

# **Productivity Toolbox User Guide**

## **Variant Assembly**

**February 2017**

## Table of Contents

1	Overview .....	1
2	Use model.....	4
2.1	Launching the utility.....	4
2.2	Basic Use model .....	5
3	Detailed description.....	6
3.1	General tab .....	6
3.1.1	Generate Assembly View for .....	6
3.1.2	Handling of DNI Parts.....	6
3.1.3	Handling of Alternate parts .....	10
3.2	Drawing Options tab.....	12
3.2.1	Component Labels .....	12
3.2.2	Component Outline .....	16
3.3	Group color tab .....	18
3.4	Advanced tab .....	21
3.4.1	Bottom view handling .....	21
3.4.2	Additional layers.....	24
3.4.3	Miscellaneous .....	25

## Table of Figures

Figure 1: Variant Assembly .....	3
Figure 2: Variant Assembly main form.....	4
Figure 3: Generate Assembly View for .....	6
Figure 4: Variant selection.....	6
Figure 5: Annotation style for DNI parts.....	6
Figure 6: DNI parts with option "Draw Cross Label" .....	8
Figure 7: Remove second label for DNI parts .....	9
Figure 8: Color option for DNI components.....	9
Figure 9: Annotation style for Alternate parts.....	10
Figure 10: Alternate parts with option "Modify Labels" .....	10
Figure 11: Alternate parts with option "Modify Color" .....	11
Figure 12: Alternate parts with option "Modify Graphics" .....	11
Figure 13: Impact of component outline subclass when hatching .....	12
Figure 14: Component label options .....	12
Figure 15: Label value options.....	13
Figure 16: Annotation of VALUE-property to variant assembly view .....	13
Figure 17: Valid subclasses for label master .....	14
Figure 18: Annotating a second label .....	14
Figure 19: Component outline drawing options.....	16
Figure 20: Component outline subclasses .....	16
Figure 21: Using fixed outline width .....	17
Figure 22: Cross scale.....	17
Figure 23: Group color specification .....	18
Figure 24: Available properties when specifying color rules.....	18
Figure 25: Example: Enable Variant color wins over group color .....	20
Figure 26: Disable Variant color wins over group color .....	20
Figure 27: Coloring example .....	20
Figure 28: Bottom view options .....	21
Figure 29: Bottom view mirroring.....	21
Figure 30: Example original board views for top and bottom .....	22
Figure 31: Example bottom view: No Action .....	22
Figure 32: Example bottom view: Mirror on bottom only .....	23
Figure 33: Example bottom view: Mirror to variant top.....	23
Figure 34: Include additional layers .....	24
Figure 35: Include additional layers example.....	25
Figure 36: Miscellaneous options .....	25
Figure 37: Color view files .....	26

# 1 Overview

Both schematic tools – *Design Entry CIS* and *Design Entry HDL* – offer the ability to create variants. The variant information is passed to PCB Editor using a file called `variants.lst`. In *PCB Editor* the variant information can be taken to create variant assembly views.

**Variant Assembly** is an application which gives customers even more flexibility when creating variant assembly views. This document describes the features and use model of this application.

**Variant Assembly** covers the following aspects:

- **Automation**  
All variant assembly views can be generated in one step. In addition it also allows to generate view for the core design.
- **Customizable label content**  
The standard function from PCB Editor only annotates the reference designator to the variant subclasses. Using **Variant Assembly** it's possible to annotate other properties such as *VALUE* and *PART\_NUMBER*. Furthermore not only one label can be annotated: A second label can be written to the variant view if necessary (e.g. Refdes & Value)
- **Customizable component outline**  
The standard function from PCB Editor always takes the information from the package geometry assembly subclasses. **Variant Assembly** enables customers to use the graphic data from other subclasses such as *Place\_Bound*, *Silkscreen*, *DFA\_Bound* instead.
- **Special options for DNI components**  
The standard function from PCB Editor removes the graphics and labels from DNI components completely. **Variant Assembly** provides additional capabilities for example, instead of removing everything it's also possible to draw a thick cross through the graphics outline or to change the color of a DNI component
- **Special options for alternate components**  
These parts are components, where certain attributes such as *VALUE* or *PART\_NUMBER* differ from the core design. PCB Editor has no function to distinguish these parts in the variant assembly view. **Variant Assembly** enables users to highlight alternate components in three ways:
  - Modify variant labels by adding a prefix or suffix
  - Use a different color for outline and label
  - Draw the component outline by hatched shape.

- **Mirror option for bottom view**

In order to improve readability users often want to mirror the bottom view. **Variant Assembly** allows mirroring the bottom view automatically on its subclass. It's even possible to arrange the top view and the mirrored bottom view side by side on one documentation subclass. The spacing and the direction (e.g. to the right) can be specified.
- **Include additional layers to variant view**

Additional layers such as *BOARD GEOMETRY/OUTLINE* or *DRAWING FORMAT/OUTLINE* can be included to the variant view, so that in the end all data is written to one documentation subclass.
- **Group colors**

**Variant Assembly** enables users also to colorize components. In a separate tab the user specifies color rules. Various properties can be chosen. As an example all components where reference designator starts with R\* shall be colored red. Several rules can be applied, wildcards are supported.
- **Update Custom Variables**

(Available in a future release)
- **Additional features**
  - Variant data (graphics and labels) are attached to the symbol instance by default. Therefore moving a component - once data have been created – will also move the data on the variant assembly view.
  - Users can specify where the text attributes for the variant labels are derived from. For example a user wants to annotate the variant value (property *VALUE*) to the variant assembly view, but the footprint (symbol \*.dra) has no *VALUE* label defined. **Variant Assembly** allows users to specify where the text attributes (text block, rotation, xy, mirror ..) are derived from. In most cases it is the *REF DES/ASSEMBLY* subclass as this one is mandatory when defining a part.
  - During update the attributes of existing variant labels can be taken into account. For example it might be possible that users – after data have been generated a first time - modify the text attributes of variant labels (such as block, xy or rotation) in order to improve readability. These attributes can be retained when recreating variant assembly views.
  - Settings are stored in database which ensures identical outputs when the application is launched next time.
  - Since **Variant Assembly** offers many features which might be suitable for the base design as well (e.g. mirroring the bottom view to top), the user can also choose the *Core design* to process.

