVY" for conductivity if the direction field is required. If spec_using is the string "RSVY" then the lossfactor is resistivity. If the string "SRVY" appears in the lossfactor field, then the lossfactor is sheet resistance at DC. The units are the default units in the project for each type of value.
direction	If nothing is entered in the direction field, then the direction of the metal is up. You may also enter "TUP" to indicate the direction is up. If the string "TDWN" appears in the direction field, then the direction of the metal is down.

`RESISTOR rdc`
Resistor loss model for planar metal. R_{dc}, the DC resistance, is a floating point number in Ohm/sq.

`NAT rdc rrf`
The Rdc/Rrf loss model for planar metal. R_{dc}, the DC resistance, is a floating point number in ohms/sq. R_{rf}, the skin effect coefficient, is a floating point number.

`SUP rdc rrf xdc ls`
General loss model for planar metal. R_{dc}, the DC resistance, is a floating point number in ohms/sq. R_{rf}, the skin effect coefficient, is a floating point number. X_{dc}, the DC Reactance, is a floating point number in ohms/sq. Ls, the kinetic inductance, is a floating point number in pH/sq.

```
RUF crosssec lossfactor thickness toprough bottomrough
curratio spec_using direction
```

Rough Metal loss model for planar metal. t

Field	Values
`crosssec`	The cross section, `crossec`, is one of two possible character strings: THK for thick and THN for thin.
`lossfactor`	The `lossfactor` is a floating point number which is either the conductivity, resistivity or sheet resistance at DC. See the `spec_using` field to determine which value is being used.
`thickness`	Thickness is a floating point number for the thickness of the metal.
`toprough`	`Toprough` is a floating point number for the roughness of the top surface of the metal in microns. Note that this value is always microns regardless of the length unit set for the project.
`bottomrough`	`Bottomrough` is the a floating point number for the roughness of the bottom surface of the metal in microns. Note that this value is always microns regardless of the length unit set for the project.
`currentratio`	The `currentratio` is a floating point number which is the ratio of current on the top surface to current on the bottom surface. Note that the `currentratio` field only appears if `crosssec` is set to THN.

Field	Values
spec_using	If nothing is entered in the spec_using field, then the lossfactor is defined as conductivity. You must enter “CDVY” for conductivity if the direction field is required. If spec_using is the string “RSVY” then the lossfactor is resistivity. If the string “SRVY” appears in the lossfactor field, then the lossfactor is sheet resistance at DC. The units are the default units in the project for each type of value.
direction	If nothing is entered in the direction field, then the direction of the metal is up. You may also enter “TUP” to indicate the direction is up. If the string “TDWN” appears in the direction field, then the direction of the metal is down.

`SEN xdc`
Sense loss model for planar metal. Xdc, the DC reactance, is a floating point number in ohms/sq.

`VOL lossfactor [solid] wallthickness spec_using`
Volume loss model for via metal. The `lossfactor` is a floating point number which is either the conductivity, resistivity, resistance per via or sheet resistance at DC. See the `spec_using` field to determine which value is being used. `Solid` appears if this volume is being modeled as a solid. If Solid does not appear the volume is modeled as hollow. `Wallthickness` is a floating point number for the wall thickness of the hollow via volume. This field is only used if the via is being modeled as hollow; it is ignored if the

Volume is Solid. The `lossfactor` value is defined by the `spec_using` field as shown in the table below. The units are the default units in the project for each type of value; note that resistance per via is always in Ohms/via.

`spec_using`	`lossfactor`
CDVY	Conductivity
RSVY	Resistivity
RPV	Resistance Per Via
SRVY	Sheet Resistance at DC

`SFC rdc rrf xdc`
Surface loss model for via metal. R_{dc}, the DC resistance, is a floating point number in ohms/sq. R_{rf}, the skin effect coefficient, is a floating point number. X_{dc}, the DC Reactance, is a floating point number in ohms/sq.

`ARR lossfactor fillfactor spec_using`
Array loss model for via metal. The `lossfactor` is a floating point number which is either the conductivity, resistivity or resistance per via. See the `spec_using` field to determine which value is being used. The Fill Factor is a floating point number indicating the percentage of metal in the array area.

The `lossfactor` value is defined by the `spec_using` field as shown in the table below. The units are the default units in the project for each type of value; note that resistance per via is always in Ohms/via.

`spec_using`	`lossfactor`
CDVY	Conductivity
RSVY	Resistivity
RPV	Resistance Per Via

Example 1 `MET “Capbot” 1 SUP 0.02439 3.1e-007 0 0`
Example 2 `MET “Metal1” 2 RES 13`

DIM Dimensions

Syntax

```
DIM STD direction sign
POS xcoord ycoord
NOM nvalue
REF1 POLY idfile 1
ivertex
REF2 POLY idfile 1
ivertex
END
```

The DIM statement defines a dimension in your geometry project. The name, type and orientation of the parameter appear on the first line. Subsequent statements define the label position, nominal value, and reference points for the dimension. This set of statements appears for each dimension in the project. The syntax is detailed below.

`STD` This is the type of dimension. Presently, there is only a standard dimension so that this field is always “`STD`”.

`direction` This field is either “`XDIR`” for a parameter in the x plane (horizontal) or “`YDIR`” for a parameter in the y plane (vertical).

`sign` This field is either "`1`" for a parameter whose reference position is greater than its anchor position and "`-1`" for a parameter whose reference position is less than its anchor position. The coordinates are based on the upper left hand corner of the substrate being point (0,0).

POS Label Position

Syntax

```
POS xcoord ycoord
```

The POS entry provides the position of the dimension label. The label always appears in the center of the dimension in the direction of the dimension. For an x directed dimension, <ycoord >is a floating point number which provides the position of the label as an offset from the first reference point in the y direction. For a y directed dimension, <xcoord> is floating point number which provides the position of the label as an offset from the first reference point in the x direction.

NOM Nominal Value of Dimension

Syntax

```
NOM nomvalue
```

This entry is the nominal value of the dimension. The keyword NOM is followed by a floating point number which is the nominal value of the dimension.

REF1 Reference Point 1

Syntax

```
REF1 POLY idfile 1
vertex
```

This entry identifies the first reference point of a dimension. The REF1 keyword is followed by the POLY statement on the same line. <idfile> is the file ID for the polygon which contains the first reference point. This is followed by the number of points on the polygon which for a dimension is always 1. The next line lists which vertex of the polygon is used as the first reference point.

REF2 Reference Point 2

Syntax

```
REF2 POLY idfile 1
vertex
```

This entry identifies the second reference point of a dimension. The REF2 keyword is followed by the POLY statement on the same line. <idfile> is the file ID for the polygon which contains the second reference point. This is followed by the number of points on the polygon which for a dimension is always 1. The next line lists which vertex of the polygon is used as the second reference point.

END End Dimension

Syntax

```
END
```

This end statement indicates the completion of defining a dimension.

BRI The specification for an isotropic dielectric material.

Syntax

```
BRI "name" pattern_id erel loss_tan diel_cond
```

A BRI line is used to specify an isotropic dielectric brick material. The BRI keyword is followed by the name of the dielectric in quotes and may contain spaces.This is followed by the index of the fill-pattern, then three floating point values. These values are the relative dielectric constant, the loss tangent and the bulk conductivity of the dielectric material.

BRA The specification for an anisotropic dielectric material.

Syntax

```
BRA "name" pattern_id x_erel  x_loss_tan  x_diel_cond
y_erel  y_loss_tan  y_diel_cond z_erel  z_loss_tan
z_diel_cond
```

A BRA line is used to specify an anisotropic dielectric brick material. The BRA keyword is followed by the name of the dielectric in quotes and may contain spaces. This is followed by the index of the fill-pattern, then nine floating point values. In order, these values are the following:

- X relative dielectric constant
- X loss tangent
- X bulk conductivity
- Y relative dielectric constant
- Y loss tangent
- Y bulk conductivity
- Z relative dielectric constant
- Z loss tangent

- Z bulk conductivity

VALVAR Variables

Syntax `VALVAR varname unittype value "description"`

The VALVAR statement defines a variable in your project. The name, units and present nominal value of the variable and a brief description are defined as detailed below.

`varname` The name of the variable. This is a character string.

`unittype` This field is one of six choices shown in the table below. The type of unit is shown here. The actual units used are the default for that type of unit in the project. For example, if the unittype is "LNG" and the project is presently using mils, then the variable's value is in mils.

unittype	Type of Units
LNG	Length
RES	Resistance
CAP	Capacitance
IND	Inductance
FREQ	Frequency
OPS	Ohms/sq
SPM	Siemens/meter
PHPM	picoHenries/meter
RRF	R_{rf}
NONE	Undefined

`value` This field is the present nominal value of the variable and is a floating point number. If the variable definition is an equation, the equation is a character string which appears in this field in quote marks. For example, you have a variable, Length that is defined as 2*Width. In that case “2*Width” would appear in the `value` field. For details on the equation syntax, please refer to ”Equation Syntax” in Help.

`“description”` This field is a description of the variable and is a character string.

GEOVAR Parameters

Syntax

```
GEOVAR parname partype direction sign scaletype
POS xcoord ycoord
NOM nvalue
REF1 POLY idfile numpol
ivertex
REF2 POLY idfile numpol
ivertex
EQN “vareqn”
PS1 numpt
POLY idfile numpt
ivertex(1)
.
.
ivertex(n)
END
PS2 numpt
POLY idfile numpt
ivertex(1)
.
.
ivertex(n)
END
END
```

The GEOVAR statement defines a dimension parameter in your geometry project. The name, type and orientation of the parameter appear on the first line. Subsequent statements define the label position, nominal value, and point sets for the parameter. This set of statements appears for each parameter in the project including linked parameters. The syntax is detailed below.

`parname` The name of the parameter. This is a character string.

`partype` This field is either ANC for an anchored parameter, SYM for a symmetric parameter or RAD for a radial parameter.

`direction` This field is either “`XDIR`” for a parameter in the x plane (horizontal) or “`YDIR`” for a parameter in the y plane (vertical).

`sign` This field is either “`1`” for a parameter whose reference position is greater than its anchor position and “`-1`” for a parameter whose reference position is less than its anchor position. The coordinates are based on the upper left hand corner of the substrate being point (0,0).

`scaletype` This field indicates if and how scaling is applied to a dimension parameter when its size is changed. This field is only used for anchored and symmetric dimension parameters. The three possible values are `NSCD`, `SCUNI`, and `SCXY`. `NSCD` is used if no scaling is being applied; for a radial parameter this value is always used. `SCUNI` indicates that scaling is only applied in the direction in which the dimension parameter is oriented. `SCXY` indicates that the dimension parameter is scaled in both the x and y direction.

POS Label Position

Syntax `POS xcoord ycoord`

The POS entry provides the position of the parameter label. The label always appears in the center of the parameter in the direction of the parameter. For an x directed parameter, ycoord is a floating point number which provides the position of the label as an offset from the first reference point in the y direction. For a y directed parameter, xcoord is floating point number which provides the position of the label as an offset from the first reference point in the x direction.

NOM Nominal Value of Parameter

Syntax `NOM nomvalue`

This entry is the nominal value of the parameter. The keyword NOM is followed by a floating point number which is the nominal value of the parameter.

