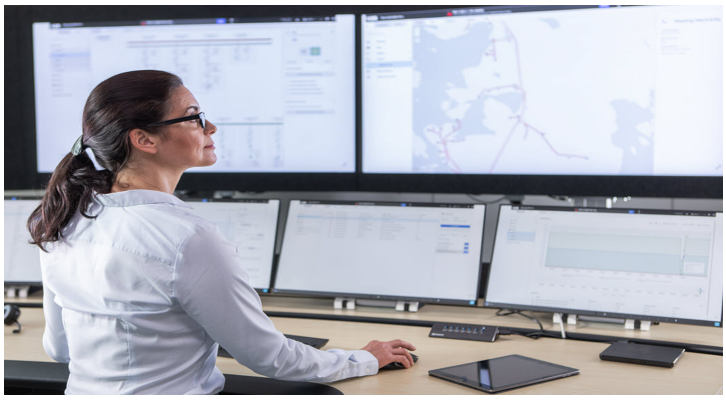
GRID AUTOMATION PRODUCTS

MicroSCADA X SYS600 10.2

Process Display Design



Document ID: 1MRK 511 478-UEN
Issued: March 2021
Revision: A
Product version: 10.2

Table of contents

Section 1 Copyrights

The information in this document is subject to change without notice and should not be construed as a commitment by Hitachi Power Grids. Hitachi Power Grids assumes no responsibility for any errors that may appear in this document.

In no event shall Hitachi Power Grids be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall Hitachi Power Grids be liable for incidental or consequential damages arising from the use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from Hitachi Power Grids, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Trademarks

ABB is a registered trademark of ABB Asea Brown Boveri Ltd. Manufactured by/for a Hitachi Power Grids company. All other brand or product names mentioned in this document may be trademarks or registered trademarks of their respective holders.

Guarantee

Please inquire about the terms of guarantee from your nearest Hitachi Power Grids representative.

Third Party Copyright Notices

List of Third Party Copyright notices are documented in "3rd party licenses.txt" and other locations mentioned in the file in SYS600 and DMS600 installation packages.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (https://www.openssl.org/). This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Section 2 Introduction

2.1 This manual

This manual gives an overview of Display Builder and process displays. The manual describes creating and editing procedures that relates to Display Builder and process displays.

For creating process pictures for Workplace X, please refer to SYS600 10.2 Workplace X Process Picture Design manual.

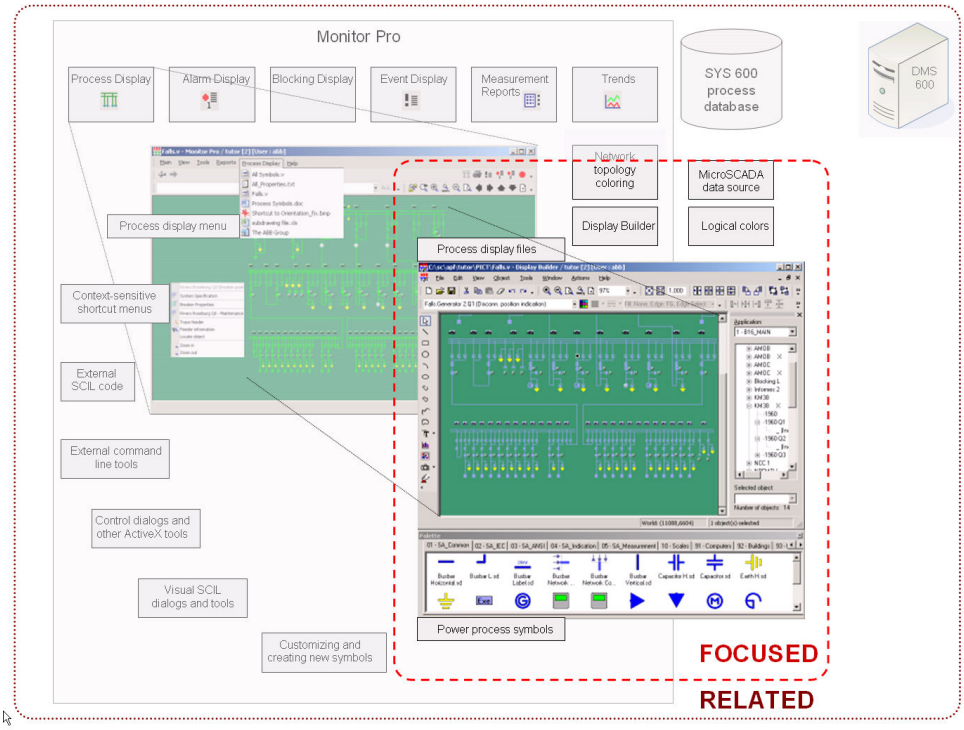


Figure 1: Focused and related topics of the manual

If the user is familiar with creating pictures for SYS600 Classic Monitor, the terminology introduced in Table 1 helps in understanding Figure 1.

Table 1: Terminology differences between Classic Monitor and Monitor Pro

Classic Monitor	Monitor Pro
Picture	Process display
Picture files (.PIC)	Process display files (.V)
Picture Editor	Display Builder
Picture function	Graphical symbol
LIB 5xx	Power Process Library, SA-LIB
Busbar coloring	Network topology coloring

Display Builder is a vector editor that can be used to create graphical user interface for monitoring and controlling live data processes. The existing Power Process Symbols can be used as building blocks of the process displays. The user can also modify the symbols or create new symbols.

2.2 Use of symbols

This publication includes the following icons that point out safety related conditions or other important information:

The caution icon indicates important information or warning related to the concept discussed in the text. It might indicate the presence of a hazard which could result in corruption of software or damage to equipment or property.

The information icon alerts the reader to relevant facts and conditions.

The tip icon indicates advice on, for example, how to design your project or how to use a certain function.

Although warning hazards are related to personal injury, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all warning and caution notices.

2.3 Related documents

Name of the manual	Document ID
SYS600 10.2 Application Design	1MRK 511 466-UEN
SYS600 10.2 Operation Manual for Workplace X	1MRK 511 500-UEN
DMS600 4.5 System Administration	1MRS257833
DMS600 4.5 System Overview	1MRS257835
SYS600 10.2 Workplace X Process Picture Design	1MRK 511 505-UEN

2.4 Document conventions

The following conventions are used for the presentation of material:

- The words in names of screen elements (for example, the title in the title bar of a window, the label for a field of a dialog box) are initially capitalized.
- Capital letters are used for the name of a keyboard key if it is labeled on the keyboard. For example, press the CTRL key. Enter and Shift keys are exceptions, for example, press Enter.
- Lowercase letters are used for the name of a keyboard key that is not labeled on the keyboard. For example, the space bar, comma key and so on.
- Press CTRL+C indicates that the CTRL key must be held down while pressing the C key (in this case, to copy a selected object).
- Press ESC E C indicates that each key is pressed and released in sequence.
- The names of push and toggle buttons are boldfaced. For example, click **OK**.
- The names of menus and menu items are boldfaced. For example, the **File** menu.
- The following convention is used for menu operations: **Menu Name** > **Menu Item** > **Cascaded Menu Item**. For example: select **Edit** > **Select** > **By Name**.
- The **Start** menu name always refers to the **Start** menu on the Windows Task Bar.
- The shortcut menu appears when, for example, a selection, a toolbar or a taskbar button is selected. It lists commands pertaining only to that screen region or selection.

2.5 Document revisions

Revision	Version number	Date	History
A	10.2	31.03.2021	New document for SYS600 10.2

Section 3 Overview of Process Displays

Typically, a process display shows the primary process, for example:

- a single line diagram of a substation
- a network overview of a power distribution network
- a district heating network

Graphical symbols are typically used to represent the primary process, but the process display can contain numerical data, graphs, buttons, controls and so on. Furthermore, the process display can represent internal system data and system self supervision information.

Navigate inside the process display by zooming or panning. The information shown at the different zoom levels can be configured by using the decluttering levels.

Process displays are saved in separate files. The number of process displays is not limited in the application.

3.1 Size and contents of process display

The process display maximum drawing area is 32767 x 32767 world coordinate points. The size of the drawing area can be defined when opening a process display. The process display can contain a small object or a whole network overview. When selecting the suitable process display, pay attention to the following issues:

1. The amount of the shown information at a time.
2. Used navigation scheme.
   Navigate between different displays by loading a new display or by moving to another area of the current display.
3. Number of objects in one display.
   The performance requirements limit the number of objects.

The network topology coloring is not dependent on having the whole network in one process display. For more information about the network topology coloring, see the SYS600 Application Design manual.

It is recommended to divide a process display if it contains more than 5000 objects.

3.2 Process display elements

Process displays can contain the following element types:

- Drawing objects:
  - lines
  - circles
  - arcs

    - rectangles
    - polylines
    - texts
- Subdrawings
  Subdrawings are complex objects, for example, breakers and measurements.
- Graphs:
    - line
    - bar
    - pie
    - area
- Input objects:
    - buttons
    - text fields
    - check boxes
    - menus
    - sliders
    - toggle keys
- Bitmaps

3.2.1 Drawing objects

Use drawing objects when drawing symbols and representations. There are both static and dynamic drawing objects regarding color, location, size, and so on. If the appearance is dynamic, it can change dynamically based on the values of the process database. Furthermore, the drawing objects can be used when creating subdrawings.

Drawing objects are:

- lines
- circles
- arcs
- rectangles
- polylines
- texts

3.2.2 Subdrawings

It is recommended to use subdrawings when the same kind of drawing is used several times in a display or in the different displays. A subdrawing is saved in a separate file and it can contain the same elements as a process display.

Typically, subdrawings are used for representing, for example, the following:

- breakers
- disconnectors
- measurements
- busbars
- transformers
- valves
- motors
- pipelines
- tanks

3.2.3 Graphs

Use graphs when presenting data in graphical formats.

Typical graphical formats are:

- line graphs
- bars
- pie diagrams
- meters
- knobs
- surfaces

3.2.4 Input objects

Use input objects to allow the operator to familiarize oneself with actions or types of data in means.

Different input object types are:

- buttons
- text fields
- check boxes
- menus
- sliders
- toggle keys

3.2.5 Bitmaps

Bitmaps can be used in the display, for example, to show maps or object photographs.

3.3 Process data and data variables

Process displays handle the internal process data by means of data variables. Data variables are grouped into logical entities that are called data sources. A data variable can be used, for example, to represent data in the process database. The data variable can also contain a static value or it can be used as a variable inside the process display to transfer data between display elements.

3.4 Designing process displays

Use the decluttering functionality to present a well-defined process display with a proper amount of information in the different zoom levels.

A process display can be designed in different ways depending on the user's needs. If the whole system is shown in a process display, pay attention to the performance aspects. Consider the following performance aspects, if the system contains more than three Monitor Pro workplaces or 1000 subdrawings.

Pay attention to the following characteristics concerning a Monitor Pro session (FrameWindow):

- A process display remains open until another process display is opened. The process display is not closed, although one of the following displays is opened:
    - Alarm Display
    - Blocking Display
    - Event Display
    - Measurement Reports Display
    - Trends Display

This feature enables a faster returning from a list to an active process display.

- Opening a large process display lasts longer than opening a small process display. A large process display contains more dynamic objects and subdrawings than a small process display.
- Moving to another area within an active process display is faster than opening a new display with the corresponding contents.
- A large process display requires more processing capacity and memory while opening.
- Tooltips in the process display elements increase the CPU load.

The system server processes the process display, although distributed workplaces are used. Therefore, pay attention to the number of workplaces during the designing of process displays.

Measure the following characteristics:

- Large displays with many workplaces require more processing capacity.
  The needed processing capacity to run a process display is dependent on the number of subdrawings and workplaces. Measure the CPU load at typical process data update rates. Based on these rates, design process displays or use the hardware that handles the CPU load.
- Opening a larger display lasts longer.
  A process display's opening time increases the processing capacity depending on the number of subdrawings. Measure the opening time on the used computer, compare the time with the requirements and design the process displays accordingly.

In case there is only one process display, it can be more effective not to divide it into smaller process displays. Even though opening a large process display lasts longer, it needs to be opened only once. After opening, navigate within the process display. Navigating within the process display is faster than opening a new process display.

Section 4 Overview of Display Builder

Display Builder is an editor where the user can create complete graphical interfaces, as well as define the graphical appearance of objects in the interface, describe the data they will show and specify their dynamic behavior.

The drawing process requires no programming. Drawings are easy to create and edit directly in the drawing area, and dialogs let the user control all aspects of the editing process, including assigning display characteristics and specifying how data is stored. The Display Builder provides default values for every dynamic element of a drawing, so that the user can quickly construct a complete dynamic drawing that can monitor and control substations as well as electrical power transmission and distribution system.

4.1 System requirements

It is recommended to have dual-display video card and two display devices, one for the Display Builder main window and another for the floating dialogs, tools and palette.

4.2 Starting Display Builder

Start the Display Builder by selecting **Tools/Engineering Tools/Display Builder**from the Monitor Pro main menu bar.

4.3 Menus

The main menu bar includes the **File**, **Edit**, **Window** and **Help** menus common to most Windows applications, plus additional menus for **View**, **Object**, **Tools** and **Actions**.

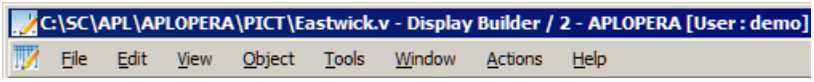


Figure 2: Main menu bar of Display Builder

The **File** menu provides standard options for working with files.

Close All: Closes all the open files.

Reload: Reopens the display. This is useful if the user has changed a referenced file and wants to see the changes in the current display.

Merge: Lets the user add the contents of a previously saved display file to the contents of the current display.

Merge special: Merges displays without modifying the existing Topology related information. This option can be used when a display, that is a part of a Topology Model, is merged to a display that is not part of a Topology Model. With this function the other display will still receive coloring from the Topology Model.

Import: Lets the user import other drawing files into Display Builder, such as Windows Meta Files (.wmf).

The **Edit** menu provides standard editing options, such as Undo, Redo and object selection. It also provides several special options. The user can cut, copy, and paste within a display and between displays.

Paste Special Pastes objects without modifying existing Topology related information. This option can be used when objects, which are a part of a Topology Model, are merged to a display that is not part of a Topology Model. With this function, the pasted objects will still receive coloring from the Topology Model.

Paste Offset: Lets the user set the offset for new copies of pasted material.

Clear Undo: Clears the undo/redo buffer.

Auto Repeat: Toggles the repeat creation mode that lets the user create multiple objects of the same type without having to click the object button again.

Data Variables: Opens the **Data Selection** dialog to let the user edit the display's data source variables.

Preferences: Opens the display's **Preferences** dialog to let the user set Display Builder's properties, such as settings for grid, color table, drag, as well as the search path.

The **View** menu allows the user to control the view of a display in the drawing area.

Pan To: Refreshes the display with the selected point in the center of the drawing area.

Refresh: Refreshes the screen to remove image debris.

Object Count: Shows information of the symbols used in the display.

Highlight Topology: Highlights the topology objects, see Section 6.3.1.

The **Object** menu allows the user to configure and manipulate graphical objects in the display.

Properties: Displays the **Object Properties** dialog for a selected object in the drawing area. The **Properties** dialog lets the user set the standard properties for a specific type of object selected. Different tabs are shown, depending on the type of the selected object.

Dynamics: Displays the **Dynamics** tab of the **Object Properties** dialog, which lets the user create dynamic features for the selected object.

Rules: Displays the **Rules** tab of the **Object Properties** dialog for a selected object.

Scale: Expands and shrinks the selected objects on a display. The scale factor for expand and shrink operations can be changed. Also, the default scale factor for new objects is defined here.

The **Tools** menu provides an option for customizing the Display Builder's menus and toolbars.

The **Window** menu lets the user control the arrangement of displays and view the palette and the errors in Display Builder.

The **Actions** menu provides Power Process specific functions, for more information see Section 6.

The **Help** menu gives the user access to the documentation and copyright information.

The shortcut menu is shown when the user right-clicks the mouse somewhere in the Display Builder. The **shortcut** menu provides a quick way to access the available commands.

4.4 Toolbars

Display Builder uses various toolbars to make the work more efficient. The toolbars allow the user to easily add graphical objects, change object attributes, format text and perform other tasks.

To show the toolbars that are not shown by default, do the following procedure:

1. Select **Tools/Customize**. The **Customize** dialog is shown (see Customizing in Section 4.8).
2. On the **Toolbars** tab, check the appropriate toolbars to be shown.
3. Click **Close**.

Toolbars:

Standard toolbar: Provides options similar to those of the Standard menu bar.

Figure 3: Standard toolbar

Format toolbar: The format toolbar lets the user control text characteristics such as size and style.

Figure 4: Format toolbar

Alignment toolbar: Displays buttons that let the user align and distribute groups of objects.

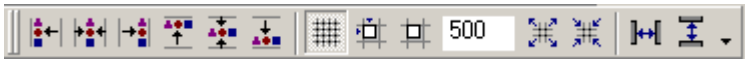


Figure 5: Alignment toolbar

Attributes toolbar: Provides buttons that control object characteristics such as line width and style, edge and fill, colors, and lets the user select objects by name.

Figure 6: Attributes toolbar

Objects toolbar: Displays buttons for each kind of graphical object that can be created.

Figure 7: Objects toolbar

Transform toolbar: Provides buttons for manipulating graphical objects in ways, such as enlarging or shrinking, rotating, mirroring, moving them to the front or back of the drawing, or moving them by small increments.

Figure 8: Transform toolbar

Zoom toolbar: Displays buttons that let the user zoom in and out of the drawing area, reset to the original drawing scale, and set a specific zoom scale.

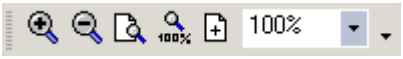


Figure 9: Zoom toolbar

4.5 Running menu or toolbar command from process display

With SAExecuteToolID it's possible to run any active menu or toolbar command in Monitor Pro and Pro+. The command is defined to custom attributes of an object in Display Builder with menu or toolbar with SAExecuteToolID as a key and the command id as a value.

The menu or toolbar command id can be retrieved in customization mode from the status bar of Monitor Pro or Pro+ by clicking the menu or toolbar command.

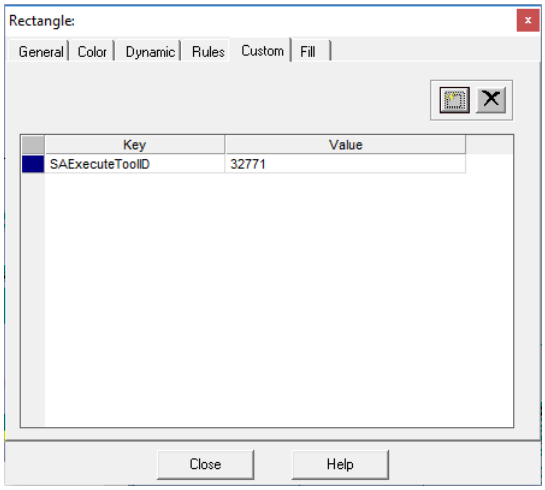


Figure 10: Example of running Logout command

In order to do user logout without showing the Login dialog, the -silentexit command line argument needs to be defined for the Monitor Pro or Pro+ instance". For more information about command line arguments please refer to Application Design manual.

4.6 Dialogs

Display Builder provides the following dialogs that assist the user in building dynamic graphics and displays:

- The **Data Selection** dialog lets the user specify and select data sources and data variables within the display. The dialog is shown whenever Display Builder requires the user to select a data variable, or whenever the user chooses to add or change a data variable attached to a Display Builder component, such as a dynamic object, dynamic feature or rule.
- The **Preferences** dialog lets the user set various configuration preferences, which specify parameters used in the drawing area. These parameters have a general effect on how the Display Builder operates. These settings are saved when the user closes the Display Builder and are reloaded when the Display Builder is started again.
- The **Object Count** dialog displays detailed information about the symbols used in the display in a tree structure.

- The **View Properties** dialog lets the user write a comment for the display, set background and off drawing colors, specify a default initial resolution for the display, add and edit display specific custom attributes, edit settings of the grid and define rules for the overall display.
- The **Object Properties** dialog lets the user customize the properties of a selected Display Builder object.
- The **Graph Properties** dialog lets the user customize the Display Builder graphs.
- The **Input Object Properties** dialog lets the user configure the Display Builder input objects.
- The **Customize** dialog lets the user determine which toolbars and commands are available in the Display Builder interface. The user can add or remove toolbars from Display Builder, add or remove specific buttons from the toolbars, and add or remove the user's own toolbars.
- The **Palette**dialog lets the user create instances of subdrawings.
- The **Error Messages** dialog shows details of situations where the Display Builder is not able to perform an operation or recover from previous errors. The dialog is shown when an error occurs.
- The **Topology Models** dialog is used to add process displays to topological models.
- The **Color Setting Tool** lets the user modify the common logical colors of displays and Power Process symbols.

See the index words at the end of this manual to find out where the specific dialog is used.