# Variant Assembly

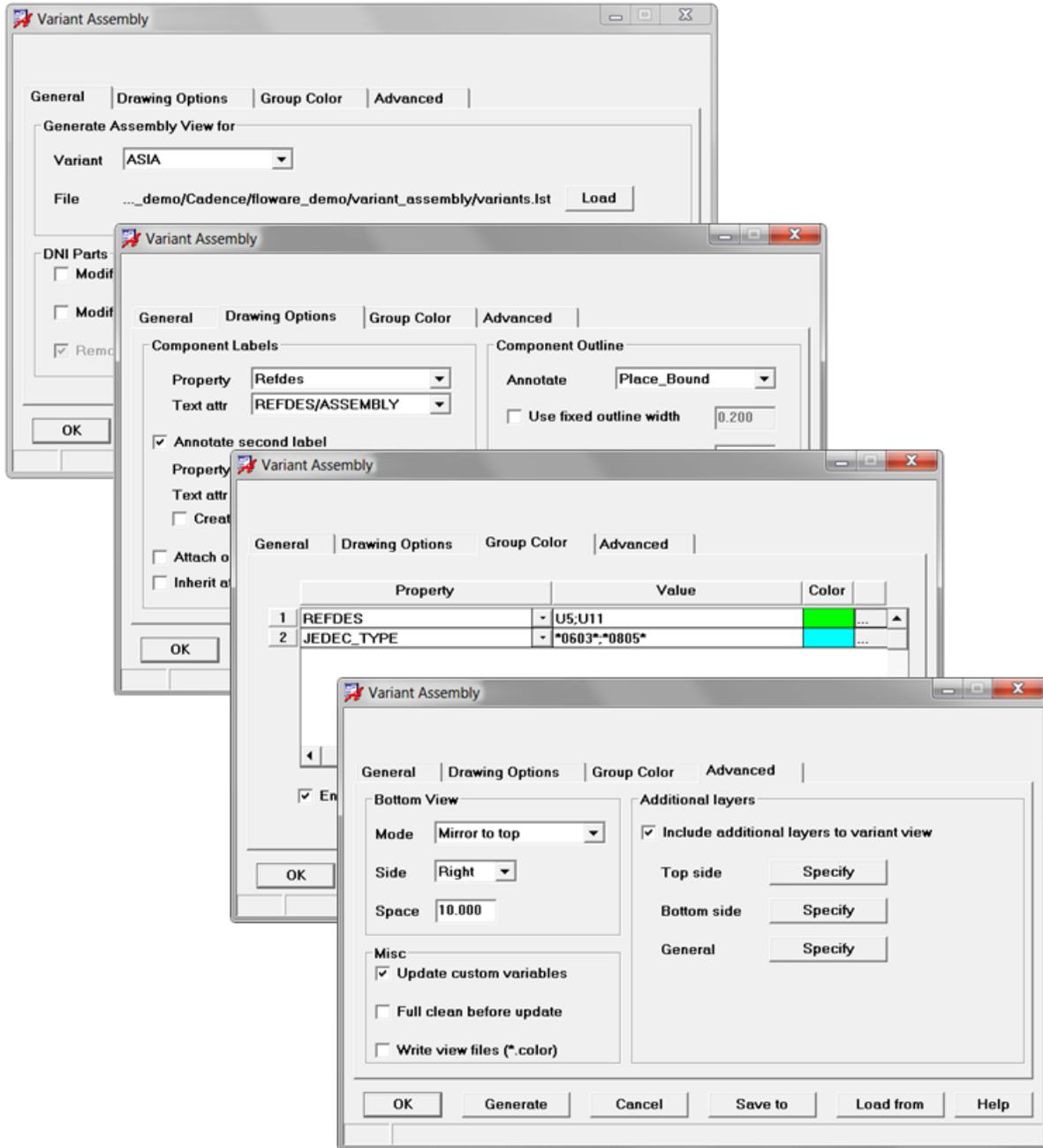


Figure 1: Variant Assembly



Note: A sample database covering the features of this application can be found under  
`<CDSROOT>/share/pcb/toolbox/getting_started/mfgdoc`

## 2 Use model

### 2.1 Launching the utility

**Variant Assembly** can be started from Pulldown menu or by entering the command `tbx variassy` in the console window.

Once the command has been launched a form appears. It contains four tabs.

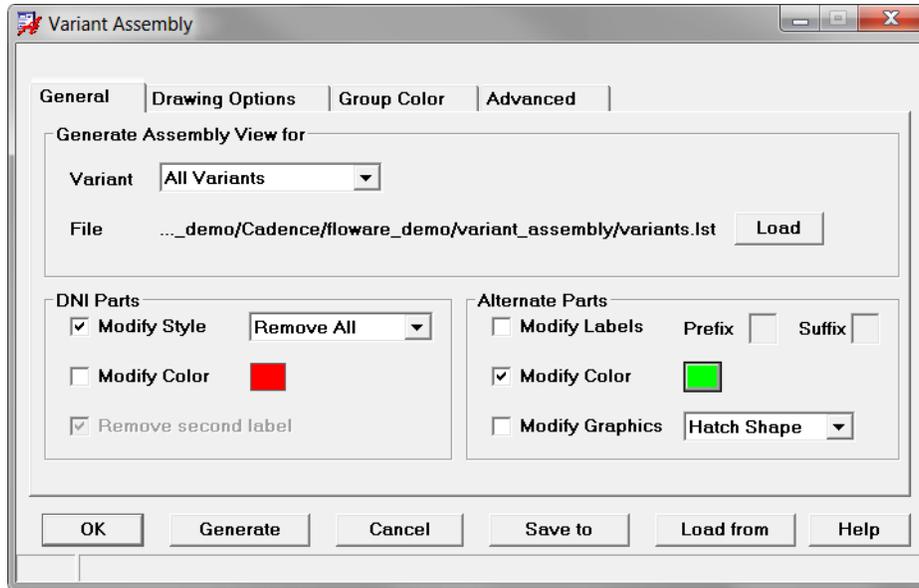


Figure 2: Variant Assembly main form

## 2.2 Basic Use model

The basic use model is as follows:

- From section *Generate Assembly View for* choose the variant you want to create an assembly variant view for. If *All Variants* is selected, **Variant Assembly** creates a variant assembly view for each variant found in file `variants.lst`. Data is written to manufacturing subclasses `MANUFACTURING/<variant_name>_TOP` and `BOTTOM`. **Variant Assembly** automatically reads the contents of file `variants.lst` file in the current working directory, the full path to the file is displayed in the form. However, if necessary, the user can load a variants file from another location using the *Load* button.
- Specify the style for components which are not installed in section *DNI parts*. Choices are *Remove All*, *Remove Label Only*, *Cross Labels* and *Cross Graphics*. In addition a color can be specified. Use appropriate checkboxes.
- Specify the style for alternate components in section *Alternate Parts*. Three switches are provided. Use this section only if you want to emphasize or distinguish alternate components from other components in the design.
  - *Modify labels*  
When checked the variant labels (e.g. `VALUE` or `PART_NUMBER`) are modified by using a prefix or suffix as specified in the corresponding fields.
  - *Modify color*  
When checked alternate components will be drawn by a different color. Use the color swatch to select a color.
  - *Modify graphics*  
When checked the component outline of an alternate component is drawn as a hatched shape.
- Click *Generate*. The variant assembly view is written to the manufacturing subclasses `MANUFACTURING/<variant_name>_TOP` and `BOTTOM`.



Note: **Variant Assembly** writes data to subclasses `MANUFACTURING/<variant_name>_TOP` and `BOTTOM`. **All data is organized in several groups (e.g. `VAS_ASIA_TOP_DEFAULT`)**. So you might add additional notes elements to the variant layer which are retained during the next update, because **Variant Assembly** only removes and updates these group objects on variant layers. In terms of migration that means if you use **Variant Assembly 3.0** and higher a first time on designs with existing variant views (either generated by standard PCB Editor command or earlier version of **Variant Assembly**), you should enable checkbox *Full clean before update* from *Advanced* tab.



Note: **Variant Assembly** stores settings in database as an attachment. This ensures identical output when the tool is launched next time. However you can use *Save to* and *Load from* to share configuration settings from one database to another.

## 3 Detailed description

### 3.1 General tab

#### 3.1.1 Generate Assembly View for

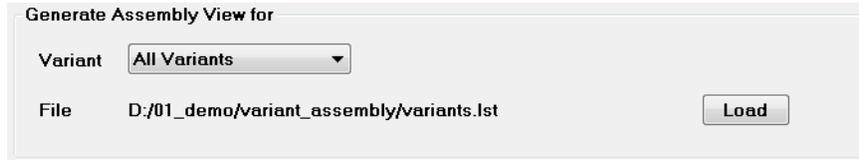


Figure 3: Generate Assembly View for

The path to the `variants.lst` file in the current working directory is displayed. Using the *Load* button another file can be loaded.

