# DipTrace PCB Layout — XML Format Specification

Generated from the DipTrace serializer, repository revision 7276 — reflects current program behaviour.

## Contents

## How to read this specification

This document describes the **DipTrace PCB Layout** flavour of the DipTrace XML format — the same format used both for `File ▸ Save/Open As ▸ DipTrace XML` and for the plug-in exchange file. It is generated directly from the DipTrace serializer and reflects what the program actually reads and writes.

Each element is given with a short description, a representative XML fragment, and a table of its attributes. The **Written** column states the condition under which the serializer emits the attribute (`always`, a guard such as `if >-1`, or a context such as `Text shapes only`); an attribute left at its default is usually omitted, so a missing attribute means *use the default*, not zero.

## Document structure

A PCB project is a single `<Source>` root holding the embedded design-cache library and the editable board:

```
<Source Type="DipTrace-PCB" Version="4.3.0.x" Units="mm">
  <Library Type="DipTrace-ComponentLibrary" ...>        <!-- design cache; nests
the pattern library -->
    <Library Type="DipTrace-PatternLibrary" ...>...</Library>
    <Categories>...</Categories> <Components>...</Components>
  </Library>
  <Board> ... </Board>                                   <!-- the editable
project -->
</Source>
```

The `<Library>` is the project's *design cache* — a full component library that **nests** the footprint (pattern) library inside it (its internals are given in the Pattern Editor specification). In a PCB file this design-cache library carries only `Type` and `Units`. The editable board data lives under `<Board>` (a plug-in's data node is `/Source/Board`).

## Value types & formatting

| Type | Encoding |
|---|---|
| `Int` | A plain decimal integer. |
| `Real` | A decimal number expressed in the file's `Units`, written with a **dot** decimal separator. The reader is tolerant (it normalises both dot and comma), but the dot is canonical and is what other tools expect. |
| `Bool` | `Y` / `N`. (A few Schematic simulator fields use `+` / `-`.) |
| `Text` | A string attribute or element text. |
| `Color` | A 24-bit integer in Windows `0x00BBGGRR` order (e.g. `255` = red). |
| enum | A fixed set of text (or, in a few cases, integer) values, listed with the element. |

## Coordinate frames, units & angles

- The document root carries `Units="mm" | "inch" | "mil"`. **Every** `Real` coordinate, length and size in the file is in those units; there is no per-object override.
- Objects inside a library symbol or footprint (pins, pads, shapes) are stored in an **object-centre** frame — coordinates are relative to the part/footprint origin and are not affected by where an instance is placed.
- Angles are in **radians, counter-clockwise**, unless an attribute is explicitly a discrete orientation enum (e.g. a footprint `Orientation` of `0/90/180/270`).
- Coordinate scaling and sign are internal to the file; read and write values as they are — do not rescale or flip them.

## Identifiers, references & list ordering

- Most list elements carry an `Id` (or `Index`/`Number`) that other objects reference. A reference value of `-1` means *none / not connected*.
- A few lists are referenced **positionally**, not by an `Id` attribute: a `ViaStyle` value indexes the in-order `<ViaStyles>` list and a `NetClass` value indexes the in-order `<NetClasses>` list. Reordering those lists silently reassigns the references.
- When a whole file is opened, top-level object lists are resolved **by position**: each array must be dense, ascending from `Id` 0, with position equal to `Id`. A file authored out of order or with gaps will bind references to the wrong objects.

## Object state flags

| Attribute | Type | Meaning |
|---|---|---|
| `Selected` | Bool | User selection state. |
| `Locked` | Bool | Edit lock. |
| `Group` | Int | Group `Id`, or `-1` for none; groups are listed in a `<Groups>` section. |
| `Enabled` | Bool | *Exists / removed.* A normal save prunes disabled objects, so a saved file rarely shows `Enabled="N"`; on the plug-in exchange / edit-import path, however, `Enabled="N"` is honoured as a **delete** flag on the editable objects. An absent flag means enabled. |

## Plug-in exchange & import merge

The same format is the **plug-in exchange** file: DipTrace exports the selected data, launches the plug-in with the file path as its argument, waits for it to exit, then re-imports the (possibly edited) file. How the result merges back depends on the import mode:

- **Whole-list (`ImpMode=All`)** — the incoming list *replaces* the project's list of that object kind; anything omitted is removed.
- **`Edit`** — each **top-level** object is matched to an existing one **by its `Id`** and overwritten in place; an object with a new or absent `Id` is added. Keep an object's `Id` to edit it, and **omit the `Id`** on

objects you add — never invent one, because over a partial export a computed "free" Id can collide with an unexported object and overwrite it.

**Nested lists without their own `Id` are REPLACED, not merged.** A net's `<Traces>`, a net's or bus's `<Wires>`, and point lists have no per-member `Id`. When such a container is present in your XML, DipTrace **discards the existing list and rebuilds it from exactly the children you supply** — so appending one trace under an existing net replaces that net's *entire* routing. To add one member, re-list every current member plus the new one; to leave the list untouched, omit the container element entirely.

**The `Selected` filter skips — it does not add.** When a plug-in runs with a `Selected` filter under `Edit`, only objects flagged `Selected="Y"` are processed at all: a marked object with an existing `Id` is edited, a marked object with a new `Id` is added, and an **unflagged** object is **skipped entirely** (neither edited nor added). Put `Selected="Y"` on every object you add or edit under such a filter, or its data is silently dropped.

To remove an object, set `Enabled="N"` on it and leave it in the file (see *Object state flags*) — do not delete the XML node. Component and Pattern *library* files use coarser Library / Component / Part import modes rather than per-object `Edit`, but the `Enabled` delete flag and the `Id` rules still apply.

**A note on completeness.** Where a newer serializer field has no counterpart in an older published table (for example the real-valued/mono font fields, or per-object shield-group ids), it is documented here as a first-class attribute. Preserve attributes and elements you do not recognise when editing a file in place — they belong to other subsystems and to future format versions.

## Cross-cutting attribute families

Two families of attributes recur across many PCB elements. They are defined once here; each element's own table then refers back to this section rather than repeating the semantics.

### ShieldGroup family

A via-stitching / shielding-group membership id. All values are `Int` indices into the board's shield-group registry; `-1` or an absent attribute means the object belongs to no shield group. Most carriers emit the attribute only when it is greater than `-1`; trace-point carriers emit it unconditionally.

| Attribute | Type | Written | Description |
|---|---|---|---|
| `ShieldGroup` | Int | if >−1 | Shield-group id on `<BoardOutline>`, `<Component>`, `<Pad>`, `<CopperPour>`, and diff-pair `<PosPoint>`/`<NegPoint>`. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `ShieldGroup` | Int | always | On a routed trace `<Point>` and on diff-pair `<PosTrace>`/`<NegTrace>` points it is emitted on every point. |
| `TraceShieldGroup` | Int | if >−1 | Second shield group carried by `<CopperPour>` (the pour's trace-side group). |
| `PosShieldGroup` | Int | if >−1 | Positive-side shield group on a diff-pair center point. |
| `NegShieldGroup` | Int | if >−1 | Negative-side shield group on a diff-pair center point. |

### Real-valued and mono font family

Every text object stores its font size twice: a legacy rounded integer `FontSize` and a companion real value. On import the real value takes precedence and overrides the rounded integer, so a reader that only understands `FontSize` still gets a usable value while a full reader keeps sub-unit precision.

| Attribute | Type | Written | Description |
|---|---|---|---|
| `FontSizeFloat` | Real | always (text) | Unrounded font size; overrides the rounded `FontSize` on import. On title-block `<Field>`, `<Settings><Markings>`, text `<Shape>`, `<Table>`, `<Cell>`, `<Dimension>`. |
| `MarkingFontSizeFloat` | Real | always | The `<Component>` marking twin of `FontSizeFloat`. |
| `FontMono` | Bool (Y/N) | vector fonts only | Monospace flag; emitted only when the object uses a vector (stroke) font. Same carriers as `FontSizeFloat`. |
| `TextWidth` | Real | text, export-only | Computed bounding-box width of the rendered text. Derived output — written for convenience, not read back. |
| `TextHeight` | Real | text, export-only | Computed bounding-box height. Derived output, not read back. On `<Field>`, text `<Shape>`, `<Cell>`, `<Dimension>`. |
| `FontColor` | Int | always | Title-block `<Field>` font color (unconditional there). |

**Note.** `Enabled="N"` is an import-honored delete flag rather than a routinely written attribute. A normal save prunes disabled objects, so it rarely appears on disk, but the importer honors it on the editable objects (shapes, nets, diff pairs, groups, …) to mark deletions in the plug-in / exchange path.