4.7 Display files

The Display Builder provides displays for various purposes, each indicated by a different file name extension. The Display Builder file dialogs filter for different extensions depending on the context of the current operation. The default extension for an ordinary display is .v. The default extension for a subdrawing display is .sd. The default extension for a display to be used as the template for an input object is .lay. The filter's drop-down menu lets the user select the type of display file to filter for, or the user can choose to filter for All Files. Display files can also be saved with no extension at all.

4.7.1 Importing display files

The user can import a .wmf file into Display Builder. Importing opens the file in the Display Builder and translates its contents into .xmlble objects.

4.7.2 Saving a partial display

The user can save complete or partial displays as files. These files can be used as display files or subdrawings.

To save a partial display:

1. Select the objects to be saved as a partial display.
2. Select **File/Save Selection**. The **Save As** dialog is shown with the file type filter set to Subdrawing Files.

4.7.2.1 Using the Save As dialog

The **Save As** dialog is shown with the file type filter set to a particular file extension, depending on the selected option. For example, if the user has selected **Save Selection**, the filter is set for the .sd subdrawing extension.

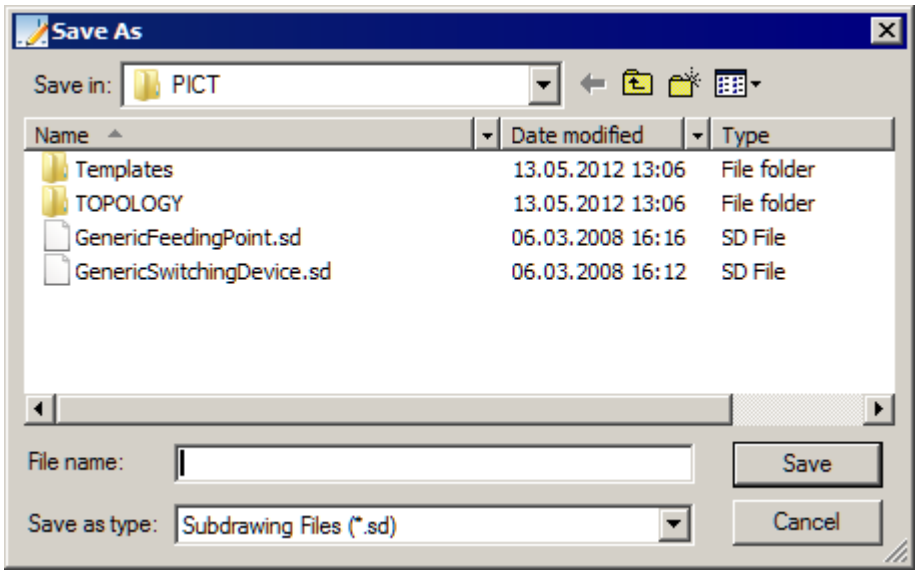


Figure 11: Save As dialog

To save the display:

1. Enter a file name to the **File name** field.
2. Select the filter from the **Save as type** drop-down list.
3. When the dialog shows the file name, click **Save**.

When a partial display is saved, the Display Builder saves only the data sources used by the selected objects.

4.8 Working with displays

There are many options to monitor and work with the displays. The following sections describe common functions to operate with the displays.

4.8.1 Customizing

The user can determine which toolbars and commands are available in the Display Builder interface. The user can add or remove toolbars from Display Builder, add or remove specific buttons from the toolbars, and add or remove the user's own toolbars.

The **Customize** dialog also lets the user show and hide the Object Browser task pane. For more information on using Object Browser, see Section 6.1. The Customize dialog is shown by selecting **Tools/Customize**.

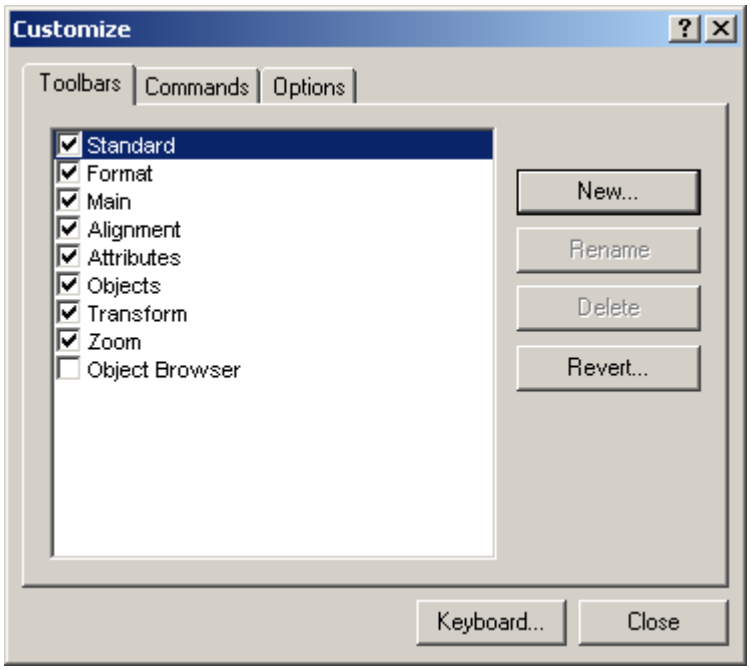


Figure 12: Customize dialog

4.8.2 Editing preferences

The Display Builder lets the user set various configuration preferences, which specify parameters used in the drawing area. These parameters have a general effect on how the Display Builder operates. These settings are saved when the Display Builder is closed and are reloaded when the Display Builder is started again.

The **Preferences** dialog opens by selecting **Edit/Preferences**.

4.8.3 View properties

The Display Builder lets the user specify the view properties of the displays. To specify the display properties, select **View/Properties**.

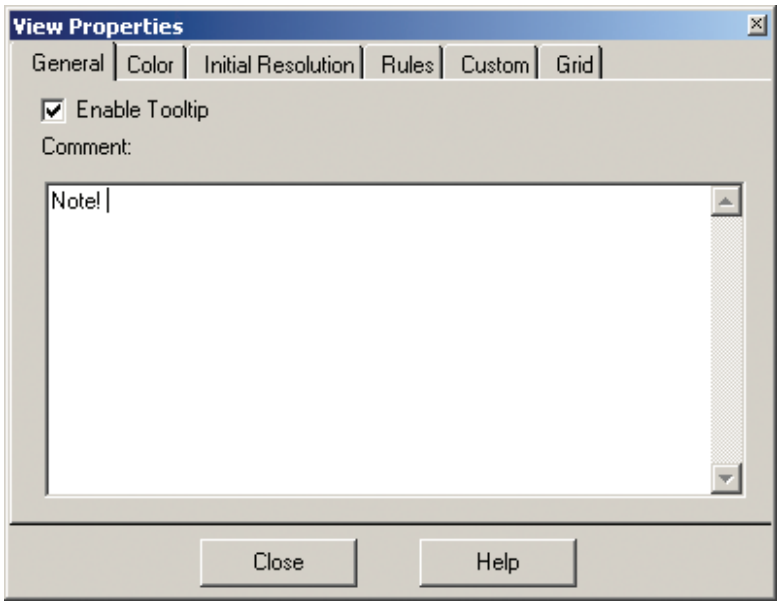


Figure 13: View Properties dialog

The **View Properties** dialog is organized in the following tabs:

- **General**: Type a comment to be saved with the display. The comment is intended to provide information about a display file if the display file contents is looked at outside the Display Builder.
- **Color**: Specify a background color for the display, and the color to be shown in the area outside the display when the display does not fill the drawing area.
- **Initial Resolution**: Specify information that controls the size of the display when it is opened in the Display Builder, or displayed in an application.
- **Rules**: Define changes that will occur in the display in response to specified user actions when specified conditions also apply.
- **Custom**: Lets the user use custom attributes in the display.
- **Grid**: Lets the user set the grid properties.

The recommended way to modify common background color of all displays is to use the Color Setting Tool.

4.8.4 Zooming

Display Builder lets the user zoom from different portions of the display at different scales. Whenever the display extends beyond the edge of the drawing area, scroll bars are shown to let the user scroll through the display. The zoom options are located in the Zoom toolbar, see Section 4.4.

The user can define the factor which the Zoom In and Zoom Out options change the scale of the display in the drawing area. Using this feature has no effect on the current appearance of the display in the drawing area. It changes how much bigger the display looks when Zoom In is used and how much smaller the display looks when Zoom Out is used.

To specify the zoom by factor:

1. Select **View/Zoom By**. The **Zoom By Factor** dialog is shown.
2. Use the slider to change the value or enter a new value in the text box.

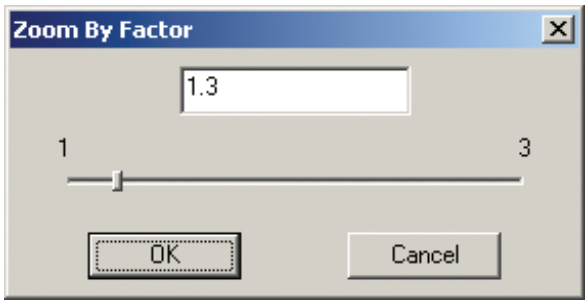


Figure 14: Zooming by factor

To refresh the display with the selected point in the center of the drawing area, click **Pan To** button in the zoom toolbar. Note that panning outside the drawing area is possible at the bottom and right side, but not at the top or left side.

4.8.5 Using grid

Display Builder provides a grid that makes it easy to align objects. The grid is shown as an array of points in the drawing area. When the grid is on, points on the grid can be used to position an object freehand or the Snap to Grid option can be used. For more information on the Snap to Grid option, see Section 4.8.5.1.

To toggle the grid on and off, select **View/Grid/Show Grid**.

A checkmark is shown when the grid is on. The user can change the distance between grid points. Increasing the distance between grid points doubles the distance between points. Decreasing cuts the distance between points in half. The distance between grid points remains constant with respect to objects in the display when the display is zoomed and panned.

To make the grid points closer or further apart:

1. Select **View/Grid**.
2. Select **Increase Spacing** or **Decrease Spacing**.

4.8.5.1 Using Snap to Grid option

The Snap to Grid option lets the user place control points only on the grid points, whether the grid is shown or not.

To turn the Snap to Grid option on and off, select **View/Grid/Snap to Grid**. A checkmark is shown when the Snap to Grid option is on.

If an object was created without the Snap to Grid option, and it should be aligned it to the grid, select the object, select the control point to be adjusted, and drag it toward a grid point. The control point snaps to the closest grid point.

4.8.6 Cursor coordinates

Display Builder supports two different coordinates. If the user selects **View/Coordinates/World**, the origin (0, 0) is located in the middle of the display. The lower left hand corner is at (-16 383, -16 383) and the upper right hand corner is at (16 383, 16 383). If **View/Coordinates/Screen** is selected, the origin is located in the lower left corner of the drawing area. The current cursor coordinates can be shown in the status bar.

The selected coordinate system also affects the representation of control points in the **Object Properties** dialog.

It is almost always recommended to use world coordinates because it does not depend on current zoom level, panning position or screen resolution.

4.9 Creating objects

Display Builder provides several object tools to create ordinary objects. These objects are shown in the Object toolbar. Creating these objects is easy and alike. The user can modify an object's characteristics from the Attribute toolbar. The following section provides a brief presentation on creating objects in the Display Builder.

4.9.1 Object count

The user can show information about used symbols in a tree structure by selecting **View/ Object Count**. The levels of the tree can be chosen using the check boxes in the Levels group box. The user can save the information as a text file and select symbols in a process display based on symbol Super type, Object type or Subdrawing file name.

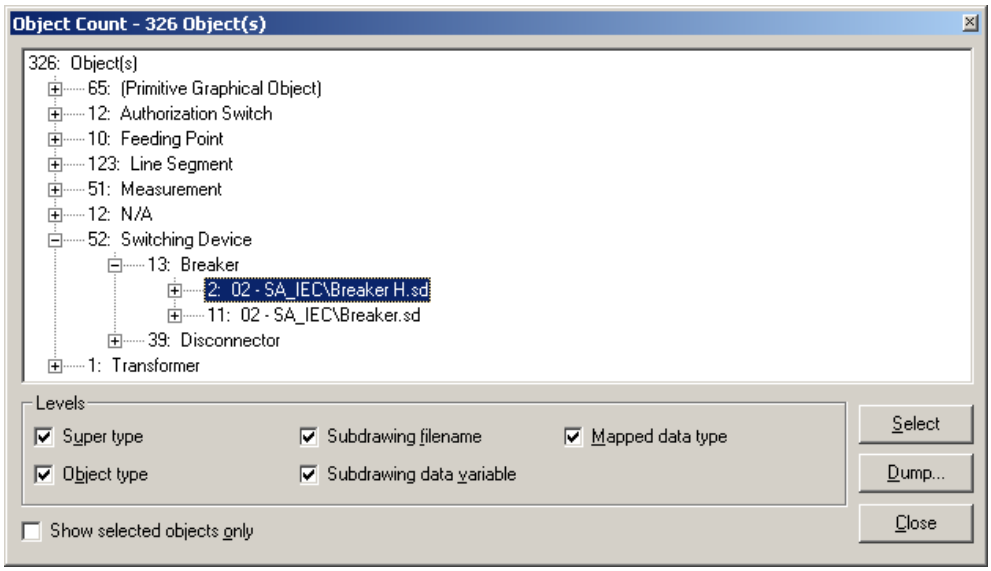


Figure 15: Object Count dialog

The selected objects of a process display are shown, if **Show selected objects only** is checked.

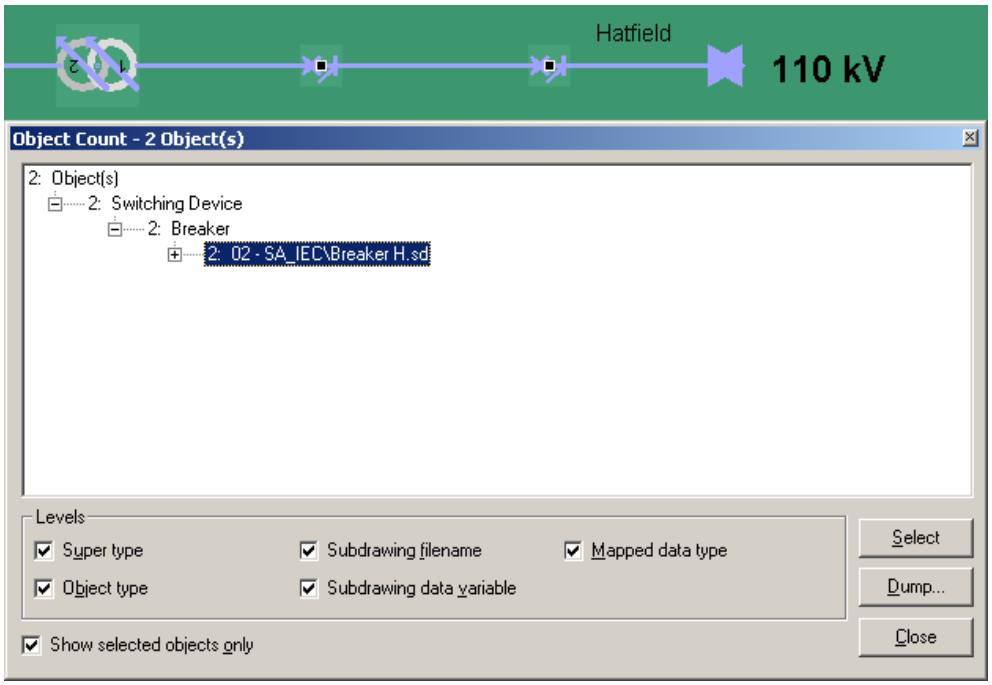


Figure 16: Object Count dialog with selected object

4.9.2 Object properties

The user can customize many properties of a selected Display Builder object. To show the Object Properties select **Object/Properties** when the certain object is selected.

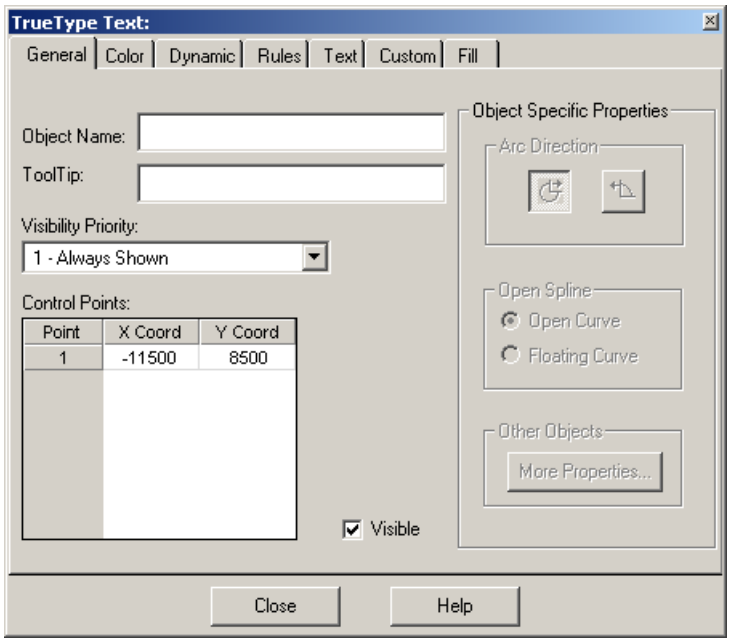


Figure 17: Specifying object properties

4.9.3 Auto repeat mode

Auto repeat mode lets the user sequentially create multiple objects of the same type without having to select the object type each time. Subdrawings or bitmaps cannot be created using the auto repeat mode.

To enter the repeat mode:

1. Select **Edit/Auto Repeat**.
2. Select the type of object to be created.

To exit the repeat mode and return the cursor function to regular object selection, click the selection cursor in the object toolbar or select **Edit/Auto Repeat** again.

4.9.4 Lines

A line object is a straight line between two points.

To create a line:

1. Click the Line button in the Object toolbar.
2. Click and drag in the drawing area to select the start point and end point.

4.9.5 Shapes

To create other shapes, such as rectangles, polygons, splines, circles, ellipses, arcs and so on, the creating procedure is similar as creating lines.

For example, to create a rectangle:

1. Click the Rectangle button in the object toolbar.
2. Click and drag in the drawing area to select points for the two opposite corners of the rectangle.

Note that the sides of a rectangle are always parallel to the sides of the drawing area.

4.9.6 Text objects

A true type text object is a string of characters that are manipulated as a unit. The text string can include carriage returns. The true type text objects can be set to scale with the drawing. The user can control character size, font style, shape, and angle. True type text objects provide a wide range of available fonts.

To create a true type text object:

1. Click the **True Type text** button in the **Object** toolbar.
2. Select an anchor point in the drawing area.
3. Type the text as it should be shown, for carriage return press CTRL+Enter.
4. After entering the text, press Enter.

When reshape handles are on, the control point is visible. The location of the control point indicates the Anchor Position setting of the text.

Vector text characters are drawn as a series of lines. A vector text object is a string of characters that are manipulated as a unit. The text string can include carriage returns. Vector text objects scale with the drawing. The user can control character size, font style, shape and angle. The vector text button is not shown by default when the Display Builder is started.

To show the vector text button:

1. Select **Tools/Customize**. The **Customize** dialog is shown.
2. On the **Command** tab, select **Object Creation**.
3. Add the **Vector text** button to the **Objects** toolbar.
4. Click **Close**.

See Section 4.3 for customizing toolbars from **Tools** menu.

To create a vector text object:

1. Click the **Vector text** button in the **Object** toolbar.
2. Select an anchor point in the drawing area.
3. Type the text as it should be shown, for carriage return press CTRL+Enter.
4. After entering the text, press Enter.

A hardware text object is a string of characters that are manipulated as a unit. The text string can include carriage returns. Hardware text objects are hardware dependent, which means they do not zoom with the rest of the drawing. Hardware text objects also do not provide a wide range of font selection. The hardware text button is not shown by default when the Display Builder is started. It can be shown the same way as the vector text button.

To create a hardware text object:

1. Click the **hardware text** button in the **Object** toolbar.
2. Select an anchor point in the drawing area.
3. Type the text as it should be shown, for carriage return press CTRL+Enter.
4. After entering the text, press Enter.

The user can use the text objects to show static strings or dynamic values. This means that the user can dynamically change the text string shown by the text objects. For more information, see Section 7.3.

4.9.7 Graphs

A graph object is a graph or chart that shows data.

To create a graph object:

1. Click the **Graph** button in the **Object** toolbar.
2. Click and drag in the drawing area to select points to define the rectangular area of the graph. The graph is drawn using the default graph format Bar Chart.

To change the graph format:

1. Double-click the graph to view the **Graph Properties** dialog, see Figure 18.
2. Select the **Types** tab.
3. Select the icon that represents the wanted graph type.