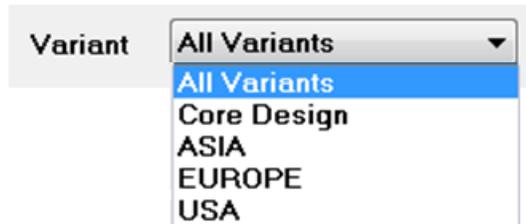


Figure 4: Variant selection

In the field *Variant* the variant to be processed can be selected. PCB Editor reads the file `variants.lst` and populates this drop down box with the variant names. When *All Variants* is selected, all variants are processed in one step. By selecting *Core Design* an assembly view for the Core design (not affected by any variant) will be created.

#### 3.1.2 Handling of DNI Parts

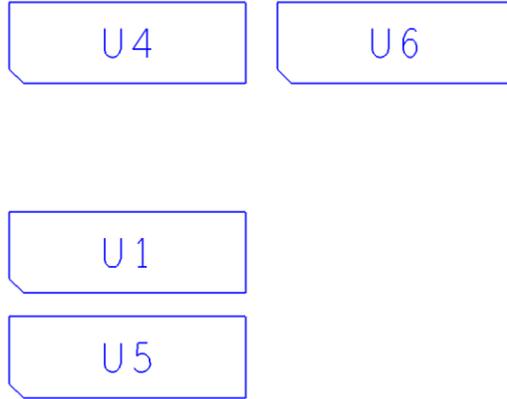
Use this section to specify how DNI Parts should appear in the variant assembly view.



Figure 5: Annotation style for DNI parts

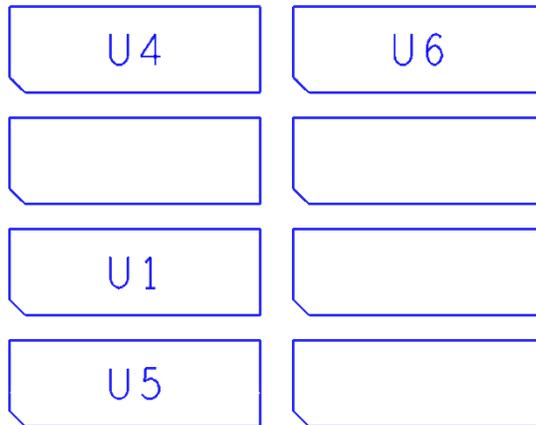
The following display styles for DNI components are available once checkbox *Modify Style* has been enabled

- *Remove All*  
Graphics and labels are completely removed.



**Figure 7: DNI Parts with option "Remove All"**

- *Remove Label Only*  
Only graphics appear but no label.



**Figure 8: DNI parts with option "Remove Label Only"**

- **Cross Graphics**  
A thick cross is drawn through the component, in order to notify that this component will not be assembled.

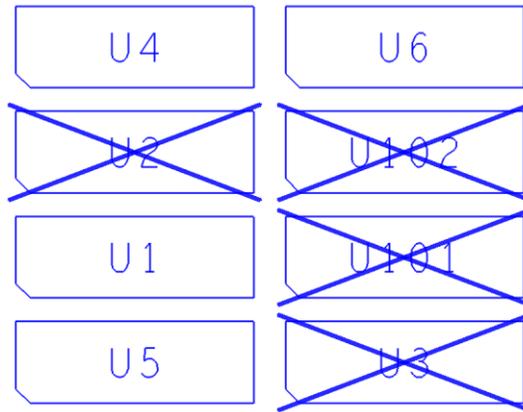


Figure 9: DNI parts with option "Draw Cross Graphics"

- **Cross Label**  
A thick cross is drawn through the component label, in order to notify that this component will not be assembled.

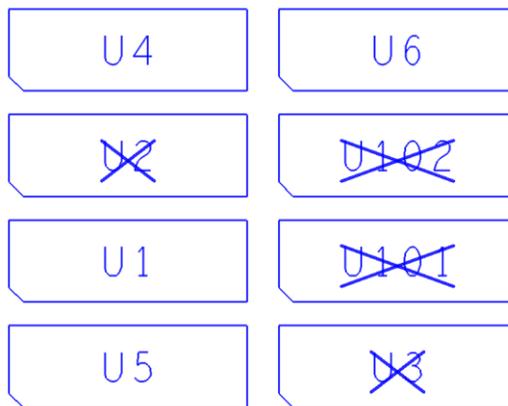


Figure 6: DNI parts with option "Draw Cross Label"



Note: Option *Remove second label* only applies if you have enabled the annotation of a second label to the variant view. (Refer to section 3.2.1). If two labels will be annotated to the same view (e.g. *Refdes* and *Value*) the second label can be suppressed when option *Draw Cross Label* or *Draw Cross Graphics* are selected.

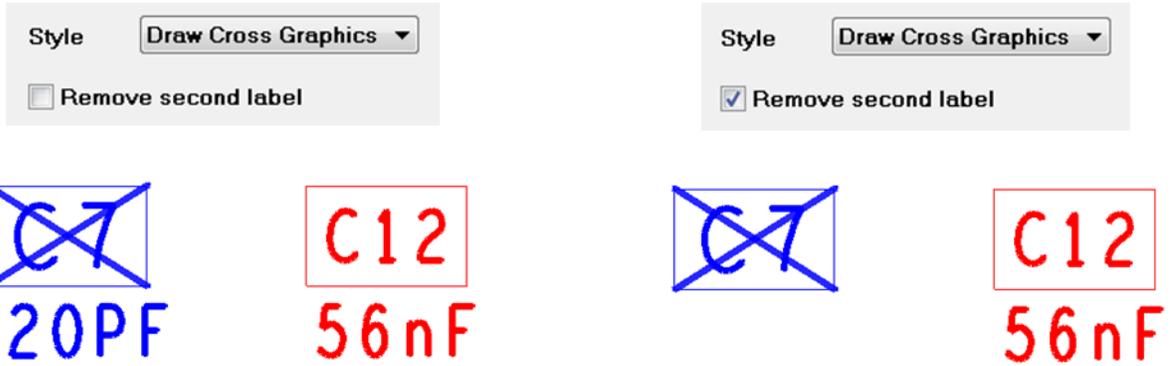


Figure 7: Remove second label for DNI parts

Furthermore a color can be specified for DNI components. Enable checkbox *Modify Color* and use the color chooser to select an appropriate color.