REF1 Reference Point 1

Syntax

```
REF1 POLY idfile 1
vertex
```

This entry identifies the first reference point of a symmetric parameter or the anchor point of an anchored parameter. The REF1 keyword is followed by the POLY statement on the same line. <idfile> is the file ID for the polygon which contains the first reference point. This is followed by the number of points on the polygon which for the reference point is always 1. The next line lists which vertex of the polygon is used as the first reference point. The vertices are numbered starting at 0.

REF2 Reference Point 2

Syntax

```
REF2 POLY idfile 1
vertex
```

This entry identifies the second reference point of a symmetric parameter or the reference point of an anchored parameter. The REF2 keyword is followed by the POLY statement on the same line. <idfile> is the file ID for the polygon which contains the second reference point. This is followed by the number of points on the polygon which for the reference point is always 1. The next line lists which vertex of the polygon is used as the second reference point.

EQN Equation or variable

This entry is optional and only appears if the nominal value of the variable assigned to the dimension parameter is defined by another variable or equation. The keyword is followed by a character string in quote marks which defines the variable or equation. For details on equation syntax and available functions, please see “Equation Syntax” in Help.

PS1 Point Set 1

Syntax

```
PS1 numpol
POLY idfile numpt
ivertex(1)
.
```

```
.
ivertex(n)
END
```

This entry starts the specification of the first point set associated with a parameter. The PS1 keyword is followed by an integer of the number of polygons which contain points in the point set. For an anchored or radial dimension parameter, there is no first point set, so numpol is always 0.

For each polygon with points in the point set, there is a POLY entry followed by vertices statements.<idfile> is the file ID for the polygon. numpt is the number of points on the polygon which are in the point set. Each point in the point set has a line which contains the vertex number on the polygon of the point. When all the points in the point set have been entered an END statement appears to indicate the end of the point set specification.

For example. If vertices 2 and 4 of polygon 13 and vertex 4 of polygon 17 are included in the point set, the input in the file would be as follows:

```
PS1 2
POLY 13 2
2
4
POLY 17 1
4
END
```

PS2 Point Set 2

Syntax

```
PS2 numpol
POLY idfile numpt
ivertex(1)
.
.
ivertex(n)
END
```

This entry starts the specification of the second point set associated with a parameter. The PS1 keyword is followed by an integer of the number of polygons which contain points in the point set. The second point set is specified using the same syntax as the first point set documented above.

END End of parameter specification.

Syntax

```
END
```

Once both point sets have been specified for a GEOVAR statement, an END statement appears to indicate the end of this GEOVAR statement. There is a GEOVAR statement for each parameter in a project including linked parameters.

BOX The dimensions of the box and the parameters of the enclosed substrates.

Syntax

```
BOX nlev xwidth ywidth xcells2 ycells2 nsubs eeff
thickness erel mrel eloss mloss esignma nzpart name
thickness erel mrel eloss mloss esignma nzpart name
.
.
thickness erel mrel eloss mloss esignma nzpart name
```

If the dielectric is anisotropic then the following syntax is appended to the end of the line and these values are used for the dielectric properties in the Z direction:

```
erel mrel eloss mloss esignma
```

The dimensions of the box and the parameters of the enclosed substrates are described with the multi-lined BOX statement. BOX is followed by at least five numbers with two more optional. First, <nlev>, is the number of metalization levels (= 1 if using standard microstrip). Next are the X dimension, <xwidth> (left to right) and Y dimension, <ywidth> (top to bottom), dimensions of the box given in the units specified by the LEN statement. These are followed by two integers: 2 times the number of cells in the X-dimension, <xcells>, and then 2 times the number of cells in the Y-dimension, <ycells2>. The first optional number is an integer which historically set the minimum density of subsections per wavelength but is no longer used. The default value of 20 should be used as a placeholder. The second optional number is the effective dielectric constant (E_{eff}) used to calculate the wavelength for satisfying the subsections/wavelength parameter. If not specified, or if it is less than 1.0, the parameter is ignored and a simple estimate of E_{eff} is used. Please note that you must specify the first optional number in order to use the second number. The last entry is the name of the dielectric used for the dielectric layer. The name needs to appear in quotes if there is a space. You may use quotes around the name when they are not required.

The lines following BOX provide information on each of the dielectric layers in the box. Note that the number of layers is one more than the number of levels, <nlev>. Each line has the thickness of the layer (in the units previously specified) followed by the relative dielectric constant and the relative permeability of the layer. The fourth and fifth parameters are the dielectric and magnetic loss tangents, respectively. The sixth parameter is the bulk conductivity for the dielectric layer. The seventh and final parameter is the number of z-partitions. This parameter is used for subsectioning dielectric bricks. All items except the substrate thickness are optional. The default is lossless free space. Only one dielectric layer per line is allowed.

VNCELLS Identifies variable(s) being used for the number of cells in the box size. This entry only appears if the number of cells in either the x or y direction or both is defined using a variable.

Syntax `VNCELLS numcellsx numcellsy`

If the number of cells in the x direction is defined using a variable then <numcellsx> is set to a character string of the name of the variable in quotes. If the number of cells is defined using a constant value then <numcellsx> is set to an integer value greater than 0. The same definition holds true for <numcellsy> but applies to the y direction.

For example:

VNCELLS “XCells” 16

means that the variable `XCells` is used to define the number of cells in the x direction and there are 16 cells in the y direction.

TECHLAY Defines a Technology Layer by identifying the name, material used, meshing properties and import/export mapping.

Syntax

```
TECHLAY lay_type lay_name mapping
type
ilevel nvertices mtype filltype debugid xmin ymin
xmax ymax conmax res res edgemesh
TOLEVEL to_level meshingfill pads
END
END
```

<laytype> is a character string which defines the type of Technology Layer. The field is a character string set to “BRICK” for Brick Technology Layers, “METAL” for Metal Technology Layers, and “VIA” for Via Technology Layers.

<lay_name> is a character string which defines the name of the Technology Layer. The name of each Technology Layer must be unique.

The next fields define the <mapping>. The syntax is shown below.

```
dxf_layer gds_stream gds_object GBR gbr_name
```

where <dxf_layer> is a character string defining the DXF Layer name to which the Technology Layer is mapped. If no DXF layer name is specified, then <dxf_layer> is set to “<Unspecified>”. <gds_stream> is an integer value defining the GDS Stream to which the Technology Layer is mapped. If this integer is negative, this indicates no values have been entered for the GDS Stream and Object fields. <gds_object> is an integer value defining the GDS Object to which the Technology Layer is mapped. GBR <gbr_name> defines Gerber file name to which the Technology Layer is mapped. The GBR keyword is used to identify the file name as Gerber and <gbr_name> is a character string.

The next three lines of the syntax are the polygon properties. Please see the keyword **NUM** on page 43 for explanations of the syntax.

The first END keyword indicates the completion of the polygon properties and the second END statement indicates the completion of the Technology Layer definition.

EVIA1 Defines an edge via in the circuit by identifying the polygon edge of origin and the level to which it extends.

Syntax

```
EVIA1
POLY poly_id 1
edge_number
TOLEVEL to_level
```

The POLY statement identifies which polygon the edge via is attached to. The poly_id is the number which identifies the polygon. The “1” is required.

The edge_number indicates the index number of the polygon vertex. The via is placed on the polygon edge specified by this vertex to the next vertex. For example, if vertex 3 is specified, the via extends from vertex 3 to vertex 4 on the polygon.

The TOLEVEL statement indicates to which level the via extends. The edge via extends from the level the polygon it is attached to is on to the level identified in to_level.

LORGN Defines the location of the (local) origin relative to the UPPER left corner of the substrate.

Syntax

```
LORGN X Y [U|L]
```

The LORGN statement defines the location of the origin relative to the upper left corner of the substrate. X is the distance in the x plane from the upper left hand corner of the substrate and Y is the distance in the y plane from the upper left hand corner of the substrate. U appears if the location of the origin is unlocked and may be dragged to a new location by the user. L appears if the origin location is locked and can only be changed by using the Local Origin Properties dialog box. The default location for the origin in a new file is the lower right hand corner of the substrate so this entry would be

LORGN 0 160 U

POR1 Defines a port in the circuit by identifying the polygon edge of origin and specifying its parameters.

Syntax

```
POR1 type
DIAGALLOWED [Y|N]
POLY ipolygon #points
ivertex
portnum resist react induct capac xcoord ycoord [reftype
rpcallen]
```

There is an POR1 statement for each port in the circuit. The POR1 statement identifies the type of port. <type> is either STD for standard, AGND for autogrounded, or CUP for co-calibrated port.

If the <type> is STD, and the port is independent then the terms <reftype> and <rpcallen> appears. These values do not appear for a STD port if it is a shared port.

If <type> is AGND, then the terms <reftype> and <rpcallen> are used. These values do not appear for a STD port.

If <type> is CUP, then the term <groupid> is used. <groupid> is a letter or letters that identify the calibration group to which the co-calibrated port belongs. If the co-calibrated port is automatically grouped then <groupid> is set to the string “Auto.”

NOTE: **If <groupid> is set to “Auto” then the CUPGRP entry which defines its calibration group must follow directly after the <portnum> line of the POR1 block entry and should be set to LOCAL.**

DIAGALLOWED is an optional entry applying only to box wall ports which only appears if the user has defined the box wall port as independent (a necessary condition in order for the reference plane to be defined as diagonal). If the entry is “DIAGALLOWED N” then the reference plane for the port is not allowed to be diagonal. The entry “DIAGALLOWED Y” indicates that the user has defined the reference plane as diagonal. This does not necessarily mean that the reference plane is diagonal, only that it is permitted to be.

The POLY statement which follows identifies the polygon. <ipolygon> is the file ID for the polygon on which the port is placed. The number of points, <#points>, is always 1 for a POR1 statement.

The <ivertex> indicates the index number of the polygon vertex. The port is placed on the polygon edge specified by this vertex to the next vertex. For example, if vertex 3 is specified, the port is placed between vertex 3 and vertex 4 on the polygon.

A port is usually placed at the edge of the substrate so that a connection to ground is readily available (the sidewall). Otherwise a port is placed at the junction of two polygons (each polygon has a line segment colinear with the other). In this case, the two terminals of the port are formed by the two polygons.

The next value, <portnum>, is the port number. Port numbers are automatically assigned starting at 1 and incrementing upward as ports are added to the circuit. This number cannot be zero, but may be any other value. It need not be consecutive. If the number is the same as the number of any other port, the ports will be connected together electrically. If the number is negative, it will be connected together with any other negative numbers of the same value and the total current going out of the negative ports will be made equal to the total current going into all the ports with the positive number.

Four optional floating point numbers follow representing the impedance to which the port S-parameters will be normalized. The first is resistance (ohms), the second is reactance (siemens), the third is inductance (henries) and the fourth is capacitance (farads). The final two values are location of the port on the substrate; the x-coordinate is first, followed by the y-coordinate. The origin (0,0) is the upper left hand corner of the substrate.

<reftype> is a character string which identifies the type of reference plane used for the autogrounded, independent co-calibrated or independent STD port. <reftype> is FIX for a reference plane and NONE for a calibration length. <rpcallen> is a floating point number which provides the length of the reference plane when <reftype> is FIX and provides the calibration length when <reftype> is NONE. If there is no reference plane or calibration length, then the entry reads “NONE 0”.

CUPGRP Defines the calibration group properties for a calibration group. The calibration group to which a co-calibrated port belongs is identified as part of the POR1 statement which defines the co-calibrated port.

