TopoR Design Language Reference

Version 1.1.3
February 2015

Contents

Chapter 1. About This Manual

TopoR PCB is an open plain-text format for definition of printed circuit boards designed in the TopoR CAD system. The format is XML-based and intended for transfer of printed circuit board designs between different versions of TopoR and for data interchange with other CAD systems.

This specification is intended for programmers developing converters between TopoR and other PCB CAD systems.

Conventions Used in the Manual

Well-known XML terminology is used for the format definition. The keywords include tags and attributes. All tags start with an uppercase letter, all attributes with a lowercase letter.

The format syntax is described using the Extended Backus-Naur Form. The following conventions are used:

- Tag attributes and nested tags are enclosed in parentheses: ()
- Possible attribute values separated by the choice sign | or attributes (tags) are enclosed in square brackets: []
- Curly braces denote that any of the enclosed attributes (tags) can be omitted or repeated any number of times: {}
- Attribute type values (listed in the Format Description chapter) are enclosed in inequality signs: <>

Some typefaces, characters and styles used in this manual have special meanings:

- `<Subwire width="0.4">`—this is how snippets from a file appear
- = **autoequ** [**None** | **Pins** | **Gates** | **Full**]—in syntax definitions, text that is identical to text in a file is **bolded**
- The ! symbol marks important additional information

See Also

Official XML 1.0 format specification.
Extended Backus-Naur Form specification.

For details about the purposes of the options and parameters described in this manual, refer to the TopoR online documentation.

Feedback

At Eremex, we strive to improve your experience of working with the TopoR PCB format, so we welcome your opinion of the format and its manual. Please send your observations and suggestions to **info@eremex.ru**.

Chapter 2. Design Language Syntax

This chapter describes the syntax and semantics of printed circuit board design in the TopoR PCB format. At the beginning of the chapter, the general structure is presented. The chapter is concluded by an alphabetic reference on the keywords (tags and attributes). Where a keyword has multiple meanings, the parent tag is provided in parentheses next to it.

Format Structure

A file written by TopoR starts with a line specifying the XML version and encoding, and comment block containing brief information about the file: file name, format name, format version, name of the program that created the file, and file creation time.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>

<!--********************************************-->
<!--   File    : pcbvar.fst                     -->
<!--   Format  : TopoR PCB file                 -->
<!--   Version : 1.1.0                          -->
<!--   Program : TopoR 8 Layer 5.3.2.13288 Alpha -->
<!--   Date    : Monday, January 2, 2012        -->
<!--   Time    : 22:06                          -->
<!--********************************************-->
```

This comment block is optional, and it serves to clarify the purpose of the file.

! TopoR support only UTF-8.

Next is the root **TopoR_PCB_File** tag, which contains all of the design information for the printed circuit board and organizes it into partitions. Partitions are in turn split into sections. Some partitions are mandatory.

! Partitions and sections within them can be in arbitrary order. However, the order of tags within sections is rigid.

TopoR creates partitions in the following order:

1. **Header**

2. **Layers** (mandatory partition)
   - **StackUpLayers** - stack of layers.
   - **UnStackLayers** - layers not included in stack.

3. **TextStyles** - text label styles.

4. **LocalLibrary** - library elements (mandatory partition).
   - **Padstacks**
   - **Viastacks**
   - **Footprints** - outer cases.
   - **Components**
   - **Packages**

5. **Constructive** - blank board.
   - **BoardOutline**
   - **Mntholes** - mounting holes.

- **MechLayerObjects** - parts on mechanical layers.
- **Texts** - text labels.
- **Keepouts**

6. **ComponentsOnBoard** (mandatory partition)
    - **Components**
    - **FreePads** - unconnected pads.

7. **NetList** - list of current nets.

8. **OriginalNetList**

! This partition is created only if there are differences between the original and current list of nets.

9. **Groups** - for objects grouping.
    - **LayersGroups**
    - **NetsGroups**
    - **ComponentsGroups**

10. **HiSpeedRules** - rules for high-speed boards.
    - **RulesImpedances** - rules for signals routing.
    - **SignalClusters**
    - **DiffSignals**
    - **SignalGroups**
    - **RulesDelay** - rules for delay equalization.
    - **SignalSearchSettings**

11. **Rules**
    - **RulesWidthOfWires**
    - **RulesClearancesNetToNet**
    - **RulesClearancesCompToComp** - clearances between components.
    - **RulesClearancesToBoard** - distance to the edge of the board.
    - **RulesViastacksOfNets** - assignment of via types to nets.
    - **RulesPlaneLayersNets** - assignment of plane layers to nets.
    - **RulesSignalLayersNets** - assignment of signal layers to nets.
    - **NetProperties** - additional net properties.

12. **Connectivity** - connections on the board.
    - **Vias**
    - **Serpents**
    - **ZippedWires**
    - **Wires**
    - **Coppers** - copper pours.
    - **NonfilledCoppers** – non-filled coppers.

13. **Settings**
    - **Autoroute**
    - **Placement** - settings for both manual and automatic placement.
    - **Labels**

14. **DisplayControl** - display options.
    - **View**
    - **ActiveLayer**
    - **Units**
    - **Colors**
    - **Show** - visibility flags.
    - **Grid**
    - **LayersVisualOptions** - layer display options.
    - **ColorNets** – net color overrides.
    - **FilterNetlines** – links visibility filter.

15. **DialogSettings**
    - **DRCSettings**
    - **GerberSettings**
    - **DXFSettings**
    - **DrillSettings**
    - **BOMSettings**
    - **MessagesFilter**

Versioning

The format version is made up of three numbers. The first number is the major version number. When it changes, converters developed for prior versions will not necessarily be able to read the files. The minor number is changed when there big changes are made to optional partitions. In this case, converters for prior versions might not be able to read the partitions with the changes. If this happens, all information from those partitions will be lost. Finally, the revision number is changed when small changes are made or additional data is included.

Changes in Version 1.1.3 of the Format

1. The **SelectFilter** tag has been removed, because the design file does not store the selection filter state any more.
2. In the **DisplayControl** section, the following tags have been added:**ColorNets** for net color overrides and **FilterNetlines** for selective display of links.

Changes in Version 1.1.2 of the Format

1. In the **Autoroute** tag, **smoothWires**, **strictCheck, recognizeBGA** attributes have been deprecated; **takeCurLayout** и **weakCheck** attributes have been added.
2. The **RedundantVia** tag has been removed.
3. The **Freestyle** tag has been removed.
4. Format of the **NetProperty** tag has been revised due to the newly-added support for flexible fixing.
5. In the **Show** tag, **showBoardOutline** has been added.
6. Format of the **SelectFilter** tag has been changed.
7. The **PowerNets** tag has been removed. It contained information about nets excluded from the signal list. For that purpose, the **ExcludedNets** tag has been added to **SignalSearchSettings**.

Changes in Version 1.1.1 of the Format

1. In the **DisplayControl** partition, **pinsName**, **showPinsName**, **pinsNet**, **showPinsNet** attributes have been added.

Changes in Version 1.1.0 of the Format

1. The **HiSpeedRules** partition has been changed considerably due to signal support in TopoR 5.3.
2. In the **Layer** tag, the **side** attribute has been deprecated. The layer side is now determined by the position of the layer in the stack.
3. In the **Layer** tag, attribute **compsOutline** has been added.
4. In the **Rules** partition, the **PowerNets** section has been added for power net definitions.
5. the definition of a **ZippedWire**, the **DiffPairRef** tag has been replaced by the **DiffSignalRef** tag. Support for the **DiffPairRef** tag is retained for compatibility with version 1.0.0.
6. In the **Pinpack** tag, the **valueType** and **delay** attributes have been added; they define signal delay within a pattern.
7. Hierarchical object grouping structures are now supported. The changes affect the **NetGroup**, **LayerGroup** and **CompGroup** tags.

Format Specifics

1. Duplication of object names within the same object type is not permitted. The one exception is the names of layers, which must be unique only within the layer type.
2. Attribute omission is permitted. If an attribute is absent, the default value is assumed.

Types of Attribute Values

<*bool*> = [**off** | **on**]
Flag. Default value: off.

<*color*> = #<*hex_byte*> <*hex_byte*> <*hex_byte*>
RGB color. Default value: 0 (black).

<*filename*> = <*string*>
String containing the full path and file name. Default value: empty string.

<*float*>
Floating-point number. Default value: 0.

<*format_version*> = <*positive_integer*> . <*positive_integer*> . <*positive_integer*>
Format version.

<*hex_byte*>
Byte in hexadecimal representation. Default value: 0.

<*integer*>
Signed integer. Default value: 0

<integer_rate>
Integer in the range 0 to 100 (percentage). Default value: 0.

<layer_type> = [**Assy** | **Paste** | **Silk** | **Mask** | **Signal** | **Plane** | **Mechanical** | **Doc** | **Dielectric**]
Type of layer.

Value	**Description**
Assy	Assembly layer (component outline layer)
Paste	Soldering paste layer
Silk	Serigraphy layer
Mask	Mask layer
Signal	Signal layer
Plane	Plane layer
Mechanical	Mechanical layer
Doc	Documentation layer
Dielectric	Dielectric layer

Default value: Signal.

<part_version> = *<positive_integer>* . *<positive_integer>*
Partition version.

<positive_integer>
Non-negative integer. Default value: 0.

<string>
String. Default value: empty string.

Keywords Descriptions

ActiveLayer

= **ActiveLayer** [**type** *<layer_type>*] (**name** *<string>*)

Sets the active layer.

align

= **align** [**LT** | **CT** | **RT** | **LM** | **CM** | **RM** | **LB** | **CB** | **RB**]

Text label parameter: how to align text

Value	Description
LT	Align to top left
CT	Align to top center
RT	Align to top right
LM	Align left
CM	Center
RM	Align right
LB	Align to bottom left
CB	Align to bottom center
RB	Align to bottom right

Default value: CM.

alignToGrid

= **alignToGrid** *<bool>*

Manual editor option: align to grid.

AllComps

= **AllComps**

Sets the rule scope: all components.

AllLayers

= **AllLayers**

Sets the rule scope: all layers.

AllLayersInner

= **AllLayersInner**

Sets the rule scope: all inner layers.

AllLayersInnerSignal

= **AllLayersInnerSignal**

Sets the rule scope: all inner signal layers.

AllLayersSignal

= **AllLayersSignal**
Sets the rule scope: all signal layers.

AllLayersOuter

= **AllLayersOuter**
Sets the rule scope: all outer layers.

AllNets

= **AllNets**
Sets the rule scope: all nets.

AllViastacks

= **AllViastacks**
Sets the viastack types available to the rule: all viastack types.

AllViastacksThrough

= **AllViastacksThrough**
Sets the viastack types available to the rule: all through viastacks.

AllViastacksNotThrough

= **AllViastacksNotThrough**
Sets the viastack types available to the rule: all non-through viastacks.

angle

= **angle** <*float*>
Sets an angle in degrees to a precision of 1 decimal digit.

Arc

= **Arc** (**Center**) (**Start**) (**End**)
Definition of an arc. The arc is drawn counterclockwise from the Start point to the End point.

Attribute (CompInstance)

= **Attribute** [**type** [**RefDes** | **PartName**] | (**name** <*string*>) (**value** <*string*>)] {(**Label**)}
Описание атрибута компонента на плате.

Attribute (Component)

= **Attribute** (**name** <*string*>) (**value** <*string*>)
Description of a component attribute.

AttributeRef

= **AttributeRef** (**name** *<string>*)
Reference to an attribute.

Attributes

= **Attributes** {(**Attribute**)}
Description of component attributes.

autoEqu

= **autoequ** [**None** | **Pins** | **Gates** | **Full**]
Autorouting option: use functional equivalence.

Value	**Description**
None	Do not use functional equivalence
Pins	Reassign component pins
Gates	Reassign component gates *(not supported)*
Full	Allow all reassignments *(not supported)*

Default value: None.

Autoroute

= **Autoroute** (**mode** [**Multilayer** | **SinglelayerTop** | **SinglelayerBottom**])
(**autoequ** [**None** | **Pins** | **Gates** | **Full**])
(**wireShape** [**Polyline** | **Arcs**])
(**teardrops** *<bool>*) (**weakCheck** *<bool>*) (**takeCurLayout** *<bool>*)
(**directConnectSMD***<bool>*) (**dontStretchWireToPolypin** *<bool>*)

Autorouting options.

background

= **background** *<color>*
Display option: background color.

backoff (Copper)

= **backoff** *<float>*
Copper area parameter: distance to the copper area.

backoff (Thermal)

= **backoff** *<float>*
Thermal pad parameter: distance from pad to the copper area.

blindVia

= **blindVia** *<bool>*
Display option: mark blind vias with special color.

board (Colors)

= **board** <*color*>
Display option: board outline color.

board (ExportObjects)

= **board** <*bool*>
Gerber file export option: output board outline.

BoardOutline

= **BoardOutline** (**Contour**) (**Voids**)
Definition of the board outline and voids.

Example:

```
<BoardOutline>
   <Contour>
      <Shape lineWidth="0.1">
         <Rect>
            <Dot x="49" y="57.8"/>
            <Dot x="144" y="182.8"/>
         </Rect>
      </Shape>
   </Contour>
   <Voids>
      <Shape lineWidth="0.1">
         <Circle diameter="3">
            <Center x="53" y="178.8"/>
         </Circle>
      </Shape>
   </Voids>
</BoardOutline>
```

bold

= **bold** <*bool*>
Text label style parameter: bold font.

BOMSettings

= **BOMSettings** (**outFile** <*filename*>) (**count** <*bool*>) (**partName** <*bool*>)
(**footprint** <*bool*>) (**refDes** <*bool*>)
{(**AttributeRef**)}
BOM file export options.

bottomHorzRotate

= **bottomHorzRotate** <*bool*>
Label orientation option: rotate horizontally-oriented labels on the bottom side.

bottomVertRotate

= **bottomVertRotate** <*bool*>
Label orientation option: rotate vertically-oriented labels on the bottom side.

burriedVia

= **burriedVia** <*bool*>
Display option: mark burried vias with special color.

Center

= **Center** (**x** <*float*>) (**y** <*float*>)
Center of a circle or oval.

checkClearances

= **checkClearances** <*bool*>
DRC option: check clearances.

checkNetIntegrity

= **checkNetIntegrity** <*bool*>
DRC option: check net integrity.

checkNetWidth

= **checkNetWidth** <*bool*>
DRC option: check wire width.

Circle

= **Circle** (**diameter** <*float*>) (**Center**)
Definition of a hollow circle.

ClearanceCompToComp

= ClearanceCompToComp (**enabled** <*bool*>) (**clrn** <*float*>) (**ObjectsAffected**)
Definition of a rule for clearance between components.

ClearanceNetToNet

= ClearanceNetToNet (**enabled** <*bool*>) (**clrnMin** <*float*>) (**clrnNom** <*float*>)
(LayersRef) (**ObjectsAffected**)
Definition of a rule for clearance between nets.

Example:

```
<ClearanceNetToNet enabled="on" clrnMin="0.5" clrnNom="0.7">
   <AllLayers/>
   <ObjectsAffected>
      <NetGroupRef name="Power"/>
      <NetGroupRef name="Power Channel"/>
   </ObjectsAffected>
</ClearanceNetToNet>
```

clrn

= **clrn** <*float*>
Component-to-component clearance rule parameter: clearance.

clrnMin

= **clrnMin** <*float*>
Net-to-net clearance rule parameter: minimal clearance.

clrnNom

= **clrnNom** <*float*>
Net-to-net clearance rule parameter: nominal clearance.

clrBlindVias

= clrBlindVias <*color*>
Display option: color of blind vias.

clrBurriedVias

= clrBurriedVias <*color*>
Display option: color of burried vias.

clrFixedVias

= clrFixedVias <*color*>
Display option: color of fixed vias.

clrThroughPads

= clrThroughPads <*color*>
Display option: color of through pads.

clrThroughVias

= clrThroughVias <*color*>
Display option: color of through vias.

color

= **color** <*color*>
Net color overrides: color of net.

colorizeCopper

= **colorizeCopper** <*bool*>
Net color overrides: apply for copper pours (polygons).

colorizeNetline

= **colorizeNetline** <*bool*>
Net color overrides: apply for links.

colorizePad

= **colorizePad** <*bool*>
Net color overrides: apply for pads.

colorizeVia

= **colorizeVia** <*bool*>
Net color overrides: apply for vias.

colorizeWire

= **colorizeWire** <*bool*>
Net color overrides: apply for wires.

ColorNets

= ColorNets (**enabled**<*bool*>) (**colorizeWire**<*bool*>) (**colorizePad**<*bool*>) (**colorizeCopper**<*bool*>) (**colorizeVia**<*bool*>) (**colorizeNetline**<*bool*>) {(**SetColor**)}
Net color overrides.

Colors (DisplayControl)

= **Colors** (**colorScheme** <*string*>) (**hilightRate** <*integer_rate*>) (**darkRate** <*integer_rate*>)
(**background** <*color*>**)** **(board** <*color*>**)** (**netLines** <*color*>**)**
(**keepoutPlaceBoth** <*color*>) (**keepoutWireAll** <*color*>)
(**keepoutPlaceTop** <*color*>) (**keepoutPlaceBot** <*color*>)
(**compsBound** <*color*>) (**compsName** <*color*>)
(**pinsName** <*color*>) (**pinsNet** <*color*>)
(**clrThroughPads** <*color*>) (**clrThroughVias** <*color*>)
(**clrBurriedVias** <*color*>) (**clrBlindVias** <*color*>) (**clrFixedVias** <*color*>)
(**drcViolation** <*color*>**)** (**narrow** <*color*>) (**trimmed** <*color*>)
Display option: general color scheme options.

Colors (LayerOptions)

= **Colors** {(**details** <*color*>) (**pads** <*color*>) (**fix** <*color*>)}
Display option: layer colors.

colorScheme

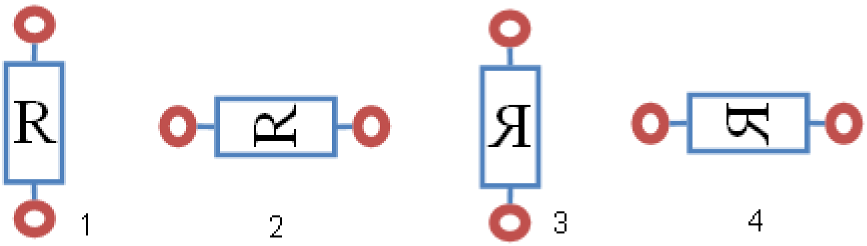
= **colorScheme** <*string*>
Display option: current color scheme.

CompGroupRef

= **CompGroupRef** (**name** <*string*>)
Reference to a component group.

CompInstance

= **CompInstance** (**name** <*string*>) (**side** [**Top** | **Bottom**]) (**angle** <*float*>) (**fixed** <*bool*>)
(**ComponentRef**) (**FootprintRef**) (**Org**)
(**Pins**) (**Mntholes**) (**Attributes**)
Definition of a component on the board.



1. Image of a library component.
2. Rotation by 90 degrees
3. Component is on the bottom side
4. Rotation by 90 degrees on the bottom side

Example:

```
<CompInstance name="C4" side="Top" angle="90" fixed="on">
   <ComponentRef name="K17B-2_1"/>
   <FootprintRef name="K17B-2_1"/>
   <Org x="184.15" y="46.65"/>
   <Pins>
      <Pin padNum="1">
         <PadstackRef name="C150R.PS"/>
         <Org x="0" y="0"/>
      </Pin>
      <Pin padNum="2">
         <PadstackRef name="C150R.PS"/>
         <Org x="5" y="0"/>
      </Pin>
   </Pins>
   <Attributes>
      <Attribute name="Type" value="K17B-2"/>
      <Attribute name="FP" value="K17B-2"/>
      <Attribute name="PDIF_TY" value="11000"/>
      <Attribute name="DEVICE" value="K17B-2"/>
   </Attributes>
</CompInstance>
```

CompInstanceRef

= **CompInstanceRef** (name <*string*>)
Reference to a component on the board.

compName

= **compName** <*string*>
Component name; used for referencing the component.

Component

= **Component** (name <*string*>) (Pins) (Attributes)
Definition of a library component.

Example:

```
<Component name="CR0402-FX-2493GLF_1">
   <Pins>
      <Pin pinNum="1" name="1" pinSymName="" pinEqual="1" gate="1"
           gateEqual="0"/>
      <Pin pinNum="2" name="2" pinSymName="" pinEqual="2" gate="1"
           gateEqual="0"/>
   </Pins>
   <Attributes>
      <Attribute name="Type" value="Resistor"/>
      <Attribute name="Available" value="Yes"/>
      <Attribute name="Manufacturer" value="Bourns"/>
      <Attribute name="Pkg_Type" value="0402"/>
      <Attribute name="Power Dissipation" value="0.062"/>
      <Attribute name="Tolerance" value="1%"/>
      <Attribute name="Value" value="24,90,00"/>
      <Attribute name="Voltage" value="25V"/>
      <Attribute name="TC" value="100ppm"/>
      <Attribute name="Stuff" value="Yes"/>
   </Attributes>
</Component>
```

CompGroup

= **CompGroup** (name <*string*>) {[(CompInstanceRef) | (CompGroupRef)]}
Component group.

CompGroups

= **CompGroups** {(CompGroup)}
Component groups.

ComponentRef

= **ComponentRef** (name <*string*>)
Reference to a component.

Components (LocalLibrary)

= Components {(Component)}
Definition of library components.

Components (ComponentOnBoard)

= **Components** {(CompInstance)}
Definition of components on the board.

Components (Signal)

= **Components** {(ComponentRef)}
Passive components in the signal path.

ComponentsOnBoard

= **ComponentsOnBoard** (Components) (FreePads)
Definition of components and free pads on the board.

comps

= **comps** <*float*>
Sets the clearance between components and the edge of the board.

compsBound

= **compsBound** <*color*>
Display option: color of component bounding boxes.

compsName

= **compsName** <*color*>
Display option: color of component icons.

compsOutline (DXFSettings\ExportLayer\ExportObjects)

= compsOutline <*bool*>
DXF file export option for layers: output component outlines.

compsOutline (Layer)

= compsOutline <*bool*>
Layer parameter: Layer contains component outlines.

Connectivity

= **Connectivity** (**version** *<part_version>*) (**Vias**) (**Serpents**) (**ZippedWires**)
(**Wires**) (**Coppers**) (**NonFilledCoppers**)

Connectivity partition.

This partition describes the specifics of the connection implementation: printed wires, vias and copper pours.

Some wire segments form special objects used in designing high-speed appliances: zipped wires and serpent-shaped delay providers. Wire segments reference these objects; parameters of the objects are described in the **ZippedWires** and **Serpents** sections, respectively. For example, in a serpent, set the location, required length and gap between turns:

```
<Serpent id="serp_0" length="15.5" gap="0.4"
         h1="1.3" h2="1.3"
         h3="1.3" h4="1.3"/>
```

If there are not references to parameter descriptions, then all information about wire and object associations is lost. Conversely, if a serpent or a pair of zipped wires is described parametrically, then TopoR calculates the shape of the corresponding wire segments automatically and ignores their descriptions in the **Wires** section.

Zipped wires are described parametrically by specifying a track line (see ZippedWire).

Example of a partial zipped wire definition:

```
<Subwire width="0.2" zipwireRef="zwire_1">
   <Start x="170.022" y="102.923"/>
   <TrackLine>
      <End x="169.866" y="102.552"/>
   </TrackLine>
   <TrackLine>
      <End x="169.514" y="101.711"/>
   </TrackLine>
   <TrackLine>
      <End x="169.391" y="100.716"/>
   </TrackLine>
</Subwire>
```

Zipped wires can also contain serpents. In this case, references to serpent objects are made at the corresponding segments of the zipped wire's track line.

connectPad

= **connectPad** [**Direct** | **Thermal**]

Stacked copper area (polygon) parameter: pad connection type.

Value	**Description**
Direct	Direct connection
Thermal	Connection through a thermal pad

Default value: Direct.

connectVia

= **connectVia** [**Direct** | **Thermal**]
Copper area (polygon) parameter: via connection type.

Value	Description
Direct	Direct connection
Thermal	Connection through a thermal pad

Default value: Direct.

connectToCopper

= **connectToCopper** [**NoneConnect** | **Direct** | **Thermal**]
Padstack parameter: copper area (polygon) connection type.

Value	Description
NoneConnect	Connection type not set; the polygon’s settings are used
Direct	Direct connection
Thermal	Connection through a thermal pad

Default value: NoneConnect.

constant

= **constant** <*float*>
Value of the constant in delay equalization rules.

! The units depend on the valueType parameter and the units set globally for the file (see Units).

Constructive

= **Constructive** (**version** <*part_version*>)
(**BoardOutline**) (**Mntholes**) (**MechLayerObjects**)
(**Texts**) (**Keepouts**)
Definition of the board constructive.

Contour

= **Contour** {(**Shape**)}
Definition of the board contour.

Copper (Connectivity\Coppers)

= **Copper** (**priority** <*integer*>) (**useBackoff** <*bool*>) (**backoff** <*float*>)
(**connectPad** [**Direct** | **Thermal**]) (**connectVia** [**Direct** | **Thermal**])
(**lineWidth** <*float*>) (**lineClr** <*float*>)
(**minSquare** <*float*>) (**precision** [**Low** | **Med** | **High**]) (**deleteUnconnected** <*bool*>)
(**state** [**Unpoured** | **Poured** | **Locked**]) (**fillType** [**Solid** | **Hatched** | **CRHatched**])
(**LayerRef**) (**NetRef**)
(**ThermalPad**) (**ThermalVia**)
(**Shape**) (**Voids**) (**Islands**) (**Fill**)

Definition of a copper pour (polygon).

! The Fill option (for line patterns) is output for use by other CAD systems. TopoR ignores it during import. The option to use solid filling (fillType = Solid) is not output.

Example:

```
<Copper priority="50" useBackoff="on" backoff="0.3"
        connectPad="Thermal" connectVia="Direct"
        lineWidth="0.2" lineClr="0.8" minSquare="0" precision="Med"
        state="Poured" fillType="Solid">
   <LayerRef name="Top"/>
   <NetRef name="AD15"/>
   <ThermalPad>
      <Thermal spokeNum="4" minSpokeNum="1" angle="45"
               spokeWidth="0.381" backoff="0.381"/>
   </ThermalPad>
   <ThermalVia/>
   <Shape>
      <FilledRect>
         <Dot x="167.845" y="90.6856"/>
         <Dot x="174.244" y="84.6483"/>
      </FilledRect>
   </Shape>
   <Voids/>
   <Islands>
      <Island>
         <Polygon>
            <Dot x="174.229" y="87.9419"/>
            <Dot x="174.244" y="87.9631"/>
            <Dot x="174.244" y="90.6856"/>
            <Dot x="167.845" y="90.6856"/>
            <Dot x="167.845" y="84.6483"/>
            <Dot x="173.382" y="84.6483"/>
            <Dot x="174.244" y="86.6247"/>
            <Dot x="174.244" y="87.847"/>
         </Polygon>
         <Voids>
            <Polygon>
               <Dot x="174.05" y="87.4237"/>
               <Dot x="173.376" y="86.75"/>
               <Dot x="172.424" y="86.75"/>
               <Dot x="171.75" y="87.4237"/>
               <Dot x="171.75" y="88.3763"/>
               <Dot x="172.424" y="89.05"/>
               <Dot x="173.376" y="89.05"/>
               <Dot x="174.05" y="88.3763"/>
            </Polygon>
            <Polygon>
               <Dot x="171.631" y="87.3901"/>
               <Dot x="170.91" y="86.669"/>
               <Dot x="169.89" y="86.669"/>
               <Dot x="169.169" y="87.3901"/>
               <Dot x="169.169" y="88.4099"/>
               <Dot x="169.89" y="89.131"/>
               <Dot x="170.91" y="89.131"/>
               <Dot x="171.631" y="88.4099"/>
            </Polygon>
         </Voids>
         <ThermalSpoke lineWidth="0.381">
            <Dot x="170.4" y="87.9"/>
            <Dot x="169.53" y="88.7705"/>
         </ThermalSpoke>
         <ThermalSpoke lineWidth="0.381">
            <Dot x="170.4" y="87.9"/>
            <Dot x="169.53" y="87.0296"/>
         </ThermalSpoke>
         <ThermalSpoke lineWidth="0.381">
            <Dot x="170.4" y="87.9"/>
            <Dot x="171.27" y="87.0295"/>
         </ThermalSpoke>
         <ThermalSpoke lineWidth="0.381">
            <Dot x="170.4" y="87.9"/>
            <Dot x="171.271" y="88.7704"/>
         </ThermalSpoke>
      </Island>
   </Islands>
</Copper>
```

Copper (Footprint\Coppers)

= **Copper** (**lineWidth** <*float*>) (**LayRef**) (Figure)
Definition of a copper pour (polygon) in a component footprint.

Coppers (Connectivity)

= **Coppers** {(**Copper**)}
Definition of copper pours (polygons).

Coppers (Footprint)

= **Coppers** {(**Copper**)}
Definition of copper pours (polygons) in a component footprint.

coppers

= **coppers** <*bool*>
Gerber and DXF file export option: output copper pours (polygons).

copperToBoard

= **copperToBoard** <*bool*>
DRC option: check the clearance between polygons and the edge of the board.

copperToCopper

= **copperToCopper** <*bool*>
DRC option: check the clearance between polygons.

copperToKeepout

= **copperToKeepout** <*bool*>
DRC option: check the clearance between polygons and keepouts.

copperToPad

= **copperToPad** <*bool*>
DRC option: check the clearance between polygons and pads.

copperToVia

= **copperToVia** <*bool*>
DRC option: check the clearance between polygons and vias.

copperToWire

= **copperToWire** <*bool*>
DRC option: check the clearance between polygons and wires.

count

= **count** <*bool*>
BOM file export option: output the number of components.

createLog

= **createLog** <*bool*>
DRC option: write a report to the specified file.

createPinPairs

= createPinPairs <*bool*>
Create pin pairs automatically.

darkRate

= **darkRate** <*integer_rate*>
Display option: how dark unselected objects are.

Date

= **Date** <*string*>
File creation date and time (in arbitrary format).

delay

= delay <*float*>
Parameter of a component pin in a pattern: signal delay within the pattern.

DelayConstant

= **DelayConstant** (**enabled** <*bool*>) (**valueType** [**Time** | **Dist**]) (**constant** <*float*>)
(**toleranceUnder** <*float*>) (**toleranceOver** <*float*>)
(**ObjectsAffected**)
Definition of a rule for the absolute delay value.

DelayEqual

= **DelayEqual** (**enabled** <*bool*>) (**valueType** [**Time** | **Dist**]) (**tolerance** <*float*>)
(**ObjectsAffected**)
Definition of rules for equalizing delays in a net group or differential pair group.

DelayRelation

= **DelayRelation** (**enabled** <*bool*>) (**valueType** [**Time** | **Dist**]) (**constant** <*float*>)
(**toleranceUnder** <*float*>) (**toleranceOver** <*float*>)
(**ObjectLeft**) (**ObjectRight**)
Definition of a rule for mutual delay equalization.

! The rule is not symmetrical for ObjectLeft and ObjectRight.

deleteUnconnected

= deleteUnconnected <*bool*>
Copper pour (polygon) parameter: delete unconnected islands.

Detail (Details, MechLayerObjects)

= **Detail** (**lineWidth**) (**LayRef**) (Figure)
Definition of a detail.

Details

= **Details** {(**Detail**)}
Definition of outer case details.

details (Colors)

= **details** <*color*>
Layer display option: color of details and wires (main color of the layer).

details (ExportObjects)

= **details** <*bool*>
Gerber file export option: output details on mechanical layers.

DialogSettings

= **DialogSettings** (**version** <*part_version*>)
(**DRCSettings**) (**GerberSettings**) (**DXFSettings**)
(**DrillSettings**) (**BOMSettings**) (**MessageFilter**)
DialogSettings partition.

diameter

= **diameter** <*float*>
Diameter of a circle or oval.

DiffSignal

= **DiffSignal** (**name** <*string*>) (**mismatch** <*float*>)
(**ImpedanceRef**) (**SignalRef**) (**SignalRef**)
Definition of a differential signal (differential pair).

DiffSignalRef

= **DiffSignalRef** (**name** <*string*>)
Reference to a differential signal.

DiffSignals

= **DiffSignals** {(**DiffSignal**)}
Definition of a differential signals.

DiffPairRef

= **DiffPaiRef** (**name** <*string*>)
Reference to a differential pair. This is equivalent to using DiffSignalRef. Support for the tag is retained for compatibility with version 1.0.0.

directConnectSMD

= **directConnectSMD** <*bool*>
Autorouting option: connect SMD pads directly.

DisplayControl

= **DisplayControl** (version *<part_version>*)
(View) (ActiveLayer) (Units) (Colors) (Show)
(Grid) (LayerVisualOptions) (ColorNets) (FilterNetlines)

DisplayControl partition.

displayScheme

= **displayScheme** *<string>*

Display option: current display scheme.

dist

= **dist** [**mkm** | **mm** | **cm** | **dm** | **m** | **mil** | **inch**]

Units for the file.

Value	Description
mkm	Micrometers
mm	Millimeters
cm	Centimeters
dm	Decimeters
m	Meters
mil	Mils (thousandths of an inch)
Inch	Inches

Default value: mm.

dontStretchWireToPolypin

= **dontStretchWireToPolypin** *<bool>*

Autorouting option: do not stretch wires to polygonal pads.

Dot

= **Dot** (**x** *<float>*) (**y** *<float>*)

Coordinates of a point or vertex.

DRCSettings

= **DRCSettings** (createLog *<bool>*) (logFileName *<filename>*)
(messageLimit *<integer>*) (tolerance *<float>*)
(checkNetIntegrity *<bool>*) (checkNetWidth *<bool>*) (checkClearances *<bool>*)
(textToCopper *<bool>*) (textToKeepout *<bool>*) (textToVia *<bool>*)
(textToWire *<bool>*) (textToPad *<bool>*) (textToBoard *<bool>*)
(copperToCopper *<bool>*) (copperToKeepout *<bool>*) (copperToWire *<bool>*)
(copperToVia *<bool>*) (copperToPad *<bool>*) (copperToBoard *<bool>*)
(wireToKeepout *<bool>*) (viaToKeepout *<bool>*) (padToKeepout *<bool>*)
(wireToWire *<bool>*) (wireToVia *<bool>*) (wireToPad *<bool>*)
(wireToBoard *<bool>*) (viaToVia) (viaToPad *<bool>*)
(viaToBoard *<bool>*) (padToPad *<bool>*) (padToBoard *<bool>*)

DRC options.

drcViolation

= **drcViolation** <*color*>
Display option: color of DRC violations.

DrillSettings

= **DrillSettings** (**outPath** <*string*>) (**units** [**mm** | **mil**])
(**intNums** <*positive_integer*>) (**fractNums** <*positive_integer*>)
{(**ExportFile**)}
Drill file export options.

DXFSettings

= **DXFfSettings** (**outFile** <*filename*>) (**units** [**mm** | **mil**])
(**outputBoardLayer** <*bool*>) (**outputDrillLayer** <*bool*>)
{(**ExportLayer**)}
DXF file export options.

enabled

= **enabled** <*bool*>
Whether a rule is enabled.

End

= **End** (**x** <*float*>) (**y** <*float*>)
Endpoint of a line or arc.

ExcludedNets

= **ExcludedNets** (**minPinsNumber** <*positive_integer*>){(**NetRef**)}
Netlist excluded from the search for signals.

ExportFile (GerberSettings)

= **ExportFile** (**fileName** <*string*>) (**output** <*bool*>) (**mirror** <*bool*>) (**negative** <*bool*>)
(**LayerRef**) (**ExportObjects**) (**Shift**)
Gerber file export options.

ExportFile (DrillSettings)

= ExportFile (**fileName** <*string*>)
Drill file export options.

ExportLayer

= **ExportLayer** (**output** <*bool*>) (**LayerRef**) (**ExportObjects**)
DXF file export options for a layer.

ExportObjects (GerberSettings\ExportFile)

= **ExportObjects** (**board** *<bool>*) (**wires** *<bool>*) (**coppers** *<bool>*)
(**padstacks** *<bool>*) (**vias** *<bool>*) (**texts** *<bool>*)
(**labels** *<bool>*) (**details** *<bool>*) (**fiducials** *<bool>*)
Gerber file export option: list of objects exported for a layer.

ExportObjects (ExportLayer)

= **ExportObjects** (**wires** *<bool>*) (**coppers** *<bool>*)
(**padstacks** *<bool>*) (**vias** *<bool>*) (**texts** *<bool>*)
(**labels** *<bool>*) (**details** *<bool>*) (**compsOutline** *<bool>*) (**fiducials** *<bool>*)
DXF file export option for a layer: list of objects exported for the layer.

fiducials

= **fiducials** *<bool>*
Gerber and DXF file export option: output fiducials.

! Fiducials are not supported by TopoR.

Figure

= [(**Arc**) | (**Circle**) | (**FilledCircle**) | (**FilledRect**) | (**Line**) | (**Polygon**) | (**Rect**)]
Definition of a figure.

fileName

= **fileName** *<string>*
Name of the Gerber or Drill file to export.

! The file name must not contain the path.

Fill

= **Fill** {(**Line**)}
How copper pours (polygons) are filled with line patterns.

! TopoR ignores this information during import and recreates the filling.

FilledCircle

= **FilledCircle** (**diameter** *<float>*) (**Center**)
Definition of a filled circle.

FilledFigure

= [(**FilledCircle**) | (**FilledRect**) | (**Polygon**)]
Definition of a filled figure.

FilledRect

= **FilledRect** (**Dot**) (**Dot**)
Definition of a filled rectangle.

fillType

= **fillType** [**Solid** | **Hatched** | **CRHatched**]
Copper pour (polygon) parameter: fill type.

Value	**Description**
Solid	Solid fill
Hatched	Hatched pattern
CRHatched	Crosshatched pattern

Default value: Solid.

FilterNetlines

= **FilterNetlines** (**enabled**<*bool*>) {[(ObjectNet) | (ObjectSignal)]}
Links visibility filter.

fix

= **fix** <*color*>
Layer display option: color of fixed objects.

fixed

= **fixed** <*bool*>
Fixed or non-fixed.

fixedVia

= **fixedVia** <*bool*>
Display option: mark fixed vias with color.

flipped

= **flipped** <*bool*>
Whether a footprint pad or pin is flipped. If the flag is not set, the planar contact pad will be on the same side as the component; otherwise it will be on the opposite side.

flexfix

= flexfix <bool>
Property of net: flex fixation.

fontName

= **fontName** <*string*>
Text label style option: font name.

Format

= **Format** <*string*>
File format name.

footprint

= **footprint** <*bool*>
BOM file export option: output the footprint name.

Footprint

= **Footprint** (**name** <*string*>)
(**Pads**) (**Texts**) (**Details**)
(**Coppers**) (**KeepoutsPlace**) (**KeepoutsTrace**)
(**Mntholes**) (**Labels**)
Definition of a footprint.

Example:

```
<Footprint name="CC7343_1">
   <Pads>
      <Pad padNum="1" name="1" angle="90">
         <PadstackRef name="SX30Y27DOT"/>
         <Org x="0" y="0"/>
      </Pad>
      <Pad padNum="2" name="2" angle="90">
         <PadstackRef name="SX30Y27DOT"/>
         <Org x="6.985" y="0"/>
      </Pad>
   </Pads>
   <Details>
      <Detail>
         <LayerRef name="TopSilk"/>
         <Polygon>
            <Dot x="0.5" y="2.4"/>
            <Dot x="-0.5" y="2.4"/>
            <Dot x="-0.5" y="1.9"/>
            <Dot x="0.5" y="1.9"/>
         </Polygon>
      </Detail>
      <Detail lineWidth="0.2">
         <LayerRef name="TopSilk"/>
         <Line>
            <Dot x="-1.8" y="1.9"/>
            <Dot x="-1.8" y="-1.9"/>
         </Line>
      </Detail>
      <Detail lineWidth="0.2">
```

```
         <LayerRef name="TopSilk"/>
         <Line>
            <Dot x="8.763" y="-1.9"/>
            <Dot x="5.334" y="-1.9"/>
         </Line>
      </Detail>
      <Detail lineWidth="0.2">
         <LayerRef name="TopSilk"/>
         <Line>
            <Dot x="8.763" y="1.9"/>
            <Dot x="5.334" y="1.9"/>
         </Line>
      </Detail>
      <Detail lineWidth="0.203">
         <LayerRef name="TopSilk"/>
         <Line>
            <Dot x="8.763" y="-1.9"/>
            <Dot x="8.763" y="1.9"/>
         </Line>
      </Detail>
      <Detail lineWidth="0.2">
         <LayerRef name="TopSilk"/>
         <Line>
            <Dot x="-1.8" y="-1.9"/>
            <Dot x="1.65" y="-1.9"/>
         </Line>
      </Detail>
      <Detail lineWidth="0.2">
         <LayerRef name="TopSilk"/>
         <Line>
            <Dot x="-1.8" y="1.9"/>
            <Dot x="1.65" y="1.9"/>
         </Line>
      </Detail>
   </Details>
   <KeepoutsTrace>
      <Keepout>
         <LayerRef name="Top"/>
         <Polygon>
            <Dot x="5.461" y="1.524"/>
            <Dot x="5.461" y="-1.524"/>
            <Dot x="1.524" y="-1.524"/>
            <Dot x="1.524" y="1.524"/>
         </Polygon>
      </Keepout>
   </KeepoutsTrace>
   <Labels>
      <Label name="RefDes" align="CM" angle="90">
         <LayerRef name="TopSilk"/>
         <TextStyleRef name="T:H60W8"/>
         <Org x="-3.165" y="0.095"/>
      </Label>
      <Label name="Value" align="LB">
```

```
         <LayerRef name="TopSilk"/>
         <TextStyleRef name="T:H80W8"/>
         <Org x="0" y="0"/>
      </Label>
      <Label name="Type" align="CM">
         <LayerRef name="TopSilk"/>
         <TextStyleRef name="T:H60W8"/>
         <Org x="1.27" y="0"/>
      </Label>
   </Labels>
</Footprint>
```

FootprintRef

= **FootprintRef** (**name** <*string*>)
Reference to a footprint.

Footprints

= **Footprints** {(**Footprint**)}
Definition of footprints.

fractNums

= **fractNums** <*integer*>
Gerber and Drill file export option for numbers: how many decimal places to use.

FreePad

= **FreePad** (**side** [**Top** | **Bottom**]) (**angle** <*float*>) (**fixed** <*bool*>)
(**PadstackRef**) (**NetRef**) (**Org**)
Definition of a free pad.

FreePads

= **FreePads** {(**FreePad**)}
Definition of free pads.

gap (LayerRule)

= **gap** <*float*>
Parameter of a rule for differential pair layout: clearance between the pair’s wires.

gap (Serpent)

= **gap** <*float*>
Serpent parameter: clearance between turns.

gate

= **gate** <*positive_integer*>
Component pin parameter: number of the pin gate.

gateEqual

= **gateEqual** <*positive_integer*>
Component pin parameter: equivalence of the pin gate.

GerberSettings

= **GerberSettings** (**outPath** <*string*>) (**units** [**mm** | **mil**])
(**intNums** <*positive_integer*>) (**fractNums** <*positive_integer*>)
{(**ExportFile**)}
Gerber file export options.

Grid

= **Grid** (**gridColor** <*color*>) (**gridKind** [**Dots** | **Lines**]) (**gridShow** <*bool*>)
(**alignToGrid** <*bool*>) (**snapToAngle** <*bool*>)
(**GridSpace**)
Grid options.

gridColor

= **gridColor** <*color*>
Grid display option: color of the grid.

gridKind

= **gridKind** [**Dots** | **Lines**]
Grid display option: grid type.

gridShow

= **gridShow** <*bool*>
Grid display option: whether the grid is shown.

GridSpace

= **GridSpace** (**x** <*float*>) (**y** <*float*>)
Grid display option: grid line spacing (**x** for horizontal, **y** for vertical).

Groups

= **Groups** (**version** < *part_version*>)
(**LayerGroups**) (**NetGroups**) (**ComponentGroups**)
Definition of object groups.

h1

= **h1** <*float*>
Serpent parameter: height **h1** (see Serpent).

h2

= **h2** <*float*>
Serpent parameter: height h2 (see Serpent).

h3

= **h3** <*float*>
Serpent parameter: height h3 (see Serpent).

h4

= **h4** <*float*>
Serpent parameter: height h4 (see Serpent).

Header

= **Header** (**Format**) (**Version**) (**Program**) (**Date**)
(**OriginalFormat**) (**OriginalFile**) (**Units**)
Header partition.

height (TextStyle)

= **height** <*float*>
Text label style parameter: character height in the current units.

height (PadRect)

= **height** <*float*>
Rectangular pad parameter: height.

hilightRate

= **hilightRate** <*integer_rate*>
Display option: brightness of selected objects.

HiSpeedRules

= **HiSpeedRules** (**version** <*part_version*>)
(**RulesImpedances**) (**SignalClusters**) (**DiffSignals**) (**SignalGroups**)
(**RulesDelay**) (**SignalSearchSettings**)
HiSpeedRules partition.

holeDiameter

= **holeDiameter** <*float*>
Diameter of a hole.

id

= **id** <*string*>
ID of unnamed objects.

Impedance

= **Impedance** (**name** <*string*>) (**Z0** <*float*>) {(**LayerRule**)}
Impedance and rules for multi-layer signal layout.

ImpedanceDiff

= **ImpedanceDiff** (**name** <*string*>) (**Z0** <*float*>) {(**LayerRule**)}
Impedance and rules for multi-layer differential pair layout.

ImpedanceRef

= I**mpedanceRef** (**name** <*string*>)
Reference to an impedance.

intNums

= **intNums** <*positive_integer*>
Gerber and Drill file export option for numbers: number of digits before decimal point.

Island

= **Island** (**Polygon**) (**Voids**) {(**ThermalSpoke**)}
Definition of a copper island.

Islands

= **Islands** {(**Island**)}
Definition of copper islands.

italic

= **italic** <*bool*>
Text label style parameter: italic font.

Keepout (Constructive\Keepouts)

= **Keepout** (**Role**) (Figure)
Definition of a keepout.

Example:

```
<Keepout>
   <Role>
      <Trace role="Wires">
         <AllLayers/>
      </Trace>
   </Role>
   <Polygon>
      <Dot x="139.2" y="177"/>
      <Dot x="140.8" y="177"/>
      <Dot x="141.8" y="178"/>
      <Dot x="141.8" y="179.6"/>
      <Dot x="140.8" y="180.6"/>
      <Dot x="139.2" y="180.6"/>
      <Dot x="138.2" y="179.6"/>
      <Dot x="138.2" y="178"/>
   </Polygon>
</Keepout>
```

Keepout (KeepoutsPlace, KeepoutsTrace)

= **Keepout** (**LayRef**) (Figure)
Definition of a keepout.

keepoutPlaceBot

= **keepoutPlaceBot** <*color*>
Display option: color of a placement keepout on the bottom side of the board.

keepoutPlaceBoth

= **keepoutPlaceBoth** <*color*>
Display option: color of a placement keepout on both sides of the board.

keepoutPlaceTop

= **keepoutPlaceTop** <*color*>
Display option: color of a placement keepout on the top side of the board.

Keepouts

= **Keepouts** {(**Keepout**)}
Definition of keepouts.

KeepoutsPlace

= **KeepoutsPlace** {(**Keepout**)}
Definition of placement keepouts on the case.

KeepoutsTrace

= **KeepoutsTrace** {(**Keepout**)}
Definition of routing keepouts on the case.

keepoutWireAll

= **keepoutWireAll** <*color*>
Display option: color of routing keepouts on all layers.

Label(CompInstance\Attributes\Attribute)

= **Label** (**angle** <*float*>) (**mirror** <*bool*>) (**align** [**LT** | **CT** | **RT** | **LM** | **CM** | **RM** | **LB** | **CB** | **RB**])
(**visible** <*bool*>) (**LayerRef**) (**TextStyleRef**) (**Org**)
Definition of a component label.

Label(Footprint\Labels)

= **Label** (**name** <*string*>) (**align** [**LT** | **CT** | **RT** | **LM** | **CM** | **RM** | **LB** | **CB** | **RB**]) (**angle** <*float*>)
(**LayerRef**) (**TextStyleRef**) (**Org**)
Definition of a footprint label.

Labels (Footprint)

= **Labels** {(**Label**)}
Definition of labels.

Labels (Settings)

= **Labels** (**rotateWithComp** <*bool*>) (**useOrientRules** <*bool*>)
(**topHorzRotate** <*bool*>) (**topVertRotate** <*bool*>)
(**bottomHorzRotate** <*bool*>) (**bottomVertRotate** <*bool*>)
Label orientation options.

labels

= **labels** <*bool*>
Gerber and DXF file export option: output labels.

Layer

= **Layer** (**name** <*string*>) (**type** <*layer_type*>) [**compsOutline** <*bool*>] [**thickness** <*float*>]
Definition of a layer.

! For signal, plane, dielectrical and documentation layers, the compsOutline parameter is not used.

! For documentation layers, the thickness parameter is not used.

LayerGroup

= **LayerGroup** (**name** *<string>*) {[(**LayerRef**) | (**LayerGroupRef**)]}
Definition of a layer group.

LayerGroupRef

= **LayerGroupRef** (**name** *<string>*)
Reference to a layer group.

LayerGroups

= **LayerGroups** {(**LayerGroup**)}
Definition of layer groups.

LayerOptions

= LayerOptions (**LayerRef**) (**Colors**) (**Show**)
Display option: layer visibility options.

LayerRange

= **LayerRange** [(**AllLayers**) | (**LayerRef**) (**LayerRef**)]
Range of layers.

LayerRef

= **LayerRef** [**type** *<layer_type>*] (**name** *<string>*)
Reference to a layer.

! If the design defines only one layer with the specified name, then the layer type is not specified.

LayerRule (Impedance)

= **LayerRule** (**width** *<float>*) (**LayerRef**)
Rule for signal routing on a layer.

LayerRule (ImpedanceDiff)

= **LayerRule** (**width** *<float>*) (**gap** *<float>*) (**LayerRef**)
Rule for differential pair routing on a layer.

Layers

=**Layers** (**version** <*part_version*>) (**StackUpLayers**) (**UnStackLayers**)
Layers partition.

Example:

```
<Layers version="1.1">
   <StackUpLayers>
      <Layer name="ASSEMBLY_TOP_ASSY" type="Assy"
             compsOutline="on"/>
      <Layer name="ASSEMBLY_TOP" type="Mechanical" thickness="0"/>
      <Layer name="SOLDERPASTE_TOP" type="Paste" thickness="0"/>
      <Layer name="SILKSCREEN_TOP" type="Silk" thickness="0"/>
      <Layer name="SOLDERMASK_TOP" type="Mask" thickness="0.508"/>
      <Layer name="1" type="Signal" thickness="0.05"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="2" type="Plane" thickness="0.018"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="3" type="Signal" thickness="0.018"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="4" type="Plane" thickness="0.018"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="5" type="Plane" thickness="0.018"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="6" type="Signal" thickness="0.018"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="7" type="Plane" thickness="0.018"/>
      <Layer name="Pre-preg" type="Dielectric" thickness="0.508"/>
      <Layer name="8" type="Signal" thickness="0.05"/>
      <Layer name="SOLDERMASK_BOT" type="Mask" thickness="0.508"/>
      <Layer name="SILKSCREEN_BOT" type="Silk" thickness="0"/>
      <Layer name="SOLDERPASTE_BOT" type="Paste" thickness="0"/>
      <Layer name="ASSEMBLY_BOT" type="Mechanical" thickness="0"/>
      <Layer name="ASSEMBLY_BOT_ASSY" type="Assy"
             compsOutline="on"/>
   </StackUpLayers>
   <UnStackLayers>
      <Layer name="Default User Layer" type="Doc"/>
      <Layer name="DXF_0" type="Doc"/>
      <Layer name="DXF_Visible narrow (iso)" type="Doc"/>
      <Layer name="DRC Assertion Assistant Layer" type="Doc"/>
      <Layer name="QEDraw" type="Doc"/>
      <Layer name="QEDraw2" type="Doc"/>
      <Layer name="QEDraw3" type="Doc"/>
      <Layer name="QEDrawText" type="Doc"/>
      <Layer name="DRILLDRAWING_THRU" type="Doc"/>
      <Layer name="Notes" type="Doc"/>
      <Layer name="ASSEMBLY_TOP" type="Doc"/>
      <Layer name="ASSEMBLY_BOTTOM" type="Doc"/>
      <Layer name="DRILLDRAWING_THRU_1" type="Doc"/>
   </UnStackLayers>
</Layers>
```

LayersRef

= [(**AllLayers**) | (**AllLayersInner**) | (**AllLayersInnerSignal**) |
(**AllLayersSignal**) | (**AllLayersOuter**) |
(**LayerGroupRef**) | (**LayerRef**) {(**LayerRef**)}]
Reference to layers.

LayersVisualOptions

= **LayersVisualOptions** {(**LayerOptions**)}
Display option: layer visibility options.

LayerTypeRef

= **LayerTypeRef** (**type** <*layer_type*>)
Reference to a layer type.

length

= **length** <*float*>
Serpent parameter: required length.

Line

= **Line** (**Dot**) (**Dot**)
Definition of a line.

lineClr

= **lineClr** <*float*>
Copper pour (polygon) parameter: spacing between hatching lines.

lineWidth

= **lineWidth** <*float*>
Width of a line.

LocalLibrary

= **LocalLibrary** (**version** <*part_version*>)
(**Padstacks**) (**Viastacks**)
(**Footprints**) (**Components**) (**Packages**)
LocalLibrary partition.

logFileName

= logFileName <*filename*>
DRC option: file to write a report to.

maxNetsInCluster

= maxNetsInCluster <*positive_integer*>
Maximum number of nets in a signal cluster. The parameter is used during automatic detection of nets in a signal cluster.

MechLayerObjects

= **MechLayerObjects** {(**Detail**)}
Objects on mechanical layers.

MessageFilter

= **MessageFilter** (**showWarnings** [**ShowChecked** | **ShowAll** | **ShowNothing**])
(**W5003** <*bool*>) (**W5012** <*bool*>) (**W5013** <*bool*>) (**W5014** <*bool*>)
(**W5015** <*bool*>) (**W5016** <*bool*>) (**W5017** <*bool*>) (**W5018** <*bool*>)
(**W5023** <*bool*>) (**W5024** <*bool*>) (**W5026** <*bool*>) (**W5034** <*bool*>)
(**W5036** <*bool*>) (**W5037** <*bool*>) (**WClrnBtwComps** <*bool*>)
(**WClrnBtwObjSameNet** <*bool*>)
Message filter configuration.

messageLimit

= **messageLimit** <*integer*>
DRC option: maximum number of messages.

metallized

= **metallized** <*bool*>
Padstack parameter: whether the hole is metallized.

minPinsNumber

= minPinsNumber <*positive_integer*>
Minimal number of pins in the power net. The parameter is used for automatic detection of power nets.

minSpokeNum

= **minSpokeNum** <*positive_integer*>
Thermal pad parameter: minimum number of spokes.

minSquare

= minSquare <*float***>**
Copper pour (polygon) parameter: minimum island area.

mirror

= **mirror** <*bool*>
Text and label parameter: whether the text is mirrored.

mirror (ExportFile)

= **mirror** <*bool*>
Gerber file export option: output a mirrored layer.

mismatch

= **mismatch** <*float*>
Differential pair parameter: acceptable mismatch between the lengths of the pair's wires.

mode

= mode [**Multilayer** | **SinglelayerTop** | **SinglelayerBottom**]
Autorouting option: routing mode.

Value	Description
Multilayer	Multilayer routing
SinglelayerTop	Single-layer routing on the top layer
SinglelayerBottom	Single-layer routing on the bottom layer

Default value: Multilayer.

Mnthole (Footprint\Mntholes)

= **Mnthole** (**id** <*string*>) (**PadstackRef**) (**Org**)
Definition of mounting hole in a component footprint.

Mnthole (CompInstance\Mntholes)

= **Mnthole** (**mntholeRef** <*string*>) (**angle** <*float*>)
(**PadstackRef**) [**NetRef**] (**Org**)
Definition of mounting hole in a component on the board.

mntholeRef

= **mntholeRef** <*string*>
Reference to a mounting hole.

MntholeInstance

= **MntholeInstance** (**angle** <*float*>) (**PadstackRef**) [**NetRef**] (**Org**)
Definition of mounting hole on the board.

Mntholes (Constructive)

= **Mntholes** {(**MntholeInstance**)}
Definition of mounting holes on the board.

Mntholes (Footprint, CompInstance)

= **Mntholes** {(**Mnthole**)}
Definition of mounting holes.

name

= **name** <*string*>
Name of an object or reference to a named object.

narrow

= **narrow** <*color*>
Display option: color of a decreased nominal clearance.

negStr

= negStr <*string*>
Parameter in a rule for naming differential signal nets: substring that defines the net of the negative signal.

negative

= **negative** <*bool*>
Gerber file export option: output an inverted layer.

Net

= **Net** (**name** <*string*>) {[(**PinRef**) | (**PadRef**)]}
Definition of a net.

NetGroup

= **NetGroup** (**name** <*string*>) {[(**NetRef**) | (**NetGroupRef**)]}
Definition of a net group.

NetGroupRef

= **NetGroupRef** (**name** <*string*>)
Reference to a net group.

NetGroups

= **NetGroups** {(**NetGroup**)}
Definition of net groups.

netLines

= **netLines** <*color*>
Display option: color of net lines.

NetList

= **NetList** (**version** *<part_version>*) {(**Net**)}
NetList partition.

NetProperties

= **NetProperties** {(**NetProperty**)}
Definition of Net Properties rules.

NetProperty

= **NetProperty** (**flexfix** <bool>) (**route** <bool>) (**NetRef**)
Net Property rule.

NetRef

= **NetRef** (**name** *<string>*)
Reference to a net.

Nets

= **Nets** {(**NetRef**)}
Nets of the signal cluster.

NonfilledCopper

= **NonfilledCopper** (**lineWidth** *<float>*) (**LayerRef**) (**NetRef**) (**Shape**)
Definition of a non-filled copper.

NonfilledCoppers

= NonfilledCoppers {(NonfilledCopper)}
Definition of a non-filled coppers.

NonfilledFigure

= [(**Arc**) | (**Circle**) | (**Line**) | (**Rect**)]
Non-filled figure.

ObjectComp

= [(**ComponentRef**) | (**CompGroupRef**) | (**AllComps**)]
Components affected by the rule.

ObjectLeft

= ObjectLeft (ObjectSignal)
First object affected by the mutual delay equalization rule.

ObjectNet

= [(**NetRef**) | (**NetGroupRef**) | (**AllNets**)]
Nets affected by the rule.

ObjectRight

= ObjectRight (ObjectSignal)
Second object affected by the mutual delay equalization rule.

ObjectSignal

= [(**SignalRef**) | (**DiffSignalRef**) | (**SignalGroupRef**)]
Signals affected by the rule.

ObjectsAffected (WidthOfWires)

= **ObjectsAffected** (ObjectNet)
Objects affected by the rule.

ObjectsAffected (ClearanceNetToNet)

= **ObjectsAffected** [(ObjectNet) | (ObjectSignal)] [(ObjectNet) | (ObjectSignal)]
Objects affected by the rule.

ObjectsAffected (ClearanceCompToComp)

= **ObjectsAffected** (ObjectComp) (ObjectComp)
Objects affected by the rule.

ObjectsAffected (PlaneLayerNets)

= **ObjectsAffected** (**NetRef**) {(**NetRef**)}
Objects affected by the rule.

ObjectsAffected (SignalLayerNets)

= **ObjectsAffected** [(**NetRef**) {(**NetRef**)} | (**NetGroupRef**) {(**NetGroupRef**)}]
Objects affected by the rule.

ObjectsAffected (DelayEqual)

= **ObjectsAffected** (**SignalGroupRef**)
Objects affected by the rule.

ObjectsAffected (DelayConstant)

= **ObjectsAffected** (ObjectSignal)
Objects affected by the rule.

ObjectsAffected (ViastacksOfNets)

= **ObjectsAffected** [(ObjectNet) | (ObjectSignal)]
Objects affected by the rule.

OriginalFile

= **OriginalFile** <*filename*>
Originally imported file. The path to the file is relative to the directory containing the project file.

OriginalFormat

= **OriginalFormat** <*string*>
Format of the originally imported file that the design came from.

OriginalNetList

= **OriginalNetList** (**version** <*part_version*>) {(**Net**)}
OriginalNetList partition.

! The original net list serves as the basis for an ECO file that contains the changes found in the current net list.

Org

= **Org** (**x** <*float*>) (**y** <*float*>)
Origin of an object.

outFile

= **outFile** <*fileName*>
Name of the output file (BOM, DXF).

outPath

= **outPath** <*string*>
Directory for output files (Gerber, Drill).

output (ExportFile)

= output <*bool*>
Gerber file export option: export the file.

output (ExportLayer)

= output <*bool*>
DXF file export option: output the layer.

outputBoardLayer

= **outputBoardLayer** <*bool*>
DXF file export option: output the layer with the board outline.

outputDrillLayer

= **outputDrillLayer** <*bool*>
DXF file export option: output the drill layer.

Package

= **Package** (**ComponentRef**) (**FootprintRef**) {(**Pinpack**)}
Definition of a package (correspondence between component pads and case pins).

Example:

```
<Package>
   <ComponentRef name="ADR03BKS-R2"/>
   <FootprintRef name="SSOP5_.65MMSP_.049X.079B_.083W__DA7"/>
   <Pinpack pinNum="2" padNum="1"/>
   <Pinpack pinNum="1" padNum="2"/>
   <Pinpack pinNum="4" padNum="3"/>
   <Pinpack pinNum="5" padNum="4"/>
   <Pinpack pinNum="3" padNum="5"/>
</Package>
```

Packages

=**Packages** {(**Package**)}
Definition of packages.

Pad

= **Pad** (**padNum** <*positive_integer*>) (**name** <*string*>)
(**angle** <*float*>) (**flipped** <*bool*>)
(**PadstackRef**) (**Org**)
Definition of a case pad (pin).

! The TopoR system supports planar pads on outer metal layers and does not support them on inner layers. This means that a planar pad can be either on the top or on the bottom side. The definition of a planar pad uses only the Top layer. Therefore, a pad will be on the same side as the component. If the pad is on the opposite side, the flipped flag must be set. This flag is set in the pattern pad definition.

PadCircle

= **PadCircle** (**diameter** <*float*>) [(**LayerTypeRef**) | (**LayerRef**)]
Definition of a circular pad.

Example:

```
<PadCircle diameter="0.6">
   <LayerRef type="Signal" name="1"/>
</PadCircle>
```

padNum

= **padNum** <*positive_integer*>
Number of a case pad (pin).

PadOval

= **PadOval** (**diameter** <*float*>) [(**LayerTypeRef**) | (**LayerRef**)]
(**Stretch**) (**Shift**)
Definition of an oval pad.

PadPoly

= **PadPoly** [(**LayerTypeRef**) | (**LayerRef**)] (**Dot**) (**Dot**) (**Dot**) {(**Dot**)}
Definition of a polygonal pad.

PadRect

= **PadRect** (**width** <*float*>) (**height** <*float*>)
[(**LayerTypeRef**) | (**LayerRef**)] (**Shift**)
Definition of a rectangular pad.

PadRef

= **PadRef** (**compName** <*string*>) (**padNum** <*positive_integer*>)
Reference to a case pad.

padToBoard

= **padToBoard** <*bool*>
DRC option: check clearances between pads and the edge of the board.

padToKeepout

= **padToKeepout** <*bool*>
DRC option: check clearances between pads and keepouts.

padToPad

= **padToPad** <*bool*>
DRC option: check clearances between pads.

Pads (Footprint)

= **Pads** {(**Pad**)}
Definition of case pads.

Pads (Padstack)

= **Pads** {[(**PadCircle**) | (**PadOval**) | (**PadRect**) | (**PadPoly**)]}
Definition of stack pads.

pads (Colors)

= **pads** <*color*>

Layer display option: color of pads.

pads (Show)

= **pads** <*bool*>

Layer display option: visibility of pads.

Padstack

= **Padstack** (**name** <*string*>) (**type** [**Through** | **SMD** | **MountHole**])
(**holeDiameter** <*float*>) (**metallized** <*bool*>)
(**connectToCopper** [**NoneConnect** | **Direct** | **Thermal**])
(**Thermal**) (**Pads**)

Definition of a padstack.

PadstackRef

= **PadstackRef** (**name** <*string*>)

Reference to a padstack.

Padstacks

= **Padstacks** {(**Padstack**)}

Definition of padstacks.

padstacks

= **padstacks** <*bool*>

Gerber, DXF file export option: output pads.

partName

= partName <*bool*>

BOM file export option: output component names.

Pin (CompInstance\Pins)

= **Pin** (**padNum** <*positive_integer*>) [**PadstackRef**)] (**Org**)

Definition of a component on board pin.

! If PadStackRef is not specified, then the padstack is taken from the pattern.

Pin (Component\Pins)

= **Pin** (**pinNum** <*positive_integer*>) (**name** <*string*>) (**pinSymName** <*string*>)
(**pinEqual** <*positive_integer*>) (**gate** <*positive_integer*>) (**gateEqual** <*positive_integer*>)

Definition of a library component pin.

pinEqual

= **pinEqual** <*positive_integer*>
Component pin parameter: equivalence.

pinName

= **pinName** <*string*>
Name of a component pin; used for reference.

pinsName

= **pinsName** <*color*>
Display option: color of pin names.

pinsNet

= **pinsNet** <*color*>
Display option: color of net names on pins.

pinNum

= **pinNum** <*positive_integer*>
Number of a component pin.

Pinpack

= **Pinpack** (**pinNum** <*positive_integer*>) (**padNum** <*positive_integer*>)
[**valueType** [**Dist** | **Time**] [**delay** <*float*>]]
Correspondence between a component pin and a case pad.

PinPairs

= **PinPairs** {(**PinPair**)}
Definition of pin pairs.

PinPair

= **PinPair** (**PinRef**) (**PinRef**)
Definition of pin pair.

PinRef

= **PinRef** (**compName** <*string*>) (**pinName** <*string*>)
Reference to a pin.

Pins (CompInstance)

= **Pins** {(**Pin**)}
Definition of a component on board pins.

Pins (Component)

= **Pins** {(**Pin**)}
Definition of a library component pins.

pinSymName

= **pinSymName** <*string*>
Symbolic name of a component pin.

Place

= **Place** (**side** [**Top** | **Bottom** | **Both**])
Keepout type: placement keepout.

Placement

= **Placement** (**PlacementArea**)
Configure automatic component placement.

PlacementArea

= **PlacementArea** (**Dot**) (**Dot**)
Area for automatic component placement.
This is a rectangular area specified by two vertices (top left and bottom right).

PlaneLayerNets

= PlaneLayerNets (**enabled** <*bool*>) (LayersRef) (**ObjectsAffected**)
Definition of a rule for assigning plane layers to nets.

Polygon

= **Polygon** [**width** <*float*>] (**Dot**) (**Dot**) (**Dot**) {(**Dot**)}
Definition of a polygon.

posStr

= posStr <*string*>
Parameter in a rule for naming differential signal nets: substring that defines the net of the positive signal.

precision

= **precision** [**Low** | **Med** | **High**]
Copper pour (polygon) parameter: outline approximation precision.

Value	Description
Low	Low precision
Med	Medium precision
High	High precision

Default value: Med.

preference

= **preference** [**Metric** | **mkm** | **mm** | **cm** | **dm** | **m** | **Imperial** | **mil** | **inch**]
Display option: measurement units.

Value	Description
Metric	Metric; the actual units used are parameter-dependent
mkm	Micrometers
mm	Millimeters
cm	Centimeters
dm	Decimeters
m	Meters
Imperial	Imperial; the actual units used are parameter-dependent
mil	Mils (thousandths of an inch)
inch	Inches

Default value: Metric

priority

= **priority** <*integer*>
Copper pour (polygon) parameters: pouring priority.

Program

= **Program** <*string*>
Name of the program that wrote the file.

ReceiverPinRef

= **ReceiverPinRef** (**compName** <*string*>) (**pinName** <*string*>)
Reference to a signal receiver pin.

Rect

= **Rect** (**Dot**) (**Dot**)
Definition of a non-filled rectangle. The top left and bottom right vertices are specified.

refDes

= **refDes** *<bool>*
BOM file export option: output the reference numerals of components.

Role

= **Role** [(**Trace**) | (**Place**)]
Keepout type.

role

= **role** [**Wires** | **Vias** | **WiresAndVias**]
Routing keepout type.

Value	Description
Wires	Wire keepout
Vias	Via keepout
WiresAndVias	Wire and via keepout

Default value: Wires.

rotateWithComp

= **rotateWithComp** *<bool>*
Label orientation option: rotate the label with the component.

route

= route *<bool>*
Property of net: routing flag for autorouter.

Rules

= **Rules** (**version** *<part_version>*) (**RulesWidthOfWires**) (**RulesClearancesNetToNet**)
(**RulesClearancesCompToComp**) (**RulesClearancesToBoard**) (**RulesViastacksOfNets**)
(**RulesPlaneLayersNets**) (**RulesSignalLayersNets**) (**NetsProperties**)
Rules partition.

! The order of the rules in a section determines the priority of the rules. The higher the priority the lower the rule is located.

RulesClearancesCompToComp

= **RulesClearancesCompToComp** {(**ClearanceCompToComp**)}
Definition of rules for clearance between components.

RulesClearancesNetToNet

= **RulesClearancesNetToNet** {(**ClearanceNetToNet**)}
Definition of rules for clearance between nets.

RulesClearancesToBoard

= **RulesClearancesToBoard** (**wires** <*float*>) (**comps** <*float*>)
Definition of rules for distance to the edge of the board.

RulesDelay

= **RulesDelay** {(**DelayEqual**)} {(**DelayConstant**)} {(**DelayRelation**)}
Definition of rules for delay equalization.

RuleDiffSignalNetsName

= **RuleDiffSignalNetsNames** (**enabled** <*bool*>) (**posStr** <*string*>) (**negStr** <*string*>)
Rule for naming differential signal nets.

RulesDiffSignalNetsNames

= **RulesDiffSignalNetsNames** {(**RuleDiffSignalNetsName**)}
Rules for naming differential signal nets.

! The order of the rules in this section determines the priority of the rules. The higher the priority the higher the rule is located.

RulesImpedances

= **RulesImpedances {[(Impedance) | (ImpedanceDiff)]}**
Impedances and rules for signals and differential pairs layout.

RulesPlaneLayersNets

= **RulesPlaneLayersNets** {(**PlaneLayerNets**)}
Definition of rules for assigning plane layers to nets.

RulesSignalLayersNets

= **RulesSignalLayersNets** {(**SignalLayerNets**)}
Definition of rules for assigning signal layers to nets.

RulesViastacksOfNets

= **RulesViastacksOfNets** {(**ViastacksOfNets**)}
Definition of rules for assigning viastacks to nets.

RulesWidthOfWires

= **RulesWidthOfWires** {(**WidthOfWires**)}
Definition of rules for wire widths.

scale

= **scale** <*float*>
Current view parameter: scale.

scrollHorz

= **scrollHorz** <*float*>
Current view parameter: horizontal scrolling.

scrollVert

= **scrollVert** <*float*>
Current view parameter: vertical scrolling.

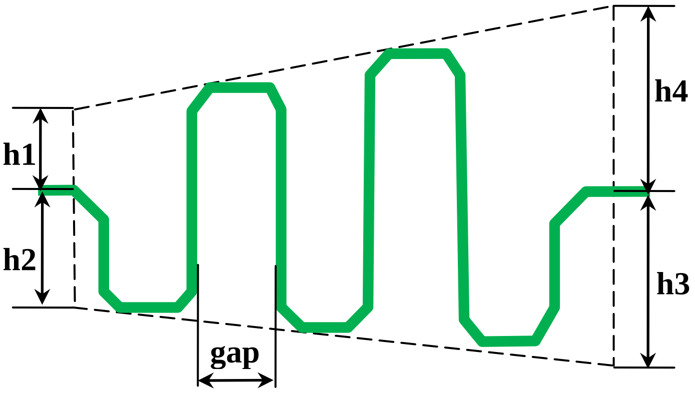
SetColor

= **SetColor** (**color** <*color*>) [(ObjectNet) | (ObjectSignal)]
Net color overrides: set color for net/ signal/ net group/ signal group.

Serpent

= **Serpent** (**id** *<string>*) (**length** *<float>*) (**gap** *<float>*)
(**h1** *<float>*) (**h2** *<float>*) (**h3** *<float>*) (**h4** *<float>*)

Definition of a serpent.



! The wires that form the serpent are defined in the **Wires** section (see Connectivity).

Serpents

= **Serpents** {(**Serpent**)}

Definition of serpents.

serpRef

= **serpRef** *<string>*

Reference to a serpent. The string must contain the ID of the Serpent.

Settings

= **Settings** (**Autoroute**) (**Placement**) (**Labels**)

Settings partition.

Shape (Contour)

= **Shape** (**lineWidth** *<float>*) (NonfilledFigure)

Definition of the board outline primitive.

Shape (Copper)

= **Shape** (FilledFigure)

Definition of the outline of a filled copper pour.

Shape (NonfilledCopper)

= **Shape** (NonfilledFigure)

Definition of the outline of a non-filled copper pour.

Shape (Voids)

= **Shape** (**lineWidth** *<float>*) (FilledFigure)
Definition of voids in the board.

Shift (ExportFile)

= **Shift** (**x** *<float>*) (**y** *<float>*)
Gerber file export option: shift of objects in the X and Y axes.

Shift (PadOval, PadRect)

= **Shift** (**x** *<float>*) (**y** *<float>*)
Pad parameter: shift of the origin in the X and Y axes.

Show (DisplayControl)

= **Show** (**displayScheme** *<string>*) (**showBoardOutline** *<bool>*)
(**showWires** *<bool>*) (**showCoppers** *<bool>*)
(**showTexts** *<bool>*) (**throughPad** *<bool>*) (**throughVia** *<bool>*)
(**burriedVia** *<bool>*) (**blindVia** *<bool>*) (**fixedVia** *<bool>*) (**showVias** *<bool>*)
(**showSignalLayers** *<bool>*) (**showTopMechLayers** *<bool>*)
(**showBotMechLayers** *<bool>*) (**showDocLayers** *<bool>*)
(**showTopMechDetails** *<bool>*) (**showBotMechDetails** *<bool>*)
(**showMetalPads** *<bool>*) (**showTopMechPads** *<bool>*)
(**showBotMechPads** *<bool>*) (**showNetLines** *<bool>*)
(**showMountingHoles** *<bool>*) (**showThinWires** *<bool>*)
(**showComponents** *<bool>*) (**showCompTop** *<bool>*) (**showCompBot** *<bool>*)
(**showCompsDes** *<bool>*) (**showPinsName** *< bool>*) (**showPinsNet** *<bool>*)
(**showCompsBound** *<bool>*) (**showLabelRefDes** *<bool>*)
(**showLabelPartName** *<bool>*) (**showLabelOther** *<bool>*)
(**showViolations** *<bool>*) (**showNarrow** *<bool>*)
(**showTrimmed** *<bool>*) (**showDRCViolations** *<bool>*)
(**showKeepouts** *<bool>*) (**showRouteKeepouts** *<bool>*) (**showPlaceKeepouts** *<bool>*)
(**showActiveLayerOnly** *<bool>*) (**showSerpentArea** *<bool>*)
Display option: object visibility settings.

Show (LayerOptions)

= **Show** (**visible** *<bool>*) (**details** *<bool>*) (**pads** *<bool>*)
Layer display option: visibility settings.

showActiveLayerOnly

= **showActiveLayerOnly** *<bool>*
Display option: show only the active layer.

showBoardOutline

= showBoardOutline *<bool>*
Display option: show board outline.

showBotMechDetails

= **showBotMechDetails** <*bool*>
Display option: show details on bottom mechanical layers.

showBotMechLayers

= **showBotMechLayers** <*bool*>
Display option: show bottom mechanical layers.

showBotMechPads

= **showBotMechPads** <*bool*>
Display option: show pads on bottom mechanical layers.

showCompBot

= **showCompBot** <*bool*>
Display option: show components on the bottom side.

showComponents

= **showComponents** <*bool*>