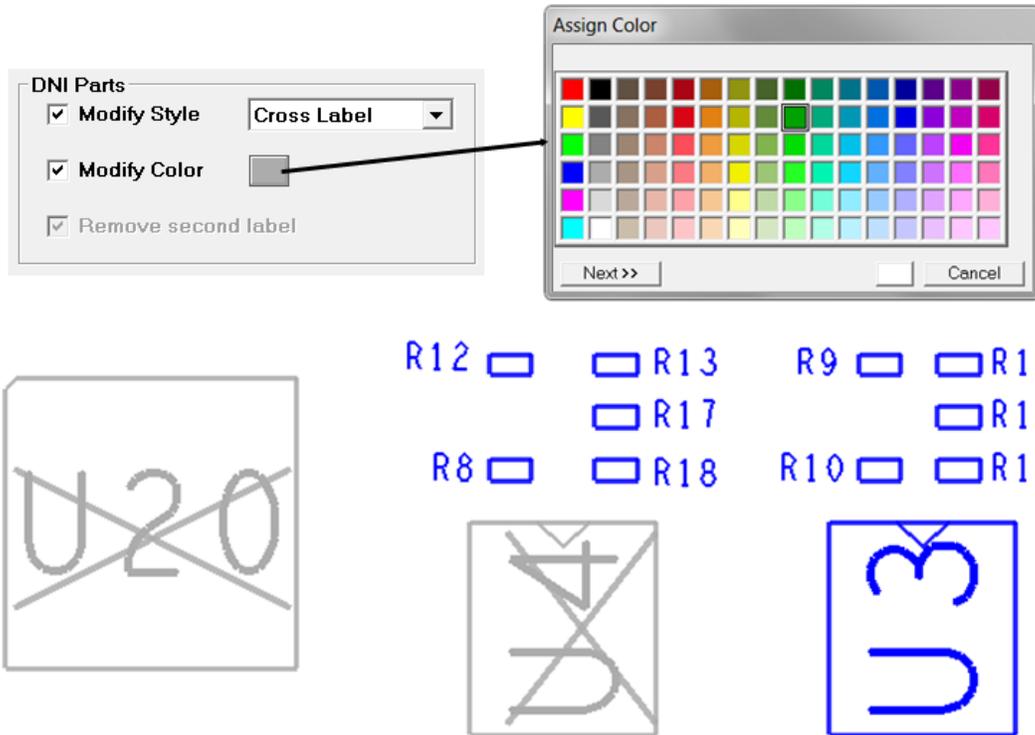


Figure 8: Color option for DNI components

### 3.1.3 Handling of Alternate parts

Use options in this section in order to distinguish alternate parts from other components in the variant assembly view. Three switches are provided.

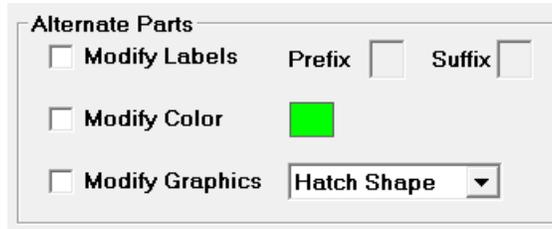


Figure 9: Annotation style for Alternate parts

- Modify Labels**  
 When checked the variant labels (e.g. *VALUE* or *PART\_NUMBER*) are modified by using a prefix or suffix as specified in the fields *Prefix* and *Suffix*.

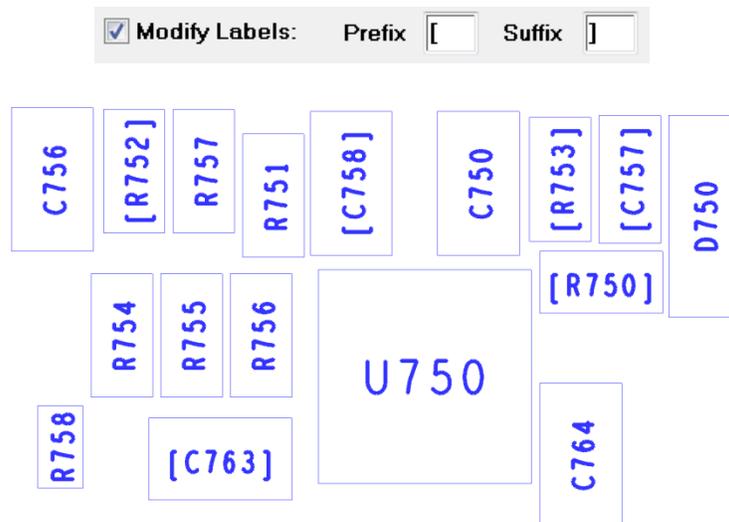


Figure 10: Alternate parts with option "Modify Labels"

- **Modify Color**  
When checked alternate components will be drawn by a specified color. The color chooser becomes active once the checkbox has been enabled.

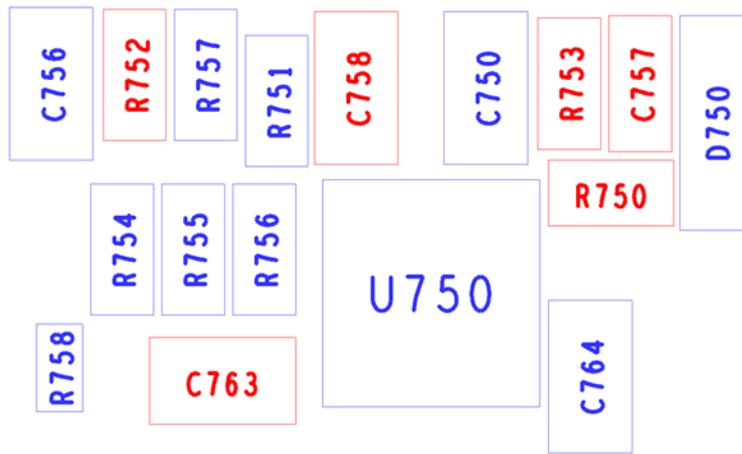


Figure 11: Alternate parts with option “Modify Color”

- *Modify graphics*  
When checked the component outline of an alternate component is drawn as a hatched shape.

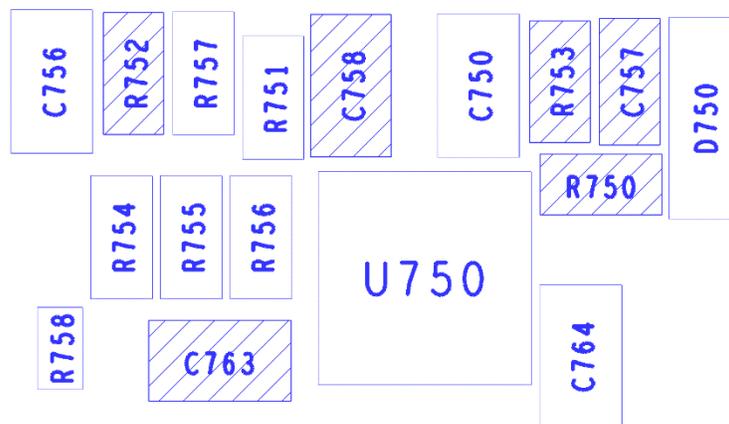


Figure 12: Alternate parts with option "Modify Graphics"



Note: This option is only available when the graphical data for the component contains shapes. This is only guaranteed for *PACKAGE GEOMETRY* subclasses *DFA\_Bound* and *Place\_Bound*. Therefore this switch can only be activated, when the field *Component Outline Annotate* (tab *Drawing Options*) is equal to *DFA\_Bound* or *Place\_Bound*. Refer to next picture.

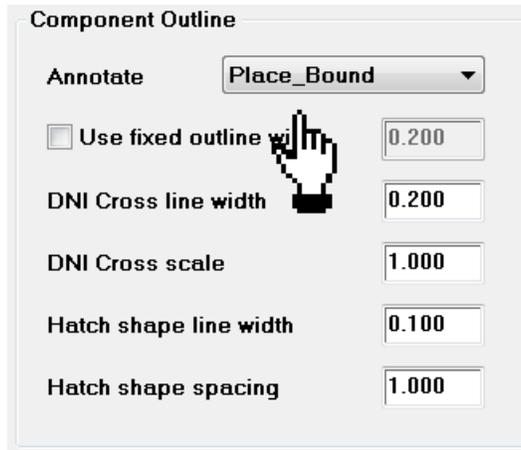


Figure 13: Impact of component outline subclass when hatching

## 3.2 Drawing Options tab

### 3.2.1 Component Labels

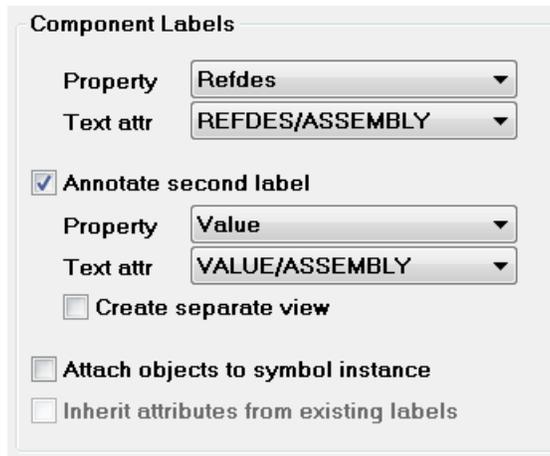


Figure 14: Component label options

- Property**  
 Specifies the label content to be written to the variant assembly view. The default value is Refdes which means that the reference designator value (e.g. R1, IC20..) is written as text label to the variant view. The following choices are available: *Part\_Number*, *Refdes*, *Tolerance*, *Value*. In case of alternate components the variant specific property values will be extracted from file `variants.lst` and written to the variant assembly view. For base components (components which are not affected at all) the core values are always written to the variant view.