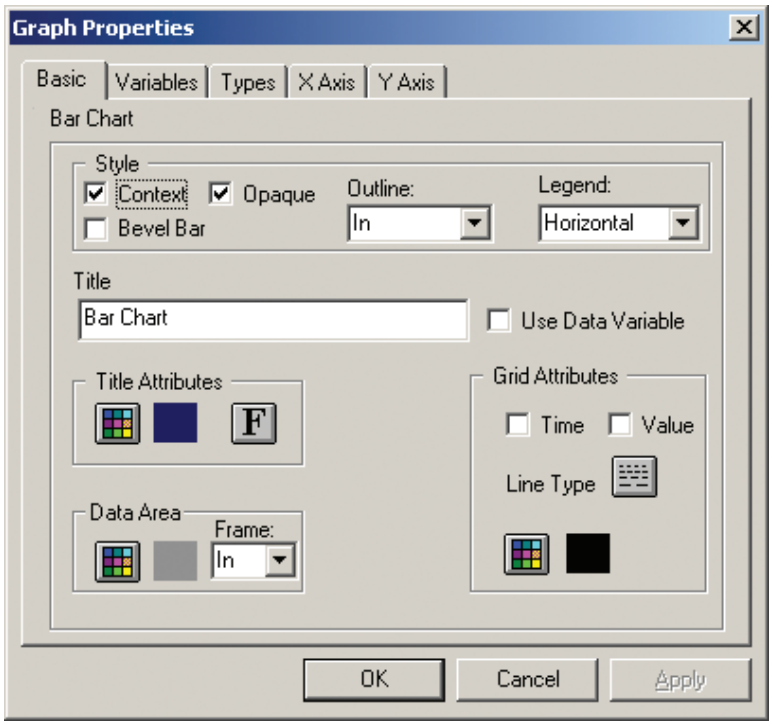


Figure 18: Graph Properties dialog

Graph dynamics can bleed through objects placed in front of them, so graphs should not be obscured by other graphical objects.

To show the **Graph Properties** dialog, the user can also right-click the graph and select **Graph Properties** from the **shortcut** menu.

The **Graph Properties** dialog is organized on the following tabs:

- **Basic**: Lets the user set most of the context components of a graph.
- **Variables**: Lets the user add, replace, or delete the data variables attached to a selected graph. The user can also control characteristics of a selected graph variable.
- **Types**: Lets the user specify a graph type for a graph.
- **X Axis**: Lets the user set context components for the horizontal axis of a graph.
- **Y Axis**: Lets the user set context components for the vertical axis of a graph.

4.9.7.1 Using contour graph type in Monitor Pro/Display Builder

Contour graph type can be used for representing surface with altitude levels. The contour lines are drawn around areas where altitude values differ from the surrounding altitude values. The graph uses matrix data type. Each value in the matrix represents an altitude value in the corresponding point. The matrix dimensions also define the resolution of the graph.

To create a contour graph:

1. Create a matrix data variable.
2. Select a matrix shape for the data variable.
3. Define row and column count.
4. Define a process object that triggers updating of the data variable. The data variable update triggers the graph update.
5. Select **Generic signal** as the signal type.
6. Write the SCIL script that constructs the data.

See an example of a data variable in Figure 19. For more information on matrix data type and writing SCIL script, see other SYS600 documentation.

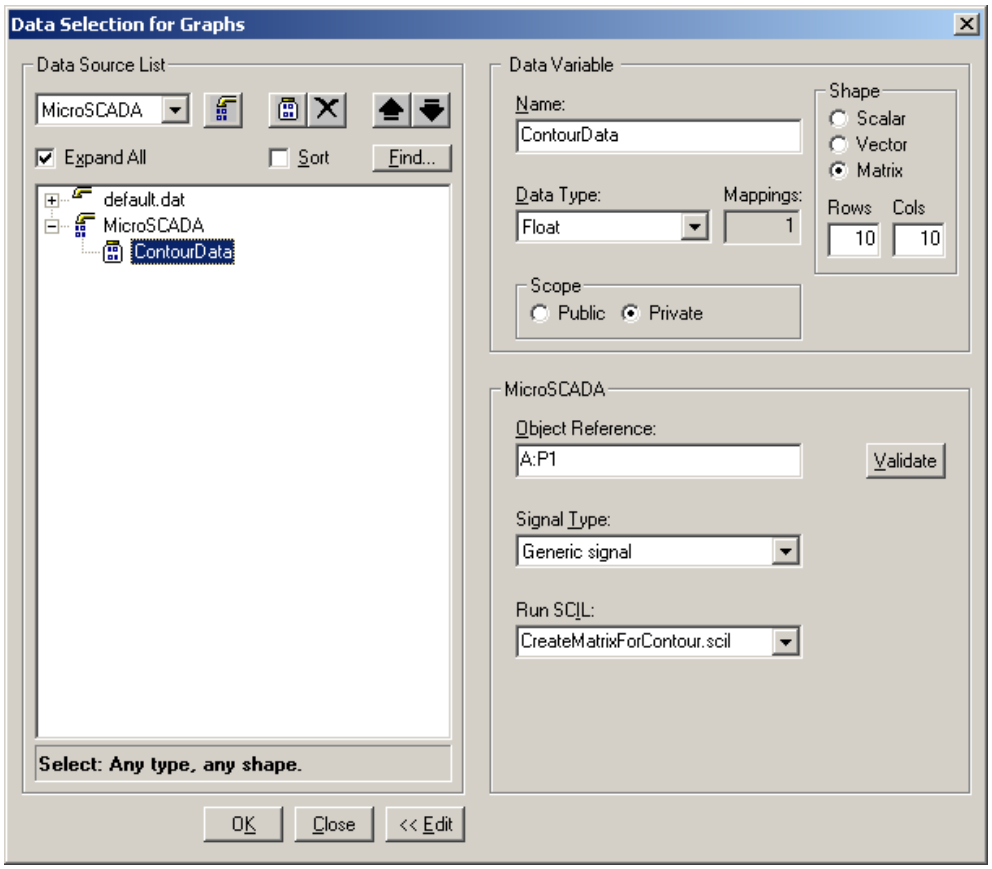


Figure 19: Matrix data variable

In the **Graph Properties** dialog, select the defined data variable. Select **Matrix** shape for data variable and dimensions that match the data variable dimensions. See Figure 20 for an example configuration.

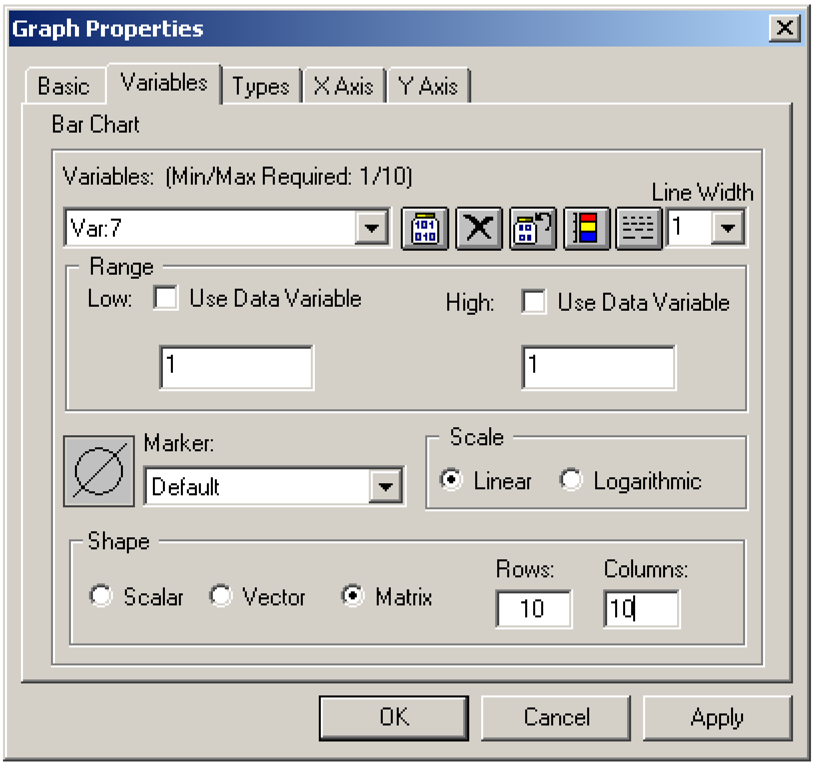


Figure 20: Graph properties

In following example, a matrix with dimensions 10,10 is used for displaying the data. The data consists of a vector with length 10 (later referred to as inner vector), each element containing vector of 10 REAL values. Values are within range 0.0-1.0. Inner vectors represent the values of vertical direction in each horizontal axis point. Indexing starts from origin that is in the lower left corner of the graph.

Example SCIL script (CreateMatrixForContour.scil):

```
; Script file template location
; <drive>:\sc\prog\graphicsEngine\support\TEMPLATEforDataChange.scil
; Template copied to <drive>:\sc\apl\<application>\PICT will be used
; instead of the default template.

; This script is executed when value of OPC item changes
(DIRECTION=="READ")
; or when data variable value changes (DIRECTION=="WRITE")
; Variable %L_INPUT contains value, name and data type of OPC item, and
direction.
; Format of the variable %L_INPUT is:
; LIST(ITEM=<Item ID of OPC item>,VALUE=<value of opc item>, -
; DATA_TYPE=<data type of OPC item>, DIRECTION=<"READ"|"WRITE">)

; <Add your code here to modify %L_INPUT.VALUE>

#RETURN -

((0.1,0.1,0.2,0.2,0.2,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
```

```
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1))
```

The example SCIL script draws a contour with three values on the vertical axis that are higher than others (indices 3-5 of the first inner vector are 0.2). In the contour graph, the contour is drawn around the values, see Figure 21.

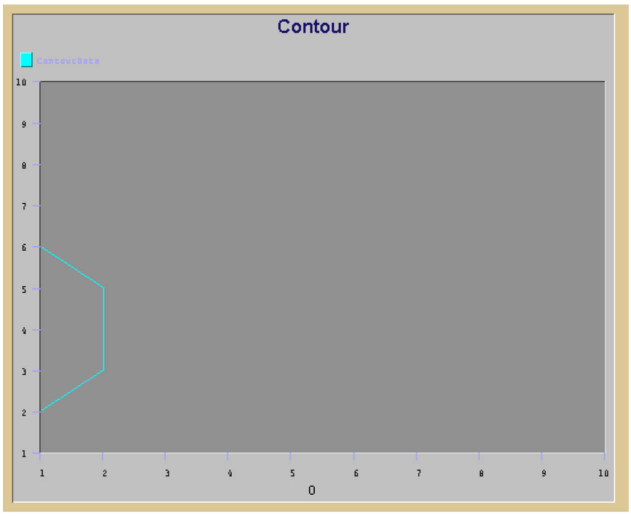


Figure 21: Example contour graph

Example 2: Points (2,5) and (3,6) are higher than others:

```
#RETURN -
```

```
((0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.2,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.2,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1))
```

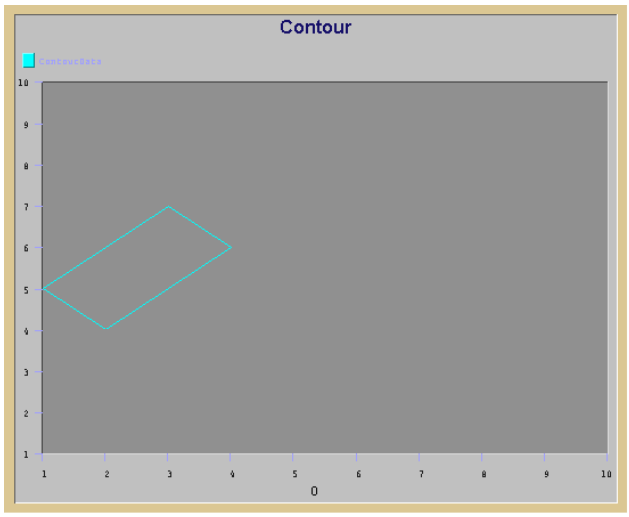


Figure 22: Example contour graph 2

In the example 3 multiple levels are used:

```
#RETURN -

((0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.2,0.2,0.2,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.2,0.2,0.3,0.3,0.2,0.2,0.1,0.1),-
(0.1,0.2,0.2,0.3,0.3,0.3,0.2,0.2,0.1,0.1),-
(0.1,0.1,0.2,0.2,0.2,0.2,0.2,0.2,0.1,0.1),-
(0.1,0.1,0.1,0.2,0.2,0.2,0.2,0.2,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1),-
(0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1))
```

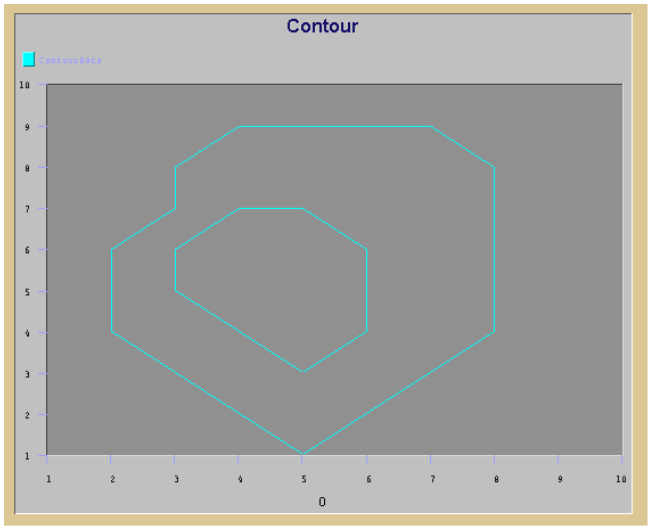


Figure 23: Example contour graph with multiple levels

4.9.8 Subdrawings

A drawing object shows another display as part of the current display. This is called a subdrawing, because it is a drawing within a drawing. A subdrawing object treats the display it points to as a unit, which means that the user cannot edit the subdrawing's individual components. The user can only edit the subdrawing if the display, which the subdrawing references, is loaded and the display is edited. The subdrawing file must already exist before the subdrawing can be created in the current display.

To create a subdrawing in the display:

1. Click the **Subdrawing** button in the Object toolbar. The **Open File** dialog opens with the file type filter set to Subdrawing Files. Click the file type drop-down menu to change the filter to Display Files or All Files as needed.
2. Select a file and click **Open**.
3. Select an anchor point in the drawing area.

The subdrawing is shown, centered around the anchor point. If the graphics in the subdrawing should be centered around the anchor point, the original file must be created with its graphical components centered around the drawing's center.

4.9.9 Bitmaps

A bitmap object shows a raster image from a file that was created and saved in a compatible pixel format. Bitmap objects have two control points and change scale with the drawing when the drawing size is changed. For example, if the background of a display is a bitmap containing a map, the map always fills the same percentage of the drawing area and more or less detail is visible as the user zooms in or out. Compatible pixel formats include the GIF format of Compuserve Corporation, the PPM format of Jef Poskanzer, and the BMP format of Microsoft Windows. Bitmaps must be 1, 4, or 8-bit DIB. To use the user's own pixel files, they must be converted to one of the compatible formats.

A bitmap object can be created from a file in the display by clicking the **Bitmap** button in the Object toolbar. To create a bitmap object from the screen, click the arrow next to the **Bitmap** button and select the **From Screen** option. When using the **From File** option, the wanted file must already exist.

To create a bitmap object from a file:

1. Select the **From File** option. The **Open File** dialog is shown with the file type filter set to Windows Bitmap Files.
2. Select an image file.
3. When the desired file is selected, click **Open**.

If Scale with Zoom is on, click and drag to select two points to define the bounding rectangle of the bitmap object. If Scale with Zoom is off, click once to specify the anchor point of the bitmap.

If Scale with Zoom is off, the bitmap has a single anchor point. The current anchor position setting has no effect on the placement of the bitmap object when it is created, but this setting can be changed after creation. If Scale with Zoom is on, the bitmap does not have an anchor point property.

To create a bitmap object from screen:

1. Select the **From Screen** option.
2. Drag the mouse to select two points on the screen to define a rectangle containing the raster image to be captured.
3. Move the captured object to a defined location in the display.

If Scale with Zoom is on, the bitmap is created with two control points. If Scale with Zoom is off, the bitmap is created with a single control point.

If Scale with Zoom is off, the bitmap has a single anchor point. The current anchor position setting has no effect on the placement of the bitmap object when it is created, but this setting can be changed after creation. If Scale with Zoom is on, the bitmap does not have an anchor point property.

The bitmap object is created at the specified location and may be indistinguishable as a separate object until its placement, scale, or color is changed with respect to its surroundings. A bitmap object created using the From screen option has no file name.

4.9.10 Input objects

An input object is used to collect input, such as a value or menu choice, from the user. Input objects are used to build interfaces where the user interactively directs or influences what information the application presents. An input object has two major elements:

- The interaction type
- The interaction template

The interaction type determines the form of interaction, such as a slider or menu. The template controls the physical layout of input objects, such as the number of items and the shape of pickable areas. The template is a graphical display created in Display Builder. The input object button is not shown by default when the Display Builder is started.

To show the input object button:

1. Select **Tools/Customize**. The **Customize** dialog opens.
2. On the **Command** tab, select **Object Creation**.
3. Add the **Input object** button to the **Objects** toolbar.
4. Click **Close**.

To create an input object:

1. Click the **Input object** button in the **Object** toolbar.
2. Click and drag to select two points that define the bounding box for the input object.

To specify the type of interaction or the file name of the template, double-click the input object to show the **Input Object Properties** dialog. Different input object menus are shown depending on the input object type.

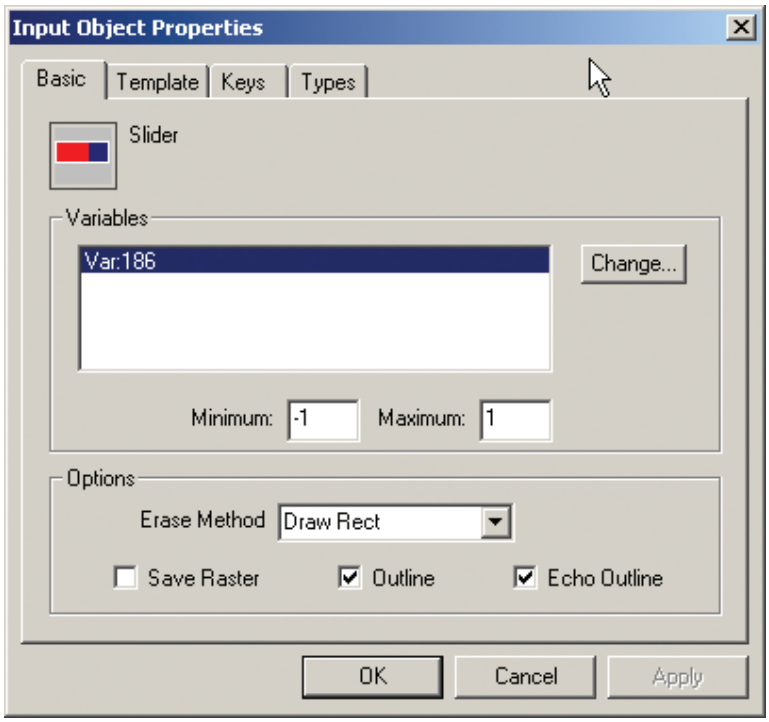


Figure 24: Input Object Properties dialog

4.10 Manipulating objects

Objects can be manipulated in many different ways. These commands can be found from the **Object** menu.

The functions that can be performed are:

- moving the objects
- copying the objects
- cutting the objects
- pasting the objects
- editing multiple objects
- rotating the objects
- reordering the objects
- scaling the objects
- flipping the objects
- aligning the objects
- distributing the objects
- control colors

Some of these features are described in more detailed in the following sections.

4.11 Tool Launcher

Actions to mouse clicking events can be configured with the Tool Launcher or by using rules.

With the Tool Launcher, the following actions can be performed:

- running SCIL code from a text file
- running a command line
- opening ActiveX tools
- opening other displays
- opening Visual SCIL tools or dialogs

Rules provide actions that are internal to the current process display, for example, changing a value of a memory data variable. For more information on rules, see Section 7.4.

Executing scripts from rules is partially supported in SYS600.

4.11.1 Configuring actions by specifying Tool Launcher settings

The Tool Launcher settings can be specified for a graphical object or for all instances of a graphical symbol. If both are specified, graphical object specific settings override the settings of the graphical symbol.

To configure actions with the Tool Launcher:

1. Open the Tool Launcher.
   To configure an action for a single graphical object:
   1.1. Right-click the object.
   1.2. Select **Tool Launcher**.

   To configure an action for all instances of a graphical symbol:
   1.1. Right-click any instance of the symbol.
   1.2. Select **Open Subdrawing**.
   1.3. Right-click the drawing area when no graphical objects are selected.
   1.4. Select **Tool Launcher...**.

   The **Tool Launcher** dialog opens.