Syntax

```
CUPGRP groupid
ID objectid
GNDREF B|F|P
TWTYPE FEED|CUST|1CELL
TWVALUE termwidth
DRP1 ...
END
```

There is an CUPGRP statement for each calibration group in the circuit. <groupid> is a letter or letters that identify the calibration group and appears in quotes, e.g, “A” which is the name of the calibration group that is displayed in the project editor on any port which is included in the calibration group. If the co-calibrated port is to be automatically grouped, then <groupid> is set to the string Auto in quotes followed by the word LOCAL as shown below:

```
CUPGRP “Auto” LOCAL
```

NOTE: **If the CUPGRP entry is set to LOCAL, then it must appear directly after the POR1 block (last entry line starting with <portnum>) which defines the co-calibrated port.**

The next entry in the CUPGRP block is the ID where the <objectid> is an integer number which identifies the calibration group. This ID number is used elsewhere in the project file to reference this component.

GNDREF defines the ground reference where B is for the Sonnet box, and F is for floating.

The TWTYPE defines the type of terminal width defined for the calibration group where FEED is for feedline width, CUST is for user defined, or 1CELL for one cell wide.

The TWVALUE entry only appears if the TWTYPE is defined as CUST. This is a floating point number which defines the width of the terminal.

If there are reference planes defined for the component, there will be DRP1 entries. For details on the syntax, please see "DRP1" on page 15.

END defines the end of the calibration group properties block.

SMD

Defines a component in the circuit by identifying the type of component, location of the schematic box, port properties, label and reference planes.

Syntax

```
SMD levelnum label
ID objectid
GNDREF B|F|P
TWTYPE FEED|CUST|1CELL
TWVALUE termwidth
DRP1 ...
SBOX leftpos rightpos toppos bottompos
PBSHW Y|N
PBOX leftpos rightpos toppos bottompos
PKG length width height
LPOS xpos ypos
TYPE ...
SMDP levelnum x y orientation portnum pinnum
```

There is an SMD statement for each component in the circuit.

<levelnum> : A positive integer value which identifies the level on which the SMD is located.

<label> : An alphanumeric string in quotes, e.g, “comp1” which is the label of the component displayed in the project editor.

The next entry in the SMD entry is the ID.

<objectid>: An integer number which identifies the SMD. This ID number is used elsewhere in the project file to reference this component.

GNDREF defines the ground reference where B is for the Sonnet box, F is for floating (default for ideal components), or P for user defined. If GNDREF is set to P additional port(s) must be defined using the SMDP keyword (see below). These ports will be assigned to a <pinnum> of zero. This setting is not allowed for a Ports Only component (TYPE NONE).

The TWTYPE defines the type of terminal width defined for the component where FEED is for feedline width, CUST is for user defined, or 1CELL for one cell wide.

The TWVALUE entry only appears if the TWTYPE is defined as CUST. This is a floating point number which defines the width of the cell.

If there are reference planes defined for the component, there will be DRP1 entries. For details on the syntax, please see "DRP1" on page 15.

SBOX determines the location of the schematic box or ideal component symbol. The four values are floating string values which represent the location of the four corners of the schematic box, left, right, top, and bottom. Note that these values are relative to the upper left hand corner of the substrate being the origin.

PBSHW indicates if the user has selected to display a package size; Y for yes, or N for no.

The PBOX entry only appears if PBSHW is set to Y. This entry determines the location of the package box using the same syntax as the SBOX entry discussed earlier.

The PKG entry only appears if PBSHW is set to Y and the user has entered dimensions for the package. The <length>, <width> and <height> are all floating point numbers which define the length, width and height of the package respectively.

LPOS provides the location of the center of the component label. <xpos> and <ypos> are floating point numbers which indicate the distance from the origin (upper left hand corner of the substrate) in the x and y direction.

The TYPE entry identifies the type of component and other parameters as detailed in below. The component type is followed by the syntax for that type of component and any parameters are defined following the syntax.

Ports Only TYPE NONE

There are no parameters.

Ideal

```
TYPE IDEAL idealtype compval
```

This entry indicates an ideal component. <idealtype> is either RES for resistor, CAP for capacitor, or IND for inductor. <compval> is a floating point number for the value of the ideal component.

Date File

```
TYPE SPARAM paramfileindex
```

This entry indicates a Data File component. <paramfileindex> is an integer value which identifies the file using the index defined in the SMDFILES block, see "SMDFILES" on page 79.

User Model

```
TYPE UMOD comp_name DLL lib_folder lib_module parameters
```

This entry indicates a User Model component. <comp_name> is a character string which is the name of the component; this name appears in the project editor window. <lib_folder> is a character string which identifies the library. <lib_module> is a character string which identifies the sub-library. <parameters> varies depending on the number of parameters for the particular model. The syntax is par_name=par_value where <par_name> is a character string which is the name of the parameter and par_value is either a floating point number or a variable name. There is no limit to the number of parameters. See the example below. Note that the entry would all be on the same line in the project file with no wrap around.

```
TYPE UMOD MODcatc100B101 DLL modelithics
Modelithics_CLR_ATC Capacitance=0.1 Sim_mode= W2
```

Sonnet Project

```
TYPE SPROJ comp_name project_name parameters
```

This entry indicates a Sonnet Project component. <comp_name> is a character string which is the name of the component; this name appears in the project editor window. <project_name> is the path and filename of the Sonnet project used for the component. The file should use the .son extension. <parameters> varies depending on the number of parameters for the particular model. The syntax is par_name=par_value where <par_name> is a character string which is the name of the parameter and par_value is either a floating point number or a variable name. There is no limit to the number of parameters. See the example below. Note that there is no path name since the component project is in the same directory as the project in which it is being used.

```
TYPE SPROJ MIM_cap_only.son Wt=65.0 Wb=W2
```

This completes the TYPE entry of the SMD definition.

```
SMDP levelnum x y orientation portnum pinnum
```

The SMDP entry defines a component port. There is an SMDP statement for each port of the component. <levelnum> is a positive integer identifying the level on which the port is placed. <x> and <y> provide the location of the port relative to the origin of the upper left hand corner. <orientation> is either T for top, B for bottom, R for right or L for left, indicating in which direction the port is oriented. This field is not used for all component ports. <portnum> is an integer value identifying the port; this number is not displayed in the project editor. <pinnum> is an integer value identifying which pin on the physical component is connected to this component port. This value is set to zero if the port is a ground reference.

NUM

The NUM statement is used to define the number and location of polygons in the circuit.

Syntax

```
NUM npoly
type
ilevel nvertices mtype filltype debugid xmin ymin
xmax ymax conmax res res edgemesh
TOLEVEL to_level meshingfill pads
TLAYNAM lay_name inherit
xvertex yvertex
    .
    .
xvertex yvertex
```

The number of polygons in the file, <npoly>, follows NUM on the same line. The lines which follow the NUM statement until the END line define an individual polygon. This is the last section of the GEO block.

The first line for each polygon section is the type. This line is optional for a metal polygon and would be set to MET POL if used. If the polygon is a dielectric brick, then this line is BRI POL If the polygon is a via polygon, then this line is VIA POLYGON.

The next line defines the metalization level, number of vertices, metal type, fill type, subsectioning constraints and edge meshing setting.

<ilevel>: Circuit metalization level index, which begins with the index 0.

<nvertices> : Number of vertices which make up the polygon.

<mtype> : Index number which identifies the planar metal, via metal or dielectric brick type for the polygon. If the polygon is metal or a via, an index of -1 indicates the default lossless metal; user-defined metals start at index 0. Indices are assigned implicitly by the location of the appropriate MET statement in the file. The first MET statement is index 0, the next index 1, and so on. Please note that only planar metal types may be used for metal polygons and via metal types for via polygons. If the polygon is a dielectric brick, an index of 0 indicates the default dielectric, air; user defined dielectrics start at index 1. Again, indices are assigned implicitly by the location of the appropriate PRI or PRA statement in the file.

<filltype>: Identifies the fill type used for the polygon. N indicates staircase fill, T indicates diagonal fill and V indicates conformal mesh. Note that filltype only applies to metal polygons; this field is ignored for dielectric brick polygons.

<debugid>: Identifies the polygon for internal debugging purposes. An integer value which should be set to “0” or to a unique value greater than 1000.

<xmin> and <ymin>: Define the minimum subsection size in number of cells for each dimension.

<xmax> and <ymax> : Define the maximum subsection size in number of cells.

The default for the minimum in both dimensions is 1, with a default for maximum of 100 for both dimensions.

<conmax>: Maximum length for a conformal mesh subsection. If this value is zero, the maximum length for a conformal mesh subsection is automatically calculated.

<res> : Reserved for future development. Set these to 0.

<edge mesh>: Y indicates edge meshing is on for this polygon. N indicates edge meshing is off.

The subsectioning information, fill type, and edge mesh setting are input in the Metalization Properties dialog box in the project editor.

The next statement, TOLEVEL, is only used for a via polygon.

<To_level >: The level to which the polygon extends. It originates on the level identified in the header line above.

<Meshingfill>: Character string which identifies the type of meshing fill used for the via polygon. Choices include `RING`, `CENTER`, `VERTICES`, `SOLID` and `BAR`. `RING` is the default for new via polygons. `Pads` is a character string which indicates whether the via polygon has via pads at the top and bottom. If there are no via pads, this field is set to `NOCOVERS`, if there are via pads, then the field is set to `COVERS`.

The next statement, TLAYNAM, identifies the Technology Layer with which the polygon is associated. This entry does not appear if the polygon is independent.

<lay_name> : Identifies the Technology Layer with which the polygon is associated. This name should be the same as the <lay_name> field in the TECHLAY entry which defines the Technology Layer.

<inherit> : should be set to the string INH if the polygon is inheriting its properties from the Technology Layer. If the polygon is set to LOCAL, then this field is NOH.

The next statements are the locations of the vertices of the polygon.

<xvertex> and <yvertex> : The following lines are x,y points pairs that define the location in terms of the substrate grid of all the vertices of the polygon. There is one line per vertex; also note that the first and last vertices are the same location in order to close the polygon off. Vertex indices are assigned implicitly by the location of the appropriate point pair statement in the file. Vertex index numbers start at zero.

The data are floating point numbers in the previously specified units (see LEN). The origin point of the grid (0,0) is located at the upper left hand corner of the substrate when viewed in the project editor. Immediately following the last line is an END statement. Subsequent polygons are presented in an identical manner, no blank lines allowed.

Once a NUM line is encountered, the other keywords, described above, are not recognized.

Polygons which have edges in common are electrically connected. Polygons with an edge in common with the edge of the substrate are automatically shorted to ground on that edge, unless there is a port at that point. The polygons may overlap; however it is recommended that they do not. Likewise, a polygon may be complex (it crosses over itself). However this is of little utility.

END End statement

Syntax `END GEO`

Indicates the end of the geometry block. Required.

Frequency Block

The frequency block contains all the frequency sweeps which have been input in a project. Which sweep is presently being used is specified in the CONTROL block. For details about the control block, see “CONTROL,” page 49.

FREQ Beginning of frequency block

Syntax

```
FREQ
```

Indicates the beginning of the frequency block. All statements following this entry are included in the frequency block until you reach the END FREQ statement. The statements in this block are generated in the Frequency Sweep dialog box in the analysis setup.