Figure 15: Label value options

 Note: If the attribute cannot be extracted (for example VALUE-property does neither exist in the database nor in file `variants.lst`, a default label “??” is written to the variant assembly view.

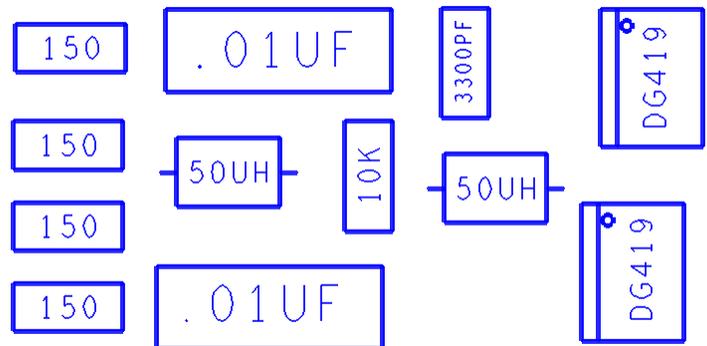


Figure 16: Annotation of VALUE-property to variant assembly view

- *Text attr*  
 This field specifies where the text attributes for the label content are derived from. For example a user wants to annotate the variant value (property VALUE) to the variant assembly view, but the footprint (symbol \*.dra) has no VALUE label defined. **Variant Assembly** allows users to specify where the text attributes (text block, rotation, xy, mirror...) are derived from. In most cases it is the *REF DES/ASSEMBLY* subclass as this one is mandatory when defining a part. The following figure shows all possible subclasses where labels can be defined. The default value is *REF DES/ASSEMBLY*.

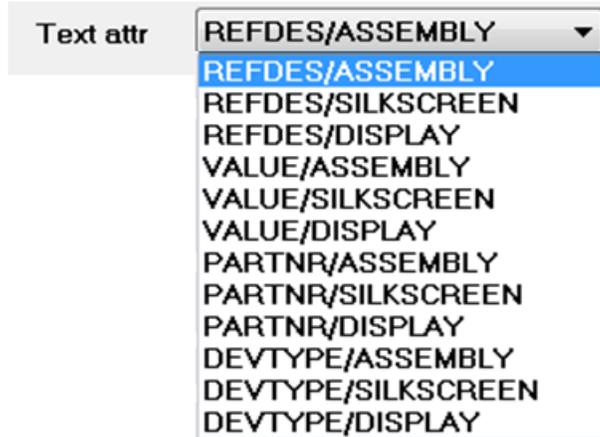


Figure 17: Valid subclasses for label master



Note: The value selected only refers to the text attributes, not the label value itself. The label value (*Refdes*, *Part\_Number*, *Value* ...) is specified in field *Property* above.

- Annotate second label**  
 By activating this checkbox a second label will be annotated to the variant assembly view. This enables users to annotate another property to the variant view, e.g. *Refdes* in conjunction with *Value*. The property and its text attributes need to be specified in a similar way. Of course the property value itself corresponds to the actual value of the selected variant. (Refer to the following example, where C20 has a variant value of 68pF)

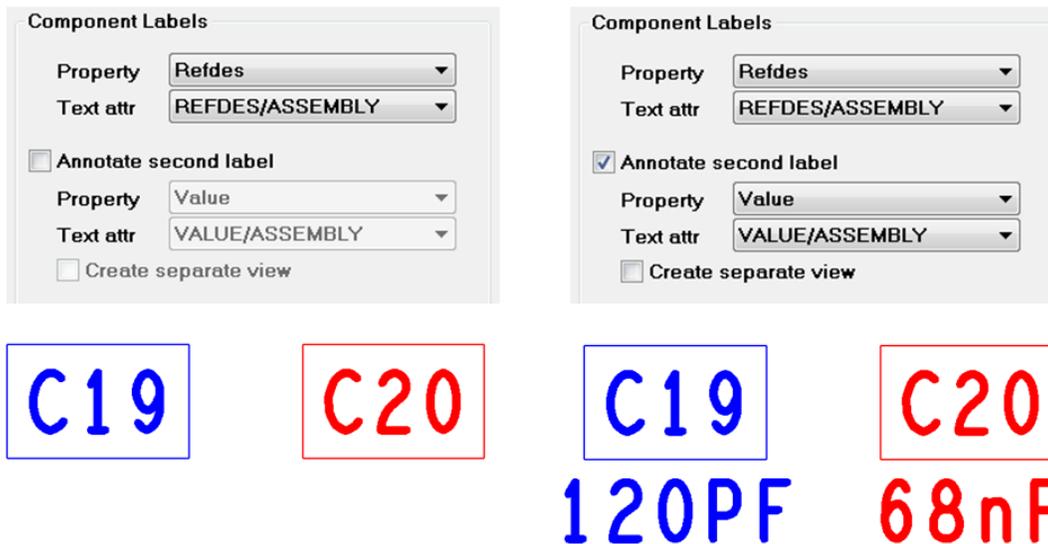


Figure 18: Annotating a second label

- Create separate view**  
 By enabling this checkbox a separate view will be written for the second label.



This gives users the flexibility to create for example one variant view containing graphics and reference designators and a second view containing graphics and Value properties. While the subclass for the first view is always *MANUFACTURING/<var\_name>\_TOP* (and *BOTTOM*) the second view has an extension “\_2”, e.g. *MANUFACTURING/<var\_name>\_2\_TOP* (and *BOTTOM*). In other words: The only difference between both views is the label content.

- *Attach objects to symbol instance*

When checked variant data (labels and graphics) are attached to the symbol instance. Therefore moving a component - once the variant assembly data have been created - will also move the data on the variant assembly view. The default value is on.



Note: When this option is checked, be careful when running *Place – Update Symbols*. This command removes all data from the corresponding variant subclasses, because all symbols are replaced with the elements from the library

- *Inherit attributes from existing labels*

During an update the attributes of existing variant labels can be taken into account. For example it might be possible that users – after data have been generated a first time - modify the text attributes of variant labels (such as block, xy or rotation) in order to improve readability. These attributes can be retained when recreating variant assembly data. The default value is on.



Note: This option is greyed out when *Attach objects to symbol instance* is unchecked. It is also greyed out when checkbox *Create separate view* for the second label is inactive, which means that the user annotates two labels per component the same subclass

### 3.2.2 Component Outline

This section provides some advanced drawing options.