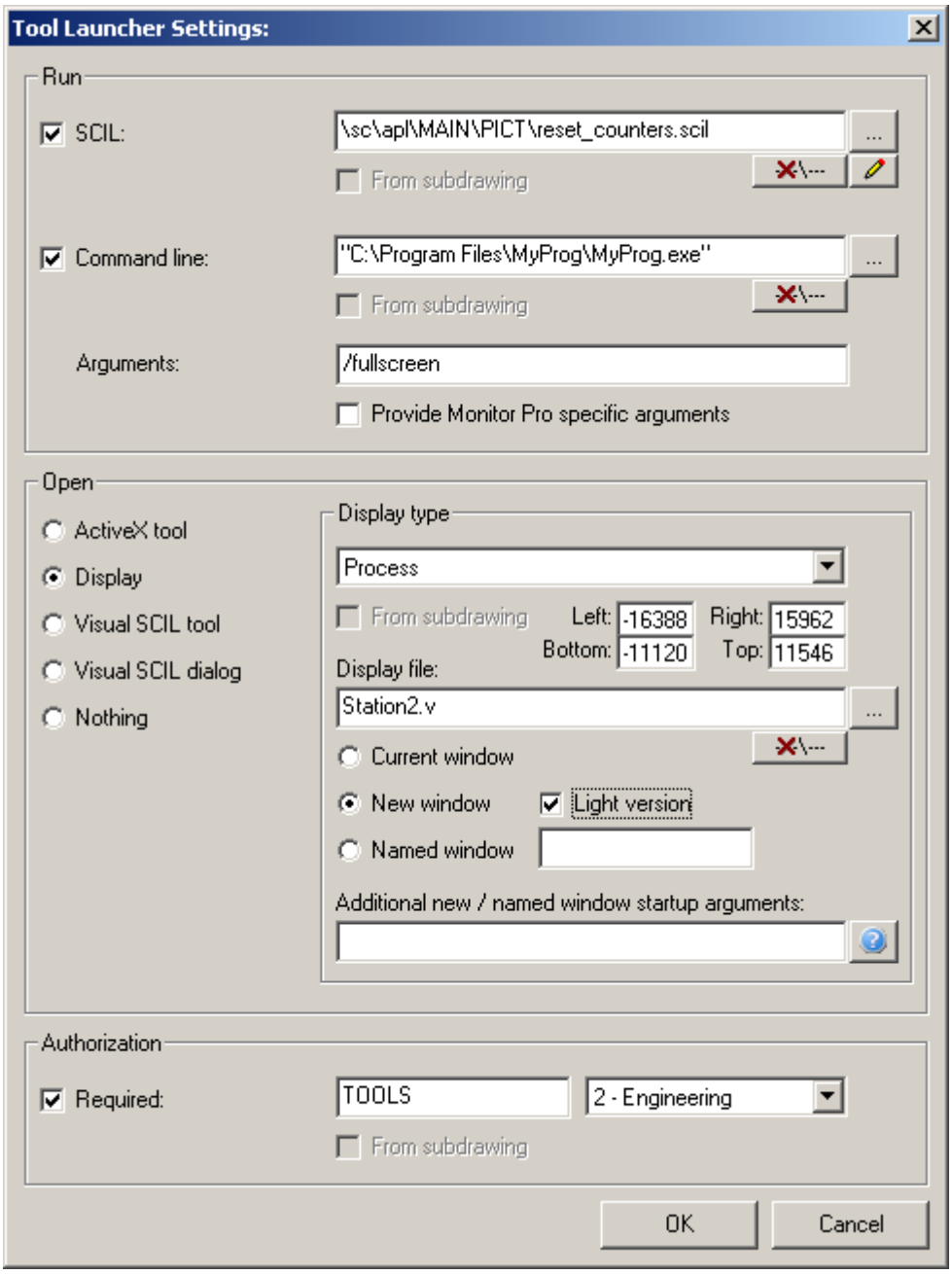


Figure 25: Tool Launcher settings

2. Specify the settings in the **Tool Launcher** dialog.
   See Table 2 for details.

Table 2: Tool Launcher dialog setting options

Field	Option	Related selections
Run	**SCIL**	Specify the SCIL file whose contents is executed.
		Check the **From subdrawing** check box, if the custom attributes should be read from the subdrawing view.
	Command line	Specify the command line to be executed.
		Select the **From subdrawing** check box, if the custom attributes should be read from the subdrawing view.
	Arguments	Specify arguments for the command.
		Select the **Provide monitor Pro specific arguments** check box, if the arguments explained in SYS600 Application Design should be passed to the executed SCIL.
Open	**ActiveX tool**	ActiveX ProgID (required). Specify the ActiveX control to be opened. Identify the tools with a programmatic ID. Monitor Pro can only open ActiveX tools that implement FrameConnectivity.CToolLauncher COM interface defined in \sc\prog\sa_lib \FrameConnectivity.dll.
	Display	Specify the **Display type**(required). Available options are Alarm (Template1), Alarm (Template2), Blocking, Event, Measurement (Graphical view), Measurement (Tabular view), Process, Trends (Graphical view) and Trends (Tabular view). **Process** is the default display type.
		Specify the preconfiguration to be loaded for Alarm (Template1), Alarm (Template2), Blocking, Event, Measurement (Graphical view), Measurement (Tabular view), Process, Trends (Graphical view) or Trends (Tabular view) in the **Display type** field. The Process Display File for the Process Display can also be specified. The display file name must always be specified if a new window is opened.
		Select the **From subdrawing** check box the custom attributes should be read from the subdrawing view. Specify the zoom area to be zoomed at for the process display, if zooming is used in Monitor Pro. By default, the coordinates of the currently visible drawing area are filled in. If no process display file is specified, the zoom is applied to the current process display.
		Select the **New window** check box if the Process Display should be opened in a new window. Additional startup arguments can be given for the New window.
		If **Light Version** is selected, the window does not contain buttons or menus, and therefore only the Process Display is displayed in a window.
		If the **Named window** check box is selected, the display will be opened to a corresponding Monitor Pro application window. The window name can be defined in the text box. The new window opens with the command line argument -windowname: [stringid]. The stringid must match the one defined in the text box. Additional startup arguments can be given for the Named window.
		Command lines can be written n the **Additional new / named window startup arguments**, or commands can be selected by clicking the **Help** button.
	Visual SCIL tool	Specify the **Tool name** of the VSCIL tool to be opened. Identify the tool with a tool name registered in Tools.ini, for example, CAL for the calendar.
	Visual SCIL dialog	Specify the VSO file to be opened with the Visual SCIL object name. Identify the dialogs with a VSO file and Visual SCIL object name, for example, \sc\sa_lib \base\bbone\use\CAL_1.VSO and DLG_CALENDAR. The type of the object must be VS_DIALOG or VS_MAIN_DIALOG. Add the argument(s) in a SCIL compatible format to the Custom argument field.
	Nothing	Select **Nothing** if, for example, the values should be used in the **Run** field only.
Authorization	**Required**	To specify the authorization level, select the **Required** check box, and specify the authorization level.
		Select the **From subdrawing** check box, if the custom attributes should be read from the subdrawing view.

Place the path names within quotation marks if they contain white-space, for example, "C:\Program Files\MyProg\\My Prog.exe". All paths can be absolute or without the drive letter. The SCIL files and process display files can be relative to the application's PICT directory.

In the **Command line** action in the **Run** field, the application searches the files from directories specified by the Windows PATH environment variable.

The **From subdrawing** function allows you to choose between the graphical object specific tool launcher settings and the graphical symbol's default settings.

With the  (Extract) button the user can remove the drive letter or directories from absolute paths.

The  (Edit) button opens the specified file for editing.

4.11.2 Example of executing a Visual Basic script

The following is an example of a procedure for executing a Visual Basic (VB) script printing all the arguments received from Monitor Pro, and for assigning the script for a graphical object in the Display Builder.

1. Save the content of the VB script to the file ..\sc\apl\<aplname>\PICT\Arguments.vbs
   The content of the script is as follows:

```
msg = "Command line arguments:" & vbCrLf
count = 1


For Each arg in WScript.Arguments 'Loop through all arguments
msg = msg & count & ": " & arg & vbCrLf
count = count + 1
Next


MsgBox msg, vbOKOnly, "Command line arguments test"
```

2. Select a graphical object in the process display.
3. Open **Tool Launcher**, and specify the settings in the **Tool Launcher Settings** dialog.
   3.1. In the **Run** field, select the **Command line** checkbox.
   3.2. Browse for the Command line file, and select ..\sc\apl\<aplname>\PICT \Arguments.vbs.
   3.3. In the **Arguments** text box, enter myarg1 myarg2.
   3.4. Select the **Provide monitor Pro specific arguments** check box.
   3.5. In the field, deselect **Required**.
4. Click **OK**.
5. Save the process display.
6. Open the process display in Monitor pro, and click the graphical object.
   The message box showing all listed arguments is displayed.
   VB Script Language Reference can be found from Microsoft MSDN: http:// msdn.microsoft.com/en-us/library/d1wf56tt.aspx.

4.12 Using palette

The **Palette** dialog is shown automatically whenever the Display Builder is started. If the palette is closed, it can be shown again by selecting **Window/Show Palette**.

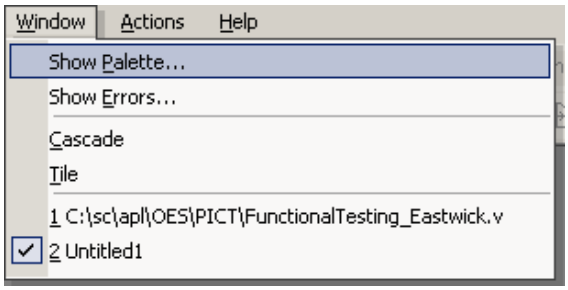


Figure 26: Showing Palette dialog

The **Palette** dialog contains the Power Process symbols and miscellaneous subdrawings that can be used to create process displays.

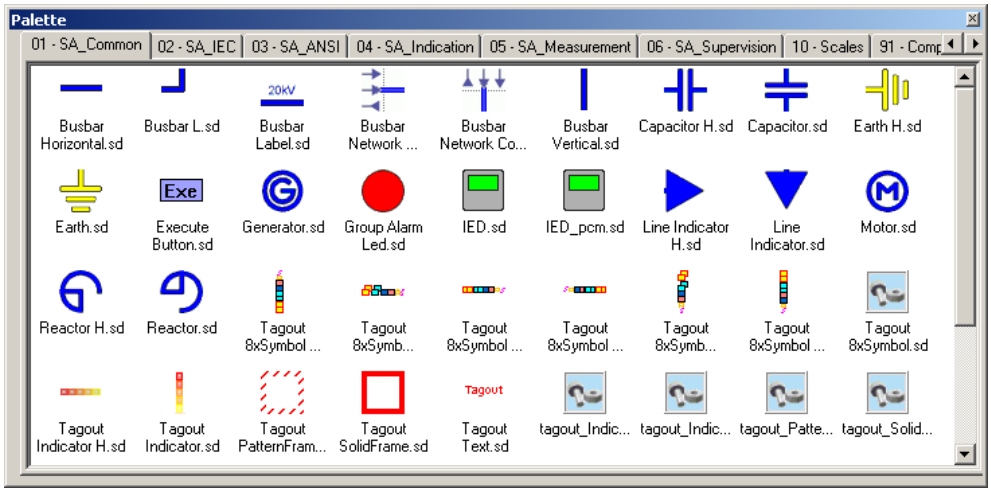


Figure 27: Palette dialog

4.12.1 Adding subdrawings to palette

The palette lets the user add complex graphics to the display in the form of single objects. Subdrawings and bitmaps can be saved as symbols in the palette, and then the palette can be used to create these images as objects in the display. The user can add subdrawing items to the palette by creating objects in Display Builder and adding them to the palette or, by copying subdrawing files into subdirectories of the palette directory. Add bitmap items to the palette by copying bitmap files into subdirectories of the palette directory. For more information on palette directory, see Section 4.12.5.

4.12.1.1 Adding subdrawing item to palette

This procedure takes the currently selected objects and creates a subdrawing file that contains a copy of those objects in the palette directory structure. You can select any kind of objects, including subdrawing objects.

If you select a single subdrawing object and no other objects, the subdrawing object will be contained in the new subdrawing file. The new subdrawing file is not a duplicate of the display file used to make the selected subdrawing object. This means that if you have applied dynamics or rules to a subdrawing object, then use this option to add the subdrawing object to the palette. The palette object contains the dynamics and rules as well. If you want to add to the palette a subdrawing file that is a duplicate of the display file used to create the selected subdrawing object, see Section 4.12.1.2.

To add a subdrawing item to the palette:

1. Go to the appropriate tab in the Palette dialog. The new item is added to the current palette tab. If a tab has not yet been created for this item, see Section 4.12.3. The new item is given a default name of GroupN.sd, where N is the next number not already used. Therefore, the first item added by using this method is named Group0.sd, the second is Group1.sd and so on. If the default names are changed, their numbers are used again.
2. Select the graphical objects for the subdrawing. A subdrawing can contain any objects from the current display. This can include dynamic objects, subdrawings, or bitmaps.
3. Select **File/Add to Palette**. A bitmap of the new symbol is added in the Palette dialog, together with the new default name.

When this method is used to add a symbol to the palette, a file is created in the subdirectory for the currently selected tab. A file for the bitmap is also added, together with the new subdrawing file. Changing the name of a palette symbol also changes the name of the corresponding bitmap file.

4.12.1.2 Adding subdrawing file to palette

This procedure takes a selected subdrawing object and creates a subdrawing file in the palette directory structure. The resulting subdrawing file is a duplicate of the display file used to make the selected subdrawing object. This means that if the user has applied dynamics or rules to a subdrawing object, then this option can be used to add the subdrawing to the palette. The palette object does not contain those dynamics or rules.

The new file in the palette directory structure will have the same name as the display file used to create the subdrawing object. This will be true even if the original display file no longer exists. If this procedure is used repeatedly with the same subdrawing, Display Builder appends a number to subsequent versions of the subdrawing file.

Using this method to create a subdrawing file in the palette does not change the reference of a referenced subdrawing. If the subdrawing object in the user's display is referenced and the user wants it to use the new file in the palette directory structure, the reference path of the subdrawing object must be edited.

To add a subdrawing object to the palette as a duplicate of the display file used by the subdrawing object:

1. Go to the appropriate tab in the **Palette** dialog. The new symbol is added to the current palette tab. If a tab has not yet been created for this item, see Section 4.12.3. The new item has the same name as the display file used to create the subdrawing object. If this procedure is used repeatedly with the same subdrawing, Display Builder appends sequential numbers to subsequent versions of the subdrawing file.
2. Select the subdrawing object for the subdrawing file.
3. Select **File/Add Subdrawing to Palette**. A bitmap of the new palette symbol is added to the Palette, together with the new default name.

When this method is used to add a symbol to the palette, a file is created in the subdirectory for the currently selected tab. A file for the bitmap is also added, together with the new subdrawing file. Changing the name of a palette symbol also changes the name of the corresponding bitmap file.

If the user wants to add to the palette a subdrawing file that contains the selected subdrawing object, as well as any dynamics or rules applied to the subdrawing object, the Add to Palette option should be used instead. That procedure is described in Section 4.12.1.1.

4.12.1.3 Adding subdrawing or bitmap symbols from file to palette

Add subdrawing or bitmap items to the Palette by copying subdrawing or bitmap files to subdirectories of the palette directory. The subdrawing or bitmap file must already exist. To add a subdrawing file to the palette by selecting the subdrawing in the display, see Section 4.12.1.1.

When copy and paste is used to add the files, the new symbols does not show in the palette until the next time the Display Builder is started.

4.12.2 Creating objects from palette

The palette provides an easy way to add subdrawing and bitmap objects to the user's display. Any symbol shown in the palette can be created in the display. The type of file associated with the symbol determines whether it is created as a subdrawing or a bitmap.

To create an object from the palette:

1. Go to the appropriate tab in the palette.
2. Click and keep the symbol in the palette pressed.
3. Drag the object into the display.
4. Release.

If the object is a bitmap, it is created at a default size. This procedure creates either a subdrawing object or a bitmap object, depending on the kind of file associated with the palette symbol.

4.12.3 Adding tabs to palette

Tabs can be added to the palette by creating new subdirectories under the palette directory. When a new subdirectory is created under the palette directory, the palette shows a new tab with that subdirectory name.

For example, to add a tab for .gif files:

1. Go to the palette directory of Display Builder.
2. Create a new directory named Gifs.

The new tab is shown the next time the Display Builder is started.

4.12.4 Hidden tabs or files from palette

If the palette shows tabs or symbols the user does not want to use, they can be hidden from the palette. This allows the user to remove tabs or objects from the palette without having to delete them from the palette directory. If the palette is crowded, this is a good way to declutter it.

4.12.4.1 Hiding tabs or files

To hide a tab or a symbol from the palette, turn on the hidden property for the directory or file using Windows Explorer. The tab or symbol is hidden from the palette the next time the Display Builder is started.

4.12.4.2 Exposing hidden tabs or files

To expose a hidden tab or a symbol to the palette, turn off the hidden property for the directory or file using Windows Explorer. The tab or symbol is shown in the palette the next time the Display Builder is started.

4.12.5 Palette directory

Files shown in the palette are located in subdirectories under the palette directory \sc\prog \graphicsEngine\Palette. Files must have the appropriate extension so that they can be created as objects in Display Builder. For example, bitmap files must have a .bmp or .gif extensions, and subdrawing files must have an .sd extension. Display Builder can only use 1, 4, or 8-bit uncompressed DIB files as objects, but the user can use any kind of bitmap file as display.

4.13 Initial zoom level

When opening a process display, Monitor Pro searches a Bounding Box graphical object. If the graphical object is found, Monitor Pro zooms in and the graphical object covers the whole Monitor Pro window completely, see Figure 28.

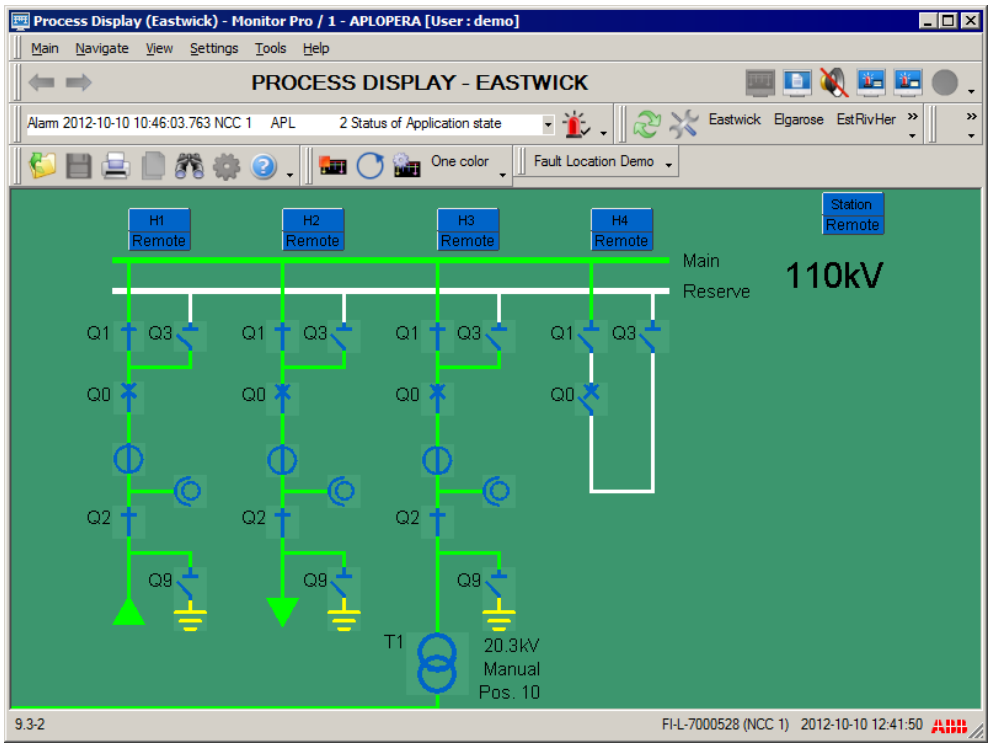


Figure 28: Bounding Box graphical object

When a process display is being built, Display Builder automatically adds a Bounding Box rectangle object to the bindings of a new process display, see Figure 29. The initial zoom level can be specified by resizing and moving the object.