## <Source> and <Board> — document root

The file root is `<Source>`. It carries the format identity and the unit system, then holds the embedded design-cache `<Library>` followed by the single `<Board>` that contains the whole project. `<Board>` itself has no attributes; it is a container.

```
<?xml version="1.0" encoding="UTF-8"?>
<Source Type="DipTrace-PCB" Version="4.3.0.5" Units="mm">
  <Library Type="DipTrace-ComponentLibrary" Units="mm"> … </Library>
  <Board> … </Board>
</Source>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Type` | Text | always | Always `DipTrace-PCB` for this dialect. |
| `Version` | Text | always | Product version that wrote the file. |
| `Units` | enum | always | File unit system: `mm`, `mil`, or `inch`. All `Real` coordinates and sizes below are expressed in these units. |

## <Library> — embedded design cache

The project's design cache: a full CompEdit-format component library that nests its pattern library inside it (it is not a second sibling library). In a PCB file it is written in cache mode, so at the outer level it carries only `Type` and `Units` — the component-library `Version`/`Name`/`Hint`/`UID32` fields are omitted.

```
<Library Type="DipTrace-ComponentLibrary" Units="mm">
  <Library Type="DipTrace-PatternLibrary" Units="mm"> … footprints … </Library>
  <Categories> … </Categories>
  <Components> … </Components>
</Library>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Type` | Text | always | Outer library is `DipTrace-ComponentLibrary`; the nested block is `DipTrace-PatternLibrary`. |
| `Units` | enum | always | Library unit system. |

**Note.** The nested `<Library Type="DipTrace-PatternLibrary">` holds the footprints that placed components reference by `PatternStyle`; its internal structure is the PattEdit dialect

and is documented there. In a PCB file the design-cache library shows only `Type` and `Units` — do not expect the standalone component-library header fields.

## <BoardOutline> — board edge polygon

The board edge. A `<Points>` list of `<Point>` vertices, each of which may be the start of an arc, plus panelization and lock flags. Y is stored flipped (screen-down positive).

```
<BoardOutline ShieldGroup="-1">
  <Points>
    <Point X="0" Y="0" Arc="N"/>
    <Point X="50.8" Y="0" Arc="N"/>
    <Point X="50.8" Y="38.1" Arc="N"/>
  </Points>
</BoardOutline>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `ShieldGroup` | Int | if >−1 | See cross-cutting family. |
| `PanelExclude` | Bool (Y/N) | if Y | Exclude this outline from panelization. |
| `Locked` | Bool (Y/N) | if Y | Outline is locked from editing. |
| `Selected` | Bool (Y/N) | if Y | Outline is selected. |

### <Point> (BoardOutline)

| Attribute | Type | Written | Description |
|---|---|---|---|
| `X` | Real | always | Vertex X. |
| `Y` | Real | always | Vertex Y (stored negated). |
| `Arc` | Bool (Y/N) | always | Whether this vertex begins an arc segment. |

## <Panel> — panelization

V-scoring or tab-routing panel parameters. All attributes below are written unconditionally.

```
<Panel Type="V-Scoring" Columns="2" Rows="2" ColumnSpacing="0" RowSpacing="0"
       PanelizeSingle="N" RailShow="Y" RailLeft="5" RailRight="5" RailTop="5"
RailBottom="5"
       LeftGap="0" RightGap="0" TopGap="0" BottomGap="0" TabWidth="5"
TabRadius="1" TabStep="10"
       HoleDiam="0.5" HoleStep="1" HoleInset="0" HoleKeepout="0" TabsDone="N"
       CombinedRadius="0" KeepMaterial="N" BorderTabs="0">
  <HorzTabsX>25.4 76.2</HorzTabsX>
```

```
  <VertTabsY>19.05</VertTabsY>
</Panel>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Type` | enum | always | Panel mode (see enum). |
| `Columns` | Int | always | Board columns in the panel. |
| `Rows` | Int | always | Board rows in the panel. |
| `ColumnSpacing` | Real | always | Gap between columns. |
| `RowSpacing` | Real | always | Gap between rows. |
| `PanelizeSingle` | Bool (Y/N) | always | Panelize a single board. |
| `RailShow` | Bool (Y/N) | always | Show border rails. |
| `RailLeft`/`RailRight`/ `RailTop`/`RailBottom` | Real | always | Rail widths on each side. |
| `LeftGap`/`RightGap`/ `TopGap`/`BottomGap` | Real | always | Board-to-rail gaps on each side. |
| `TabWidth` | Real | always | Routing-tab width. |
| `TabRadius` | Real | always | Tab corner radius. |
| `TabStep` | Real | always | Spacing between tabs. |
| `HoleDiam` | Real | always | Mouse-bite hole diameter. |
| `HoleStep` | Real | always | Mouse-bite hole spacing. |
| `HoleInset` | Real | always | Mouse-bite hole inset. |
| `HoleKeepout` | Real | always | Mouse-bite keepout. |
| `TabsDone` | Bool (Y/N) | always | Tabs have been generated. |
| `CombinedRadius` | Real | always | Combined-tab radius. |
| `KeepMaterial` | Bool (Y/N) | always | Keep material at tabs. |
| `BorderTabs` | Real | always | Border-tab parameter. |

| Value | Name |
|---|---|
| `0` | V-Scoring |
| `1` | Tab Routing |

**Note.** `<HorzTabsX>` and `<VertTabsY>` are lists of **bare numeric values**, not `<Item>`-wrapped children — the serializer writes the values inline with no per-value tag. Contrast the title-block `<ColumnWidths>`/`<RowHeights>` lists, which really do use `<Item>`.

## <SheetSettings> — drawing sheet, zones, title blocks

Sheet geometry, border-zone lettering, and up to six title blocks. Coordinate children use lower-case-p tags `<Xpos>`/`<Ypos>` (the reader is case-insensitive). The title-visibility children and the border-zone / title-block sub-tree are written only in a full document export.

```
<SheetSettings>
  <DisplayTitles>Y</DisplayTitles>
  <DisplaySheet>Y</DisplaySheet>
  <Xpos>0</Xpos> <Ypos>0</Ypos> <Scale>100</Scale>
  <SheetWidth>297</SheetWidth> <SheetHeight>210</SheetHeight>
  <LeftMargin>10</LeftMargin> <TopMargin>10</TopMargin>
  <RightMargin>10</RightMargin> <BottomMargin>10</BottomMargin>
  <BorderZones> … </BorderZones>
  <BottomRightBlock Width="120" Height="40"> … </BottomRightBlock>
</SheetSettings>
```

| Child | Type | Written | Description |
|---|---|---|---|
| `DisplayTitles` | Bool (Y/N) | full export | Show title blocks. |
| `DisplaySheet` | Bool (Y/N) | full export | Show sheet border. |
| `Xpos`/`Ypos` | Real | always | Sheet origin on screen. |
| `Scale` | Real | always | Sheet scale, percent. |
| `SheetWidth`/ `SheetHeight` | Real | always | Sheet size. |
| `LeftMargin`/ `TopMargin`/ `RightMargin`/ `BottomMargin` | Real | always | Sheet margins. |

### <BorderZones>

Border-zone lettering (the A/B/C… 1/2/3… grid around the sheet).

| Child | Type | Written | Description |
|---|---|---|---|
| `Visible` | Bool (Y/N) | always | Show zone lettering. |
| `HorzZones` | Int | always | Number of horizontal zones. |

| Child | Type | Written | Description |
|---|---|---|---|
| `VertZones` | Int | always | Number of vertical zones. |
| `Standard` | Int | always | Numbering standard. |
| `FontName` | Text | always | Zone-label font. |
| `FontSize` | Real | always | Zone-label font size (fractional values allowed). |
| `FontLineWidth` | Real | always | Stroke width; a negative value is a relative-to-size ratio, a positive value is an absolute width. |
| `Border` | Bool (Y/N) | always | Draw the zone border frame. |
| `HorzBorderSize`/ `VertBorderSize` | Real | always | Border frame sizes. |

> **Note.** `FontSize` and `FontLineWidth` here are `Real`, not integers — they are written unrounded and can carry fractional values.