SWEEP Linear frequency sweep with stated interval.

Syntax

```
SWEEP f1 f2 fstep
```

Linear frequency sweep. <f1> is the starting frequency. <f2> is the ending frequency. <fstep> is the interval between frequencies.

ESWEEP Exponential frequency sweep.

Syntax

```
ESWEEP f1 f2 Nfreq
```

Exponential frequency sweep from <f1> to <f2> with a common ratio between the <Nfreq> frequency points. <f1> is the starting frequency. <f2> is the ending frequency. <Nfreq> is the number of points in the sweep.

LSWEEP Linear frequency sweep with number of points.

Syntax

```
LSWEEP f1 f2 Nfreq
```

Linear frequency sweep from <f1> to <f2>. <f1 is the starting frequency. <f2> is the ending frequency. <Nfreq> is the number of analysis frequencies. <Nfreq> is used to calculate the step size in the following manner:

```
Step Size = (f2-f1)/(Nfreq-1)
```

LIST Frequency list of analysis frequencies.

Syntax `LIST f1 f2 ... fN`

List of individual analysis frequencies. <f1> is a floating point number representing the first frequency at which the circuit is analyzed. <f2> is a floating point number representing the second frequency at which the circuit is analyzed. There is no limit to the number of frequencies which may be entered.

STEP Discrete analysis frequencies

Syntax `STEP f1`

Discrete frequency. The value is a floating point number. Only one frequency value may be entered per Step statement.

ABS_ENTRY Adaptive Band Synthesis Sweep

Syntax `ABS_ENTRY startfreq stopfreq`

An Adaptive Band Synthesis (ABS) from <startfreq> to <stopfreq>. These two values define the band across which the ABS is being performed.

ABS_FMIN Find the minimum frequency response.

Syntax `ABS_FMIN NET= param f1 f2`

ABS_FMIN finds the frequency at which the minimum frequency response occurs. The <param> field specifies the parameter whose minimum you wish to find. Only S-Parameters may be used. The format is S followed by a pair of port indices. For example, S_{24} would be S24. If one of the two port indices has more than one digit, you use an underscore to separate the two port indices. For example, S10_27.

<f1> is the starting frequency and <f2> is the ending frequency for the ABS sweep that is performed before determining the minimum.

ABS_FMAX Find the maximum frequency response.

Syntax `ABS_FMAX NET= param f1 f2`

ABS_FAX is identical to ABS_FMIN except that it finds the frequency at which the maximum frequency response occurs.

DC_FREQ Analyze at a DC frequency point.

Syntax `DC_FREQ fcalc frequency`

DC_FREQ specifies an analysis at a DC Point. The fcalc field is “AUTO” if you wish to allow ***em*** to automatically calculate the analysis frequency. When AUTO is used, there is no `frequency` entry. The fcalc field is set to “MAN” if the analysis frequency has been entered by the user. This option is followed by the desired analysis frequency in KHz.

END End statement

Syntax `END FREQ`

Indicates the end of the frequency block. Required.

Control Block

The control block specifies the type of analysis sweep presently defined for the project. All the analysis controls which have been input to the project are available in the FREQ block. The control block identifies which sweep would be used in the actual analysis of the project.

The CONTROL block contains a statement which indicates the type of sweep presently selected in the Analysis Control drop list in the Analysis Setup dialog box. Only one type of sweep is entered in this block at any one time. Choices include SIMPLE, STD, ABS, RES_ABS, OPTIMIZE, VARSWP, and EXTFILE. See the keyword entry below for details.

CONTROL Beginning of control block

Syntax `CONTROL`

Indicates the beginning of the control block. All statements following this entry are included in the control block until you reach the END CONTROL statement. The statements in this block are generated in the Analysis Setup dialog box.

SIMPLE Simple Sweep

Syntax `SIMPLE`

This selects a simple sweep. The simple sweep values are entered under the keyword SIMPLE in the FREQ block.

STD Standard Sweep

Syntax `STD`

This selects the standard sweep. The standard sweep values are entered under the SWEEP keyword in the FREQ block. There may be multiple STD statements in the frequency block.

ABS Adaptive Band Synthesis (ABS)

Syntax This selects an Adaptive Band Synthesis. If a Manual resolution has been entered by the user, then it appears in the RES_ABS statement in this block.

RES_ABS ABS Resolution

Syntax `RES_ABS [N|Y] resolution`

This controls the ABS resolution. If the resolution is automatically calculated by em, then an N appears in the second field. If the manual resolution is selected in the Advanced Options dialog box, then Y appears in the second field. <resolution> is a floating point number which is the minimum gap between data points to be used in the ABS analysis. This value is input by the user.

OPTIMIZE Optimization

Syntax `OPTIMIZE`

This selects an optimization. The optimization is specified in the OPT block. For details on the OPT block, see “OPT,” page 56.

VARSWP Parameter Sweep

Syntax `VARSWP`

This selects a parameter sweep. The parameter sweep is specified in the VARSWP block. For details on the VARSWP block, see “VARSWP,” page 58.

EXTFILE External Frequency File

Syntax

```
EXTFILE
```

This selects an external frequency file. The external frequency file is identified in the FILENAME statement which also appears in the control block. For details about the FILENAME statement, see “FILENAME,” page 53.

OPTIONS Analysis Control options

Syntax

```
OPTIONS [-<option identifiers>]
```

Command line options for the analysis run. Options appear depending on which checkboxes are selected in the Analysis Setup dialog box and Advanced Options dialog box. Multiple analysis options can be combined. For example, if both Generate Current Density and De-embedding are selected then the entry would be OPTIONS -dj. The table below shows all the options along with the checkboxes that generate them. In addition, any options entered in the Additional Options text entry box in the Advanced Options dialog box appear here. Note that run options are case sensitive.

Checkbox	Option
Generate Current Density	j
Multi-Frequency Caching	A
Memory Saver	m
Box Resonance	b
De-embedding	d
Q-factor Accuracy	

SUBSPLAM Subsections per Lambda

Syntax

```
SUBSPLAM use subslambda
```

This statement provides the maximum subsections/lambda and is optional. If no SUBSPLAM entry appears, the default value of 20 is used. This value is entered in the Advanced Subsectioning dialog box. <use> indicates if the value is being used. The field is “Y” for yes and “N” for no. <subslambda> is an positive integer value which defines the maximum number of subsections/lamba. The minimum legal value is 6. In release 6.0a, this value was input with the Box parameters.

EDGECHECK Polygon Edge Checking

Syntax `EDGECHECK use numlevels checktype`

This statement provides the type of edge checking and the number of levels or Technology Layers for which polygon edge checking is performed. If no EDGECHECK entry appears, the default value of 1 is used. This value is entered in the Advanced Subsectioning dialog box. <use> indicates if the value is being used. The field is “Y” for yes and “N” for no. <numlevels> is a positive integer value which defines the number of levels or Technology Layer for which polygon edge checking is performed. <checktype> defines whether the edge checking is done based on metal levels or Technology Layer. If the field does not appear, then metal levels are used. If it is set to the string “TECHLAY” then edge checking is based on Technology Layers.

CFMAX Maximum Subsectioning Frequency

Syntax `CFMAX use subfreq`

This statement provides the frequency used to calculate the maximum subsection size. If no CFMAX entry appears, the value is automatically calculated by ***em***. This value is entered in the Advanced Subsectioning dialog box. <use> indicates which frequency is being used. The field is “N” for the highest frequency in the present analysis only, “Y” for a fixed frequency which is entered in the <subfreq> field, “L” for the highest frequency in a previous analysis only, and “B” for the highest frequency in the present or a past analysis. <subfreq> is a floating point value which defines the frequency used to calculate the maximum subsection size when a Fixed Frequency is selected; this would be indicated by a “Y” in the use field. If the use field is another value then <subfreq> is not used. The units for the frequency are those stated in the DIM statement of the GEO block.

CEPSY Estimated Epsilon Effective

Syntax `CEPSY use epsilon`

This statement provides the estimated epsilon effective. If no CEPSY entry appears, the value is automatically calculated by ***em***. This value is entered in the Advanced Subsectioning dialog box. <use> indicates if the value is being used. The field is “Y” for yes and “N” for no. <epsilon> is a floating point value which defines the estimated epsilon effective. This value must be greater than zero and should be greater than one.

FILENAME External Frequency File

Syntax `FILENAME filename`

The filename statement provides the name of the external frequency file used to control the analysis. This file is only used if the EXTFILE statement also appears in the control block. The filename is a character string. This character string may include an absolute or relative path name.

SPEED Analysis Speed/Memory Control - Valid only for geometry projects.

Syntax `SPEED setting`

This entry is the position of the Speed/Memory slider in the Analysis/Speed Control dialog box. This entry is only valid for geometry projects. See the table below for values of setting.

Setting Value	Meaning
0	Fine/Edge Meshing (Left on Slider)
1	Coarse/Edge Meshing (Middle on Slider)
2	Coarse/No Edge Meshing (Right on Slider)

RES_ABS Target for Manual Frequency Resolution for ABS Sweep

RES_ABS [Y|N] number

This entry is the number of frequencies entered as the target for an ABS sweep when Manual is selected for the ABS frequency resolution in the Advanced Options dialog box. The number of frequencies is an integer value. For a new project in which the Manual option has not been selected, this entry does not appear. Once the Manual option has been selected for the Frequency Resolution, then the RES_ABS statement appears with a “Y” indicating that Manual is presently selected and a “N” indicating that Automatic is presently selected. If Manual is selected, the TARG_ABS statement is ignored.

CACHE_ABS ABS Caching Level

```
CACHE_ABS setting
```

This entry is the selected setting of the ABS caching level in the Advanced Options dialog box. See the table below for values of setting:

Setting Value	Meaning
0	None
1	Stop/Restart
2	Multi-sweep plus Stop/Restart

TARG_ABS Target for Automatic Frequency Resolution for ABS Sweep

```
TARG_ABS number
```

This entry is the number of frequencies entered as the target for an ABS sweep when Automatic is selected for the ABS frequency resolution in the Advanced Options dialog box. The number of frequencies is an integer value. For a new project, this is the only entry which appears for the ABS frequency resolution. If the Manual option has even been selected for the Frequency Resolution, then the RES_ABS statement also appears. In that case, this statement is only used, if the RES_ABS statement indicates that “Manual” is not selected. See “RES_ABS,” page 53 for details.

Q_ACC Q-Factor Accuracy

```
Q_ACC use
```

This statement determines if the Q-Factor Accuracy option in the Advanced Options dialog box is being used. <use> indicates if the value is being used. The field is “Y” for yes and “N” for no.

DET_ABS_RES Enhanced Resonance Detection

```
DET_ABS_RES use
```

This statement determines if the Enhanced Resonance Detection option in the Advanced Options dialog box is being used. <use> indicates if the value is being used. The field is “Y” for yes and “N” for no. This entry is optional if the option is not being used.

PUSH Hierarchy Sweep - Valid only for netlist projects.

Syntax

```
PUSH
```

The presence of this keyword indicates that the Hierarchy sweep option has been selected in the Analysis Setup dialog box for a netlist project. A Hierarchy sweep imposes its frequency sweep on all subprojects during an analysis. This option is only available for a netlist project.

END End statement

Syntax

```
END CONTROL
```

Indicates the end of the control block. Required.