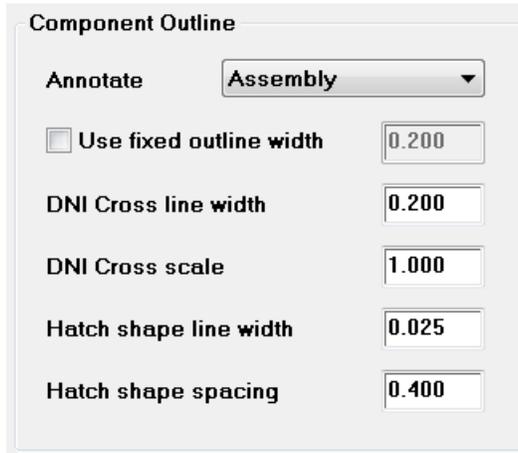


Figure 19: Component outline drawing options

- Annotate**  
 Specifies the package geometry subclass where the graphics information such as outline is taken from. The following choices are available: *Assembly*, *DFA\_Bound*, *Display*, *Place\_Bound* and *Silkscreen*. The graphics data from these subclasses (TOP and BOTTOM) are read and written to the assembly variant view. The default value is *Assembly* which refers to *PACKAGE GEOMETRY/ASSEMBLY\_TOP* and *BOTTOM* subclass.

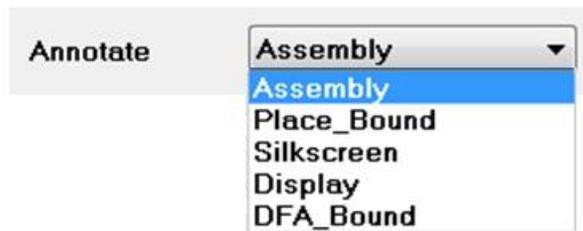


Figure 20: Component outline subclasses

 Note: If there are no graphics data available on the specified package geometry subclass nothing is written to the variant assembly view.

 Note: If graphics data from the selected package geometry subclass contain filled shapes (which is always the case for *DFA\_Bound* and *Place\_Bound*), the shapes are drawn as unfilled shapes to the variant subclass. Otherwise text labels would not be readable.

- Use fixed outline width*  
 Using this option it's possible to define a fixed line width to be used for the component outline so the existing width from outline segments will be overridden. This option applies also to shapes which might exist on *PACKAGE GEOMETRY* subclasses (e.g. *PLACE\_BOUND*). In this case the shape outline will be decomposed to line objects.

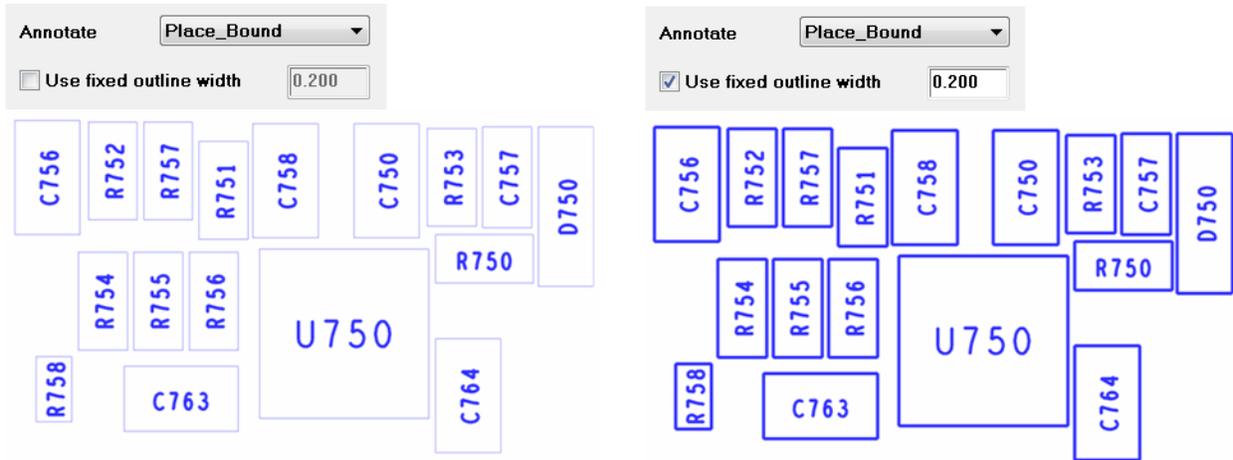


Figure 21: Using fixed outline width

- DNI Cross line width*  
 Specifies the line width of the cross drawn through the graphics or label bounding box.
- DNI Cross scale*  
 The start and end coordinates of the cross drawn through the label or graphics are derived from the corresponding bounding box corners. Specifying a value in percent stretches start end coordinates of the cross with respect to the center of the bounding box. The default value is 1.0, which means that no scaling occurs. A value < 1.0 moves the start end coordinates inside the bounding box.

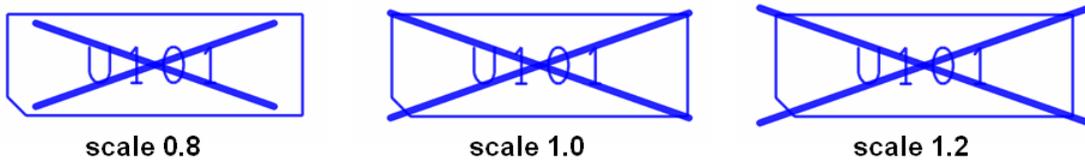


Figure 22: Cross scale

- Hatch shape line width*  
 Specifies the line width for the hatch shape.
- Hatch shape spacing*  
 Specifies the spacing for the hatch shape.

### 3.3 Group color tab

In this tab the user specifies the color to be used for the components.

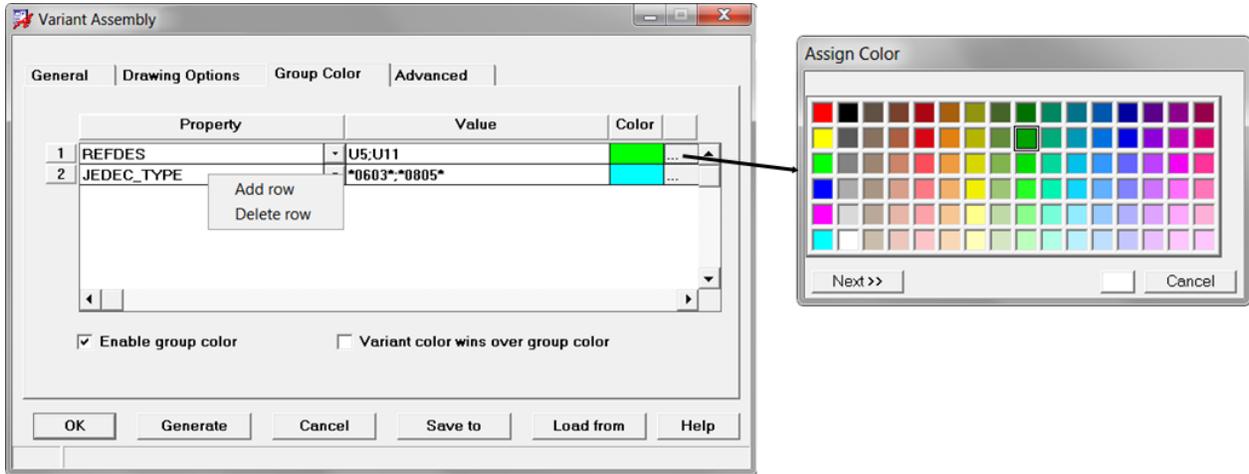


Figure 23: Group color specification

Each row in the grid represents a color rule to be used when generating variant views. Using context menu *Add row* and *Delete row* rules can be added and removed.

In the example above the first rule states that component U5 has to be drawn in green. The second rule states that components where the footprint name matches 0805 or 0603 have to be drawn in blue.