### Title blocks and <Field>

Six title-block elements share one structure: `<BottomRightBlock>`, `<BottomLeftBlock>`, `<TopRightBlock>`, `<TopLeftBlock>`, `<ExtTopLeftBlock>`, `<ExtBottomLeftBlock>`. A block is written only when its name is non-empty. It carries `Width`/`Height` attributes and a `<Name>`, a `<ColumnWidths>`/`<RowHeights>` pair (each an `<Item>` list), a `<Cells>` grid of border-flags, and a `<Fields>` list of text fields.

```
<BottomRightBlock Width="120" Height="40">
  <Name>Default</Name>
  <ColumnWidths><Item>30</Item><Item>90</Item></ColumnWidths>
  <RowHeights><Item>10</Item></RowHeights>
  <Cells><Item><Item FieldId="0" TopLine="Y" BottomLine="Y" LeftLine="Y"
RightLine="Y"/></Item></Cells>
  <Fields>
    <Field Id="0" FontVector="Y" FontMono="N" FontSize="10" FontSizeFloat="10"
FontWidth="-2"
           FontScale="1" LineSpacing="1.2" TextWidth="14" TextHeight="2.5"
TextShow="Text"
           TextAlign="Left" FontColor="0" X1="0" Y1="0" X2="30" Y2="10">
      <TextLines><TextLine>Title</TextLine></TextLines>
      <FontName>Tahoma</FontName>
    </Field>
  </Fields>
</BottomRightBlock>
```

| Attribute (Field) | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Field index within the block. |
| `FontVector` | Bool (Y/N) | always | Vector (stroke) vs TrueType font. |
| `FontMono` | Bool (Y/N) | vector only | See font family. |
| `FontSize` | Int | always | Rounded font size (legacy). |
| `FontSizeFloat` | Real | always | Overrides `FontSize` on import. |
| `FontWidth` | Real | always | Stroke width; negative = ratio to size, positive = absolute. |
| `FontScale` | Real | always | Font aspect scale. |
| `LineSpacing` | Real | always | Multi-line spacing. |
| `TextWidth`/ `TextHeight` | Real | text fields, export-only | Computed extents; see font family. |
| `TextShow` | enum | always | Field content mode (see enum). |
| `TextAlign` | enum | always | Text alignment (see enum). |
| `FontColor` | Int | always | Field color. |
| `X1`/`Y1`/`X2`/`Y2` | Real | always | Field rectangle within the block. |

| Value | Name (TextShow) |
|---|---|
| `0` | Text |
| `1` | Sheet |
| `2` | File |

| Value | Name (TextAlign) |
|---|---|
| `0` | Left |
| `1` | Center |
| `2` | Right |

The `<Cells>` grid nests two levels of `<Item>` (rows then cells); each leaf cell carries `FieldId` (Int) and `TopLine`/`BottomLine`/`LeftLine`/`RightLine` (Y/N) border flags. `<TextLines>`/`<TextLine>` and `<FontName>` are written only for text fields (`TextShow="Text"`).

## <Settings> — global project settings

The largest settings block. It opens with a `<Markings>` font+field sub-tree, then grid, origin, snap, line-width, routing, layer-panel, and a run of scalar flag children.

```
<Settings>
  <Markings CompRotate="N" FontVector="Y" FontMono="N" FontSize="10"
FontSizeFloat="10"
            FontWidth="-2" FontScale="1"> … <FontName>Tahoma</FontName> … field
nodes … </Markings>
  <Grid> … </Grid> <Origin> … </Origin> <Assembly> … </Assembly>
  <LayerDisplayMode>Current Only</LayerDisplayMode>
  <LineWidth> … </LineWidth> <Routing> … </Routing>
  <LayerPanel …/> <RelatedSchem> … </RelatedSchem>
  <SchemNetClasses>N</SchemNetClasses>
  <JumperLayer>Silk</JumperLayer> <ProjectDir>C:\proj</ProjectDir>
</Settings>
```

**<Markings>**

Default component-marking font plus per-field silk/assembly show & align.

| Item | Type | Written | Description |
|---|---|---|---|
| `CompRotate` | Bool (Y/N) | always | Rotate markings with the component. |
| `FontVector` | Bool (Y/N) | always | Vector vs TrueType default font. |
| `FontMono` | Bool (Y/N) | vector only | See font family. |
| `FontName` | Text | always | Default marking font. |
| `FontSize` | Int | always | Rounded default size. |
| `FontSizeFloat` | Real | always | Overrides `FontSize` on import. |
| `FontWidth` | Real | always | Stroke width. |
| `FontScale` | Real | always | Aspect scale. |

Inside `<Markings>` follow the per-field display nodes `<RefDesGlobal>`, `<NameGlobal>`, `<ValueGlobal>`, `<PatternGlobal>`, `<ManufacturerGlobal>`, `<DatasheetGlobal>`, and `<AddFieldsGlobal>`/`<AddField>` (user fields, each with a `<Name>`). Every field node carries the same four attributes:

| Attribute | Type | Written | Description |
|---|---|---|---|
| `SilkShow` | enum | always | Silkscreen show mode. |
| `SilkAlign` | enum | always | Silkscreen alignment. |
| `AssyShow` | enum | always | Assembly show mode. |
| `AssyAlign` | enum | always | Assembly alignment. |

| Value | Name (Show) |
|---|---|
| 0 | Common |
| 1 | Show |
| 2 | Hide |

| Value | Name (Align) |
|---|---|
| 0 | Common |
| 1 | Center |
| 2 | Top |
| 3 | Bottom |
| 4 | Left |
| 5 | Right |
| 6 | Corner |
| 7 | Auto |
| 8 | Position |

**<Grid>, <Origin>, <Assembly> (auto-snap)**

| Item | Type | Written | Description |
|---|---|---|---|
| Grid · Visible | Bool (Y/N) | always | Grid on. |
| Grid · Snap | Bool (Y/N) | always | Snap to grid. |
| Grid · Size | Real | always | X grid step. |
| Grid · YSize | Real | always | Y grid step. |
| Grid · YIdentical | Bool (Y/N) | always | Y step equals X step. |
| Origin · Visible | Bool (Y/N) | always | Show axes. |
| Origin · AxisColor | Int | always | Axis color. |
| Origin · X/Y | Real | always | Origin position. |
| Assembly · Pads/Silk/CompBorders/BoardOutline | Bool (Y/N) | always | Auto-snap targets. |

### <LineWidth> and <Routing>

| Item | Type | Written | Description |
|---|---|---|---|
| LineWidth · Silk/Table/Titles/BoardOutline/Assembly/CompOutline/Courtyard | Real | always | Default line widths per class. |
| Routing · Router | enum | always | Router grid mode (see enum). |
| Routing · TraceWidth/TraceClearance/BoardClearance/ViaSize/ViaHole/FanoutLength | Real | always | Routing defaults. |

| Value | Name (Router) |
|---|---|
| 0 | Shape |
| 1 | Grid |

### <LayerPanel> and scalar settings

<LayerPanel> holds per-layer visibility as Y/N attributes: TopAssy, TopMask, TopPaste, BotAssy, BotMask, BotPaste, TopTerminals, BotTerminals, TopCourtyard, BotCourtyard, TopOutline, BotOutline, TopDimensions, BotDimensions, plus a <NonSignal> child list of Y/N <Item>s (one per non-signal layer).

The remaining <Settings> children are scalars:

| Child | Type | Written | Description |
|---|---|---|---|
| LayerDisplayMode | enum | always | Current Only / All Layers / Contrast. |
| EditInactiveLayer | Bool (Y/N) | always | Allow editing on inactive layers. |
| StackLength | Bool (Y/N) | always | Include stack in length calc. |
| SignalDelayLength | Bool (Y/N) | always | Use signal-delay length. |
| SolderMaskSwell | Real | always | Global mask swell. |
| PasteMaskShrink | Real | always | Global paste shrink. |

| Child | Type | Written | Description |
|---|---|---|---|
| ShowCompFiducials | Bool (Y/N) | always | Show component fiducials. |
| RelatedSchem | Path/ Var | always | Linked schematic (<Path>/<Var>). |
| SchemNetClasses | Bool (Y/N) | always | Net classes originated from the related schematic. |
| TopComponentLock/BottomComponentLock | Bool (Y/N) | always | Side component locks. |
| LockNetStructure | Bool (Y/N) | always | Lock net structure. |
| FlipTextAuto | Bool (Y/N) | always | Auto-flip bottom text. |
| JumperLayer | enum | always | Silk / Assembly / Signal / Do Not Show. |
| ProjectDir | Text | always | Project directory. |

| Value | Name (LayerDisplayMode) |
|---|---|
| 0 | Current Only |
| 1 | All Layers |
| 2 | Contrast |

| Value | Name (JumperLayer) |
|---|---|
| 0 | Silk |
| 1 | Assembly |
| 2 | Signal |
| 3 | Do Not Show |

### <ProjectLibs> — library folders and files

The project's library search paths. Two child lists — <Folders> of <Folder> and <Libs> of <Lib> — each entry being a <Path>/<Var> pair (a literal path plus an environment-variable form).

```
<ProjectLibs Sorted="Y">
  <Folders><Folder><Path>C:\lib</Path><Var>$(LIB)</Var></Folder></Folders>
  <Libs><Lib><Path>C:\lib\res.eli</Path><Var></Var></Lib></Libs>
</ProjectLibs>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Sorted` | Bool (Y/N) | always | Whether the library list is sorted. |

### <CopperLayers> / <Lay> — signal and plane layers

The copper stack. Each `<Lay>`'s `Id` is the target of every copper `Lay`/`LayId` reference elsewhere in the file.