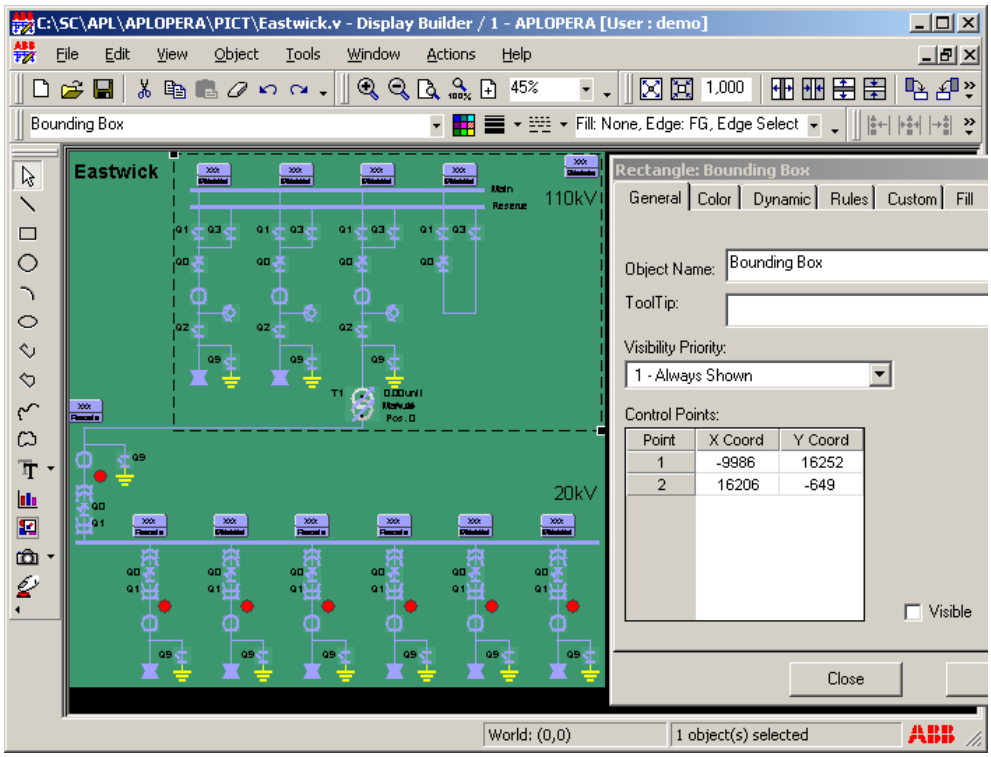


Figure 29: Bounding Box rectangle object

If the Bounding Box graphical object is deleted, the user can create a new object or copy it from a new process display. The object name is case sensitive and the object's shape can be anything, although a rectangle is recommended. If several objects with the same name are found, the bottom object is used for zooming.

Section 5 Power process symbols

SYS600 provides a set of power process symbols. The symbols are implemented as subdrawings. Several data variables and custom attributes define the behavior of the symbols, their appearance in Monitor Pro and how Display Builder handles the symbols during editing.

To view all the symbols at a time, copy the All Symbols.v file to the application's PICT folder. The file is located in \sc\prog\graphicsEngine\.

To view and modify the data variables and custom attributes in Display Builder:

1. Right-click a symbol in a process display.
2. Select **Open Subdrawing** from the shortcut menu.
3. Select **Edit/Data Variables** or **View/Properties/Custom** from the menu bar.

The symbols can be grouped either physically or logically.

The symbols are grouped physically by:

- the **Palette** tab
- the file name

The symbols are grouped logically by:

- the super type
  value of the SASuperType custom attribute, for example, Switching Device or Transformer.
- the object type
  value of the SAObjectType custom attribute, for example, Breaker or 2-Winding Transformer.
- the representation
  value of the SARepresentation custom attribute, for example, ANSI or IEC.
- the orientation
  value of the SAOrientation custom attribute, for example, Horizontal or Vertical.

The symbols can contain one or more of the attributes shown in Section 5.7.

5.1 Common symbols

5.1.1 Arrow 3D

File name	Palette icon
Arrow 3D.sd	

Data variables	Value
Horizontal	Float: 0 = horizontal gradient, 1 = vertical gradient

5.1.2 Door

File name	Palette icon
Door.sd	
Data variables	**Value**
Position	Float: 0 = Intermediate, 1 = Closed, 2 = Open, 3 = Faulty
Status color	Float: Clut index 0-255

5.1.3 Custom status

File name	Palette icon
Custom Status.sd	
Data variables	**Value**
Position	Float: 0 .. 9
Status color	Float: Clut index 0-255
Text 1 .. Text 9	Text: Text according to value of Position

5.1.4 Button

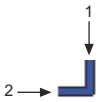
File name	Palette icon
Button.sd	
Data variables	**Value**
Top Text	Text
Middle text	Text
Bottom text	Text
Status color	Float: Clut index 0-255
Show Status Color	Float: 0 = Invisible, 1 = Visible as a border, 2 = Visible as a border when selected or alarming, 3 = Visible as face color, 4 = Visible as face color when selected or alarming
Normal Text Visibility	Float: Value range is 0-7. Values 1-3 are used to make top, middle or bottom text element visible. Values 4-6 control the visibility of pairs. 0 = Invisible, 1 = Top, 2 = Middle, 3 = Bottom, 4 = Top, Bottom, 5 = Top, Middle, 6 = Middle, Bottom, 7 = Top, Middle, Bottom
Bold Text Visibilty	Float: Value range is 0-7. Values 1-3 are used to make top, middle or bottom text element visible. Values 4-6 control the visibility of pairs. 0 = Invisible, 1 = Top, 2 = Middle, 3 = Bottom, 4 = Top, Bottom, 5 = Top, Middle, 6 = Middle, Bottom, 7 = Top, Middle, Bottom
State Color	Float: Clut index 0-255
Table continues on next page	

File name	Palette icon
Show State Color	Float: 0 = Invisible, 1 = Visible
Sunken	Float: 0 = Button raised, 1 = Button Sunken
Toggle	Float: 0, 1

5.1.5 Execute button

File name	Palette icon
Execute Button.sd	
Data variables	**Value**
Executable	Text (max 256 characters): Path to the executable program file
Arguments	Text (max 256 characters): Command line arguments for the program
Caption	Text (max 16 characters): Caption shown in the symbol

5.1.6 Line segments

File name	Palette icon	Hotspots
Busbar Horizontal.sd		
Busbar L.sd		
Busbar Vertical.sd		
Busbar Network Connecting H.sd, Obsolete, do not use		
Busbar Network Connecting.sd, Obsolete, do not use		
Data Variables	**Value**	
Topology Color	Float: Color index 1 to 255	
Remarks		
Symbol can be added to a network topology model. This symbol can be snapped to middle of an another symbol and other symbols can be snapped to middle of this symbol.		

Line segments are used for connecting device symbols in process displays. Line segments are similar to, for example, busbars and cables that are used for connecting real devices in primary processes. Busbars can be connected to line indicators in embedded process displays.

Network topology coloring uses Topology Color data variable to set the color of the line segment. In the dynamic coloring, the color value also reflects the electric state of the line segment and therefore the value can be used to add some customized logic to line segments, for example, a dashed line segment may be used when the line segment is in earthed state, see Section 5.9. Note that the powered state can have several colors depending on network topology coloring modes and that looped state is only applicable if the loop indication is enabled. Powered state is a group of color indexes where as only one color index value determines electric states such as unpowered, uncertain, error, and earthed. For more information, see Table.

File name	Palette icon	Hotspots
Busbar Label.sd		
Data variables	**Value**	
Caption	Text (max 255 characters): Busbar Label's caption.	
Topology Color	Float: Color index 1 to 255	
Remarks		
Symbol can be added to a network topology model. This symbol can be snapped to middle of a symbol.		

Busbar Label follows the color of the nearest actual line segment. Primitive lines and polylines can be used instead of busbars.

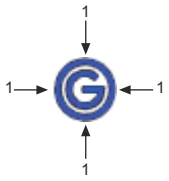
5.1.7 Capacitor

File name	Palette icon	Hotspots
Capacitor H.sd		
Capacitor.sd		
Remarks		
Symbol can be added to a network topology model.		

5.1.8 Earth

File name	Palette icon
Earth.sd	
Data variables	**Description**
Status Color	Float: (clut index 0 - 255)
Remarks	
Symbol can be added to a network topology model. Symbols default color is 58.	

5.1.9 Generator

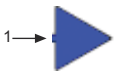
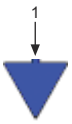
File name	Palette icon	Hotspots
Generator.sd		
Data variables	**Value**	
External Color	Float: Color code given by a external coloring component, for example DMS600. Value defines the active feed state of the component. Value greater than 0=On, otherwise=Off. Feed state of unmapped component is On.	
Voltage Level (kV)	Float: Voltage level in kilovolts	
Remarks		
Symbol can be added to a network topology model. Process Object value can be attached to Voltage Level Data Variable. This way, for example using measurement value, the voltage level coloring can be made dynamic.		

5.1.10 IED

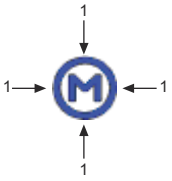
File name	Palette icon
IED.sd	
IED_pcm.sd	
Data variables	**Value**
Alarm Signal	Float: 0=Normal, 1=Unacknowledged Alarm (Blinking), 2=Alarm
Caption	Text (max 16 characters)
Remarks	
IED.sd On click program identified by ProgId LIB510RT.LIB510IEDTools is launched. Symbols defined additional custom attribute called LIBObjectName. **IED_PCM.sd** On click program identified by ProgId IedPcm.Ied is launched. Symbols defined additional custom attribute called PCMObjectPath.	

The IED symbols act as a link to the MicroSCADA LIB 510 relay tools or the PCM relay tools. After adding a symbol to a process display, select **Object/Properties/Custom**. Type LIBObjectName or PCMObjectPath required by the relay tools.

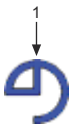
5.1.11 Line indicator

File name	Palette icon	Hotspots
Line Indicator H.sd		
Line Indicator.sd		
Data variables	**Value**	
Position	Float: Less than 0=Outgoing, 0=Unpowered, greater than 0=Incoming. The absolute value is a color code given by an external coloring component, for example DMS600.	
Status Color	Float: Clut index 0-255	
Voltage Level (kV)	Float: Voltage level in kilovolts	
Remarks		
Symbol can be added to a network topology model. If this symbol is in an embedded subdrawing, symbols in outer display can be snapped to this. Process Object value can be attached to Voltage Level Data Variable. This way, for example using measurement value, the voltage level coloring can be made dynamic.		

5.1.12 Motor

File name	Palette icon	Hotspots
Motor.sd		
Remarks		
Symbol can be added to a network topology model.		

5.1.13 Reactor

File name	Palette icon	Hotspots
Reactor H.sd		
Reactor.sd		
Remarks		
Symbol can be added to a network topology model.		

5.1.14 Truck

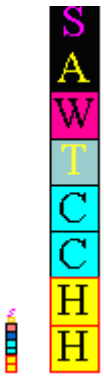
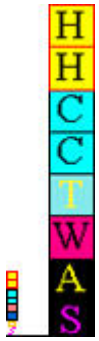
File name	Palette icon	Hotspots
Truck H.sd		
Truck.sd		
Data variables	**Value**	
Position	Float: 0=Intermediate, 1=Open, 2=Closed, 3=Faulty	
Status Color	Float: Color index 1...111	
Remarks		
Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.		

5.1.15 Tagout

File name	Palette icon
Tagout Indicator.sd	
Tagout Indicator H.sd	
Data variables	**Value**
Value	Float: Priority value 2...63

File name	Palette icon
Tagout Solid Frame.sd	
Tagout Pattern Frame.sd	
Data variables	**Value**
Value	Float: Priority value 1...255

File name	Palette icon
Tagout Text.sd	
Data variables	**Value**
Value	Float: Priority value 1...255
TagText	Text: Tagout Text

File name	Palette icon
Tagout 8xSymbol BottomUp.sd	
Tagout 8xSymbol LeftRight.sd	
Tagout 8xSymbol LeftRight Overlay.sd	
Tagout 8xSymbol RightLeft.sd	
Tagout 8xSymbol TopDown.sd	
Tagout 8xSymbol TopDown overlay	
Data variables	**Value**
Value	Float: Priority value 1...255

These symbols are used to represent the state of active tagouts added to a switching device.

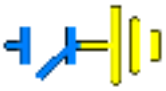
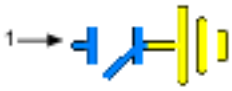
They can be installed as a group together with the corresponding switching device or separate.

5.2 IEC symbols

5.2.1 Switching devices

File name	Palette icon	Hotspots
Breaker H.sd		
Breaker.sd		
Load Breaker H.sd		
Load Breaker.sd		
Contactor H.sd		
Contactor.sd		
Disconnector H.sd		
Disconnector.sd		
Data variables	**Value**	
Position	Float: 0 = Intermediate, 1 = Open, 2 = Closed, 3 = Faulty	
Status Color	Float: Color index 1...111	
Remarks		
Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.		

5.2.2 Earth switch

File name	Palette icon	Hotspots
Earth Switch H.sd		
Earth switch.sd		
Data variables	**Value**	
Position	Float: 0 = Intermediate, 1 = Open, 2 = Closed, 3 = Faulty	
Status Color	Float: Color index 1...111	
Remarks		
Many earth switches are connected to the same process object and appear in the same Topology Model. Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.		

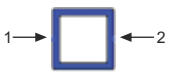
5.2.3 Transformers

File Name	Palette icon
Transformer2w.sd	
Data variables	**Value**
Bounding Box	Float: 0=Visible, 1=Invisible
Primary Color	Float: Color index 1...111
Primary Tap Changer	Float: 0=Invisible, 1=Visible
Primary Voltage Level (kV)	Float: voltage level in kilovolts
Secondary Color	Float: Color index 1...111
Secondary Tap Changer	Float: 0=Invisible, 1=Visible
Secondary Voltage Level (kV)	Float: voltage level in kilovolts
Transformer3w.sd	
Data variables	**Value**
Bounding Box	Float: 0=Visible, 1=Invisible
Table continues on next page	

File Name	Palette icon
Primary Color	Float: Color index 1...111
Primary Voltage Level (kV)	Float: voltage level in kilovolts
Secondary Color	Float: Color index 1...111
Secondary Voltage Level (kV)	Float: voltage level in kilovolts
Tertiary Color	Float: Color index 1...111
Transformer4w.sd	
Data variables	**Value**
Bounding Box	Float: 0=Visible, 1=Invisible
Primary Color	Float: Color index 1...111
Primary Voltage Level (kV)	Float: voltage level in kilovolts
Secondary Color	Float: Color index 1...111
Secondary Voltage Level (kV)	Float: voltage level in kilovolts
Tertiary Color	Float: Color index 1...111
Tertiary Voltage Level (kV)	Float: voltage level in kilovolts
Quaternary Color	Float: Color index 1...111
Quaternary Voltage Level (kV)	Float: voltage level in kilovolts
Remarks	
Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.3 ANSI symbols

5.3.1 Switching devices

File name	Palette icon	Hotspots
Breaker H.sd		
Breaker.sd		
Disconnector H.sd		
Disconnector.sd		
Table continues on next page		

File name	Palette icon	Hotspots
Data variables	**Value**	
Position	Float: 0=Intermediate, 1=Open, 2=Closed, 3=Faulty	
Status Color	Float: Color index 1...111	
Remarks		
Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.		

5.3.2 Earth switch

File name	Palette icon	Hotspots
Earth Switch H.sd		
Earth switch.sd		
Data variables	**Value**	
Position	Float: 0 = Intermediate, 1 = Open, 2 = Closed, 3 = Faulty	
Status Color	Float: Color index 1...111	
Remarks Many earth switches are connected to the same process object and appear in the same Topology Model. Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.		

5.3.3 Transformers

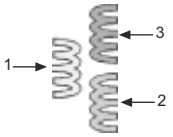
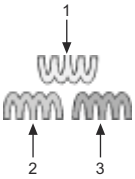
File name	Palette icon	Hotspots
Transformer2w H.sd		
Transformer2w.sd		
Transformer3w H.sd		

Table continues on next page

File name	Palette icon	Hotspots
Transformer3w.sd		
Data variables	**Value**	
Tap Changer Value	Float: Tap position	
Primary Color	Float: Color index 1...111	
Secondary Color	Float: Color index 1...111	
Tertiary color	Float: Color index 1...111	
Primary Voltage Level (kV)	Float: Voltage level in kilovolts	
Secondary Voltage Level (kV)	Float: Voltage level in kilovolts	
Tertiary Voltage Level (kV)	Float: Voltage level in kilovolts	
Primary Tap Changer	Float: O=Hide arrow, 1=Show arrow on primary winding	
Secondary Tap Changer	Float: O=Hide arrow, 1=Show arrow on secondary winding	
Transformer4w.sd		
Data variables	**Value**	
Quarternary Color	Float: Color index 1...111	
Quaternary Voltage Level (kV)	Float: Voltage level in kilovolts	
Remarks		
Symbol can be added to a network topology model. On click program identified by ProgId ControlDlgs.Launcher is launched.		

5.4 Indication symbols

5.4.1 8xAlarm indicator

File name	Palette icon
8xAlarm Led.sd	
Data variables	**Value**
Status Color 1...8	Float: Color index less than 38=Inactive (gray), 38=Not Sampled (indicated by a question mark), 39...44=Inactive, 45=Alarm, 46=Unacknowledged Alarm (blinking), greater than 46=Inactive
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

8xAlarm Led.sd combines up to eight status color signals. The priority order of the statuses differs from the normal, thus Unacknowledged Alarm status has higher priority than Not Sampled status.

Note that when an 8xAlarm symbol is added to a process display by dragging and dropping from Object Browser, all eight data variables are always mapped to indexes 10...17, the invalid mappings will appear as question marks as in Not Sampled status. To fix this, select the symbol and then **Object/Properties/Subdrawing** and manually unmap the data variables lacking corresponding indexes.

5.4.2 Alarm indicator

File name	Palette icon
Alarm A.sd	
Alarm Bell.sd	
Alarm Led.sd	
Alarm Star.sd	
Data variables	**Value**
Status Color	Float: Color index 1...111

5.4.3 Auto-reclosing tag (analog input or 5 binary inputs)

File name	Palette icon
Auto-Reclosing.sd	
Data variables	**Value**
Value 1	Float: 0=Invisible greater than 0=Visible (Blinking)
Value 2...5	Float: The text AR and the data variable's number is shown in the symbol. For example, the text AR3 is shown when Value 3 = 1.

5.4.4 Auto-reclosing tag (binary input)

File name	Palette icon
Auto-Reclosing BI.sd	
Data variables	**Value**
Value	Float: 0=Invisible, 1=Visible (Blinking)

5.4.5 Bay (analog input)

File name	Palette icon
Bay AI.sd	
Data variables	**Value**
State Signal	Float: 0 = Off, 1 = Local, 2 = Remote, 3 = Error, 4 = Error,
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters):
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.4.6 Bay (binary input)

File name	Palette icon
Bay BI.sd	
Data variables	**Value**
State Signal	Float: 0 = Local, 1 = Remote
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters):
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.4.7 Bay (double indication)

File name	Palette icon
Bay.sd	
Data variables	**Value**
State Signal	Float: 0 =Disabled, 1= Local, 2 = Remote, 3 = None
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters):
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.4.8 Operator place

File name	Palette icon
Operator Place.sd	
Data variables	**Value**
State Signal	Float: 0 = None, 1 = Remote, 2 = Station, 3 = Station/ Remote
Table continues on next page	

File name	Palette icon
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters):
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.4.9 Station (binary input)

File name	Palette icon
Station BI.sd	
Data variables	**Value**
State Signal	Float: 0 = Remote, 1 = Station
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters):
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.4.10 Station (double indication)

File name	Palette icon
Station.sd	
Data variables	**Value**
State Signal	Float: 0 = Local, 1 = Station, 2 = Remote, 3 = Out of Use
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters):
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.4.11 Tripping tag

File name	Palette icon
Tripping.sd	
Data variables	**Value**
Trip Signal	Float: Color index less than 45 = Invisible, 45 = Alarm 46 = Unacknowledged Alarm (blinking) greater than 46 = Invisible
Caption	Text (max 16 characters):

5.5 Measurement symbols

5.5.1 Measurement graphs

Display Builder provides several different graph objects that can be configured and customized in many ways. In the power process symbols, two different graphs are wrapped inside subdrawings. They can be used in process displays as such, or developed further, for example, by adding limit markers, scales, ticks, labels, suffixes and so on if needed. The minimum and maximum levels of the measurement graphs are calculated automatically by using the following formulas:

Min = Low Alarm - 0.1 * (High Alarm - Low Alarm)

File name	Palette icon
Bar Graph.sd	
Line Graph.sd	
Data variables	**Value**
Status color	Float: less or greater than 38 = No affect 38 = Not sampled
Value	Float: Measured value
High Alarm	Float: Limit between high warning and high alarm levels
High Warning	Float: Limit between normal level and high warning level
Low Warning	Float: Limit between low warning level and normal level
Low Alarm	Float: Limit between low alarm and low warning levels
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

5.5.2 Measurement values

File name	Palette icon
Value.sd	
Value FG basic.sd	
Value BBox.sd	
Value BG.sd	
Value FG.sd	
Data variables	**Value**
Table continues on next page	

File name	Palette icon
Value	Float: Measured value
Status Color	Float: Color index 1...111
Unit	Text (max 16 characters): Measured unit
Decimals	Float: Amount of decimals. By default, two decimals are shown, if not otherwise specified.
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

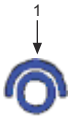
Value.sd shows a measurement value and a unit with the static text color. The Not Sampled status is indicated by a question mark. The other measurement value symbols have the following additional features:

- Value FG Basic
  The text color indicates all statuses. The blinking text indicates the Unacknowledged Alarm status.
- Value BBox
  The symbol has a bounding box. The blinking bounding box indicates the Unacknowledged Alarm status. The static red bounding box indicates the Acknowledged Alarm status.
- Value BG
  The symbol has a bounding box and the box color indicates all statuses. The blinking bounding box indicates the Unacknowledged Alarm status.
- Value FG
  The symbol has a bounding box and the text color indicates all statuses. The blinking bounding box indicates the Unacknowledged Alarm status.