In the *Property* column the property name has to be defined. The predefined ones are shown in the picture below:

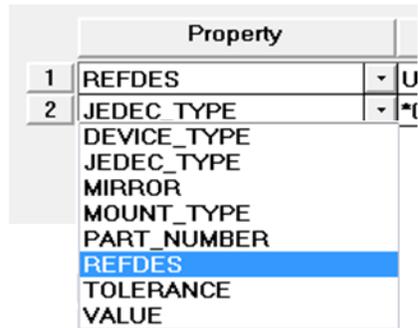


Figure 24: Available properties when specifying color rules



Note: This field is editable. Beyond the predefined properties you may also enter your own property if appropriate

In the *Value* column the match pattern can be defined. You can enter more than one match pattern if necessary. The “,” character acts as delimiter. In addition to that you can use wildcards “\*” in any combination.



Note: When using property *MIRROR*, the mirror status of the component will be checked. Valid values are “YES” and “NO”



Note: When using property *MOUNT\_TYPE*, the component will be checked whether it is an SMT or a THT component. If the property with the same name is attached to the component definition (or instance) the value will be taken. In addition – when the property is not found on the component - Variant Assembly automatically extracts the mounting type by analyzing the electrical pins of the footprints (not mechanical pins!!).

SMD pads	THT pads	MOUNT_TYPE
NO	NO	UNDEF
NO	YES	THT
YES	NO	SMT
YES	YES	UNDEF

In the *Color* column, the color name will be assigned.

Moreover, two switches are provided:

- *Enable group color*  
When checked the components will be colored as specified by the color rules.
- *Variant color wins over group color*  
The situation might occur where on the one hand a component is a variant alternate component (with *Modify Color* enabled) and on the other hand the same component matches one of the color rules. This switch can then be used to define which color wins.



## 3.4 Advanced tab

### 3.4.1 Bottom view handling

This section specifies how data for the bottom view is handled



Figure 28: Bottom view options

- *Mode*

Three choices are available:

- When choosing *No Action*, variant data for the bottom view is written to the documentation subclass `MANUFACTURING/<variant_name>_BOTTOM` without further modification.
- When choosing *Mirror on bottom only*, variant data for the bottom view is mirrored to the documentation subclass `MANUFACTURING/<variant_name>_BOTTOM`. This improves readability significantly.
- When choosing *Mirror to top*, variant data for the bottom view is mirrored to the documentation subclass `MANUFACTURING/<variant_name>_TOP`. In order to avoid any overlapping with data from variant top, the direction and the spacing needs to be specified in order to arrange top and bottom view data side by side.



Figure 29: Bottom view mirroring

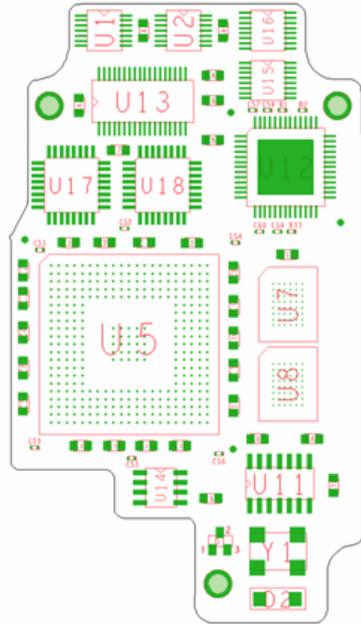
- *Side*

Specifies the side to which the bottom view will be mirrored. Choices are *Left*, *Right*, *Top* and *Bottom*. This option is only available when *Mode* is set *Mirror to variant top*.

- *Space*

Specifies the spacing between the variant top and bottom view on documentation subclass `MANUFACTURING/<variant_name>_TOP`. The default value is 10 mm. This option is only available when *Mode* is set *Mirror to top*.

Original board top view



Original board bottom view

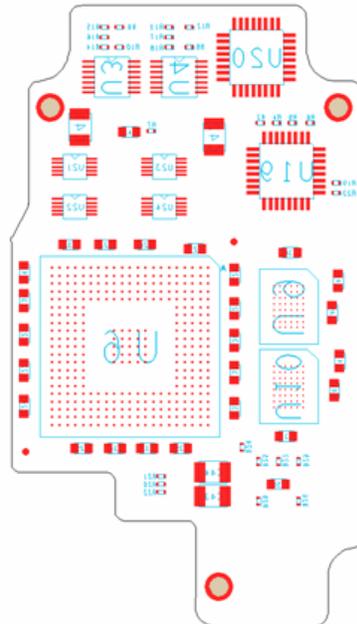


Figure 30: Example original board views for top and bottom

Subclass <var\_name>\_BOTTOM

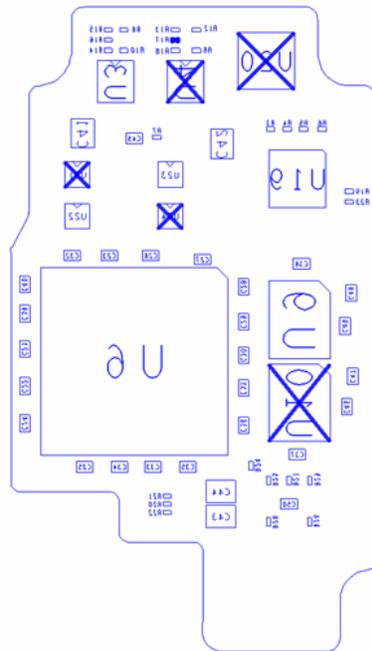


Figure 31: Example bottom view: No Action

Subclass <var\_name>\_BOTTOM

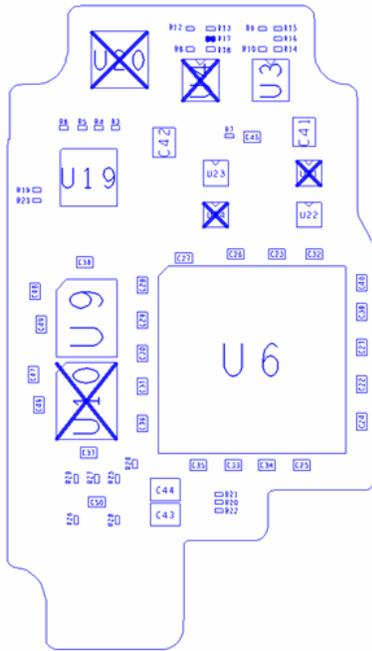


Figure 32: Example bottom view: Mirror on bottom only

Subclass <var\_name>\_TOP

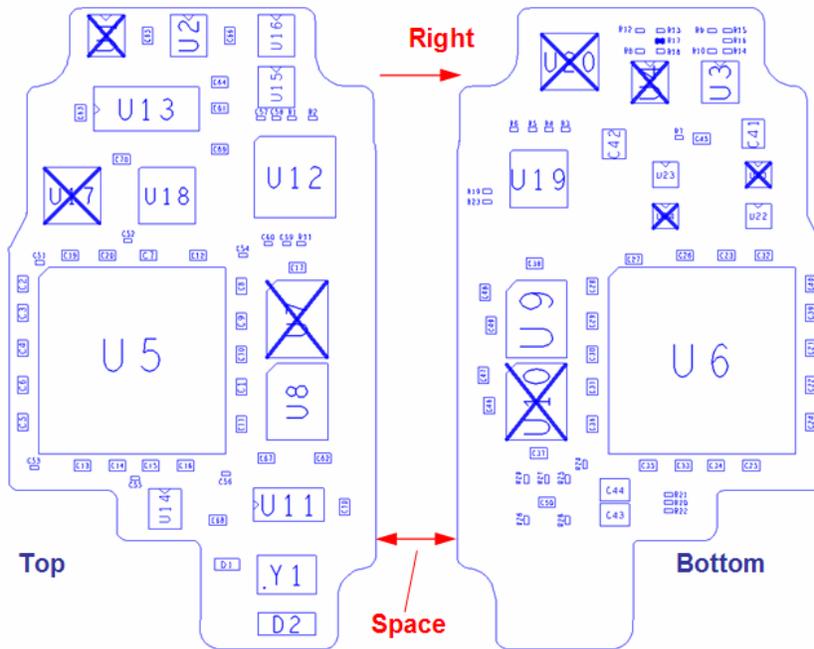


Figure 33: Example bottom view: Mirror to variant top

### 3.4.2 Additional layers

This section can be used to include additional layer information to the variant view on the documentation subclass *MANUFACTURING/<var\_name>\_TOP* and *BOTTOM*. When checked the layers can be specified for *Top side*, *Bottom side* and *General*