```
<CopperLayers>
  <Lay Id="0" Type="Signal" PlanePad="By Pads" PlaneRing="0.3"
Color="255"><Name>Top</Name></Lay>
  <Lay Id="1" Type="Plane" PlanePad="By Pads" NetId="0" PlaneRing="0.3"
Color="128"><Name>GND</Name></Lay>
</CopperLayers>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Layer id; reference target for `Lay`/`LayId`. |
| `Type` | enum | always | Signal or Plane. |
| `PlanePad` | enum | always | Plane thermal mode (By Pads / Fixed Ring). |
| `NetId` | Int | if >−1 | For a plane layer, the net it belongs to. |
| `PlaneRing` | Real | always | Plane thermal-relief ring. |
| `Color` | Int | always | Layer display color. |
| `Locked` | Bool (Y/N) | if Y | Layer locked. |

| Value | Name (Type) |
|---|---|
| `0` | Signal |
| `1` | Plane |

| Value | Name (PlanePad) |
|---|---|
| `0` | By Pads |
| `1` | Fixed Ring |

### <NonSignals> / <NonSignal> — custom non-signal layers

```
<NonSignals>
  <NonSignal Id="0" Side="Top" Color="65280"><Name>Notes</Name></NonSignal>
</NonSignals>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| Id | Int | always | Non-signal layer id; target of a Non-Signal LayId. |
| Side | enum | always | None / Top / Bottom. |
| Color | Int | always | Display color. |

| Value | Name (Side) |
|---|---|
| 0 | None |
| 1 | Top |
| 2 | Bottom |

### <LayerStackName> / <LayerStackItems> — stack-up and materials

<LayerStackName> is a bare text element naming the stack. <LayerStackItems> holds one <LayerStackItem> per physical layer, each pointing at a copper layer by Lay and carrying a <Material>.

```
<LayerStackName>Default</LayerStackName>
<LayerStackItems>
  <LayerStackItem Lay="0">
    <Material Type="Conductor" VariableThickness="N" Thickness="0.035"
Constant="0" TraceWidth="0"><Name>Copper</Name></Material>
  </LayerStackItem>
</LayerStackItems>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| LayerStackItem · Lay | Int | always | Copper-layer index this item covers. |
| Material · Type | enum | always | Conductor / Plane / Dielectric. |
| Material · VariableThickness | Bool (Y/N) | always | Thickness varies. |
| Material · Thickness | Real | always | Layer thickness. |
| Material · Constant | Real | always | Dielectric constant. |
| Material · TraceWidth | Real | always | Reference trace width for impedance. |

| Value | Name (Material Type) |
|---|---|
| 0 | Conductor |
| 1 | Plane |
| 2 | Dielectric |

## <HierarchySheets> — hierarchy blocks

Registered hierarchy sheets. Each `<HSheet>` has a number and name and an optional `<UpdateIds>` list linking instances (`<UId>`, carrying a `UId` and a `<Name>`).

```
<HierarchySheets>
  <HSheet Number="0"><Name>PowerBlock</Name>
    <UpdateIds><UId UId="12"><Name>U1</Name></UId></UpdateIds>
  </HSheet>
</HierarchySheets>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `HSheet` · `Number` | Int | always | Sheet number. |
| `UId` · `UId` | Int | always | Instance update id. |

## <ViaStyles> / <ViaStyle> — via styles

The in-order list of via styles. A via-style reference elsewhere (a trace point's `ViaStyle="N"`) is a **positional index** into this list; `-1` means no style. Do not reorder the list — that silently reassigns every reference.

```
<ViaStyles>
  <ViaStyle Id="0" Size="0.6" HoleSize="0.3"><Name>Default</Name></ViaStyle>
  <ViaStyle Id="1" Size="0.4" HoleSize="0.2" Lay1="0" Lay2="1"><Name>Buried</
Name></ViaStyle>
</ViaStyles>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Equal to the style's position in the list. Written for readability; the reader ignores it and resolves references positionally. |
| `Size` | Real | always | Via pad diameter. |
| `HoleSize` | Real | always | Via drill diameter. |
| `Lay1`/`Lay2` | Int | if not through | Start/end copper layers for a blind/buried via. Emitted only when the style does not span the full stack; when absent the via is a through via. |

> **Note.** There is no `Through` attribute — through vs blind/buried is derived from `Lay1`/`Lay2`: a style with no `Lay1`/`Lay2` (or one spanning the full stack) is a through via, otherwise it is blind/buried.

### <NetClasses> / <NetClass> — net-class rules

The in-order list of net classes. A net's `NetClass="N"` is a **positional index** into this list — reordering silently reassigns rules. Each class carries per-layer width/clearance properties, an allowed-via list, and autorouting constraints.

```
<NetClass Id="0" UpdateId="-1" Type="Normal" AllLayers="Y" CheckLength="N"
AllVias="Y"
          PerformDRC="Y" Phase="0" Phase_ErrorLength="0" LengthDelta="0"
          MaxUncoupledLength="0" Tolerance="0">
  <Name>Default</Name>
  <LayProperties><LayProperty MinWidth="0.15" MaxWidth="5"> … </LayProperty></
LayProperties>
  <AllowedVias><ViaStyle>0</ViaStyle></AllowedVias>
  <Autorouting RoutePriority="N" PriorityValue="0" RouteMaxVias="N"
MaxViasValue="0"
               RouteMaxIncorrectWay="N" MaxIncorrectWayValue="0"
RouteAllLayers="Y"/>
</NetClass>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Equal to list position; written for readability, ignored on read (references are positional). |
| `UpdateId` | Int | always | Cross-reference id (schematic net-class link). |
| `Type` | enum | always | Normal or Differential Pair. |
| `AllLayers` | Bool (Y/N) | always | One property row for all layers vs per-layer rows. |
| `CheckLength` | Bool (Y/N) | always | Enforce length limits. |
| `AllVias` | Bool (Y/N) | always | Allow all via styles. |
| `PerformDRC` | Bool (Y/N) | always | Include class in DRC. |
| `Phase`/ `Phase_ErrorLength`/ `LengthDelta` | Real | always | Diff-pair phase and length tolerances. |
| `FixedLength` | Real | if >−1 | Target fixed length; omitted (= longest-connection mode) when `-1`. |
| `MaxUncoupledLength`/ `Tolerance` | Real | always | Diff-pair uncoupled-length limit and matching tolerance. |

Each <LayProperty> carries Width (if >−1), MinWidth, MaxWidth, Clearance (if >−1), the diff-pair-only neck fields (Neck_Width, Neck_DifClearance, Neck_MaxLength, DifClearance, written only when Type=Differential Pair), an optional <LayerName> (only in per-layer mode), and a <ClearanceDetails> object-to-object clearance sub-matrix. The <Autorouting> child holds RoutePriority/PriorityValue, RouteMaxVias/MaxViasValue, RouteMaxIncorrectWay/MaxIncorrectWayValue, RouteAllLayers, and a <RouteLayers> Y/N <Item> list.

| Value | Name (Type) |
|---|---|
| 0 | Normal |
| 1 | Differential Pair |

> **Note.** MaxIncorrectWayValue is a Real (written and read unrounded, with no unit coefficient), not an integer.

## <ClassToClass> — class-to-class clearances

The upper-triangular matrix of clearances between net classes. Each <Cell> names two classes by index. When the pair uses one clearance for all layers, a single Clearance attribute is written; otherwise a <LayerClearances> child lists a per-layer <Lay> for each copper layer. A clearance of -1 means "not set / inherit".

```
<ClassToClass Enabled="Y">
  <CTC_Cells>
    <Cell NetClass1="0" NetClass2="0" Clearance="0.2"/>
    <Cell NetClass1="0" NetClass2="1">
      <LayerClearances><Lay Id="0" Clearance="0.25"/><Lay Id="1" Clearance="0.3"/
></LayerClearances>
    </Cell>
  </CTC_Cells>
</ClassToClass>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| Enabled | Bool (Y/N) | always | Whether class-to-class checking is on. |
| Cell · NetClass1/NetClass2 | Int | always | The two net-class indices. |
| Cell · Clearance | Real | all-layers case | Single clearance for the pair (-1 = unset). Present only when the pair is all-layers. |
| Lay · Id/Clearance | Int / Real | per-layer case | Copper-layer index and its clearance. |