The width of the symbols, especially symbols with frames, does not change according to text length. In most cases, the amount of digits is already known when editing the process display. Use the **Expand Horizontal** button to change the width of the symbol, or resize it manually.

5.5.3 Measurement symbols

File name	Palette icon
Circle.sd	
Data variables	**Value**
Status Color	Float: Color index 1...111
Caption	Text (max 16 characters): Caption shown in the symbol
Remarks	
On click program identified by ProgId ControlDlgs.Launcher is launched.	

File name	Palette icon	Hotspots
Current Transducer ANSI H.sd		
Current Transducer ANSI.sd		
Current Transducer IEC H.sd		
Current Transducer IEC.sd		
Voltage Transducer ANSI H.sd		
Voltage Transducer ANSI.sd		
Voltage Transducer IEC H.sd		
Voltage Transducer IEC.sd		
Data variables	**Value**	
Status Color	Float: Color index 1...111	
Remarks		
Symbol can be added to a network topology model. This symbol can be snapped to middle of a symbol. On click program identified by ProgId ControlDlgs.Launcher is launched.		

5.6 System self supervision symbols

A status symbol can be used to show generic statuses with proper icons.

Different presentations of statuses are possible. For example, a status can be shown with the appropriate color, shape, icon or an image.

Table 3: Icons and colors for status symbols

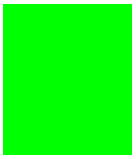
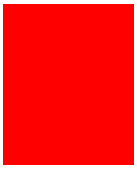
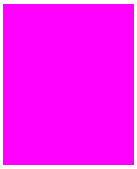
Status	Color	Icon / Image
OK		
Warning		
Not OK / Alarm		
Uncertain		

Table 4: Generic status symbol (for example, for IEC 61850 Health)

File name	Palette icon
Status 1234.sd	See Table 3.
Data variables	**Value**
Status	Float: 1 = OK, 2 = Warning, 3 = Alarm, 4 = Uncertain, All other = Uncertain
Status Color	Float: see Table 3.

Table 5: Boolean status symbol

File name	Palette icon
Status -101.sd	See Table 3.
Data Variables	**Value**
Status	Float:0 = NOT OK, other OK
Status Color	Float: See Table 3.

Table 6: Status symbol for OPC quality

File name	Palette icon
Quality OPC.sd	See Table 3.
Data variables	**Value**
Status	Float: 0-63 = Bad, 64-127 = Uncertain, 128-191 = Unknown, 192- = Good

Table 7: Status symbol for SYS600 OS quality

File name	Palette icon
Quality OS.sd	See Table 3.
Data variables	**Value**
Status	Float: See Table 3.

5.6.1 Label

A label symbol is a generic text symbol, which can have some textual process value.

The symbol has following properties:

- Text scales when zoomed.
- It is possible to set forecolor.
- It is possible to set backcolor.

Table 8: Label symbol

File name	Palette icon
Label L.sd (left aligned), Label R.sd (right aligned), Label C.sd (centered)	
Data variables	**Value**
Text	String: for example, DuoDriver
Forecolor	
Backcolor	

Table 9: Status with label

File name	Palette icon
Status 1234 with Label.sd and Status -101 with Label.sd	
Data variables	**Value**
Status	Float: See Table 4 and See Table 5.
Status Color	Float: See Table 3.
Text	String
Text Align	Float: 0 = Left aligned text, 1 = Right aligned text
Forecolor	Float
Backcolor	

Same symbol is used for left and right aligned texts. In Display Builder, this symbol shows text 'XXX' in both side of the symbol. The default setting is left aligned text, so in Monitor Pro the right side text is only shown. To use right aligned text, set the value of Text Align data variable to 1.

Table 10: Communication line

File name	Palette icon
Comm Line H.sd Comm Line V.sd	
Data variables	**Value**
Status	Integer: 0 = Bad, Other = Good
Status Color	Integer: See Table 3.
Remarks	
Symbol can be added to a network topology model. This symbol can be snapped to middle of an another symbol and other symbols can be snapped to middle of this symbol. Symbols default color is 0. On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 11: Communication Port

File name	Palette icon
Communication Port.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT OK, other = OK

Table 12: Communication Port x 4

File name	Palette icon
Communication Port x 4.sd	
Data Variables	**Value**
Port 1..4 Status	Integer: 0 = NOT OK, other = OK

Table 13: Workstation

File name	Palette icon
Workstation.sd	

Table 14: Servers

File name	Palette icon
Server.sd	
Communication Server.sd	
Report Server.sd	
Security Server.sd	
Database Server.sd	
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 15: Printer

File name	Palette icon
Printer.sd	

Table 16: Video Projector

File name	Palette icon
Video Projector.sd	

Table 17: Widescreen

File name	Palette icon
Widescreen.sd	

Table 18: LCD Monitor

File name	Palette icon
LCD Monitor.sd	

Table 19: Web Camera

File name	Palette icon
Webcam.sd	

Table 20: Router

File name	Palette icon
Router.sd	

Table 21: RuggedCom Router

File name	Palette icon
Router - RuggedRouter.sd	

Table 22: Wireless Router

File name	Palette icon
Wireless Router.sd	

Table 23: Ethernet Switched Hub

File name	Palette icon
Switched Hub.sd	

Table 24: RuggedCom Ethernet Switched Hub

File name	Palette icon
Switched Hub - RuggedSwitch.sd	

Table 25: Network Cloud

File name	Palette icon
Network Cloud.sd	
Data Variables	**Value**
Text	

Table 26: Ethernet Hub

File name	Palette icon
Hub.sd	

Table 27: Modem

File name	Palette icon
Modem.sd	

Table 28: Firewall

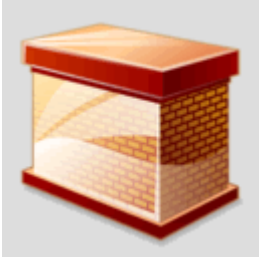
File name	Palette icon
Firewall.sd	

Table 29: Media Converter

File name	Palette icon
Media Converter.sd	

Table 30: Uninterruptible Power Supply (UPS)

File name	Palette icon
UPS.sd	

Table 31: Global Positioning System (GPS)

File name	Palette icon
GPS.sd	

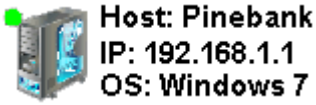
Table 32: Meinberg Lantime GPS

File name	Palette icon
GPS - Meinberg Lantime.sd	

Table 33: SYS600C symbol

File name	Palette icon
SYS600C.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT OK, other OK
Status Color	Float: See Table 3.
Text	String
Hide Status	Boolean: 0 = Visible, 1 = Invisible
Hide Texts	Boolean: 0 = Visible, 1 = Invisible
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 34: SYS600 Server symbol

File name	Palette icon
SYS600 Server.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT OK, other OK
Table continues on next page	

Status Color	Float: See Table 3.
Text1	String
Text2	String
Text3	String
Hide Status	Boolean: 0 = Visible, 1 = Invisible
Hide Texts	Boolean: 0 = Visible, 1 = Invisible
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 35: SYS600 Workplace symbol

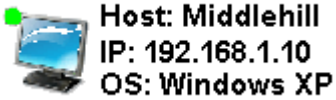
File name	**Palette icon**
SYS600 Worplace.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT OK, other OK
Status Color	Float: See Table 3.
Text1	String
Text2	String
Text3	String
Hide Status	Boolean: 0 = Visible, 1 = Invisible
Hide Texts	Boolean: 0 = Visible, 1 = Invisible
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 36: SYS600 Application symbol

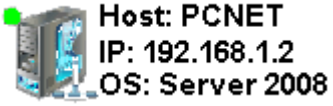
File name	**Palette icon**
SYS600 Application.sd	
Data Variables	**Value**
ApplicationStatus	Float: See Table 25.
ApplicationStatusText	String: for example, Hot
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 37: Application status codes and texts for system self supervision

Event	OV	Color	Comment
NONE	0	Grey	Not in use
COLD	1	Grey	Single system
WARM	2	Yellow	Single system
Table continues on next page			

Event	OV	Color	Comment
HOT	3	Green	Single system
HOT%	4	Green	HSB, AS = HOT, sys_event or cyclical check
TO_WARM_SEND TO_WARM_SD	5	Green, blinking	HSB, sys_event or cyclical check
WARM_SEND WARM_SD	6	Green, blinking	HSB, sys_event or cyclical check
TO_HOT_SEND TO_HOT_SD	7	Green, blinking	HSB, sys_event or cyclical check
HOT_SEND HOT_SD	8	Green	HSB, sys_event or cyclical check
TO_WARM_RECEIVE TO_WARM_RC	9	Yellow, blinking	HSB, apl_event or cyclical check
WARM_RECEIVE WARM_RC	10	Yellow, blinking	HSB, apl_event or cyclical check
TO_HOT_RECEIVE TO_HOT_RC	11	Yellow, blinking	HSB, apl_event or cyclical check
HOT_RECEIVE HOT_RC	12	Yellow	HSB, apl_event or cyclical check
Unknown	13	Red	

Table 38: SYS600 PC-NET symbol

File name	Palette icon
SYS600 PC-NET.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT OK, other OK
Status Color	Float: See Table 3.
Text1	String
Text2	String
Text3	String
Hide Status	Boolean: 0 = Visible, 1 = Invisible
Hide Texts	Boolean: 0 = Visible, 1 = Invisible
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

5.6.2 DuoDriver

There is no special symbol for DuoDriver. Instead, DuoDriver is presented using a status symbol with a label for each port.

In the SSS display, DuoDriver symbols are normally positioned next to the device they are related to. In Figure 30 the labels are positioned next to the server symbol.

Figure 30: DuoDriver status labels next to the server symbol

Table 39: RTU 560 symbol

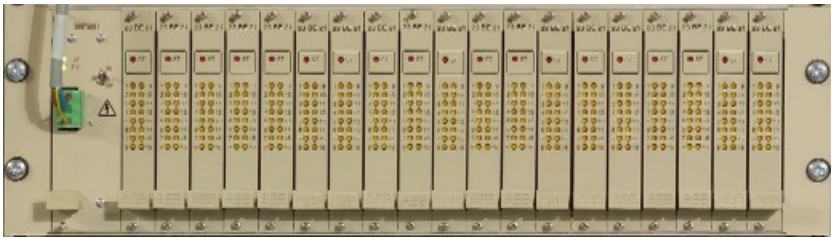
File name	Palette icon
RTU 560.sd	

Table 40: COM600 symbol

File name	Palette icon
COM600.sd	

Table 41: Generic IED

File name	Palette icon
IED - Generic.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT, other = OK
Status Color	Float: Table 3.
Text	String: "REF 615", "AAJ1Q1D1" etc.
Hide Status	Boolean: 0 = Visible, 1 = Invisible
Hide Text	Boolean: 0 = Visible, 1 = Invisible
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 42: Generic ABB and Hitachi Power Grids IED

File name	Palette icon
IED - ABB Generic.sd	
Data Variables	**Value**
Status	Integer: 0 = NOT, other = OK
Status Color	Float: See Table 3.
Text	String: "REF615_2", "AAJ1Q1D1" etc.
Hide Status	Boolean: 0 = Visible, 1 = Invisible
Hide Text	Boolean: 0 = Visible, 1 = Invisible
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

Table 43: ABB and Hitachi Power Grids IEDs

File name	Palette icon
IED - <ied name>.sd	Different presentations, where <ied name> has the following values: IED RED 54x Series, IED REF 542+, IED Relion 605 Series, IED Relion 610 Series, IED Relion 615 Series, IED Relion 630 Series, IED Relion 650 Series, IED Relion 670 Series
Remarks	
On click VSO dialog identified by SYS_TOOL/SSS_CtrlDlg.VSO@1MAIN@2LIST(LN="%LN%",IX=%IX %)@31SGTS:0 is opened.	

5.6.3 Creating IED symbols

There is one generic symbol for IEDs and one for ABB and Hitachi Power Grids IEDs, but the user can create their own IED symbols with the actual IED faceplate by replacing the image inside the generic symbol. The image should be shot from up front.

To create a symbol for a specific IED with its faceplate:

1. Open the symbol file IED - Generic.sd to Display Builder.
2. Select **File/Save as** and rename the file to IED - <ied name>.sd.
3. Select Image object in the display and open **Object/Properties/Bitmap**.
4. Select an image file having dimensions 128 x 128 pixels and click **Close**.
5. Select **View/Properties/Custom** and rename SAObjectType attribute to IED <ied name>.
6. Save the symbol file.
7. To add the symbol to the Palette, see Section 4.12. See also Section 5.10.

5.7 Custom attributes

Custom attributes can be used to control the behavior of the symbol. Normally, custom attributes are configured automatically when symbols are installed to the display, or when a

symbol is configured using different items in the graphical user interface. Normally, process graphics designer does not have to use custom attributes directly. However, they are needed when new libraries or symbols with new behavior is implemented. Most important custom attributes are listed below in Table 44.

Table 44: Custom attributes inside power process symbols

Network topology coloring	
SATopology	Indicates that the symbol has 1...5 hotspots and the symbol can be a part of the network topology. Hotspots are points where the symbol can be connected to its neighboring symbols. Only these symbols can be handled by network topology coloring related functions.
SASnapToMiddle	A symbol with the SASnapToMiddle attribute can snap to • the exact hotspots of the neighbor. • a virtual line between two hotspots of the neighbor. If the symbol is snapped to the virtual line, the symbol and the neighbor form a T-branch.
SAAllowSnapToMiddle	A symbol with the SAAllowSnapToMiddle attribute allows its neighbor to snap to • the exact hotspots of the symbol. • a virtual line between two hotspots of the symbol. If the neighbor is snapped to the virtual line, the symbol and the neighbor form a T-branch.
SAAllowSnapToEmbedded	If a symbol is in an embedded process display and has this attribute, the symbol allows other symbols in a network overview display to snap to it.
Tool starting	
SAAuthLevel	These attributes define the symbol's default behavior when the user clicks the symbol in Monitor Pro. Because the attribute values may contain complex syntax, the values can be defined by using the Tool Launcher Settings dialog. To open the dialog: • Open a subdrawing. • Right-click the drawing area when no graphical objects are selected. • Select **Tool Launcher** from the **shortcut** menu.
SAAuthGroup	
SARunSCILOnClick	
SARunCmdLineOnClick	
ToolLauncherProgId	
SAOpenDisplayOnClick	
SAOpenVSCILOnClick	
SAOpenVSCILOnClickVso	
ChangePictureOnClick	Obsolete version and syntax for opening or zooming a process display in Monitor Pro. The Tool Launching dialog automatically converts the attribute to the attribute SAOpenDisplayOnClick.
Miscellaneous	
SADefaultColor	The SADefaultColor attribute's value defines a color index that is used when a new symbol is created and foreground color is initially assigned. Possible values are 0...255.

Table continues on next page

SAAddCustomAttribute	If a symbol contains the SAAddCustomAttribute attribute inside the subdrawing, a new custom attribute is added outside of the subdrawing object when the symbol is created. The internal value is used as a key name of the new custom attribute. If more than one custom attributes need to be created to the subdrawing object, the internal attributes can be named SAAddCustomAttribute, SAAddCustomAttribute2, SAAddCustomAttribute3, and so on.
SAUpdateRate	Process Display specific update rate in milliseconds.
SADisableZoomPan	Process Display specific disabling of all zooming and panning actions.

5.8 Status colors

If not otherwise mentioned, all power process symbols containing a status color data variable have the following functions:

- If a status color value is less than 1 or greater than 111, the Unmapped Device color is used (color index = 29). Typically, the value is zero, because the data variable has no mapping. Transformers have their own Unmapped Winding colors for each winding (color indexes = 20, 21 and 22).
- If the status color value is 38, the mapping is not valid or the mapped process object has the Not Sampled status. In this case, a question mark is shown, see Table 45.
  - Primary Color data variable is used for transformers.
  - In measurement graphs, the question mark is shown next to the measurement data. See Bar Graph.sd in Table 45.
  - In bay and station symbols, the question mark is shown below the caption. See Bay.sd in Table 45.
  - The question mark covers other symbols completely. See Truck.sd in Table 45.

Table 45: Not Sampled status indications

		
Bar Graph.sd	Bay.sd	Truck.sd

- If the status color value is 42 (selected, under command), white blinking occurs in:
  - switching devices.
  - transformers.
  - bay and station symbols.
- If the status color value is 46 (Unacknowledged Alarm), red blinking occurs.
- Other status color values are indicated by using the corresponding color index as a foreground or background color.

5.9 Modifying existing symbols

All Power Process symbols can be freely modified in the Display Builder. The only exception is the locations of the Hotspot objects inside the symbols. Moving these may cause the existing process display to stop working.

All modified symbols should be saved to the subdirectories of the ..\sc\apl\<application name>\AplMod4\Palette\ directory, from where the symbols will be automatically shadowed in Hot Stand-by systems. For example, if the default 01 - SA_Common\Generator.sd is modified, it should be saved as \sc\apl\<application name>\AplMod4\Palette\01 - SA_Common\Generator.sd. In this directory, modified symbols are not overwritten when a new SYS600 version is installed, and the modified version will automatically be used in process displays because of relative paths. Relative symbol search paths in the order of priority are as follows:

```
<drive>:\sc\apl\<application name>\PICT\
<drive>:\sc\apl\<application name>\AplMod4\Palette\
<drive>:\sc\prog\graphicsEngine\Palette\
<drive>:\sc\prog\graphicsEngine\etc\
<drive>:\sc\prog\graphicsEngine\lib\
<drive>:\sc\prog\graphicsEngine\lib\views\
<drive>:\sc\prog\graphicsEngine\lib\fonts\
<drive>:\sc\prog\graphicsEngine\lib\templates\
<drive>:\sc\prog\graphicsEngine\lib\templates\drawings\
<drive>:\sc\prog\graphicsEngine\support\
```

It is recommended to maintain a project specific record that explains the modifications that have been made to each symbol and that this document will be stored to same location as the modified symbols.

Example: Dash line for earthed line segments:

Earthed state has a color index 58 as seen in Table.

1. Open existing Vertical Busbar.sd symbol in Display Builder.
2. Save this symbol to ..\sc\graphicsEngine\<application name>\AplMod4\Palette\01 - SA_Common\ directory with the same file name.
3. Create a new primitive line to the symbol.
4. Open object properties and give it a name Earthed dash line, modify line style (dash) and width (3).
5. Select Object foreground color and set it to index 23 (process display background).
6. Select Dynamic tab and enable Visible dynamics (Data variable=Topology Color, In Min=57, In Max=59).
7. Edit threshold for this item as shown in Figure 31.

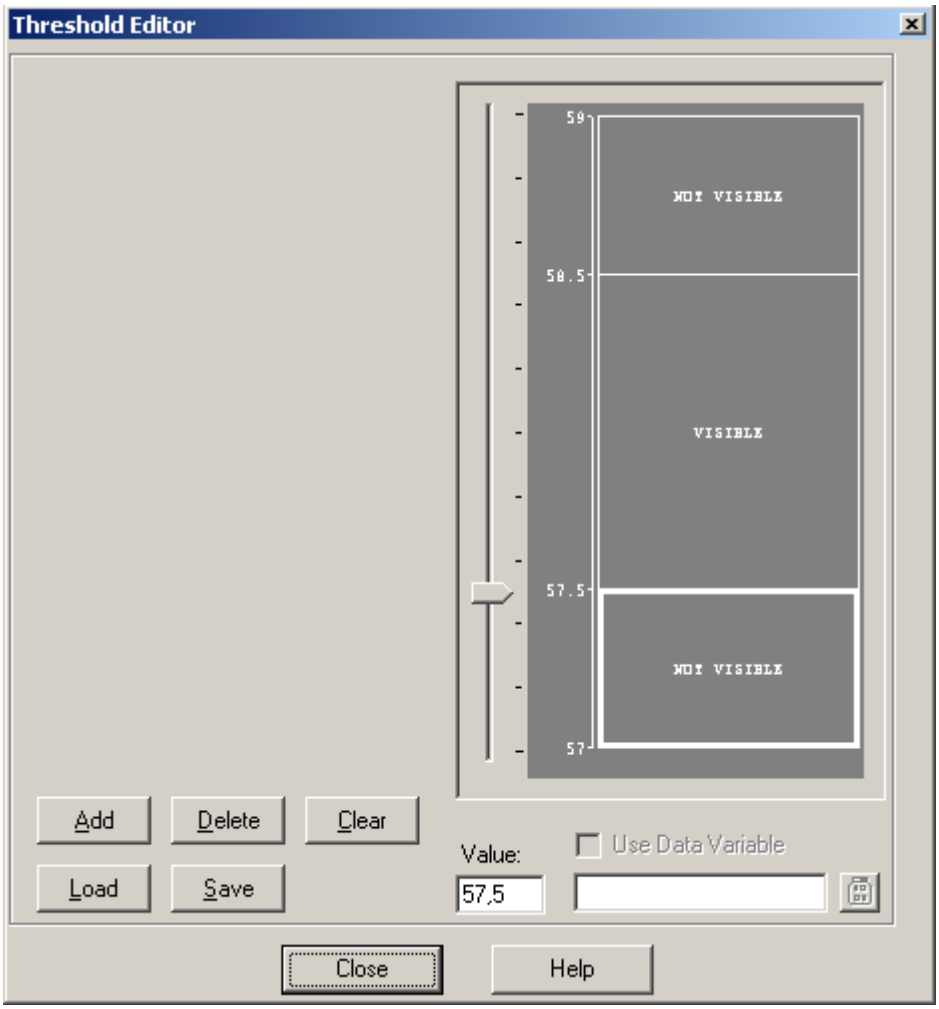


Figure 31: Editing threshold for earthed state

8. Close the dialogs and move Earthed dash line object above the object named Line. Save the symbol and open a process display where there are vertical line segments, which are earthed.