Display option: show components.

showCompTop

= **showCompTop** <*bool*>
Display option: show components on the top side.

showCompsBound

= **showCompsBound** <*bool*>
Display option: show components bounding boxes.

showCompsDes

= **showCompsDes** <*bool*>
Display option: show reference designations of components.

showCoppers

= **showCoppers** <*bool*>
Display option: show copper pours (polygons).

showDocLayers

= **showDocLayers** <*bool*>
Display option: show documentation layers.

showDRCViolations

= showDRCViolations *<bool>*
Display option: show DRC violations.

showKeepouts

= showKeepouts *<bool>*
Display option: show keepouts.

showLabelOther

= showLabelOther *<bool>*
Display option: show labels of custom attributes.

showLabelPartName

= showLabelPartName *<bool>*
Display option: show labels of the PartName attribute.

showLabelRefDes

= showLabelRefDes *<bool>*
Display option: show labels of the RefDes attribute.

showMetalPads

= showMetalPads *<bool>*
Display option: show pads on metal layers.

showMountingHoles

= showMountingHoles *<bool>*
Display option: show mounting holes.

showNarrow

= showNarrow *<bool>*
Display option: show nominal clearance decreases.

showNetLines

= showNetLines *<bool>*
Display option: show net lines.

showPinsName

= showPinsName *<bool>*
Display option: show pin names.

showPinsNet

= **showPinsNet** *<bool>*

Display option: show net names on pins.

showPlaceKeepouts

= **showPlaceKeepouts** *<bool>*

Display option: show placement keepouts.

showRouteKeepouts

= **showRouteKeepouts** *<bool>*

Display option: show routing keepouts.

showSerpentArea

= **showSerpentArea** *<bool>*

Display option: show serpent areas.

showSignalLayers

= **showSignalLayers** *<bool>*

Display option: show signal layers.

showTexts

= **showTexts** *<bool>*

Display option: show text labels.

showThinWires

= **showThinWires** *<bool>*

Display option: show wires as thin lines.

showTopMechDetails

= **showTopMechDetails** *<bool>*

Display option: show details on top mechanical layers.

showTopMechLayers

= **showTopMechLayers** *<bool>*

Display option: show top mechanical layers.

showTopMechPads

= **showTopMechPads** *<bool>*

Display option: show pads on top mechanical layers.

showTrimmed

= **showTrimmed** <*bool*>
Display option: show wire width decreases.

showVias

= **showVias** <*bool*>
Display option: show vias.

showViolations

= **showViolations** <*bool*>
Display option: violations.

showWarnings

= **showWarnings** [**ShowChecked** | **ShowAll** | **ShowNothing**])
Message filter option: warning display mode.

Value	**Description**
ShowChecked	Show only warning of selected types
ShowAll	Show all warnings
ShowNothing	Do not show any warnings

Default value: ShowChecked.

showWires

= **showWires** <*bool*>
Display option: show wires.

side

= **side** [**Top** | **Bottom** | **Both**]
Side of an object.

! The Both value is valid only for placement keepouts.

Signal

= **Signal** (**name** <*string*>) (**ReceiverPin**) (**Components**)
Definition of a signal.

SignalCluster

= **SignalCluster** (**ImpedanceRef**) (**SourcePin**) (**Nets**) (**PinPairs**) {(**Signal**)}
Definition of a signal cluster.

SignalClusters

= **SignalClusters** {(**SignalCluster**)}
Definition of a signal clusters.

SignalGroup

= **SignalGroup** (**name** *<string>*) {[(**SignalRef**) | (**DiffSignalRef**) | (**SignalGroupRef**)]}
Definition of a signal group.

SignalGroupRef

= **SignalGroupRef** (**name** *<string>*)
Reference to a signal group.

SignalGroups

= **SignalGroups** {(**SignalGroup**)}
Definition of a signal groups.

SignalLayerNets

= SignalLayerNets (**enabled** *<bool>*) (LayersRef) (**ObjectsAffected**)
Definition of a rule for assigning signal layers to nets.

SignalRef

= **SignalRef** (**name** *<string>*)
Reference to a signal.

SignalSearchSettings

= **SignalSearchSettings** (**maxNetsInCluster** *<positive_integer>*)
(**createPinPairs** *<bool>*) (**RulesDiffSignalNetsNames**)
(**ExcludedNets**)
Signals automatic search options.

snapToAngle

= **snapToAngle** *<bool>*
Manual editing option: snap rotations to 45-degree angles.

SourcePinRef

= **SourcePinRef** (**compName** *<string>*) (**pinName** *<string>*)
Reference to a signal source pin.

spokeNum

= **spokeNum** <*positive_integer*>
Thermal pad parameter: number of spokes.

! TopoR support only one value - 4.

spokeWidth

= **spokeWidth** <*float*>
Thermal pad parameter: width of a spoke.

StackUpLayers

= **StackUpLayers** {(**Layer**)}
Definition of layers in a stack.

Start

= **Start** (**x** <*float*>) (**y** <*float*>)
Starting point of a line of arc.

state

= **state** [**Unpoured** | **Poured** | **Locked**]
Copper pour (polygon) parameter: state.

Value	Description
Unpoured	Unpoured
Poured	Poured
Locked	Poured and locked

Default value: Unpoured.

Stretch

= **Stretch** (**x** <*float*>) (**y** <*float*>)
Oval pad parameter: stretch rate in the X and Y axes.

Subwire

= **Subwire** (**fixed** <*bool*>) (**width** <*float*>) [**zipwireRef** <*string*>]
[**Teardrops**] (**Start**) (Track){(Track)}
Definition of a subwire (series of segments that have the same width and fix state).

! The zipwireRef attribute (reference to a zipped wire pair) is used if the subwire in question is included in a ZippedWire pair (see this example of a differential wire pair definition).

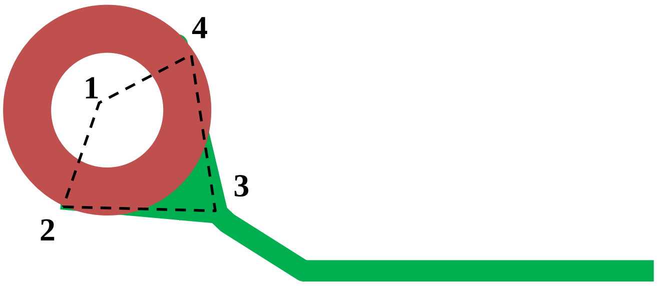
takeCurLayout

= **takeCurLayout** <*bool*>
Autoroting parameter

Teardrop

= **Teardrop** (**Dot**) (**Dot**) (**Dot**) (**Dot**)

Definition of a drop shape by means of a quadrangle. The first vertex corresponds to the origin of the pad or via. The rest of the vertices specify the outline of the quadrangle in a counterclockwise order.



! During import, TopoR ignores information about teardrops.

Teardrops

= **Teardrops** [**Teardrop**] [**Teardrop**]

Definition of teardrops for Subwire.

teardrops

= **teardrops** <*bool*>

Autorouting parameter: create teardrops.

Text

= **Text** (**text** <*string*>) (**align** <*align_type*>) (**angle** <*float*>) (**mirror** <*bool*>) (**LayerRef**) (**TextstyleRef**) (**Org**)

Definition of a text label.

text

= **text** <*string*>

Text label parameter: the text.

Texts

= **Texts** {(**Text**)}

Definition of text labels.

texts

= **texts** <*bool*>

Gerber and DXF file export option: output text labels.

TextStyle

= **TextStyle** (**name** <*string*>) (**fontName** <*string*>) (**height** <*float*>)
(**bold** <*bool*>) (**italic** <*bool*>)
Definition of a text label style.

TextStyleRef

= **TextStyleRef** (**name** <*string*>)
Reference to a text label style.

TextStyles

= **TextStyles** (**version** <*part_version*>) {(**TextStyle**)}
TextStyles partition.

textToBoard

= **textToBoard** <*bool*>
DRC option: check clearances between text labels and the edge of the board.

textToCopper

= **textToCopper** <*bool*>
DRC option: check clearances between text labels and copper pours (polygons).

textToKeepout

= **textToKeepout** <*bool*>
DRC option: check clearances between text labels and keepouts.

textToPad

= **textToPad** <*bool*>
DRC option: check clearances between text labels and pads.

textToVia

= **textToVia** <*bool*>
DRC option: check clearances between text labels and vias.

textToWire

= **textToWire** <*bool*>
DRC option: check clearances between text labels and wires.

Thermal

= **Thermal** (**spokeNum** <*positive_integer*>) (**minSpokeNum** <*positive_integer*>)
(**angle** <*float*>) (**spokeWidth** <*float*>) (**backoff** <*float*>)
Definition of a thermal pad.

ThermalPad

= **ThermalPad** (**Thermal**)

Definition of a thermal pad for connecting pads to a copper pour.

ThermalSpoke

= **ThermalSpoke** (**lineWidth** <*float*>) (**Dot**) (**Dot**)

Definition of a spoke in a thermal pad that is present on the board.

ThermalVia

= **ThermalVia** (**Thermal**)

Definition of a thermal pad for connecting vias to a copper pour.

thickness

= **thickness** <*float*>

Layer parameter: thickness.

throughPad

= **throughPad** <*bool*>

Display option: mark through pads with color

throughVia

= **throughVia** <*bool*>

Display option: mark through vias with color.

time

= **time** [**fs** | **ps** | **ns** | **us**]

Time units for the file.

Value	Description
fs	Femtoseconds
ps	Picoseconds
ns	Nanoseconds
us	Microseconds

Default value: ps.

tolerance (DRCSettings)

= **tolerance** <*float*>

DRC option: tolerance.

tolerance (DelayEqual)

= **tolerance** <*float*>
Parameter of a rule for delay equalization within a net group: tolerance.

! The measurement units depend on the valueType parameter and the units set for the file (see Units).

toleranceOver

= **toleranceOver** <*float*>
Parameter of a rule for delay equalization: upper bound for the tolerance.

! The measurement units depend on the valueType parameter and the units set for the file (see Units).

toleranceUnder

= **toleranceUnder** <*float*>
Parameter of a rule for delay equalization: lower bound for the tolerance.

! The measurement units depend on the valueType parameter and the units set for the file (see Units).

topHorzRotate

= **topHorzRotate** <*bool*>
Label orientation option: rotate horizontally-oriented labels on the top side.

topVertRotate

= **topVertRotate** <*bool*>
Label orientation option: rotate horizontally-oriented labels on the top side.

TopoR_PCB_File

= **TopoR_PCB_File** (**Header**) (**Layers**) (**TextStyles**) (**LocalLibrary**) (**Constructive**)
(**ComponentsOnBoard**) (**NetList**) [**OriginalNetList**]
(**Groups**) (**HiSpeedRules**) (**Rules**) (**Connectivity**)
(**Settings**) (**DisplayControl**) (**DialogSettings**)
Root tag; nests all partitions of the file.

Trace

= **Trace** (**role** [**Wires** | **Vias** | **WiresAndVias**]) (LayersRef)
Keepout type: routing keepout.

Track

= [(**TrackLine**) | (**TrackArc**) | (**TrackArcCW**)]
Definition of a wire segment.

TrackLine

= **TrackLine** (**End**) [**serpRef** <*string*>]
Definition of a linear wire segment.

! The starting point of the segment is taken from the preceding segment of set by the Start tag specified in SubWire.

! If the segment is included in a serpent, a serpent reference serpRef is used.

TrackArc

= **TrackArc** (**Center**) (**End**) [**serpRef** <*string*>]
Definition of an arc-shaped wire segment (counterclockwise arc).

! The starting point of the segment is taken from the preceding segment of set by the Start tag specified in SubWire.

! If the segment is included in a serpent, a serpent reference serpRef is used.

TrackArcCW

= **TrackArcCW** (**Center**) (**End**) [**serpRef** <*string*>]
Definition of an arc-shaped wire segment (clockwise arc).

! The starting point of the segment is taken from the preceding segment of set by the Start tag specified in SubWire.

! If the segment is included in a serpent, a serpent reference serpRef is used.

trimmed

= **trimmed** <*color*>
Display option: color of wire width decreases.

type (Attribute)

= **type** [**RefDes** | **PartName**]
Type of predefined component attribute.

Значение	Описание
RefDes	reference designation
PartName	component name

Значение по умолчанию – RefDes.

type (Layer, LayerRef, LayerTypeRef)

= **type** <*layer_type*>
Layer type.

type (Padstack)

= **type** [**Through** | **SMD** | **MountHole**]
Padstack type.

Value	Description
Through	Through
SMD	Surface-mounted
MountHole	Mounting hole

Default value: Through.

Units (Header)

= **Units (dist** [**mkm** | **mm** | **cm** | **dm** | **m** | **mil** | **inch**]) (**time** [**fs** | **ps** | **ns** | **us**])
Measurement units for the file.

Units (DisplayControl)

=**Units** (**preference** [**Metric** | **mkm** | **mm** | **cm** | **dm** | **m** | **Imperial** | **mil** | **inch**])
Display option: measurement units.

units

= **units** [**mm** | **mil**]
Gerber, DXF and Drill file export option: measurement units.

Value	Description
mm	Millimeters
mil	Mils (thousandths of an inch)

Default value: mm.

UnStackLayers

= **UnStackLayers** {(**Layer**)}
Definition of unstacked layers.

useBackoff

= **useBackoff** <*bool*>
Copper pour (polygon) parameter: use the specified backoff.

useOrientRules

= **useOrientRules** <*bool*>
Label editing option: use orientation rules.

value

= **value** <*string*>
Value of an attribute.

valueType

= **valueType** [**Dist** | **Time**]
Parameter of a rule for delay equalization: type of constant and tolerance values.

Value	Description
Dist	Distance
Time	Time

Default value: Dist.

version

= **version** <*part_version*>
Version of a partition.

Version

= **Version** <*format_version*>
Version of the format.

Via

= **Via** (**fixed** <*bool*>) (**ViastackRef**) (**NetRef**) (**Org**)
Via on the board.

Example:

```
<Via>
   <ViastackRef name="Via Round 25 Drill 13_NSM"/>
   <NetRef name="GND"/>
   <Org x="-5.6896" y="-10.9728"/>
</Via>
```

viaOnPin

= **viaOnPin** <*bool*>
Via type parameter: whether the via can be put on a pin.

ViaPads

= **ViaPads** {(**PadCircle**)}
Definition of viastack pads.

Vias

=**Vias** {(**Via**)}
Vias on the board.

vias

= **vias** <*bool*>
Gerber and DXF file export option: output vias.

Viastack

= **Viastack** (**name** <*string*>) (**holeDiameter** <*float*>) (**viaOnPin** <*bool*>)
(**LayerRange**) (**ViaPads**)
Definition of a via (viastack) type.

Example:

```
<Viastack name="Via Round 25 Drill 13_NSM" holeDiameter="0.3302">
   <LayerRange>
      <AllLayers/>
   </LayerRange>
   <ViaPads>
      <PadCircle diameter="0.635">
         <LayerTypeRef type="Signal"/>
      </PadCircle>
      <PadCircle diameter="0.5">
         <LayerTypeRef type="Plane"/>
      </PadCircle>
   </ViaPads>
</Viastack>
```

ViastackRef

= **ViastackRef** (**name** <*string*>)
Reference to a via (viastack) type.

Viastacks (LocalLibrary)

= **Viastacks** {(**Viastack**)}
Definition of via (viastack) types.

Viastacks (ViastacksOfNets)

= **Viastacks** [**AllViastacks** | **AllViastacksThrough** | **AllViastacksNotThrough** | {(**ViastackRef**)}]
Assigned via types.

ViastacksOfNets

= **ViastacksOfNets** (**enabled** <*bool*>) (**ObjectsAffected**) (**Viastacks**)
Definition of a rule for assigning viastacks to nets.

viaToBoard

= **viaToBoard** <*bool*>
DRC option: check clearances between vias and the edge of the board.

viaToKeepout

= **viaToKeepout** <*bool*>
DRC option: check clearances between vias and keepouts.

viaToPad

= viaToPad <*bool*>
DRC option: check clearances between vias and pads.

viaToVia

= **viaToVia** <*bool*>
DRC option: check clearances between vias.

View

= View (**scale** <*float*>) (**scrollHorz** <*float*>) (**scrollVert** <*float*>)
Display option: current view parameters.

visible

= **visible** <*bool*>
Whether a layer is visible.

Voids (BoardOutline)

= **Voids** {(**Shape**)}
Voids in the board.

Voids (Copper)

= **Voids** {(FilledFigure)}
User-defined voids in copper pours (polygons).

Voids (Island)

= **Voids** {(**Polygon**)}
Voids in a copper island.

W5003

= **W5003** <*bool*>
Message filtering option: show message 5003.

W5012

= **W5012** <*bool*>
Message filtering option: show message 5012.

W5013

= **W5013** <*bool*>

Message filtering option: show message 5013.

W5014

= **W5014** <*bool*>

Message filtering option: show message 5014.

W5015

= **W5015** <*bool*>

Message filtering option: show message 5015.

W5016

= **W5016** <*bool*>

Message filtering option: show message 5016.

W5017

= **W5017** <*bool*>

Message filtering option: show message 5017.

W5018

= **W5018** <*bool*>

Message filtering option: show message 5018.

W5023

= **W5023** <*bool*>

Message filtering option: show message 5023.

W5024

= **W5013** <*bool*>

Message filtering option: show message 5013.

W5026

= **W5026** <*bool*>

Message filtering option: show message 5026.

W5034

= **W5034** <*bool*>

Message filtering option: show message 5034.

W5036

= **W5036** <*bool*>
Message filtering option: show message 5036.

W5037

= **W5037** <*bool*>
Message filtering option: show message 5037.

WClrnBtwComps

= **WClrnBtwComps** <*bool*>
Message filtering option: quick checking of clearances between components.

WClrnBtwObjSameNet

= **WClrnBtwObjSameNet** <*bool*>
Message filtering option: quick checking of clearances between objects in the same net.

weakCheck

= **weakCheck** <*bool*>
Autorouting parameter: weak checking of clearances.

width (Subwire, LayerRule)

= **width** <*float*>
Width of a wire.

width (PadRect)

= **width** <*float*>
Width of a rectangular pad.

widthMin

= **widthMin** <*float*>
Wire width rule parameter: minimum wire width.

widthNom

= **widthNom** <*float*>
Wire width rule parameter: nominal wire width.

WidthOfWires

= **WidthOfWires** (**enabled** <*bool*>) (**widthMin** <*float*>) (**widthNom** <*float*>)
(LayersRef) (**ObjectsAffected**)
Definition of a wire width rule.

Wire

= **Wire** (LayerRef) (NetRef) {(Subwire)}
Definition of a wire.

Example of a definition of a wire in a differential pair:

```
<Wire>
   <LayerRef name="Top"/>
   <NetRef name="MEM_CLK#"/>
   <Subwire width="0.2">
      <Start x="171.65" y="105.4"/>
      <TrackLine>
         <End x="170.417" y="104.391"/>
      </TrackLine>
   </Subwire>
   <Subwire width="0.2" zipwireRef="zwire_1">
      <Start x="170.417" y="104.391"/>
      <TrackLine>
         <End x="170.417" y="102.836"/>
      </TrackLine>
      <TrackLine>
         <End x="170.417" y="102.773"/>
      </TrackLine>
      <TrackLine>
         <End x="170.379" y="102.654"/>
      </TrackLine>
      <TrackLine>
         <End x="170.343" y="102.604"/>
      </TrackLine>
      <TrackLine>
         <End x="169.138" y="100.911"/>
      </TrackLine>
   </Subwire>
   <Subwire width="0.2">
      <Start x="169.138" y="100.911"/>
      <TrackLine>
         <End x="170.4" y="100.4"/>
      </TrackLine>
   </Subwire>
</Wire>
```

Wires

= **Wires** {(Wire)}
Definition of wires.

wires (RulesClearancesToBoard)

= **wires** <*float*>
Sets the clearance between wires and the edge of the board.

wires (ExportObjects)

= **wires** <*bool*>
Geber and DXF file export option: output wires.

wireShape

= **wireShape** [**Polyline** | **Arcs**]
Autorouting parameter: wire shape.

wireToBoard

= **wireToBoard** <*bool*>
DRC option: check clearances between wires and the edge of the board.

wireToKeepout

= wireToKeepout <*bool*>
DRC option: check clearances between wires and keepouts.

wireToWire

= **wireToWire** <*bool*>
DRC option: check clearances between wires.

wireToPad

= **wireToPad** <*bool*>
DRC option: check clearances between wires and pads.

wireToVia

= **wireToVia** <*bool*>
DRC option: check clearances between wires and vias.

Z0

= **Z0** <*float*>
Parameter of a rule for differential pair layout: impedance value in ohms.

ZippedWire

= **ZippedWire** (**id** <*string*>) (**fixed** <*bool*>) (**LayerRef**) (**DiffSignalRef**) (**Start**) (Track) {(Track)}
Definition of a zipped wire pair.

! Tracks specify the pair's track line. The shape of wires is calculated automatically.

Example:

```
<ZippedWire id="zwire_1">
   <LayerRef name="Top"/>
   <DiffSignalRef name="MEM_CLK"/>
   <Start x="170.217" y="104.391"/>
   <TrackLine>
      <End x="170.217" y="102.772"/>
   </TrackLine>
   <TrackLine>
      <End x="168.975" y="101.027"/>
   </TrackLine>
</ZippedWire>
```

ZippedWires

= **ZippedWires** {(**ZippedWire**)}
Definition of zipped wire pairs.

zipwireRef

= **zipwireRef** <*string*>
Reference to a zipped wire pair. The string must contain the ID of the existing ZippedWire pair being referenced.

Index