## <DRC> — design-rule set

The DRC configuration: a run of check-enable flags, then per-layer clearance and size records. When `AllLayers="Y"`, the per-layer lists collapse to a single record.

```
<DRC AllLayers="Y" CheckClearance="Y" CheckSize="Y" CheckJumpers="Y"
CheckCopperPours="Y"
     CheckClassToClass="Y" CheckSilk="Y" CheckLength="N" CheckKeepouts="Y"
CheckSameNet="N"
     CheckSameComponentPads="N" CheckCourtyard="Y" SilkClearance="0.2">
  <ShowList>Y</ShowList> <RealTimeMode>1</RealTimeMode> <DRCDone>N</DRCDone>
  <LayClearances><LayClearance Lay="0" SameTraceToTrace="0" SameSmdToVia="0"
SameSmdToPad="0" SameSmdToSmd="0"> … </LayClearance></LayClearances>
  <LaySizes><LaySize Lay="0" MinTrace="0.15" MinDrill="0.3" MinRing="0.1"
MaxRing="0" MaxPlatedHole="0" MaxNonPlatedHole="0"/></LaySizes>
</DRC>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `AllLayers` | Bool (Y/N) | always | Single ruleset for all layers. |
| `CheckClearance`/ `CheckSize`/ `CheckJumpers`/ `CheckCopperPours`/ `CheckClassToClass`/ `CheckSilk`/ `CheckLength`/ `CheckKeepouts`/ `CheckSameNet`/`CheckSameComponentPads`/ `CheckCourtyard` | Bool (Y/N) | always | Individual DRC check toggles. |
| `SilkClearance` | Real | always | Silk-to-copper clearance. |
| `ShowList` | Bool (Y/N) | always | Show the error-list panel. |
| `RealTimeMode` | Int | always | Real-time DRC mode. |
| `DRCDone` | Bool (Y/N) | always | DRC has been run. |

Each `<LayClearance>` carries `Lay` and the same-net clearances `SameTraceToTrace`, `SameSmdToVia`, `SameSmdToPad`, `SameSmdToSmd` (all Real), plus a `<ClearanceDetails>` sub-matrix of the object-pair clearances (`TraceToTrace`, `TraceToVia`, … `DrillToBoard` — the upper triangle over the object types Trace/Via/Pad/Smd/Copper/Drill/Board). Each `<LaySize>` carries `Lay`, `MinTrace`, `MinDrill`, `MinRing`, `MaxRing`, `MaxPlatedHole`, `MaxNonPlatedHole` (all Real).

**Note.** When `AllLayers="Y"` both `<LayClearances>` and `<LaySizes>` emit exactly one record with `Lay="0"` and stop — there is no per-layer expansion. Read the single record as applying to every layer.

### <ConnectivityCheck> — connectivity/obstacle flags

Three flags governing what the interactive connectivity check treats as an obstacle.

```
<ConnectivityCheck Traces="Y" Shapes="N" CopperPours="Y"/>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Traces` | Bool (Y/N) | always | Consider traces. |
| `Shapes` | Bool (Y/N) | always | Consider shapes. |
| `CopperPours` | Bool (Y/N) | always | Consider copper pours. |

### <MainLengthRule> / <LengthRules> — length matching

`<MainLengthRule>` is the single primary rule; `<LengthRules>` is the list of additional `<LengthRule>`s. The two share the same shape: a target/type header plus a `<Rules>` list of measured connections.

```
<MainLengthRule Type="By User" Delta="0.1" AnyError="N" NetClass="-1">
  <Name>Main</Name>
  <Rules>
    <Rule NetId="0" Comp1="1" Pad1="1" Comp2="2" Pad2="1" Length="25.4"
Error="N"/>
  </Rules>
</MainLengthRule>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Type` | enum | always | By User or By NetClass. |
| `FixedLength` | Real | if >−1 | Target length; omitted (= longest-connection target) when `-1`. |
| `Delta` | Real | always | Allowed deviation. |
| `AnyError` | Bool (Y/N) | always | Rule has any error. |
| `NetClass` | Int | always | Net-class index (or `-1`). |
| `Rule` · `NetId` | Int | always | Net of the measured connection. |
| `Rule` · `Comp1`/`Pad1`/`Comp2`/`Pad2` | Int | always | The two endpoints (component id + pad id). |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Rule` · `Length` | Real | always | Measured length. |
| `Rule` · `Error` | Bool (Y/N) | always | Endpoint is in error. |

| Value | Name (Type) |
|---|---|
| `0` | By User |
| `1` | By NetClass |

**Note.** `<LengthRule>` (inside `<LengthRules>`) uses the identical attribute set and the same `<Rules>`/`<Rule>` children as `<MainLengthRule>`.

## <Groups> / <Group> — group registry

The registry of object groups. Only enabled groups are written. A `Group="N"` reference on any object points at a `<Group>` by its `Id`; `-1` means ungrouped.

```
<Groups>
  <Group Id="0"/>
  <Group Id="1" Selected="Y"/>
</Groups>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Group id (reference target for `Group` attributes). |
| `Selected` | Bool (Y/N) | if Y | Group is selected. |

## <Components> / <Component> — placed components

The `<Components>` list holds every placed `<Component>` — footprints, plus the special stand-alone objects (free pad, mounting hole, static via, fiducial). Each component carries its placement, marking display, and its own `<Pads>` list.

```
<Component Id="0" UpdateId="-1" PatternStyle="PatType0" X="15.24" Y="22.86"
           Angle="0" Side="Top" MarkingFontSize="10" MarkingFontSizeFloat="10"
           GridAlign="Pad" Group="-1">
  <RefDes>C1</RefDes>
  <Name>10SVP10M</Name>
  <Value>47</Value>
  <Pads>
    <Pad Id="1" NetId="0" InternalConnection="-1"/>
    <Pad Id="2" NetId="-1" InternalConnection="-1"/>
  </Pads>
```

```
  <RefDesMarking> <Silk Show="Common" .../> <Assy Show="Common" .../> </
RefDesMarking>
</Component>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Component index; what nets/ratlines/diff pairs reference together with a pad `Id`. |
| `UpdateId` | Int | always | Hidden id linking back to the source schematic part (`-1` = none). |
| `Type` | enum | if `tp>10` | Special object kind (`Pad`/`MtHole`/`Via`/`Fiducial`). Absent for ordinary footprints. Distinct from `PatternStyle`. |
| `PatternStyle` | Text | if not a static via | Footprint-style name referencing a footprint in the embedded pattern library. A free-text style name, not an enum value. |
| `ViaStyle` | Int | Via objects only | For a static-via component (`Type="Via"`): positional index into the `<ViaStyles>` list. Written in place of the placement/marking block. |
| `X` / `Y` | Real | always | Placement origin. |
| `Angle` | Real | if ≠0 | Rotation, radians CCW. |
| `Side` | enum | ordinary components | `Top` or `Bottom`. |
| `Flip` | Bool (Y/N) | if set | Mirror flag; written only when `Y`. |
| `HorzFlip` | Bool (Y/N) | if set | Horizontal-mirror flag; written only when `Y`. |
| `ShowFiducials` | enum | if >0 | `Common`/`Show`/`Hide`. |
| `CustomMarkingFont` | Bool (Y/N) | if set | Component overrides the global marking font. |
| `MarkingFontSize` | Int | ordinary components | Marking font height, rounded. |
| `MarkingFontSizeFloat` | Real | ordinary components | Unrounded twin of `MarkingFontSize`; overrides the rounded value on import. |
| `MarkingFontAll` | Bool (Y/N) | if set | Font applies to all marking fields. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `GridAlign` | enum | ordinary components | `Pad` or `Origin` — the point the grid snaps to. |
| `PlacementClearance` | Real | if >0 | Extra placement clearance around the component. |
| `PanelExclude` | Bool (Y/N) | if set | Excluded from panelization. |
| `Group` | Int | if >−1 | Group registry id. |
| `ShieldGroup` | Int | if >−1 | Shared shield/stitching-group id. |
| `Locked` | Bool (Y/N) | if set | Placement locked. |
| `Selected` | Bool (Y/N) | if set | Selection state. |

Text children: `<RefDes>`, `<Name>`, `<Value>`. Ordinary components also emit optional add-field lists and the marking sub-elements below.

| Value | Name |
|---|---|
| `14` | Pad |
| `15` | MtHole |
| `16` | Via |
| `20` | Fiducial |

> **Note.** `Type` and `PatternStyle` are unrelated. `Type` (written only for the special `tp>10` objects) names the object kind; `PatternStyle` is the footprint-style name an ordinary placed component references. A static via (`Type="Via"`, internal `tp=16`) and a mounting hole (`Type="MtHole"`, `tp=15`) skip the placement/marking block entirely.

### <Component> marking sub-elements

Six per-field marking blocks describe how each component field is drawn on silk and on the assembly layer: `<RefDesMarking>`, `<NameMarking>`, `<ValueMarking>`, `<PatternMarking>`, `<ManufacturerMarking>`, `<DatasheetMarking>`. Each contains a `<Silk>` and an `<Assy>` child.

```
<RefDesMarking>
  <Silk Show="Common" Align="Common" Horz="Center" Vert="Center" X="0" Y="0"
Angle="0"/>
  <Assy Show="Common" Align="Common" Horz="Center" Vert="Center" X="0" Y="0"
Angle="0"/>
</RefDesMarking>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Show` | enum | always | `Common`/`Show`/`Hide` — visibility of this field on the layer. |
| `Align` | enum | always | Placement anchor (`Common`/`Center`/`Top`/`Bottom`/`Left`/`Right`/`Corner`/`Auto`/`Position`). |
| `Horz` | enum | always | Horizontal text alignment (`Center`/`Right`/`Left`). |
| `Vert` | enum | always | Vertical text alignment (`Center`/`Bottom`/`Top`). |
| `X` / `Y` / `Angle` | Real | always | Field offset and rotation (`Angle` radians). |

**<Pad>**

Each `<Pad>` under a component's `<Pads>`. The pad's `NetId` is the authoritative carrier of net membership.

```
<Pad Id="1" NetId="0" InternalConnection="-1" SignalDelay="0.5"/>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Pad number within the component. |
| `NetId` | Int | always | Owning net's `Id` (`-1` = unconnected). Authoritative — DipTrace derives rat-lines and the net's `<Pads>` mirror from this. |
| `SignalDelay` | Real | if >0 | Extra pin/package delay added to length-matching. Omitted when 0. Not written for static-via objects. |
| `InternalConnection` | Int | ordinary pads | Internal-connection group index within the component (`-1` = none). |
| `ShieldGroup` | Int | if >−1 | Shared shield/stitching-group id. |

Optional pad children: `<BlockLayers>`/`<BlindLayers>` (layer lists), `<CustomSpoke>`, `<TeardropParams>`, and — for static-via objects — `<MaskPaste>`.

**<MaskPaste>**

Per-pad mask and paste overrides, emitted under a static-via pad when any state or swell/shrink differs from default. States and swell/shrink are attributes; explicit mask/paste opening geometry lives in `<TopSegments>`/`<BotSegments>`.