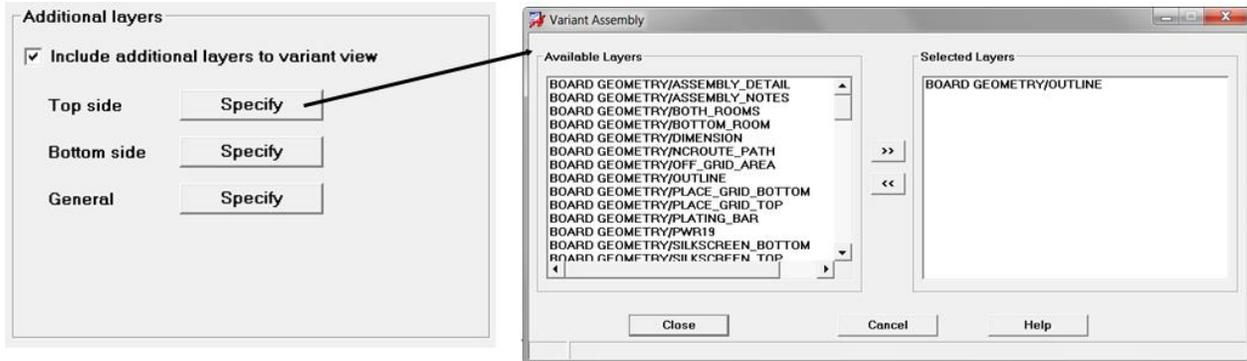


Figure 34: Include additional layers

**Variant Assembly** distinguishes between layers for Top side, Bottom side and General layers.

- *Top side*  
Layers specified in this section will be annotated together with variant component data for TOP side. The default entry is *BOARD GEOMETRY/OUTLINE*
- *Bottom side*  
Layers specified in this section will be annotated together with variant component data for BOTTOM side. The default entry is *BOARD GEOMETRY/OUTLINE*. However information on these layers will be mirrored together with component data, if one of the mirror option have been selected.
- *General*  
Data from these layers are simply copied to the documentation subclass without any mirroring. The default entry is *DRAWING FORMAT/OUTLINE*.



Note: Usually the layers specified for *Top side* and *Bottom side* are identical (e.g. *BOARD GEOMETRY/OUTLINE*). But in some cases let's say fiducials appear on top and bottom side at different locations while the mirror option has been selected, separate definitions must be possible.

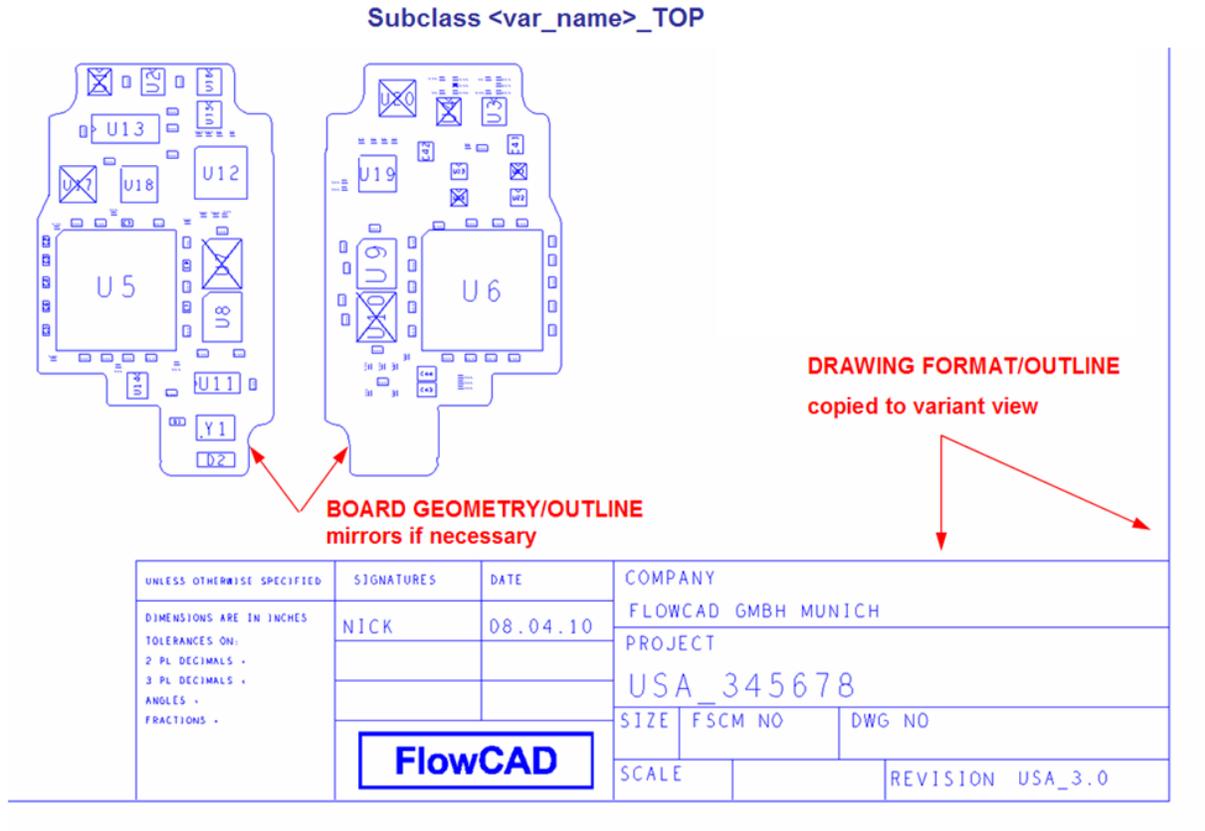


Figure 35: Include additional layers example

### 3.4.3 Miscellaneous

The following options are available



Figure 36: Miscellaneous options

- Full clean before update*  
 When disabled **Variant Assembly** only updates the contents on the variant layers that was generated by itself. Any additional data which was added manually by the user (e.g. notes, graphics) are retained. When enabled Variant Assembly clears the layer completely.

- *Update custom variables*  
When checked custom variables will be updated for the current variant. Refer to **Custom Variables** user guide for more information. Usually the text labels reside on layers other than the layer of the current variant view. If *Copy labels to variant view* is checked, the text labels will be copied to the layer of the current variant view.
- *Write view files (\*.color)*  
When checked a colorview file with the extension “.color” will be written to the current working directory for the selected variant and can be used for review or plotting purposes in the Visibility panel.

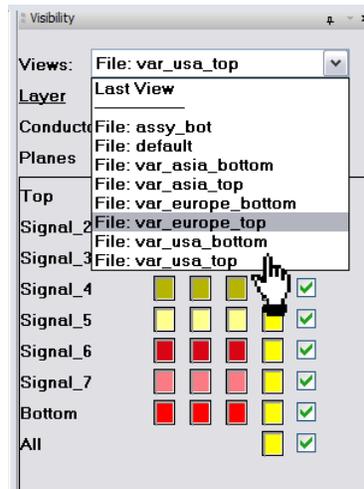


Figure 37: Color view files