Optimization Block

The optimization block specifies the sweep, parameter data range and optimization goals for an optimization. This block is only used when the control block contains the OPTIMIZE entry. Any frequency sweep which may be defined for an ***em*** analysis may be used for an optimization. For details about these entries, see the "Frequency Block" on page 46.

OPT Beginning of optimization block

Syntax

```
OPT
```

Indicates the beginning of the optimization block. All statements following this entry are included in the optimization block until you reach the END OPT statement. The statements in this block are generated in the Analysis Setup dialog box.

MAX Maximum number of iterations.

Syntax

```
MAX num
```

The keyword MAX is followed by an integer number, <num>, which is used as the maximum number of iterations for an optimization.

VARS Optimization Parameter Settings

Syntax

```
VARS
VAR varname(1) [N|Y] minval maxval granularity
VAR varname(2) [N|Y] minval maxval granularity
.
.
var varname(n) [N|Y] minval maxval granularity
END
```

The VARS command defines which parameters are to be used in an optimization and their data range. Each parameter in a project must have an entry line which starts with VAR. This is followed by the name of the parameter, <varname>. The <varname> is followed by an N if it is not used in the optimization. If the parameter is used in the optimization this field is Y. The default is N. <minval> is the minimum value of the parameter and <maxval> is the maximum value. If values have not been entered both fields are UNDEF. The `granularity` field is set to UNDEF if Auto is entered. If a manual value has been entered for granularity this is a floating point number. The END statement indicates the end of the list of variables.

Optimization Goals

An optimization can have one or more goals. Each goal consists of a frequency sweep statement followed by an error statement.

The following eight keywords are all the types of sweeps available with an optimization. There is some type of sweep statement associated with each optimization goal over which error statements are calculated.

SWEEP Linear frequency sweep with stated interval. For details see the SWEEP keyword in the "Frequency Block" on page 46.

ESWEEP Exponential frequency sweep. For details see the ESWEEP keyword in the "Frequency Block" on page 46.

LSWEEP Linear frequency sweep with number of points. For details see the LSWEEP keyword in the "Frequency Block" on page 46.

STEP Discrete analysis frequencies. For details see the STEP keyword in the "Frequency Block" on page 46.

ABS Adaptive Band Synthesis Sweep. For details see the ABS keyword in the "Frequency Block" on page 46.

ABS_FMIN Find the minimum frequency response. For details see the ABS_FMIN keyword in the "Frequency Block" on page 46.

ABS_FMAX Find the maximum frequency response. For details see the ABS_FMAX keyword in the "Frequency Block" on page 46.

DC_FREQ Analyze at a DC frequency point. For details see the DC_FREQ keyword in the "Frequency Block" on page 46.

NET Optimization Goal Error Statement

Syntax

```
NET=[GEO|netname] restype[respar] rel tartype tarvalue
weight
```

There is a net statement for each optimization goal entered in the Analysis Setup. If the project is a geometry project NET=GEO is used. NET=GEO also identifies the main network of a netlist. For a netlist project, <netname> is the name of the network whose response you wish to use. <restype> is the response type: choices are DB, ANG, MAG, RE and IM. <respar> is the response. <respar> consists of either S, Y, or Z followed by a pair of port indices. For example, S_{24} would be S24. If one of the two port indices has more than one digit, you use an underscore to separate the two port indices. For example, S10_27. <rel> is the relationship between the response and target: choices include "=", "<" or ">". <tartype> is the type of target: choices include VALUE, NET, and FILE.

If the <tartype> is VALUE then the <tarvalue> is a floating point number which specifies the desired target value.

If <tartype> is FILE it is followed by a pathname for the file using this syntax: "`<pathname>`". In this case, <tarvalue> is a response specified using the syntax of `restype[respar]` as described above.

If <tartype> is NET then tarvalue is specified using the "`NET[GEO|netname] restype[respar]`" specified above.

END End statement

Syntax

```
END OPT
```

Indicates the end of the optimization block. Required.

Parameter Sweep Block

The parameter sweep block specifies a frequency sweep and data range for each parameter sweep. This block is only used when the control block contains the VARSWP entry.

VARSWP Beginning of parameter sweep block

Syntax

```
VARSWP
sweeptype(1) sweepparameters
VAR parameter(1) [N|Ytype] min max step
VAR parameter(2) [N|Ytype] min max step
.
.
END
.
.
sweeptype(n) sweepparameters
VAR parameter(n) [N|sweeptype] min max step
.
```

```
.
END
END VARSWP
```

Indicates the beginning of the parameter sweep block. All statements following this entry are included in the parameter sweep block until you reach the END VARSWP statement. The statements in this block are generated in the Analysis Setup dialog box.

The sweeptype entry defines a parameter sweep. There is a sweeptype entry for each parameter sweep defined in the Analysis Setup dialog box. Each sweeptype statement defines the frequency sweep. After each sweep statement there is a line for each parameter defined in the project.

The sweeptype entry can be any of the Frequency Control options followed by its parameters. The available entries are SWEEP, ESWEEP, LSWEEP, STEP, ABS_ENTRY, DC_FREQ. For details about each sweep type and its parameters, see "Frequency Block" on page 46.

There is a VAR entry for each parameter defined in your project. The VAR keyword is followed by the name of the parameter. Then next field is N if the parameter is not used in the parameter sweep. In this case, the nominal value of the parameter is used for all analysis frequencies. If the next field is a `Ytype` value, then the parameter is used in the parameter sweep. The table below shows the possible values for Ytype and their definitions.

Ytype	Definition
Y	Linear sweep
YN	Linear sweep # steps
YC	Corner Sweep
YS	Sensitivity Sweep
YE	Exponential Sweep

<min> is a floating point number which is the minimum value of the parameter. <max> is a floating point number which defines the maximum value of the parameter. <step> is a floating point number which defines the interval between parameter values.

END End statement which indicates the end of the sweep entry. There is one END statement for each sweeptype entry.

END VARSWP End statement

Syntax `END VARSWP`

Indicates the end of the parameter sweep block. Required.

For example, you have a project with two variables, Lstub and Sstub and have defined two parameters sweeps. The first sweep is an ABS sweep from 2.0 to 10.0 GHz with Lstub varying from 120 to 280 mils in steps of 160.0 mils. Sstub is not used in this parameter sweep. The second sweep is an ABS sweep from 5.0 to 7.0 GHz, with Sstub varying from 180.0 mils to 240.0 mils in steps of 10.0 mils. The entry for these two parameter sweeps is annotated below.

```
VARSWP ;Start of parameter sweep block
ABS_ENTRY 2.0 10.0 ;ABS entry for first sweep
VAR Lstub Y 120.0 280.0 160.0 ;Lstub range defined for first sweep
VAR Sstub N 220.0 220.0 UNDEF ;Sstub entry, not used in this parameter sweep
END ;end of first parameter sweep
ABS_ENTRY 5.0 7.0 ;ABS entry for second sweep
VAR Lstub N 220.0 220.0 UNDEF ;Lstub entry not used in this parameter sweep
VAR Sstub Y 180.0 240.0 10.0 ;Sstub range defined for second sweep
END ;end of second parameter sweep
END VARSWP ;End of parameter sweep block
```

Output File Block

The output file block specifies an output file which allows you to store response data from your analysis outside your project file. After the initial FILEOUT statement in the beginning of the block, each line specifies an output file. These files are specified in the Output Files dialog box.

FILEOUT Beginning of output file block

Syntax `FILEOUT`

Indicates the beginning of the output file block. All statements following this entry are included in the output file block until you reach the END FILEOUT statement. The statements in this block are generated in the Generate Default Output Files dialog box. Each output file has its own entry. There are five different syntaxes for the output file; response data, PI Spice, N-coupled Line Spice, Broadband Spice. and Inductor model. All are detailed below.

Response `filetype [NET=network] embed ABS_inc filename comments sig partype parform ports`

`filetype` The following options are available for this field.

filetype entry	Definition
TS	Touchstone
TOUCH2	Touchstone v2
DATA_BANK	Databank
SC	SCompact
CSV	Spreadsheet
CADENCE	Cadence
MDIF	MDIF (S2P)
EBMDIF	MDIF (ebridge)

`network` The NET=network is omitted in a geometry project. For a netlist project, network is the network for which you wish to export data.

`embed` This field is “`D`” for de-embedded data or “`ND`” for non-de-embedded data.

`ABS_inc` This field is “Y” to include the ABS adaptive data or “N” to include only the discrete data. Default is to include the adaptive data.

`filename` The filename consists of a basename and extension. If the basename of the project file is used, the variable "`$BASENAME`" may be substituted in the filename. For example, in the project file steps.son if an output file steps.s2p is entered, the filename would appears as "`$BASENAME.s2p`" in the fileout block. The user may enter any filename they wish and are not restricted in their use of extensions.

`comments` This field is "NC" for no comments or "IC" to include comments.

`sig` This value is "15" if High precision is on and "8" if High Precision is not selected.

`partype` This field is "`S`" for S-Parameters, "`Y`" for Y-Parameters, and "`Z`" for Z-Parameters.

`parform` The following options are available for this field:

parform entry	Definition
`MA`	Mag-Angle
`DB`	DB-Angle
`RI`	Real-Imag

`ports` There are three different syntaxes for port information for output files.

If all ports in the circuit use real impedance with the same resistance and all other values 0, then <ports> is as follows:

```
R resist
```

where <resist> is a floating point number for the resistance.

If all ports in the circuit use complex impedance with the same resistance and all other values 0, then <ports> is as follows:

```
Z rresist iresist
```

where <rresist> is a floating point number for the real part of the resistance and <irresist> is a floating point number for the imaginary part of the resistance.

```
TERM resist(1) react (1) resist(2) react(2) ... resist(n) react(n)
```

where <`resist(1)`> is a floating point number for the resistance of the first port in the circuit, <`react(1)`> is a floating point number for the reactance of the first port in the circuit. Pairs of values, for resistance and reactance are repeated for each port in the circuit. If the number of ports is large, the continuation character (&) is used for additional lines that are part of this file's specification.

If a port or ports in the circuit have a non-zero value for either the inductance or capacitance, then each port displays four values using the following syntax:

```
FTERM resist(1) react(1) induct(1) cap(1) ... resist(n) react(n) induct(n) cap(n)
```

where <`resist(1)`> is a floating point number for the resistance of the first port in the circuit, <`react(1)`> is a floating point number for the reactance of the first port in the circuit, <`induct(1)`> is a floating point number for the inductance of the first port in the circuit, and <`cap(1)`> is a floating point number for the capacitance of the first port in the circuit. Four values, for resistance, reactance, inductance and capacitance are repeated for each port in the circuit. If the number of ports is large, the continuation character (&) is used for additional lines that are part of this file's specification.

This completes the syntax for the file specifications for response files.

PIMODEL This is the entry for an PI Model Spice file which is specified in the PI Model File Entry dialog box.

Syntax

```
PIMODEL [NET=network] embed ABS_inc filename comments sig
PINT=pint RMAX=rmax CMIN=cmin& LMAX=lmax KMIN=kmin
RZERO=rzero format
```

`network` The NET=network is omitted in a geometry project. For a netlist project, <network> is the network for which you wish to export data.

`embed` This field is "`D`" for de-embedded data or "`ND`" for non-de-embedded data.