```
<MaskPaste TopMask="..." TopPaste="..." CustomSwell="0.1" CustomShrink="0.05">
  <TopSegments>
```

```
    <Item X1="0" Y1="0" X2="1.0" Y2="0.4"/>
  </TopSegments>
  <BotSegments> <Item X1="0" Y1="0" X2="1.0" Y2="0.4"/> </BotSegments>
</MaskPaste>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| TopMask / BotMask | enum | if state >0 | Top/bottom solder-mask opening mode. |
| TopPaste / BotPaste | enum | if state >0 | Top/bottom paste opening mode. |
| Segment_Percent | Real | segmented paste | Paste coverage percent (when a paste state selects the segmented mode). |
| Segment_EdgeGap / Segment_Gap / Segment_Side | Real | segmented paste | Segmented-paste edge gap, inter-segment gap, and side margin. |
| CustomSwell | Real | if set | Mask swell override (absent = inherit). |
| CustomShrink | Real | if set | Paste shrink override (absent = inherit). |

**<TopSegments> / <BotSegments> <Item>**

Explicit mask/paste opening rectangles. Each <Item> is a segment given by two corner points.

| Attribute | Type | Written | Description |
|---|---|---|---|
| X1 / Y1 | Real | always | First corner. |
| X2 / Y2 | Real | always | Second corner. |

> **Note.** These <Item> children are the one place a segment/point container carries *geometry* as <Item>. Elsewhere <Item> denotes a reference (net <Pads>), and geometry point lists use <Point>.

## <Ratlines> / <Ratline> — unrouted connections

The ratline list is auto-generated by DipTrace from pad NetId membership; a plug-in does not hand-build it. Each <Ratline> is a straight guide between two pads.

```
<Ratline Id="0" Hidden="N" X1="10.0" Y1="5.0" X2="12.5" Y2="5.0"
         Comp1="1" Pad1="1" Comp2="0" Pad2="1"/>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| Id | Int | always | Ratline index. |
| Hidden | Bool (Y/N) | always | Hidden from display. |
| X1 / Y1 / X2 / Y2 | Real | always | Endpoint coordinates. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Comp1` / `Pad1` | Int | always | First pad: component `Id` and pad `Id`. |
| `Comp2` / `Pad2` | Int | always | Second pad: component `Id` and pad `Id`. |

### <Nets> / <Net> — nets, pads, teardrops, traces

A `<Net>` gathers its member pads (a derived mirror of the pads' own `NetId`), teardrop parameters and polygons, and the routed `<Traces>`.

```
<Net Id="0" HiddenId="0" NetClass="0" RouteMode="Ratlines">
  <Name>AP-WAKE-BT</Name>
  <Pads> <Item Comp="1" Pad="1"/> <Item Comp="0" Pad="1"/> </Pads>
  <Traces> ... </Traces>
</Net>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Net index; referenced by pad/pour/shape `NetId`. |
| `HiddenId` | Int | always | Stable hidden id. |
| `NetClass` | Int | always | **Positional** index into the in-order `<NetClasses>` list (not an Id lookup). |
| `RouteMode` | enum | if >0 | `Ratlines`/`Correct Traces`/`Full Reroute`/`Dont Route`. Omitted for the default. |
| `HideRatlines` | Bool (Y/N) | if set | Suppress this net's ratlines. |
| `ShowLength` | Bool (Y/N) | if set | Show routed length. |
| `Highlighted` | Bool (Y/N) | if set | Highlight state. |
| `AllowLoops` | Bool (Y/N) | if set | Loops permitted in this net's routing. |
| `CustomColor` | Bool (Y/N) | if custom color | Uses a per-net trace color. |
| `TraceColor` | Int | with CustomColor | The custom color value. |
| `MeanderGap` | Real | if ≥0 | Meander spacing for this net. |
| `Locked` | Bool (Y/N) | if set | Net locked. |

Text child `<Name>`. Reference child `<Pads>` holds `<Item Comp="" Pad=""/>` entries (component `Id` + pad `Id`). Optional `<TeardropParams>` and a `<Teardrops>` polygon list follow.

**<Traces> / <Trace>**

The routed segments of a net. Endpoints are declared with paired attributes — there is no `Pad=` attribute.

```
<Trace Id="0" Connected1="Pad" Object1="1" SubObject1="1" Point1="-1"
              Connected2="Pad" Object2="0" SubObject2="1" Point2="0"
              PairSeparateTrace="-1">
  <Points> ... </Points>
</Trace>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Trace index within the net. |
| `Connected1` | enum | always | End-1 connection kind (see enum). |
| `Object1` | Int | always | End-1 object. For a pad end: the component `Id`. |
| `SubObject1` | Int | always | End-1 sub-object. For a pad end: the pad `Id`. |
| `Point1` | Int | always | End-1 point index (for trace/segment ends). |
| `Connected2` / `Object2` / `SubObject2` / `Point2` | enum/ Int | always | End-2 quad, same meaning as end-1. |
| `Group` | Int | if >−1 | Group registry id. |
| `PairSeparateTrace` | Int | always | Link to the paired separate trace (diff-pair coupling), `-1` = none. |
| `Selected` | Bool (Y/N) | if set | Selection state. |

| Value | Name |
|---|---|
| `0` | Pad |
| `1` | Trace |
| `5` | Segment |
| `6` | Separate Trace |
| `7` | Free |

### <Points> / <Point> (trace)

The trace polyline. Per-segment properties (layer, width, via, jumper, arc, meander) live on the segment's **second** point; the first point's non-geometry attributes are ignored on import. `Id`, `X`, `Y` always matter.

```
<Point Id="0" X="20.955" Y="12.7" Lay="0" Width="0.33" ShieldGroup="-1"
       Arc="Y" ViaStyle="-1" Meander="0" MeanderAngle="0"/>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Point index. |
| `X` / `Y` | Real | always | Point coordinates. |
| `Lay` | Int | always | Numeric copper-layer id of the segment entering this point. |
| `Width` | Real | always | Segment width. |
| `ShieldGroup` | Int | always | Shared shield/stitching-group id (written unconditionally here). |
| `Jumper` | Int | if ≠0 | Jumper flag (0/1/2). |
| `Arc` | Bool (Y/N) | if set | Segment is an arc. |
| `ViaStyle` | Int | if >−1 | Via placed at this point: **positional** index into `<ViaStyles>` (`-1` = none, omitted). |
| `PairPoint` / `PairSubPoint` | Int | always | Diff-pair coupling references. |
| `Meander` | Int | if ≠0 | Meander segment kind. |
| `MeanderAngle` | Real | if ≠0 | Meander angle. |
| `Selected` | Bool (Y/N) | if set | Selection state. |

**Note — connectivity model.** Electrical membership lives on the **pad**: each pad's own `NetId` is authoritative and is preserved across import. The net's `<Pads>` list is a derived mirror that DipTrace rebuilds, and the `<Ratlines>` are auto-generated from pad `NetId`s — so a pad keeps its net even when unrouted, and to move a pad between nets you set its `NetId`, not the `<Pads>` list. A trace connects through its declared endpoint quads, not by where it visually sits; a trace endpoint that lands on a pad belonging to a different net **merges those two nets into one**. Connect deliberately.