5.10 Creating new symbols

When creating new symbols, it is recommended to draw them centered to the origin (0, 0) and so that the graphics cover approximately 1000x1000 world coordinates, that is, from (-500, -500) to (500, 500). Enable grid feature to ease positioning of the objects. Any of the existing symbols can be used as a template. It is recommended that a descriptive SAObjectType custom attribute should be assigned to the symbol, and if applicable, SASuperType, SARepresentation and/or SAOrientation custom attributes. Also, other custom attributes listed in Section 5.7 can be used depending on the symbol functionality.

Save custom symbols to \sc\apl\<application name>\AplMod4\Palette\<tab name>\. The symbols will be automatically shadowed in Hot Stand-by systems and are not overwritten when installing new SYS600 version.

5.10.1 Adding new or modified symbols to Palette window

New or modified symbols saved to the \sc\apl\<application name>\AplMod4\Palette\<tab name>\ directory are not automatically added to the Display Builder's Palette window. The

symbols can be used from Palette window by creating a shortcut to the actual symbol in the application directory. Here are the instructions:

1. Create a new tab in the Palette window by adding new directory \sc\prog\graphicsEngine \Palette\<tab name>. Use exactly the same name as <tab name> directory mentioned above.
2. Copy \sc\prog\graphicsEngine\support\Palette Stub.sd as \sc\prog\graphicsEngine \Palette\<tab name>\<symbol name>.sd. Use the exact name of the <symbol name>.sd located in \sc\apl\<application name>\AplMod4\Palette\<tab name>\ directory, for example My Busbar.sd.
3. Copy a 32x32 pixel 256 color .bmp file, created for example with Windows Paint, as \sc \prog\graphicsEngine\Palette\<tab name>\<symbol name>.bmp to be used as palette item bitmap. Use hidden file attribute to hide <symbol name>.bmp from the Palette window.
4. Restart Display Builder to take new tab and symbol into use.

It is recommended that the Palette stub symbol is used as a shortcut to the custom symbol to ensure the correct version of the symbol is used. Missing symbols will be indicated as shown in Figure 32.

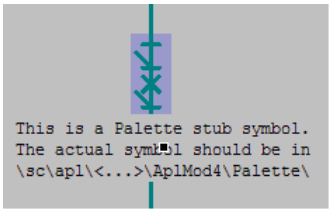


Figure 32: Symbol missing in application directory

To enable adding custom symbols from Object Browser of Display Builder, modify \sc\prog \graphicsEngine\etc\ObjNav.ini.

5.11 Primitive symbols

5.11.1 Text symbols

When **Scale with Zoom** is enabled, the text object resizes when zooming the view. When **Scale with Zoom** is disabled, the text object maintains the same size when zooming the view. Note that this attribute can only be set before creating a true type text object.

To set the resize behavior:

1. Pull down the **Options** menu next to the **true type text** object button.
2. Toggle **Scale with Zoom** on or off.

5.11.2 Graph types

5.11.2.1 Bar graphs

Bar graphs display data values using one bar for each data element. Additional variables are displayed using additional bars, lines, or whole bar graphs. The dimension of the bar is proportional to the variable value.

The color of each bar or line is determined by the color or color threshold table associated with the variable. Each bar or line is divided into sections of different colors depending on the

variable's value and the colors assigned in the threshold table. The exception is the Solid Bar Graph, in which the entire bar appears in the threshold color of the current variable value.

Bar graphs display scalar, vector, or matrix data. A vector variable whose length equals the number of samples updates most efficiently. A matrix whose total number of elements equals the number of samples also updates efficiently.

Bar graphs can display an unlimited number of data samples.

Most bar graphs scroll from right to left, or wrap around to the left edge of the graph. The exception is the Horizontal Bar Graph, which scrolls down, or wraps around to the bottom of the graph.

Bar Chart
Maximum Number of Variables: 10

Draws one vertical bar for each variable element.

Bar-Line
Maximum Number of Variables: 11

Displays the first variable as a bar chart and all subsequent variables as lines.

If all variables have the same range, only the left value axis is displayed. If the range of the second variable (the first to be displayed as a line) is different from the first, a second value axis is displayed on the right.

If only one variable is specified, this graph type displays an overlapping bar and line graph using one variable.

The legend of this graph type lists only the first variable, displayed as a bar. The remaining variables do not appear in the legend. To list all of the variables in the legend, turn the legend off in the Bar-Line graph and use the Legend graph type to display the variables.

The minimum number of samples allowed is 2.

Centered Bar Chart
Maximum Number of Variables: 10

Draws one bar for each variable element. The bar is centered vertically in the graph. The vertical center point of the graph represents the bottom of the range of the variable. The bar extends the same distance in both directions vertically.

The range of the first variable determines the value of the center line.

For this graph type to be meaningful, all variables should have the same range.

Horizontal Bar Chart
Maximum Number of Variables: 10

Draws one bar for each variable element, starting at the bottom of the graph.

Scrolls down or wraps around to the bottom of the graph.

Packed Bar Chart
Maximum Number of Variables: 10

Draws one bar for each variable element. This graph type differs from the Bar Chart in having no gaps between the bars.

Packed Bar-Line
Maximum Number of Variables: 11

Displays the first variable as a bar chart, and all subsequent variables as lines. This graph type differs from the Bar-Line Graph in having no gaps between the bars.

If all variables have the same range, only the left value axis is displayed. If the range of the second variable (the first to be displayed as a line) is different from the first, a second value axis is displayed on the right.

If only one variable is specified, this graph type displays an overlapping bar and line graph using one variable.

The legend of this graph type lists only the first variable, displayed as a bar. The remaining variables do not appear in the legend. To list all of the variables in the legend, turn the legend off in the Packed Bar-Line graph and use the Legend graph type to display the variables.

The minimum number of samples allowed is 2.

Piggyback Bar Chart
Maximum Number of Variables: 10

Draws one bar for each sample. Variable values are stacked vertically within the bar, adding each variable value to the sum of the values beneath it. The first variable is on the bottom and the last variable is on the top.

The height of the bar is proportional to the sum of the values of all the variables.

For this graph type to be meaningful, all variables should have the same range.

The range of the graph equals the sum of the variable ranges. The variables should be either all logarithmic or all linear.

Piggyback Bar Distribution
Maximum Number of Variables: 10

Draws one bar for each sample. Variable values are stacked vertically within the bar, adding each variable value to the sum of the values beneath it. The first variable is on the bottom and the last variable is on the top.

This graph type is useful for displaying statistical distribution.

All variables must have the same range. The range of the graph equals the range of the attached variables. For this graph type to be meaningful, the sum of the variable values for each sample should not exceed the maximum range value. For example, if a graph has three variables with a range of 0 to 10, the range of the graph is 0 to 10. A given sample of the three variables might have values of 2, 3, and 5 or 2, 3, and 4 but not 2, 3, and 6.

See also "Piggyback Bar Chart".

Solid Bar Chart
Maximum Number of Variables: 10

Draws a standard bar chart in which the entire bar appears in the threshold color of the current variable value.

Use distinct color thresholds to avoid confusion between adjacent variable values.

Stacked Packed Bar-Line
Maximum Number of Variables: 32

Displays each variable pair as a Packed Bar-Line Graph, stacking each graph above the previous one.

The first variable of each pair is displayed as a packed bar; the second as a line.

There is no space between the bars.

Since the maximum number of variables allowed is 10, and each graph uses 2 variables, the maximum number of stacked graphs is 5.

The value axis of each graph is displayed on the left side of the graph. The value axis is determined by the first variable of the pair. If the range of the second variable (the one displayed as a line) is different from the first, a second value axis is displayed on the right side of that graph.

If there is an odd number of variables, the last graph displays an overlapping bar and line graph using one variable.

This graph type displays a single title and a single legend for the stack of graphs. The legend lists all of the variables in the stack of graphs.

The minimum number of samples allowed is 2.

5.11.2.2 Block graphs

Block graphs display data values using a box (a filled rectangular area) for each variable element. As the variable value changes, the color of the box changes according to the variable thresholds.

The color of each block is determined by the color or color threshold table associated with the variable.

These graph types work well with color thresholds.

Block graphs can display scalar, vector, or matrix data. The shape of the variable determines the number of boxes displayed.

Only one data sample is displayed at a time.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. For matrix data, both the horizontal and vertical axis ticks and tick labels apply. For vector data, only the horizontal ticks and tick labels apply. For scalar data, neither the horizontal nor vertical axis ticks and tick labels apply.

The value axis tick label displays the variable value centered in each block.

The time axis value is equal to the current sample number.

The maximum number of variables allowed is 5.

Block graph
The maximum number of variables for the block graph is 5.

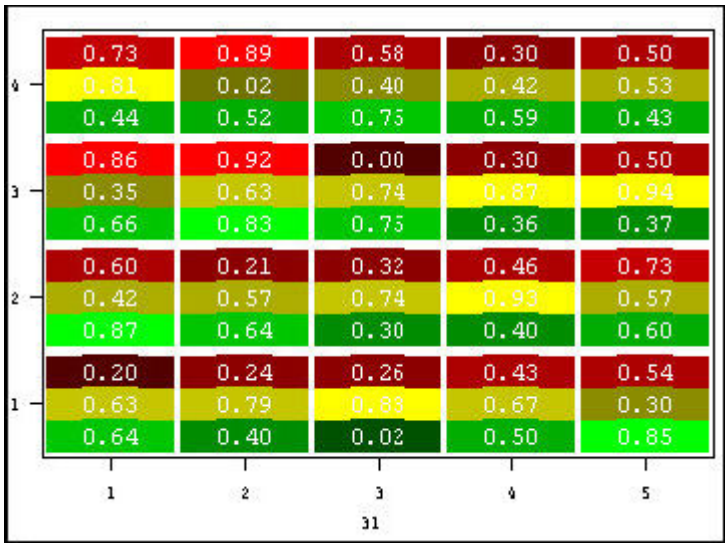


Figure 33: Block graph variable elements

The block graph draws a box (a filled rectangular area) for each variable element. The boxes of each element set are outlined in the background color.

Packed block
The maximum number of variables for the packed block is 5.

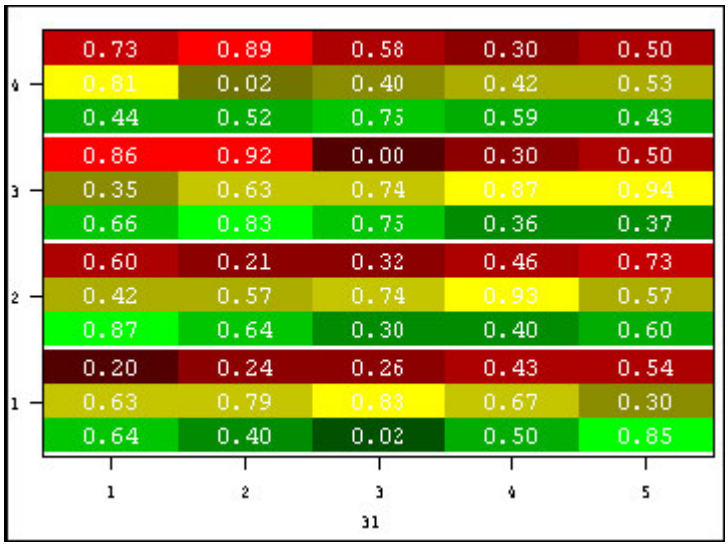


Figure 34: Packed block variable elements

The packed block draws a box (a filled rectangular area) for each variable element without a separating outline between the boxes.

5.11.2.3 Bullseye graph

The maximum number of variables for the bullseye graph is 5.

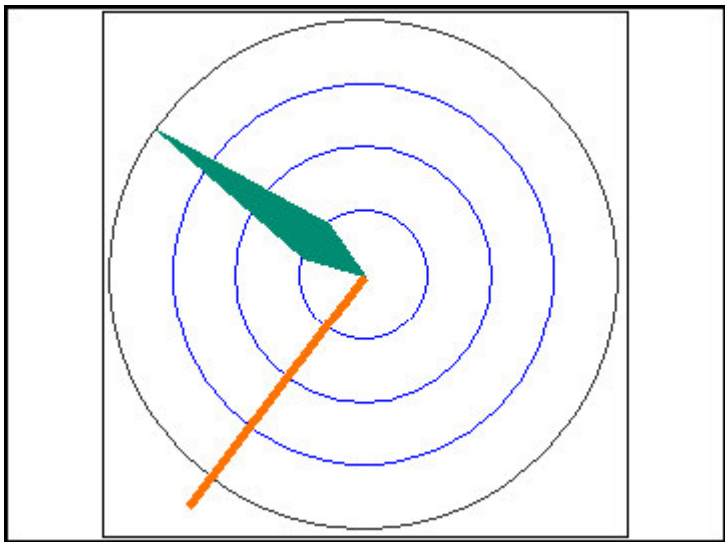


Figure 35: Bullseye graph

The graph displays x,y points on a Cartesian graph.

The bullseye graph accepts either scalar or vector[2] variables. If scalar variables are used, two variables are required for each graph point to provide the x and y values, respectively. If vector variables are used, the variable can only have two elements, for the x and y values respectively.

The range of the variables must be symmetrical around zero. All variables must have the same range.

To display the axes, turn on the **Value Grid** option on the **Basic** tab of the **Graph Properties** dialog.

The grid attributes options determine the line type and color. The grid consists only of the X and Y axes.

To display radial target lines, turn on the **Ticks** option on the **Y Axis** tab of the **Graph Properties** dialog. To display rectilinear target lines, turn on the **Time Grid** option on the **Basic** tab of the **Graph Properties** dialog.

To display both the axes and rectilinear target lines, turn on both grids on the **Basic** tab of the **Graph Properties** dialog.

To vary the number of target lines, enter the desired number of target lines in the **Time Start** text box on the **X Axis** tab of the **Graph Properties** dialog.

When using vector variables, the first *n* variables provide target line values where *n* is the number of target lines specified. If the target lines are radial, the first element of the vector is the radius of the circle and the second element is ignored. If the target lines are rectilinear, the first element of the vector provides the x position and the second element provides the y position. The remaining variables are plotted as x,y coordinates.

When using scalar variables, the first *n* variables provide target line values for radial target lines where *n* is the number of target lines specified. Each target line value provides the radius of a circle. If the target lines are rectilinear, *2n* variables are required for the target line values. In each pair of variables, the first variable provides the x position and the second provides the y position. The remaining variables are plotted as x,y coordinates, so there must be an even number of graph variables. The graph attributes are determined by the second variable (the y variable) in each pair.

The graph variables are plotted alternately as a solid vector and a clock hand, starting with a solid vector, to prevent vector pairs from overlapping. The hour hand can be hollow or filled. To

make the hour hand filled, turn on the **Ticks** option on the **Y Axis** tab of the **Graph Properties** dialog.

The color of the solid vectors and clock hands is determined by the color associated with the variable if a vector variable is being used, or by the color associated with the second variable if scalar variables are being used. If the determining variable has a color threshold table, the color is determined by that variable value and the corresponding color in that threshold table.

The scale of the graph corresponds to the range of the variables. This scale is applied to both the x and y axes. The bullseye graph does not currently allow separate scale control of the X and Y axes.

The bullseye graph supports color threshold tables when using radial target lines. The number of entries in the color threshold table should be one more than the number of target lines. This provides a color range between each pair of target lines and beyond the innermost and outermost target lines. The actual numerical values of the thresholds are ignored since thresholds are set to equal the range bar values.

When the variable value crosses a range bar value, the color of the vector or clock hand changes. If there are not enough thresholds, no color dynamics are used. If there are extra thresholds, the graph starts with the lowest threshold and uses only as many thresholds as it needs.

The number of history samples displayed is determined by the number specified in the **Samples** option on the **X Axis** tab of the **Graph Properties** dialog. To display history, the **Samples** value must be greater than 1 and the variable must have any marker except the null marker. If both of these conditions are not true, no history is displayed. When using history, each new vector and clock hand leaves a history marker in the color of the vector or clock hand.

5.11.2.4 Context graphs

Context graphs display variable information, not variable values. Only one sample is displayed at a time. Because their position and size can be controlled independently, these graphs can sometimes be more useful than using the legend of other graph types.

Context graphs can use scalar, vector, or matrix data.

Color bar
The maximum number of variables for the color bar is 1.

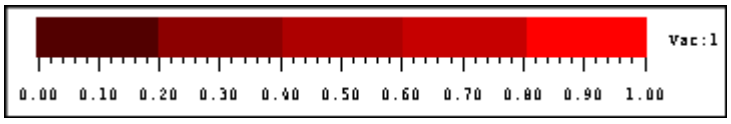


Figure 36: The color bar

The color bar displays the color threshold table of the variable as a horizontal legend.

The color bar appears at the top edge of the graph area. The height of the color bar is proportional to the width of the graph area, not to its height. If the graph area is not as high as the color bar, the complete color bar and axis still displays correctly.

This graph type works best with color thresholds.

The axis displays the value range of the color threshold table. The axis and variable name cannot be turned off.

Legend graph
The maximum number of variables for the legend graph 20.

Figure 37: The legend graph

The legend graph draws a legend listing the name and color threshold table of each variable attached to the graph. The legend is static, as it is drawn once and is not updated while running. The legends appear as a centered column in the graph. There is no context except the outline.

To use the legend graph in conjunction with other graphs being used to display data:

1. Select the data display graph, select **Edit**, and turn off the **Legend** option on the **Basic** tab of the **Graph Properties** dialog.
2. Attach the same variables to both the data display graph and the legend graph.
3. Position the legend graph where you want the legend of the data display graph to appear.

5.11.2.5 Contour graphs

Contour graphs display a contour plot of a matrix variable. Matrix element values are located at the midpoints of the grid, with intermediate values mapped between one value and another. Contour lines are drawn through all points where the values correspond to the threshold values in the color threshold table.

Only one sample is displayed at a time. These graph types work best with matrix variables and with a color threshold table.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. The time axis tick label displays the iteration number centered below the horizontal axis.

Contour graph
The maximum number of variables for the contour graph is 1.

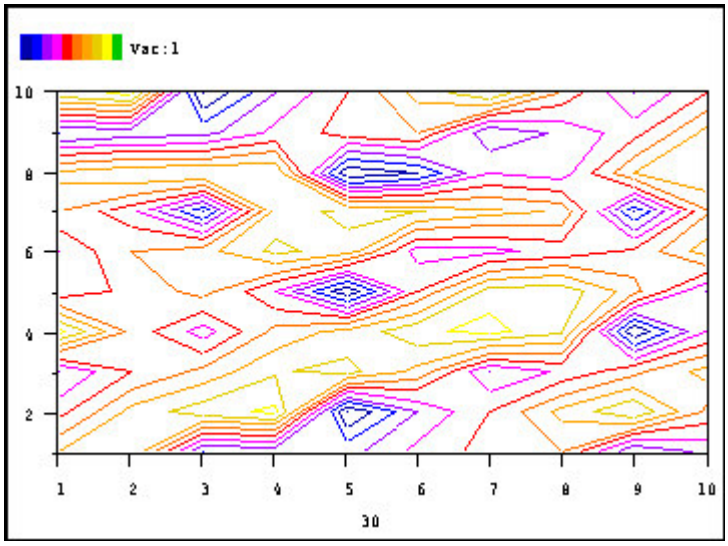


Figure 38: The contour graph

The contour graph displays a contour plot. If there is no color threshold table, the graph calculates two or more equidistant contours, depending on the size of the data area.

Filled contour graph
The maximum number of variables for the filled contour graph is 1.

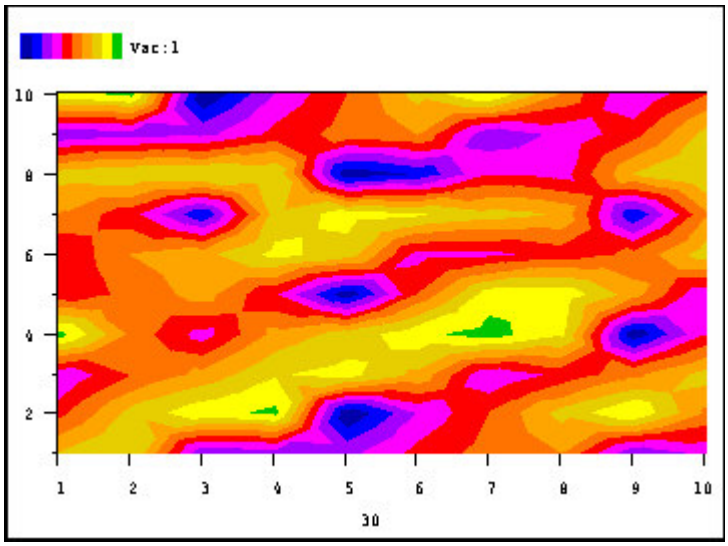


Figure 39: The filled contour graph

The filled contour graph displays a filled contour plot. The areas between contour lines are filled with the corresponding color threshold color.