`ABS_inc` This field is "Y" to include the ABS adaptive data or "N" to include only the discrete data. Default is to include the adaptive data.

`filename` The filename consists of a basename and extension. If the basename of the project file is used, the variable "`$BASENAME`" may be substituted in the filename. For example, in the project file steps.son if an output file steps.lib is entered, the filename would appears as "`$BASENAME.lib`" in the fileout block. The user may enter any filename they wish and are not restricted in their use of extensions.

`sig` This value is "15" if High precision is on and "8" if High Precision is not selected.

`pint` This is a floating point number for the percentage used to determine the intervals between the two frequencies used to determine each SPICE model.

`rmax` This is a floating point number for the maximum allowed resistance (Rmax). Default value is 1000.0

`cmin` This is a floating point number for the minimum allowed capacitance (Cmin). Default value is 0.1.

`lmax` This is a floating point number for the maximum allowed inductance (Lmax). Default value is 100.0.

`kmin` This is a floating point number for the minimum allowed mutual inductance (Kmin). Default value is 0.01.

`rzero` This is a floating point number for the resistor to go in series with all lossless inductors (Rzero). Default value is 0.0.

`format` This is the format for the PI Model output file. This field is "PSPICE" for PSpice and "SPECTRE" for Spectre.

NCLINE This is the entry for an N-coupled line Spice model file which is specified in the N-coupled Line Model File Entry dialog box.

Syntax `NCLINE [NET=network] embed ABS_inc filename comments sig format`

`network` The NET=network is omitted in a geometry project. For a netlist project, <network> is the network for which you wish to export data.

`embed` This field is "`D`" for de-embedded data or "`ND`" for non-de-embedded data.

`ABS_inc` This field is "Y" to include the ABS adaptive data or "N" to include only the discrete data. Default is to include the adaptive data.

`filename` The filename consists of a basename and extension. If the basename of the project file is used, the variable "`$BASENAME`" may be substituted in the filename. For example, in the project file steps.son if an output file steps.lib is entered, the filename would appears as "`$BASENAME.lib`" in the fileout block. The user may enter any filename they wish and are not restricted in their use of extensions.

`sig` This value is "15" if High precision is on and "8" if High Precision is not selected.

`format` This is the format for the N-coupled Line model output file. This field is "PSPICE" for PSpice and "SPECTRE" for Spectre.

BBEXTRACT This is the entry for a Broadband Spice Model which is specified in the Broadband Spice Model File Entry dialog box. The entry specified above is followed by an OPTIONS block.

Syntax `BBEXTRACT [NET=network] embed ABS_inc filename comments sig format`

`network` The NET=network is omitted in a geometry project. For a netlist project, <network> is the network for which you wish to export data.

`embed` This field is "`D`" for de-embedded data or "`ND`" for non-de-embedded data.

`ABS_inc` This field is "Y" to include the ABS adaptive data or "N" to include only the discrete data. Default is to include the adaptive data.

`filename` The filename consists of a basename and extension. If the basename of the project file is used, the variable "`$BASENAME`" may be substituted in the filename. For example, in the project file steps.son if an output file steps.lib is entered, the filename would appears as "`$BASENAME.lib`" in the fileout block. The user may enter any filename they wish and are not restricted in their use of extensions.

`sig` This value is "15" if High precision is on and "8" if High Precision is not selected.

`format` This is the format for the BBExtract Model output file. This field is "PSPICE" for PSpice or HSPICE and "SPECTRE" for Spectre.

OPTIONS Options block for Broadband Spice Model specification. These options are specified in the Broadband Model File Entry dialog box and the Advanced Broadband Model Options dialog box.

Syntax `OPTIONS`

ErrThresh Error Threshold for Broadband Spice Model

Syntax `ErrThresh threshold`

`threshold` This field is a floating point value used for the error threshold.

TotalOrder The maximum order of the rational polynomial which produces the Broadband Spice Model.

Syntax `TotalOrder order`

`order` This field is an integer value used as the maximum order.

CurveFit Generate Predicted S-parameter data file

Syntax `CurveFit Y|N`

This field indicates whether a Predicted S-parameter data file is generated when you create the Broadband Spice model. 'Y" is for yes and "N" is for no.

DCPoint N Generate data at a DC Point for the Predicted S-parameter data

Syntax `DCPOINT Y|N`

This field indicates whether analysis data at a DC Point should be included in the predicted S-Parameter data. “Y” is for yes and “N” is for no.

SingleFile Generate a file for each parameter combination.

Syntax `SingleFile Y|N`

This field indicates whether a separate output file is created for each parameter combination or if a single output file contains the data for all parameter values. “Y” is for yes and “N” is for no.

INDMODEL This is the entry for a Inductor Model which is specified in the Inductor Model File Entry dialog box. The entry specified above is followed by an OPTIONS block.

Syntax

```
INDMODEL embed ABS_inc filename comments sig format
gen_data reserve modeltype
```

`embed` This field is “`D`” for de-embedded data or “`ND`” for non-de-embedded data.

`ABS_inc` This field is always “Y” to include the ABS adaptive data or “N” to include only the discrete data. Default is to include the adaptive data.

`filename` The filename consists of a basename and extension. If the basename of the project file is used, the variable “`$BASENAME`” may be substituted in the filename. For example, in the project file steps.son if an output file steps.scs is entered, the filename would appears as “`$BASENAME.scs`” in the fileout block. The user may enter any filename they wish and are not restricted in their use of extensions.

`sig` This value is “15” if High precision is on and “8” if High Precision is not selected.

`format` This is the format for the InductorModel output file. This field is “PSPICE” for PSpice or HSPICE ,“SPECTRE” for Spectre and “NETLIST” for Netlist.

`gen_data`	This field indicates whether a predicted S-Parameter data file should also be generated. Set to "Y" for yes and "N" for No.
`reserve`	This field is reserved for future use and should always be set to "N."
`modeltype`	Indicates the what type of inductor model is being generated: Untapped or Center Tapped. Set to "SKIN_EFFECT" for Untapped and "CENTER_TAP" for Center Tapped.
FREQBAND	Entry which determines the frequency band over which the inductor model will be generated.
Syntax	`FREQBAND source start stop`
`source`	Determines if the frequency band is automatically generated by the software or input by the user. Set to "AUTO" to have the software generate the band. The `start` and `stop` field do not appear for this setting. Set to "CUSTOM" to use values input by the user.
`start`	Floating point number for the starting frequency of the bandwidth. Only appears if `source` is set to "CUSTOM."
`stop`	Floating point number for the ending frequency of the bandwidth. Only appears if `source` is set to "CUSTOM."
OPTIONS	Options block for Inductor Model specification. There are presently no options so only the keyword "OPTIONS" needs to appear.
Syntax	`OPTIONS`
END	Indicates the end of the Inductor Model block.
Syntax	`END`
FUNIT	Frequency units
Syntax	`FUNIT unit`
	This field defines the frequency units used when requesting extra frequencies be output to the Predicted S-parameter data. "unit" is the frequency units being used. This unit is always the same as specified for the project file. Possible values are Hz, KHz, MHz, GHZ, THz,

FOLDER Identifies the output directory in which the output files are created.

Syntax `FOLDER pathname`

This field identifies the directory in which the output files are created. The pathname is relative to the project source directory.

END End statement

Syntax `END OPTIONS`

Indicates the end of the Options block for the Broadband Spice Model file. Required

END End statement

Syntax `END FILEOUT`

Indicates the end of the output file block. Required.

Parameter Block for Netlist Project

The parameter block specifies parameters for a netlist project. After the initial VAR statement in the beginning of the block, each line specifies a parameter and its nominal value. The end of the VAR block is indicated by the END VAR statement.

VAR Parameters for a Netlist Project

Syntax

```
VAR
parname = nomvalue
```

Indicates the beginning of the parameter block in a netlist project. There is an entry line for each parameter defined in a netlist project. The END VAR statement indicates the end of the parameter block.

`parname` A character string which provides the name of the parameter.

`nomvalue` A floating point number which is the nominal value for the parameter.

END End statement

Syntax `END VAR`

Indicates the end of the VAR block. Required.

Circuit Block for Netlist Project

The circuit block specifies all the elements and networks in a netlist project. These entries correspond to the netlist which appears in the netlist editor window. This is equivalent to the GEO block of a geometry project.

CKT Circuit elements and netlist

Syntax `CKT`

Indicates the beginning of the circuit block in a netlist project. The END CKT statement indicates the end of the parameter block.

RES Resistor Element

Syntax `RES nodenum1 [nodenum2] R = resvalue`

The RES statement defines a resistor element in your netlist circuit. The <nodenum1> is the first, and possibly only, node number in the net to which the resistor is connected. <nodenum2> only appears if the resistor is connected between two nodes in the network, and is the second node to which the resistor is connected. The resistor is included in the first network whose entry (DEFnP) appears after the RES statement in the file. <resvalue> is the value of the resistor and can either be a floating point number or a character string representing a parameter defined in the netlist project.

IND Inductor Element

Syntax `IND nodenum [nodenum2] L = indvalue`

The IND statement defines an inductor element in your netlist circuit. The <nodenum1> is the first, and possibly only, node number in the net to which the inductor is connected. <nodenum2> only appears if the inductor is connected between two nodes in the network, and is the second node to which the inductor is connected. The inductor is included in the first network whose entry (DEFnP) appears after the IND statement in the file. <indvalue> is the value of the inductor and can either be a floating point number or a character string representing a parameter defined in the netlist project.

CAP Capacitor Element

Syntax

```
CAP nodenum [nodenum2] C = capvalue
```

The CAP statement defines a capacitor element in your netlist circuit. The <nodenum1> is the first, and possibly only, node number in the net to which the capacitor is connected. <nodenum2> only appears if the capacitor is connected between two nodes in the network, and is the second node to which the capacitor is connected. The capacitor is included in the first network whose entry (DEFnP) appears after the CAP statement in the file. <capvalue> is the value of the capacitor and can either be a floating point number or a character string representing a parameter defined in the netlist project.

TLIN Transmission Line element

Syntax

```
TLIN prt1nd prt2nd Z=imped E=length F=freq
```

The TLIN statement defines a transmission line element in your netlist circuit. The transmission line is included in the first network whose entry (DEFnP) appears after the TLIN statement in the file. <prt1nd> is an integer number; port 1 of the transmission line is connected to this node in the netlist. <prt2nd> is an integer number; port 2 of the transmission line is connected to this node in the netlist. <imped> is a floating point number used for the impedance of the transmission line. <length> is a floating point number used for the electrical length in degrees of the transmission line. <freq> is a floating point number which is the frequency for the transmission line.

TLINP Physical Transmission Line element

Syntax

```
TLINP prt1nd prt2nd Z=imped L=length K=eeff F=freq
A=atten
```

The TLINP statement defines a physical transmission line element in your netlist circuit. The physical transmission line is included in the first network whose entry (DEFnP) appears after the TLINP statement in the file. <prt1nd> is an integer number; port 1 of the transmission line is connected to this node in the netlist. <prt2nd> is an integer number; port 2 of the transmission line is connected to this node in the netlist. <imped> is a floating point number used for the impedance of the transmission line. <length> is a floating point number used for the length of the transmission line. <freq> is a floating point number which is the frequency for the transmission line. <atten> is a floating point number which is the attenuation (dB/length unit) of the physical transmission line.