### <DifferentialPairs> / <RemovedDifferentialPairs>

Differential-pair definitions with their coupled-segment geometry. `<RemovedDifferentialPairs>` holds pairs the user unpaired but whose geometry is retained; its `<DifferentialPair>` structure mirrors the active one.

```
<DifferentialPair Id="0" NetClass="1" PosNet="3" NegNet="4"
                  RouteMode="Full Reroute" AutoPadPoints="Y">
  <Name>USB_D</Name>
  <Segments><Segment><CenterPoints>
    <CenterPoint X="10" Y="5" PosShieldGroup="-1" NegShieldGroup="-1">
      <PosPoints><PosPoint X="10" Y="4.8" ShieldGroup="-1"/></PosPoints>
      <NegPoints><NegPoint X="10" Y="5.2" ShieldGroup="-1"/></NegPoints>
    </CenterPoint>
  </CenterPoints></Segment></Segments>
  <PosSeparateTraces><PosTrace><Points>
    <Point X="9" Y="4.8" Lay="0" Width="0.2" ShieldGroup="-1"/>
  </Points></PosTrace></PosSeparateTraces>
</DifferentialPair>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Pair index. |
| `NetClass` | Int | always | Positional index into `<NetClasses>`. |
| `PosNet` / `NegNet` | Int | always | Positive / negative member net `Id`s. |
| `RouteMode` | enum | if >0 | Same enum as net `RouteMode`. |
| `AutoPadPoints` | Bool (Y/N) | always | Auto-generate the pad break points. |
| `CustomColor` | Bool (Y/N) | if custom color | Uses a per-pair color. |
| `TraceColor` | Int | with CustomColor | Custom color value. |

Text child `<Name>`. Geometry lives in `<Segments>` (coupled center points), `<PosSeparateTraces>`/`<NegSeparateTraces>` (the uncoupled legs), and a `<DifPoints>` pad-break list.

> **Note — ShieldGroup placement on diff-pair geometry.** A coupled `<CenterPoint>` carries `PosShieldGroup` and `NegShieldGroup` (each written only when `>-1`). Its `<PosPoint>`/`<NegPoint>` children carry a single `ShieldGroup` (when `>-1`). The separate-trace `<Point>` children (under `<PosTrace>`/`<NegTrace>`) carry `ShieldGroup` **unconditionally** and have **no `Id`** — the reader indexes them by position.

### <CopperPours> / <CopperPour>

Copper pours / planes: an outline polygon plus fill rules. The pour belongs to a numeric copper layer and (optionally) a net.

```
<CopperPour Id="0" NetId="0" Lay="0" Priority="0" Poured="Y" Type="Solid"
            Clearance="0.33" LineWidth="0.1" LineSpacing="0.1" Spoke="Direct"
            SpokeWidth="0.33">
  <Points> <Point X="20.79" Y="11.42"/> ... </Points>
</CopperPour>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Pour index. |
| `NetId` | Int | always | Net the pour connects to (`-1` = none). |
| `Lay` | Int | always | Numeric copper-layer id. |
| `Priority` | Int | always | Fill priority. |
| `Poured` | Bool (Y) | only when poured | Present (`Y`) only when the pour is filled; absent means unpoured — never written as `N`. |
| `Type` | enum | always | Fill pattern (see enum). |
| `Clearance` | Real | always | Fill clearance. |
| `UseNetClearance` | Bool (Y/N) | always | Use the net's clearance instead of the local one. |
| `BoardClearance` | Real | always | Clearance to the board edge. |
| `LineWidth` | Real | always | Hatch/fill line width. |
| `LineSpacing` | Real | always | Hatch line spacing. |
| `MinimumArea` | Real | always | Minimum island area kept. |
| `Spoke` | enum | always | Thermal spoke style (see enum). |
| `SpokeWidth` | Real | always | Thermal spoke width. |
| `ViaDirect` | Bool (Y/N) | always | Direct-connect vias. |
| `SMD_Separate` | Bool (Y/N) | always | Separate SMD thermal settings. |
| `SMD_Spoke` | enum | always | SMD spoke style (Spoke enum). |
| `SMD_SpokeWidth` | Real | always | SMD spoke width. |
| `RatlineMode` | enum | always | `Automaticaly`/`All Ratlines`/`Do Not Hide`. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| SnapToBoard | Bool (Y/N) | always | Snap outline to board edge. |
| IslandRegion / IslandInternal / IslandConnection | Bool (Y/N) | always | Island-removal flags. |
| RegionsDone | Bool (Y/N) | always | Fill-region computation state. |
| ShieldGroup | Int | if >−1 | Shared shield/stitching-group id. |
| TraceShieldGroup | Int | if >−1 | Shield group of the pour's stitching traces. |
| Group | Int | if >−1 | Group registry id. |
| PanelExclude | Bool (Y) | if set | Excluded from panelization. |
| Locked | Bool (Y) | if set | Pour locked. |
| Selected | Bool (Y) | if set | Selection state. |

Outline child <Points> holds geometry <Point X="" Y=""/> vertices.

| Value | Name |
|---|---|
| 0 | Solid |
| 1 | Horizontal Lines |
| 2 | Vertical Lines |
| 3 | Cross 45 |
| 4 | Cross 90 |

| Value | Name (Spoke) |
|---|---|
| 0 | Direct |
| 1 | 2 spoke 90 |
| 2 | 2 spoke |
| 3 | 4 spoke 45 |
| 4 | 4 spoke |

### <Shapes> / <Shape> — free graphics & text

Free graphics, text, pictures and QR codes on any documentation or copper layer — the most common plug-in target. The `Type` and `Layer` attributes select the shape kind and target layer; which further attributes appear depends on the kind.

```
<Shape Id="0" Type="Polyline" AllLayers="N" Layer="Top Assy" LayId="0"
       LineWidth="0.2">
  <Points> <Point X="86.995" Y="21.59"/> ... </Points>
</Shape>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Shape index. |
| `Type` | enum | always | Shape kind (see enum). |
| `AllLayers` | Bool (Y/N) | always | Place a copper shape on all copper layers. |
| `Layer` | enum | always | Target layer (single-literal tokens, see enum). |
| `LayId` | Int | always | Copper-layer id (for `Signal/Plane`) or non-signal-layer id (for `Non-Signal`). |
| `LineWidth` | Real | non-text, non-filled, if >−1 | Line width for outline shapes; not written for Text/Picture. |
| `Angle` | Real | Text/Picture/QR | Radians CCW. |
| `HorzAlign` / `VertAlign` | enum | Text/Picture/QR | Anchor alignment. |
| `TextAlign` | enum | Text/QR | Multi-line justification (`Center`/`Right`/`Left`). |
| `Inverted` | Bool (Y/N) | Text/Picture/QR | Inverted (knockout) rendering. |
| `FontVector` | Bool (Y/N) | Text | Vector (stroke) font vs TrueType. |
| `FontMono` | Bool (Y/N) | Text, vector | Monospaced vector font. |
| `FontSize` | Int | Text | Font height, rounded. |
| `FontSizeFloat` | Real | Text | Unrounded font height; overrides `FontSize` on import. |
| `FontWidth` / `FontScale` | Real | Text | Stroke width and horizontal scale. |
| `LineSpacing` | Real | Text | Inter-line spacing. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `TextWidth` / `TextHeight` | Real | Text, computed | Derived text bounding size (export-only). |
| `PictureWidth` / `PictureHeight` | Real | Picture/QR | Picture box size. |
| `PictureProportions` | Bool (Y/N) | Picture/QR | Keep aspect ratio. |
| `PictureRaster` | Bool (Y/N) | Picture/QR | Raster source present (vs vector-only). |
| `PictureTransparent` | Int | Picture/QR | Transparent-color threshold. |
| `PictureFlipped` | Bool (Y/N) | Picture/QR | Mirrored picture. |
| `Group` | Int | if >−1 | Group registry id. |
| `NetId` | Int | if >−1 | Net a copper shape belongs to. |
| `PanelExclude` | Bool (Y) | if set | Excluded from panelization. |
| `Locked` | Bool (Y) | if set | Shape locked. |
| `Selected` | Bool (Y) | if set | Selection state. |

Children: `<Points>` with geometry `<Point X="" Y=""/>` (Line/Rectangle/Obround = 2 pts, Arc = 3, Polyline/Polygon = n); for Text/QR a `<FontName>` and a `<TextLines>`/`<TextLine>` list; for Picture/QR a `<PictureFile>`; and a vector `<PictureVector>` (with `Width`/`Height` and `<Polygons>`/`<Polygon>`/`<Points>`/`<Point>`) for pictures, QR, and inverted text.

| Value | Name |
|---|---|
| `0` | Line |
| `1` | Arc |
| `2` | Rectangle |
| `3` | FillRect |
| `4` | Obround |
| `5` | FillObround |
| `6` | Text |
| `7` | Picture |

| Value | Name |
|---|---|
| 8 | Polyline |
| 9 | Polygon |
| 10 | QR (serialized as Type="10") |

| Value | Name (Layer) |
|---|---|
| 0 | Top Assy |
| 1 | Top Silk |
| 2 | Route Keepout |
| 3 | Signal/Plane |
| 4 | Bottom Silk |
| 5 | Bottom Assy |
| 6 | Top Mask |
| 7 | Top Paste |
| 8 | Bottom Paste |
| 9 | Bottom Mask |
| 10 | Board Cutout |
| 11 | Placement Keepout |
| 12 | None |
| 13 | Top Dimension |
| 14 | Bottom Dimension |
| 15 | Non-Signal |
| 16 | Top Courtyard |
| 17 | Bottom Courtyard |
| 18 | Top Outline |
| 19 | Bottom Outline |
| 20 | Top Terminals |
| 21 | Bottom Terminals |

**Note.** Each `Layer` value is one literal token — there is no combined `Top/Bottom X` form, and an unrecognized string resolves to no layer. The `Type` enum has no `None` value; a QR shape (internal `tp=10`, a combined picture+text object) has no symbolic name and is written

as `Type="10"`. Geometry point lists use `<Point>` throughout (including the vector-picture polygons).

### <DesignErrors> / <DesignError>

DRC results as placed markers. Each error records its location, kind, the offending objects, and the layers involved.