SnP Data Response file element

Syntax
```
SnP prtnode(1) prtnode(2) .. prtnode(n) [gndnode] filename
```

The SnP entry defines a data response file element in your netlist circuit. The response file is included in the first network whose entry (DEFnP) appears after the SnP statement. The keyword changes depending upon the number of ports in the data file. A response file for a 2 port circuit would use the keyword S2P. A response file for a 4 port circuit would use the keyword S4P. A node number, <prtnode>, appears for each port in the data file. If there are four ports in a circuit, then there will be four integers representing the nodes to which the ports are connected. <gndnode> is optional. If the ground for the response file is connected to GND than this field is omitted. However, if ground for the response file is connected to a node, than that node appears here. Last in the statement is a character string, <filename>, identifying the response file you wish to include. This string may include an absolute or relative path.

PRJ Project File Element

Syntax
```
PRJ prt1node .. prt(n)node filename numprt swpcon
[parameter = parvalue]
```

The PRJ entry defines a subproject element in your netlist circuit. The subproject is included in the first network whose entry (DEFnP) appears after the PRJ statement. A node number, <prt(n)node>, appears for each port in the project. If there are four ports in a circuit, then there will be four integers representing the nodes to which the ports are connected. <gndnode> is optional. If the ground for the project is connected to GND than this field is committed. However, if ground for the project is connected to a node, than that node appears here. Next in the statement is a character string, <filename>, identifying the subproject you wish to include. This string may include an absolute or relative path. <numprt> is an integer value for the number of ports in the project. <swpcon> is 0 to indicate that you use the sweep from this project or 1 to indicate that you use the sweep from the subproject. This setting is overridden if Hierarchy Sweep is on. If the subproject contains any parameters, then there is a `<parameter>=<parvalue>` entry for each parameter. <parameter> is a character string which is the name of the parameter. <parvalue> is a floating point number or a character string representing a parameter defined in the netlist project.

DEFnP Network or Network Element

Syntax `DEFnP prtnode(1) prtnode(2) .. prtnode(n) netname ports`

The DEFnP entry defines a network element or main network in your netlist circuit. The main network is identified by the last DEFnP statement which occurs in the project file. The keyword changes depending upon the number of ports in the network. A network for a 2 port circuit would use the keyword DEF2P. A network for a 4 port circuit would use the keyword DEF4P. A node number, <prtnode>, appears for each port in the network. If there are four ports in a network, then there will be four integers representing the nodes to which the ports are connected. Last in the statement is a character string, <netname>, identifying the network. In the case of a network element, this string may include an absolute or relative path. <ports> defines the port terminations for the network. The syntax is detailed below.

`ports` There are three different syntaxes for port information for output files.

If all ports in the circuit use real impedance with the same resistance and all other values 0, then <ports> is as follows:

```
R resist
```

where <resist> is a floating point number for the resistance.

If all ports in the circuit use complex impedance with the same resistance and all other values 0, then <ports> is as follows:

```
Z rresist iresist
```

where <rresist> is a floating point number for the real part of the resistance and <irresist> is a floating point number for the imaginary part of the resistance.

If a port or ports in the circuit have a non-zero value for the reactance, then each port displays two values using the following syntax:

```
TERM resist(1) react (1) resist(2) react(2) ... resist(n) react(n)
```

where `<resist(1)>` is a floating point number for the resistance of the first port in the circuit, `<react(1)>` is a floating point number for the reactance of the first port in the circuit. Pairs of values, for resistance and reactance are repeated for each port in the circuit. If the number of ports is large, the continuation character (&) is used for additional lines that are part of this file's specification.

If a port or ports in the circuit have a non-zero value for either the inductance or capacitance, then each port displays four values using the following syntax:

```
FTERM resist(1) react(1) induct(1) cap(1) ... resist(n) react(n) induct(n) cap(n)
```

where `<resist(1)>` is a floating point number for the resistance of the first port in the circuit, `<react(1)>` is a floating point number for the reactance of the first port in the circuit, `<induct(1)>` is a floating point number for the inductance of the first port in the circuit, and `<cap(1)>` is a floating point number for the capacitance of the first port in the circuit. Four values, for resistance, reactance, inductance and capacitance are repeated for each port in the circuit. If the number of ports is large, the continuation character (&) is used for additional lines that are part of this file's specification.

This completes the syntax for the file specifications for response files.

END End statement

Syntax `END CKT`

Indicates the end of the circuit block. Required.

Subdivider Block for Geometry Project

The subdivider block specifies the subdividers and resulting filenames for a geometry project. After the initial SUBDIV statement in the beginning of the block, entries specify the output file names and subdivider's positions. The end of the subdivider block is indicated by the END SUBDIV statement.

MAIN Main netlist name

Syntax `MAIN filename`

This entry defines the name for the resulting netlist file produced from performing the subdivide on this geometry project. This is the name entered in the Circuit Subdivision dialog box which appears when you select the *Tools* ⇒ *Subdivide Circuit* command. The filename is a character string which identifies the main netlist. This string may include an absolute or relative path. The default for this filename is $Basename_net.son.

REFPLANE Reference Planes for Subprojects

Syntax REFPLANE reftype reflength

This statement defines the reference planes added to the geometry subprojects which result from performing the subdivide. This selection is made in the Subproject Specification dialog box. The reftype is the type of reference plane you wish to add to the geometry subprojects. The options and meanings are shown in table below.

reftype	Meaning
A	Automatic - Suggested Length used
F	Fixed - Entered length used
N	None - No reference planes added

<reflength> is a floating point number which is the length of the automatically added reference planes. If the reftype is set to “N” then reflength is equal to zero.

NAME Geometry Subproject Name

Syntax `NAME secnum prjname`

The NAME statement defines the name for a geometry subproject created when a subdivide is performed. There is a name statement for each geometry subproject created. A geometry subproject is created for each section of the geometry produced when the subdividers are added. These names are entered in the Subproject Specification dialog box.

<secnum> is the integer value for the section number from which the geometry subproject is produced. secnum always starts at a value of 1 and increments for each NAME statement. The <prjname> is a character string which contains the name for the resultant geometry subproject. This string may include an absolute or relative path. The default is $BASENAME_net_s(secnum).son. For example, for section 3 of the file spiral.geo, the <prjname> would be $BASENAME_net_s3.son.

LINE Subdivider Line Location

Syntax `LINE linenum coordinate dir`

The LINE statement defines the location of a subdivider in the project. There is a LINE statement for each subdivider in a geometry project. The <linenum> is an integer number which identifies the subdivider. <linenum> starts at a value of 1 for each geometry file and increases sequentially for each subdivider. The <coordinate> is a floating point number which indicates the position of the subdivider on the substrate. If the subdivider is vertical, indicated by “`V`” in the dir field, then the subdivider occurs at this distance from the origin in the x (horizontal) direction. If the subdivider is horizontal, indicated by “`H`” in the <dir> field, then the subdivider occurs at this distance from the origin in the y (vertical) direction. The direction for all subdividers in a circuit must be the same. The origin (0,0) is located at the upper left hand corner of the substrate.

Quick Start Guide Block for a Geometry Project

The QSG block is used to specify the settings in the Quick Start Guide. The following seven lines specify which tasks in the Quick Start Guide have been done by the user. Each line corresponds to an entry in the Quick Start Guide. If the task is done a YES appears after the keyword. If the task has NOT been done, a NO appears after the keyword.

QSG Indicates the beginning of the Quick Start Guide block.

Syntax `QSG`

This statement is followed by statements detailed below which provide the state of entries in the Quick Start Guide. All the entries in this block are required.

IMPORT Specifies if a DXF or GDS import has been performed in the project editor.

Syntax `IMPORT [Y|N]`

A “Y” appears if an import has been done. A “N” appears if no import has been down.

EXTRA_METAL Specifies if extra metal has been removed from the circuit.

Syntax `EXTRA_METAL [Y|N]`

UNITS Specifies if the user has changed the units used in the project.

Syntax `UNITS [Y|N]`

A “Y” appears if the units have been changed. A “N” appears if the units have not been changed.

ALIGN Specifies if the user has aligned the circuit to the grid.

Syntax `ALIGN [Y|N]`

A “Y” appears if the circuit has been aligned to the gird. A “N” appears if the circuit has not been aligned.

REF Specifies if reference planes have been added to the circuit.

Syntax `REF [Y|N]`

A “Y” appears reference planes have been added to the circuit. A “N” appears if no reference planes have been added.

VIEW_RES Specifies if the user had viewed response data.

Syntax `VIEW_RES [Y|N]`

A “Y” appears if the user has viewed response using the response viewer, the current density viewer or the far field viewer. A “N” appears if the user has not viewed response.

METALS Specifies if the user has defined any new metal types.

Syntax `METALS [Y|N]`

A “Y” appears if the user has defined any new metal types. A “N” appears if no metal types have been defined.

METALS Specifies if the user has defined any new metal types.

Syntax `METALS [Y|N]`

A “Y” appears if the user has defined any new metal types. A “N” appears if no metal types have been defined.

USED Indicates if the Quick Start Guide is enabled for this project.

Syntax `USED [Y|N]`

A “Y” appears if the Quick Start Guide is being displayed. A “N” appears if the Quick Start Guide is not being displayed.

END QSG Indicates the end of the Quick Start Guide Block.

Syntax `END QSG`

This indicates the end of the Quick Start Guide block.

Component Data Files Block

The Component Data files block is used to specify the response data files associated with data file type components (SPARAM). Each line corresponds to data file and assigns an index number to the file that is used in the component definition specified in the SMD keyword. For more details about the SMD keyword, please see "SMD" on page 40.

SMDFILES Indicates the beginning of the Component Data Files block

Syntax

```
SMDFILES
```

This statement is followed by an entry for each data file used by a component.

Data File Specifies a response data file used by a component and assigns an index number.

Syntax

```
<indexnumber> "<filename>"
```

<indexnumber> is a non-zero positive integer. If the data file is used by more than one component, only one index number is assigned to that file. The same index number is used in the different component “SMD” entries. <filename> specifies the location of the response data file. Relative paths are used.

END SMDFILES Indicates the end of the Component Data Files Block.

Syntax

```
END SMDFILES
```

This indicates the end of the Component Data Files block.

Translators Block

The Translators block is used to specify the options for a DXF, GDSII or Gerber export of your project. These entries control the defaults for the Export Options dialog box in the project editor. Setting flags here controls the properties in this dialog box. This allows you use the same settings each time you export this project. All entries are optional and only appear in the project file if an export command has been selected. There may be up to three transtype statements followed by options; one for each of the translators.

TRANSLATOR Indicates the beginning of the Translators block

Syntax `TRANSLATOR`

transtype Indicates the type of export being preformed.

Syntax [DXFEXPORT|GDSEXPORT|GERBEREXPORT]

Indicates to which translator the options that follow apply. This field is DXFEXPORT for a DXF export, GDSEXPORT for a GDSII export and GERBEREXPORT to export a Gerber file.

SepObj Indicates that a separate layer should be created for each object type (metal polygons, via polygons or dielectric bricks).

Syntax SepObj [TRUE|FALSE]

If this option is selected in the project editor, then the keyword is followed by `TRUE`. If the option is not selected, the field is set to `FALSE`.

SepMat Indicates that a separate layer should be created for each material type (metal type or dielectric brick material). For example if the project uses two different metal types, a separate layer is created for each metal type.