```
<DesignError X="12.0" Y="8.0" Type="Clearance" Value="0.2" Rule="0.25"
             ObjectType1="Trace" ObjectType2="Pad">
  <Lays> <Item>Y</Item> <Item>N</Item> </Lays>
</DesignError>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `X` / `Y` | Real | always | Marker location. |
| `Type` | enum | always | Error kind (see enum). |
| `Value` | Real | always | Measured value at the violation. |
| `Rule` | Real | always | The rule limit that was violated. |
| `RuleType` | enum | always | Which rule set produced it (`DesignRules`/`NetClass1`/`ClassToClass`/…). |
| `RuleLay` | Int | always | Layer the rule applies to. |
| `ObjectType1` / `ObjectType2` | enum | always | Kinds of the two offending objects (`Trace`/`Via`/`Pad`/`CopperPour`/…). |
| `cn1`/`cl1`/`cp1`, `cn2`/`cl2`/`cp2` | Int | always | Object reference quads for the two objects. |
| `ext1` / `ext2` | Int | always | Extra object qualifiers. |
| `nclass1` / `nclass2` | Int | always | Net-class indices involved. |

Child `<Lays>` holds a per-copper-layer `<Item>` `Y`/`N` presence list.

| Value | Name |
|---|---|
| `0` | Clearance |
| `1` | MinTrace |
| `2` | MinHole |
| `3` | MinRing |
| `4` | MaxRing |
| `5` | MaxPlatedHole |
| `6` | MaxNonPlatedHole |

| Value | Name |
|---|---|
| 7 | ShiftedVia |
| 9 | Drill |
| 10 / 11 | TraceLength |
| 12 | HiddenPad |
| 13 | PrimaryGap |
| 14 | UncoupledLength |
| 15 | LengthTolerance |
| 16 | PhaseTolerance |
| 17 | MaxNeckLength |
| 18 | DiffPairLoopback |

### <Tables> / <Table>

BOM, pick-and-place, layer-stack, hole-size and free tables. A table is a placed box with default cell font settings, an `<AutoUpdate>` generation spec, and a `<Cells>` grid in which each cell is its own text box.

```
<Table Id="0" X1="10" Y1="10" X2="60" Y2="40" Layer="Top Assy"
       Orientation="0" FontSize="8" FontSizeFloat="8">
  <Name>BOM</Name>
  <AutoUpdate Type="BOM" Units="mm" RowType="Name and Value" Header="Y">
    <Supplier>...</Supplier>
    <Columns><Column Type="0" Align="Left" Width="20"><Name>RefDes</Name></
Column></Columns>
  </AutoUpdate>
  <RowComponents><RowCom><Item>0</Item><Item>1</Item></RowCom></RowComponents>
  <Cells><Cell><Cell X1="10" Y1="10" X2="30" Y2="14">
    <TextLines><TextLine>RefDes</TextLine></TextLines>
  </Cell></Cell></Cells>
</Table>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Table index. |
| `X1`/`Y1`/`X2`/`Y2` | Real | always | Table box corners. |
| `Layer` | enum | always | Documentation layer (see enum). |
| `NonSignal` | Int | always | Non-signal-layer id (when on a custom layer). |
| `Orientation` | enum | always | `0`/`90`/`180`/`270`. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| HideBorder | Bool (Y/N) | always | Hide the table border. |
| TextAlign | enum | always | Default cell alignment (Left/Center/Right). |
| FontVector | Bool (Y/N) | always | Default vector font. |
| FontMono | Bool (Y/N) | vector default | Default monospaced vector font. |
| FontSize | Int | always | Default cell font height, rounded. |
| FontSizeFloat | Real | always | Unrounded default font height. |
| FontWidth / FontScale | Real | always | Default stroke width / scale. |
| LineSpacing | Real | always | Default cell line spacing. |
| Group | Int | if >−1 | Group registry id. |
| PanelExclude | Bool (Y) | if set | Excluded from panelization. |
| Locked / Selected | Bool (Y) | if set | State flags. |

Text children <FontName> and <Name>.

| Value | Name (Layer) |
|---|---|
| 0 | Top Assy |
| 1 | Top Silk |
| 2 | Bottom Silk |
| 3 | Bottom Assy |
| 4 | Non-Signal |

**<AutoUpdate>**

The generation spec. RowType is written for every document type (its value comes from a different enum for Pick-and-Place than for other types).

| Attribute | Type | Written | Description |
|---|---|---|---|
| Type | enum | always | Document type: Text/LayerStack/BOM/Pick and Place/Hole Size. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| Units | enum | always | Common/inch/mil/mm. |
| RowType | enum | always | For Pick-and-Place: All/Top/Bottom. Otherwise: Components/Name/Name and Value/Name and Pattern/Name, Value and Pattern. |
| Header | Bool (Y/N) | always | Include a header row. |
| assyvariant | Int | always | Assembly-variant index. |
| bomindex / bomtotal | Bool (Y/N) | always | BOM index / total columns. |
| pickoff X/Y, pickmirror, pickorigin | Real/ Bool | always | Pick-and-place origin offset, mirror and origin flags. |

Text children: <AssemblyName>, <Separator>, and <Supplier> (unconditional). Child <Columns> holds <Column> entries — each with a Type (column-kind index), Align, Width, a <Name> and a <Title>.

**<RowComponents> / <RowCom> / <Item>**

Per-row component-id lists: each <RowCom> is one table row and holds the component Ids it aggregates as text <Item> values.

**<Cells> / <Cell>**

The grid. <Cells> holds one <Cell> per row, and each row <Cell> holds the row's cell <Cell> boxes. Every cell is a self-contained text box.

| Attribute | Type | Written | Description |
|---|---|---|---|
| X1/Y1/X2/Y2 | Real | always | Cell box corners. |
| TextAlign | enum | always | Cell alignment (Left/Center/Right). |
| FontVector / FontMono | Bool (Y/N) | always / vector | Cell font family. |
| FontSize / FontSizeFloat | Int / Real | always | Cell font height, rounded and un-rounded. |
| FontWidth / FontScale | Real | always | Stroke width / scale. |
| LineSpacing | Real | always | Cell line spacing. |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `TextWidth` / `TextHeight` | Real | with text, computed | Derived text bounding size (export-only). |

Cell text children: `<FontName>` and a `<TextLines>`/`<TextLine>` list.

### <Dimensions> / <Dimension>

Dimensions and pointer callouts. A dimension records its measured points, the dimension-line origin, its layer, the objects it snaps to, and text/font settings.

```
<Dimension Id="0" Type="Horizontal" X1="10" Y1="5" X2="30" Y2="5"
           XD="20" YD="8" Layer="Top Silk" Units="mm" ShowUnits="Y"
           Connected1="Pad" Connected2="Pad">
  <PointerText>20.00 mm</PointerText>
</Dimension>
```

| Attribute | Type | Written | Description |
|---|---|---|---|
| `Id` | Int | always | Dimension index. |
| `Type` | enum | always | `Horizontal`/`Vertical`/`Free`/`Radius`/`Pointer`. |
| `PointerMode` | enum | Pointer only | `Coordinates` or `Comment`. |
| `X1` / `Y1` | Real | always | First measured point. |
| `X2` / `Y2` | Real | always | Second measured point. |
| `XD` / `YD` | Real | always | Dimension-line origin. |
| `Layer` | enum | always | Board layer (board-layer enum, e.g. `Top Silk`). |
| `NonSignal` | Int | always | Non-signal-layer id (when applicable). |
| `Angle` | Real | always | Text angle. |
| `Size` | Real | always | Arrow/extension size. |
| `ExternalRadius` | Real | always | Radius-dimension extension factor. |
| `Units` | enum | always | `Common`/`inch`/`mil`/`mm`. |
| `ShowUnits` | Bool (Y/N) | always | Append the unit label. |
| `Connected1` / `Connected2` | enum | always | Kind of object each end snaps to (`Pad`/`Pattern Shape`/`Pattern Hole`/`Net`/`Plane`/`Shape`/`Board Outline`/`Origin`/`Panel`). |

| Attribute | Type | Written | Description |
|---|---|---|---|
| `cn1`/`cl1`/`cp1`, `cn2`/`cl2`/`cp2` | Int | always | Object reference quads for the two connection ends. |
| `FontVector` / `FontMono` | Bool (Y/N) | always / vector | Font family. |
| `FontSize` / `FontSizeFloat` | Int / Real | always | Font height, rounded and unrounded. |
| `FontWidth` / `FontScale` | Real | always | Stroke width / scale. |
| `TextWidth` / `TextHeight` | Real | computed | Derived text bounding size (export-only). |
| `Group` | Int | if >−1 | Group registry id. |
| `PanelExclude` | Bool (Y) | if set | Excluded from panelization. |
| `Locked` / `Selected` | Bool (Y) | if set | State flags. |

Text children: `<FontName>` and `<PointerText>` (the dimension/pointer caption).