Syntax SepMat [TRUE|FALSE]

If this option is selected in the project editor, then the keyword is followed by `TRUE`. If the option is not selected, the field is set to `FALSE`.

DivideMulti Indicates that vias which extend over multiple metal levels are divided into multiple vias, each of which span only one level. For example, if you have a via which extends downward from Level 0 to level 2, then two vias are created during the translation. One via extends from Level 0 to Level 1 and the other extends from Level 1 to Level 2.

Syntax `DivideMulti [TRUE|FALSE]`

If this option is selected in the project editor, then the keyword is followed by `TRUE`. If the option is not selected, the field is set to `FALSE and vias are translated without being divided`.

Circles Indicates that vias in your project are converted to circular vias in your exported file. converts the vias in your Sonnet project to circular vias in your DXF output. If you select the Auto radio button, then the circle is the biggest that can fit in the bounding box of the via polygon

Syntax `Circles [TRUE|FALSE]`

If this keyword is set to TRUE, then the **Convert Vias to Circles** option is selected in the project editor. If the keyword is set to `FALSE`, then the option is not selected and vias are translated without the shape being changed.

CircleType Indicates whether the circular via is created automatically by the software or manually based on user inputs. This entry is only used is the Circles entry is set to TRUE.

Syntax `CircleType cirtype`

If the size of the circular via is determined automatically, then <cirtype> is set to `inscribed`. In this case, the project editor creates the circular via based on the largest circle which can be inscribed in the bounding box of the original via. If the size of the circular via is manually input by the user, then <cirtype> is set to `manual` and the size of the circular via is determined by the CircleSize entry (see below.)

CircleSize Indicates the radius of the translated circular vias. Please note that if you enter a manual radius, then ALL the vias in your circuit are converted to circular vias of that size.

Syntax `CircleSize size`

Size is the desired radius of the translated circular vias. This entry is only used if the Circles entry is set to TRUE and the Circletype is set to `manual`. The units are the present Length units being used in the project.

KeepMetals Indicates whether metal polygons should be translated into the exported file.

Syntax KeepMetals [TRUE|FALSE]

If this keyword is set to TRUE, then the **Metal Polygons** option is selected in the project editor. If the keyword is set to FALSE, then the option is not selected and metal polygons are not exported.

KeepVias Indicates whether via polygons should be translated into the exported file.

Syntax KeepVias [TRUE|FALSE]

If this keyword is set to TRUE, then the **Via Polygons** option is selected in the project editor. If the keyword is set to FALSE, then the option is not selected and via polygons are not exported.

KeepViaPads Indicates whether via pads should be used for exported vias.

Syntax KeepViaPads [TRUE|FALSE]

If this keyword is set to TRUE, then the **Via Pad** option is selected in the project editor. If the keyword is set to FALSE, then the option is not selected and via pads are not added to the exported via polygons. This statement is only used if `Keepvias` is `TRUE`.

KeepBricks Indicates whether dielectric bricks should be translated into the exported file.

Syntax KeepViaPads [TRUE|FALSE]

If this keyword is set to TRUE, then the **Brick Polygons** option is selected in the project editor. If the keyword is set to FALSE, then the option is not selected and dielectric bricks are not exported.

KeepParent Indicates whether a metal polygon with an edge via is translated to the exported file.

Syntax `KeepParent [TRUE|FALSE]`

If this keyword is set to `TRUE`, then the **Metal Polygons with Edge Vias** option is selected in the project editor and metal polygons are translated into the export file. If the keyword is set to `FALSE`, then the option is not selected and metal polygons with edge vias are not translated.

NOTE: **The KeepEdgeVia, ConvertParent and AllEdgeViasAsVia keywords are mutually exclusive. Only one should be set true at a time. If all are** `TRUE` **ConvertParent overrides the other keywords. If ConvertParent is** `FALSE` **and the other two are** `TRUE,` **then AllEdgeViasAsVia overrides the KeepEdgeVia keyword.**

KeepEdgeVias Indicates whether edge vias should be translated into the exported file.

Syntax `KeepEdgeVias [TRUE|FALSE]`

If this keyword is set to TRUE, then the **Output edge vias as one cell wide via** option is selected in the project editor. If the keyword is set to FALSE, then the option is not selected, and edge vias are not exported.

ConvertParent Indicates whether a polygon with an edge via should be translated into a via polygon the same size and shape as the metal polygon to which the edge via is attached.

Syntax `ConvertParent[TRUE|FALSE]`

If this keyword is set to TRUE, then the **Convert polygons with edge vias to vias** option is selected in the project editor, and a via polygon the same size and shape as the original metal polygon to which the edge via was attached is exported.

AllEdgeViasAsVia Indicates that if a metal polygon has edge vias on every side, then the metal polygon is converted to a via polygon in the exported file. If there are edge vias only on some sides, then the metal polygon is exported and the edge vias are converted individually.

Syntax `AllEdgeViasAsVia[TRUE|FALSE]`

If this keyword is set to TRUE, then the **If all polygon edges have vias, convert polygon to a via otherwise convert edge vias individually** option is selected in the project editor and the metal polygon is output as a via polygon, if not overridden by another option. See the note on page 83.

GbrWholeDigits Indicates the number of whole digits you wish to use for length values in the Gerber out files. The whole number portion is those digits to the left of the decimal place. This option only applies to Gerber exports.

Syntax `GbrWholeDigits numwhole`

The GbrWholeDigits statement defines the number of whole digits you wish to use for length values in the Gerber output file in which `numwhole` is an integer number which defines the number of digits.

GbrDecimalDigits Indicates the number of decimal digits you wish to use for length values in the Gerber out files. The decimal number portion is those digits to the right of the decimal place. This option only applies to Gerber exports.

Syntax `GbrDecimalDigits numpart`

The GbrDecimalDigits statement defines the number of decimal digits you wish to use for length values in the Gerber output file in which `numpart` is an integer number which defines the number of digits.

GbrFilenamePrefix The GbrFilenamePrefix statement is one of three options that determine how to name the exported Gerber Files. This option allows you to define a Prefix such that all Gerber files created in the translation start with that prefix. Please refer to the Gerber Export Options dialog box in Help for a detailed explanation of the files generated by a Gerber export.

Syntax GbrFilenamePrefix Customprefix

When the option **Custom Prefix** is selected, then the exported files are named <Customprefix>_#. gbr where <Customprefix> is a character string entered by the user for the prefix. # is an integer value that uniquely identifies each file. For example, for the custom prefix "galaxy_" the Gerber output with project name selected would be as follows:

SONNET_TOP: galaxy_1.gbr
METAL_Lossless_0: galaxy_2.gbr
SONNET_BOT: galaxy_3.gbr

The prefix is only used if `GbrFilenameType` is set to “custom.”

GbrFilenameExt This statement defines the file extension used for all the Gerber files created by the export.

Syntax

```
GbrFilenameExt Extension
```

This statement defines the file extension for all files created by the Gerber export. `Extension` is a character string entered by the user.

GbrFilenameType This statement defines the naming convention for the Gerber files output by the export.

Syntax

```
GbrFilenameType nametype
```

The GbrFilenameType determines how the output Gerber files are named. The `nametype` field determines the basename for all the Gerber files out put from the export. There are three possible character string values for `nametype`: `default`, `project` or `custom`.

If nametype is set to default, then the default names, detailed in the table below are used for the files created by the export.

Layer Type	Output File Name	Definition
Metal	METAL_MAT_#	MAT is an optional field which identifies the metal type and # is the Sonnet metal level from which the metal was exported. The MAT field is only used if there is more than one metal type.
Via	VIA_MAT_B#_T#	MAT is an optional field which identifies the metal type. B# is the Sonnet metal level from which the via originates and T# is the Sonnet metal level where the via terminates. The MAT field is only used if there is more than one metal type.
Box Top	SONNET_TOP	This is a special file name that identifies this metal layer as the Sonnet box top. This special layer name is recognized if you are importing Gerber files into Sonnet
Box Bottom	SONNET_BOT	This is a special file name that identifies this metal layer as the Sonnet box bottom. This special layer name is recognized if you are importing Gerber files into Sonnet.
Dielectric Brick	BRICK_MAT_#	MAT is an optional field which identifies the dielectric type. # is the Sonnet metal level from which the dielectric brick was exported. The MAT field is only used if there is more than one metal type.

If nametype is set to `project`, then the output files are named <Project Base Name>_#. gbr where <Project Base Name> is the basename of the project from which you are exporting and # is an integer value that uniquely identifies each file. The Export Gerber window displayed at the end of the export identifies the layers using the default naming scheme in the table above and their correspondence to the output file names. For example, for the project steps.son the Gerber output with project name selected would be as follows:

SONNET_TOP: steps_1.gbr
METAL_Lossless_0: steps_2.gbr
SONNET_BOT: steps_3.gbr

If nametype is set to `custom`, then the output files are named <Custom_prefix>_#. gbr where <Customprefix> is the prefix identified in the `GbrFilenamePrefix` statement and # is an integer value that uniquely identifies each file. The Export Gerber window displayed at the end of the export identifies the layers using the default naming scheme in the table above and their correspondence to the output file names. For example, for the custom prefix “galaxy_” the Gerber output filenames would be as follows:

SONNET_TOP: galaxy_1.gbr
METAL_Lossless_0: galaxy_2.gbr
SONNET_BOT: galaxy_3.gbr

GbrJobFilenameType This statement defines the naming convention for the Netex-G Job file output by the export.

Syntax GbrJobFilenameType jobnametype

A job file is output as part of the export that provides the setup for the output Gerber files if you need to re-import the Gerber files into Sonnet. You may do a Job File import using this file. The `GbrJobFilenameType` determines how the output Netex-G job file is named. The `jobnametype` field determines the basename for the job file; the file extension is always .njb. There are two possible character string values for `jobnametype`: project or custom.

If `jobnametype` is set to `project`, then the job file is named <Project Base Name>. njb where <Project Base Name> is the basename of the project from which you are exporting. For example, if you are exporting the project steps.son then the job file is named steps.njb.

If `jobnametype` is set to `custom`, then the job file is named <Custom Prefix>. njb where <Custom Prefix> is the whatever is entered by the user which is defined in the GbrJobPrefix statement described below. For example, for the custom prefix "galaxy" then the job file is named galaxy.njb.

GbrJobPrefix The GbrJobPrefix statement is one of two options that determine how to name the exported Netex-G job file. This option allows you to define a Prefix such that the job file created in the translation starts with that prefix.

Syntax GbrJobPrefix jobprefix

When the **GbrJobFilenameType statement is set to custom**, then the exported job file is named <jobprefix>. gbr where <jobprefix> is a character string entered by the user for the prefix. For example, for the custom prefix "galaxy" the Netex-G job file would be “galaxy.njb.”

GbrUnits This statement defines the length unit used for the exported Gerber files: mm or inches.

Syntax GbrUnits unit

This statement defines the length unit used for the exported Gerber files where unit is either the character string “mm” for millimeters or “inch” for inches.

END The end statement for a particular translator block.

Syntax `END`

This statement defines the end of a DXFEXPORT, GDSEXPORT or GERBEREXPORT block.

END TRANSLATOR The end statement for the TRANSLATORS block.

Syntax `END TRANSLATOR`

This statement defines the end of the translator block which may contain up to three blocks, one for each of the possible export types: DXF, GDSII or Gerber.