5.11.2.6 Face graph

The maximum number of variables for the face graph is 5.

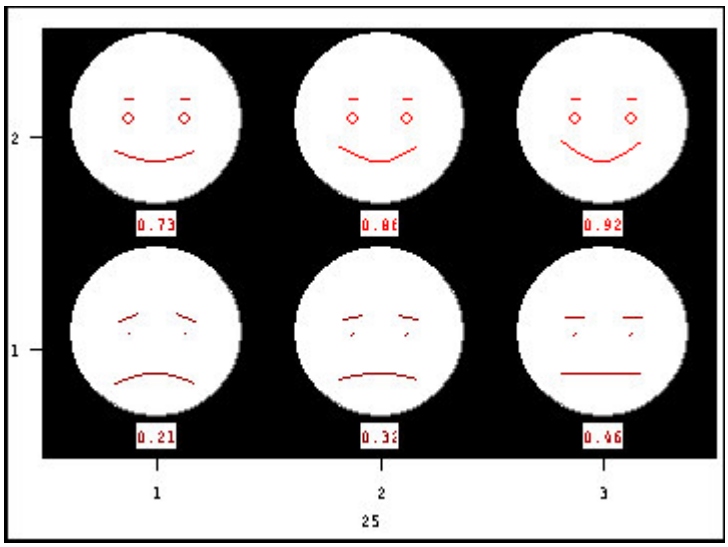


Figure 40: The face graph

The face graph draws stylized faces with eyes, eyebrows, and mouth. The greater the value, the more the corners of the mouth point up, the larger the eyes become, and the more the eyebrows rise. The lower the value, the more the corners of the mouth point down, the smaller the eyes become, and the more the eyebrows tilt down.

The face graph can display scalar, vector, or matrix data. Only one data sample is displayed at a time.

The shape of the variable determines the number of faces displayed.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. For matrix data, both the horizontal and vertical axis ticks and tick labels apply. For vector

data, only the horizontal ticks and tick labels apply. For scalar data, neither the horizontal nor vertical axis ticks and tick labels apply.

The time axis tick label displays the iteration number centered below the horizontal axis. The value axis tick label displays the variable value centered below each face.

The color of the features is determined by the color or color threshold table associated with the variable.

The face graph works best with one variable. Multiple variables are displayed on top of each other, making the values difficult to distinguish.

5.11.2.7 High low graph

High low graphs display the first three variables as a vertical line with a close line. Additional variables display as open lines, bar graphs, or line graphs.

The color of the vertical bar is determined by the color or color threshold table associated with the first variable.

High low graphs can display scalar, vector, or matrix data. A vector variable whose length equals the number of samples updates most efficiently. Matrix data is the least effective.

High low graphs can display an unlimited number of data samples.

The maximum number of variables for the high low graph is 4.

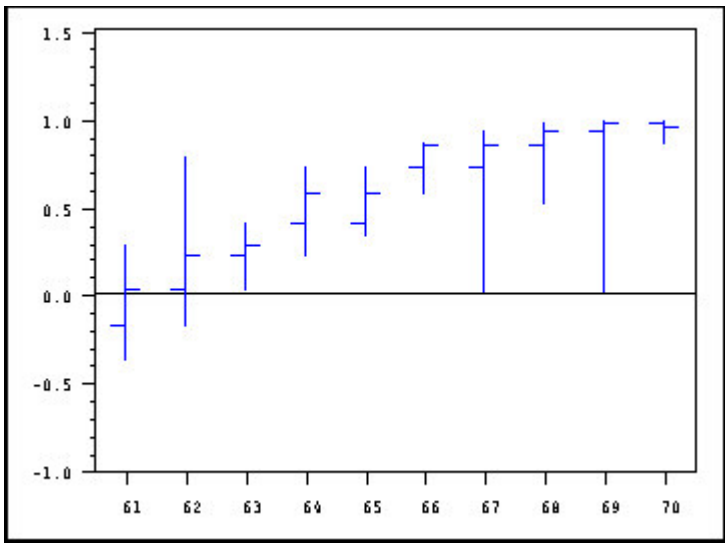


Figure 41: The high low graph

Displays four variables as a high-low-open-close graph.

In each time slot, the formatter displays a vertical line with the upper and lower end points representing the highest and lowest data values. If three variables are used, one horizontal line marks the intermediate value. If four values are used, two lines mark the intermediate values, in the manner of open and close market values. In this case, the vertical line is located in the center of the time slot, with the two horizontal lines on either side.

If two variables are used, they determine the high and low values of the vertical line, and the second variable determines the value of the horizontal line. If only one variable is used, only a horizontal line appears.

For the high low graph to be meaningful, all variables should have the same range.

High low bar

The maximum number of variables for the high low bar is 13.

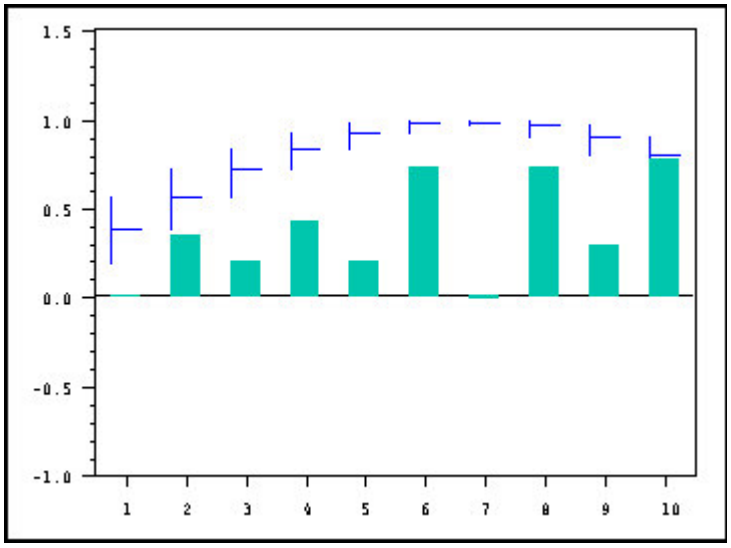


Figure 42: The high low bar

The high low bar displays the first three variables as a high-low-close graph and all subsequent variables as bars.

If all variables have the same range, only the left value axis is displayed. If the fourth variable (the first variable displayed as a bar) has a different range from the first variable, a second value axis is displayed on the right.

This graph type only lists the first variable in the legend. Additional variables do not appear in the legend. To show the legend for all variables, turn the legend off in the high low bar graph and use the legend graph type to display the variables.

If only two variables are used, the first variable is displayed as the close line (the horizontal line) of the high-low-close graph and the second as a bar chart. If the two ranges are different, both value axes are displayed.

When fewer than four variables are specified, this graph type uses the last variable for the bar and the remaining variables for the high low graph.

The maximum number of variables for the high low portion of the graph is 3, the maximum number for the bar portion is 10.

High low line

The maximum number of variables for the high low line is 13.

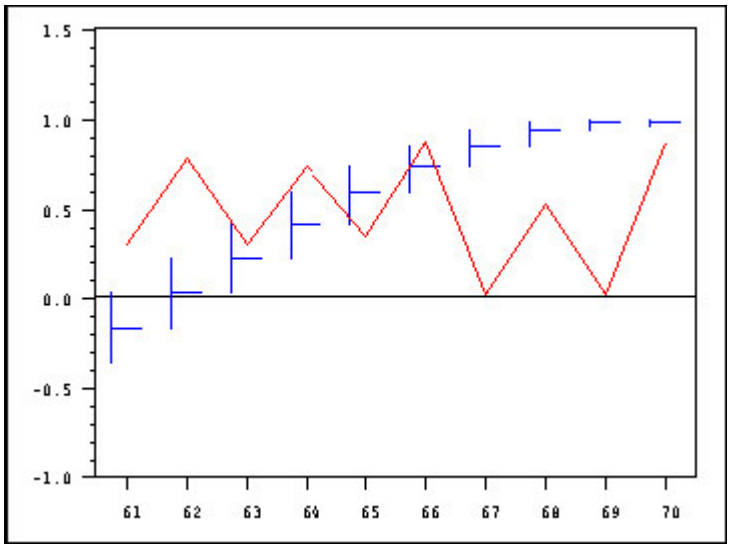


Figure 43: The high low line

The high low line graph displays the first three variables as a high-low-close graph and all subsequent variables as lines.

If all variables have the same range, only the left value axis is displayed. If the fourth variable (the first variable displayed as a line) has a different range from the first variable, a second value axis is displayed on the right.

This graph type only lists the first variable in the legend. Additional variables do not appear in the legend. To show the legend for all variables, turn the legend off in the High Low Line graph and use the legend graph type to display the variables.

If only two variables are used, the first variable is displayed as the close line (the horizontal line) of the high-low-close graph and the second as a line. If the two ranges are different, both value axes are displayed.

When fewer than four variables are specified, this graph type uses the last variable for the line and the remaining variables for the high-low graph.

The maximum number of variables for the High Low portion of the graph is 3, the maximum number for the Line portion is 10. The minimum number of samples allowed is 2.

5.11.2.8 Instrument graphs

Instrument graphs display a single instrument.

Instrument graphs only display scalar data.

The data display color is determined by the color or color threshold table associated with the variable.

Artificial Horizon Graph
Maximum Number of Variables: 4.

This graph type displays a horizon line, runway, the representation of air plane wings, and a track circle within a 360-degree dial-shaped graph. Four variables can be used. Their values determine the roll, pitch, roll error, and pitch error respectively. Roll error and pitch error are optional.

The first variable value determines the roll angle. The roll value is represented by rotation of the horizon, sky, runway, pitch axis, and a red arrowhead indicator. The indicator points to the

current roll value on the circumference of the circle. A positive roll value rotates these objects counter-clockwise, a negative value rotates them clockwise.

The circumference of the circle is always divided -180 to 180 with zero at the top. If the range of the roll variable values is smaller than -180 to 180, the tick labels are limited correspondingly. If the range is set to be greater than -180 to 180, values wrap around the dial. For example, a value of 240 appears as -120. Values outside the range are clipped to the range limits.

To turn the roll value ticks and tick labels on and off, use the **X Axis** tab of the **Graph Properties** dialog.

The second variable determines the pitch angle. The pitch value is represented by the position of the horizon with respect to the pitch axis. Movement of the horizon changes the proportion of the sky area to the ground area. Positive values move the horizon down the scale, drawing more sky than ground, negative values move it up the scale, drawing more ground than sky.

The maximum pitch value is mapped to the bottom of the scale and produces 100% sky and 0% ground. The minimum pitch value is mapped to the top of the scale and produces 0% sky and 100% ground. Zero is located at the mid-point of the pitch axis.

The pitch angle tick marks are drawn along the pitch axis. To turn the pitch value ticks and tick labels on and off, use the **X Axis** tab of the **Graph Properties** dialog.

The third variable value determines the roll error. The roll error value is represented by a short yellow vertical line perpendicular to the dial's horizontal axis. The roll error variable uses the range of the roll variable. The range is mapped to the horizontal axis, with zero in the center. Positive values move the line proportionally to the left, and negative values move it to the right.

The fourth variable value determines the pitch error. The pitch error value is represented by a short yellow horizontal line perpendicular to the dial's vertical axis. The pitch error variable uses the range of the pitch variable. The range is mapped to the vertical axis, with zero in the center. Positive values move the line down proportionally, and negative values move it up.

If a variable range is not symmetrical around zero, this graph type interprets it as if it were, using the larger absolute value for both the positive and negative limits. For example, a variable range of -45 to 90 is interpreted as -90 to 90.

Controller Graph
Maximum Number of Variables: 12

Draws a combination of bar graphs and point graphs. Displays only one data sample at a time.

Each variable displays as either a vertical bar or as a point, as determined by the variable's graph marker type. If the marker is null (0), the variable is represented by a vertical bar proportional in height to the value of the variable. If the marker is a symbol, the variable is represented by a marker with its center point at a vertical position proportional to the value of the variable.

This graph type can be used either with or without range bars. To turn range bars on, turn on the **Ticks** option on the **X Axis** tab of the **Graph Properties** dialog. Note that this graph type does not support the time axis, so only the value axis is labeled.

If range bars are used, the first two variables supply the values for the range bars and the remaining variables provide the values for the graphs. Therefore, three variables are required when using range bars. If range bars are not used, all variables provide values for the graphs and only one variable is required. Range bars can be used a s a visual cue that the data is inside or outside a critical range.

The data values used for range bars should be constants.

Each variable can have a separate dynamic color threshold table. If range bars are used, the first two color thresholds of every color threshold table are set equal to the range bar values and subsequent threshold values are ignored. If the range bars move, so do the color threshold values.

Each bar or symbol displays in a single solid color. If the variable value crosses a threshold value, the whole bar or symbol is redrawn in the new color.

Fader
Maximum Number of Variables: 1

Displays the variable value in a format that resembles a stereo equalizer control. The position of the horizontal bar is proportional to the variable value.

The fader bar color is determined by the color or color threshold table of the variable.

Horizontal Controller Graph
Maximum Number of Variables: 12

Draws a combination of horizontal bar graphs and indicator graphs.

Each variable displays as either a horizontal bar or as a point, as determined by the variable's graph marker type. If the marker is null (0), the variable is represented by a horizontal bar proportional in height to the value of the variable. If the marker is a symbol, the variable is represented by a marker with its center point at a horizontal position proportional to the value of the variable.

This graph type can be used either with or without range bars. To turn range bars on, turn on the **Ticks** option on the **X Axis** tab of the **Graph Properties** dialog. Note that this graph type does not support the time axis, so only the value axis is labeled.

If range bars are used, the first two variables supply the values for the range bars and the remaining variables provide the values for the graphs. Therefore, three variables are required when using range bars. If range bars are not used, all variables provide values for the graphs and only one variable is required. Range bars can be used as a visual cue that the data is inside or outside a critical range.

The data values used for range bars should be constants.

Each variable can have a separate color threshold table. If range bars are used, the first two color thresholds of every color threshold table are set equal to the range bar values and subsequent threshold values are ignored. If the range bars move, so do the color threshold values.

Each bar or symbol displays in a single solid color. If the variable value crosses a threshold value, the whole bar or symbol is redrawn in the new color.

Indicator Graph
Maximum Number of Variables: 10

Displays the current value of a variable as a marker. The horizontal position of the marker is proportional to the variable value.

To choose a new marker, select the variable on the **Variables** tab of the **Graph Properties** dialog and use the **Marker** option. The height of the marker is determined by the height of the graph.

The example shows one sample. Multiple samples are displayed one above the other, starting at the bottom of the graph. The vertical position of each marker represents a new time sample, not a spatial (x,y) value. Scrolls vertically down or wraps around to the bottom of the graph.

Multiple variables can overlap in the same slot space if the markers are large and the values are close together.

Radial Graph
Maximum Number of Variables: 10

This graph type plots a line graph in polar coordinates. The variable value is the distance from the center of the graph to the current point on the line. The graph divides the circle by the number of samples specified and plots that number of values.

The graph wraps in a counter-clockwise direction, starting at the zero-degree line, which is at 3 o'clock.

The current value erases any previous value at that position so the current value is always preceded by a blank space.

The time axis tick label displays the iteration number centered below the graph.

The range of the variable is mapped between a circle around the center of the graph and a line touching the outer edge of the graph. To display ticks for the variable value, turn on the **Ticks** option on the **Y Axis** tab of the **Graph Properties** dialog.

Different colors can be assigned to different lines to make it easier to distinguish between them. The color of each line is determined by the color or color threshold table associated with its variable.

Changing the line color has no effect on a monochrome system.

Changing the line type has no effect.

Radial Graph (No Erase)
Maximum Number of Variables: 10

This graph type plots a line graph in polar coordinates. The variable value is the distance from the center of the graph to the current point on the line. The graph divides the circle by the number of samples specified and plots that number of values. Previous values are not erased, but remain as history.

The graph wraps in a counter-clockwise direction, starting at the zero-degree line.

The range of the variable is mapped between a circle around the center of the graph and a line touching the outer edge of the graph. To display ticks for the variable value, turn on the **Ticks** option on the **Y Axis** tab of the **Graph Properties** dialog.

The time axis tick label displays the iteration number centered below the graph.

Different colors can be assigned to different lines to make it easier to distinguish between them. The color of each line is determined by the color or color threshold table associated with its variable.

Changing the line color has no effect on a monochrome system.

Changing the line type has no effect.

If the graph is redrawn, only the most recent number of samples specified in the **Samples** option on the **X Axis** tab of the **Graph Properties** dialog are redrawn. Previous values are not preserved.

5.11.2.9 Instrument array graphs

Instrument array graphs display an array of instruments. Instrument graphs can display scalar, vector, or matrix data. The shape of the variable determines the number of boxes displayed.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. For matrix data, both the horizontal and vertical axis ticks and tick labels apply. For vector data, only the horizontal ticks and tick labels apply. For scalar data, neither the horizontal nor vertical axis ticks and tick labels apply.

The time axis tick label displays the iteration number centered below the horizontal axis. The value axis tick label displays the variable value centered below each instrument.

The data display color is determined by the color or color threshold table associated with the variable.

For all instrument graphs except the digits graph, the range of the variable maps to the circumference.

To display ticks around the circumference, turn on the **Ticks** option on the **Y Axis** tab of the Graph of the Properties dialog.

Clock
The maximum number of variables for the clock is 2.

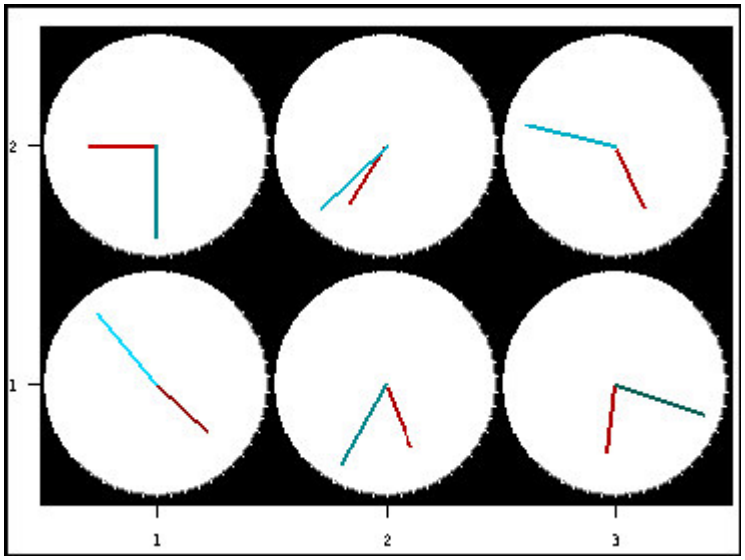


Figure 44: Simulated analog clock graph

Draws a simulated analog clock. The hour hand displays the value of the first variable and the minute hand displays the value of the second variable. The shape of the second variable determines the number of clocks displayed.

Dial 360
The maximum number of variables for dial 360 is 1.

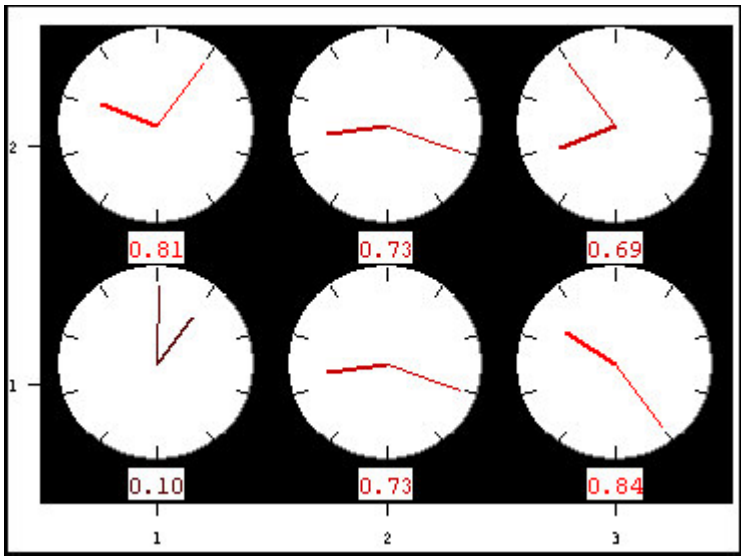


Figure 45: Dial encompassing 360 degrees

The variable is represented by two hands pointing to the corresponding value. The small hand encodes the most significant digit and the large hand encodes the complete value. For example, if the data range is [0,1000], a value of 550 is displayed by a small hand at 500 and a large hand at 550.

The dial displays ten tick marks. Variable range values should be powers of 10.

Dial graph
The maximum number of variables for the dial graph is 5.

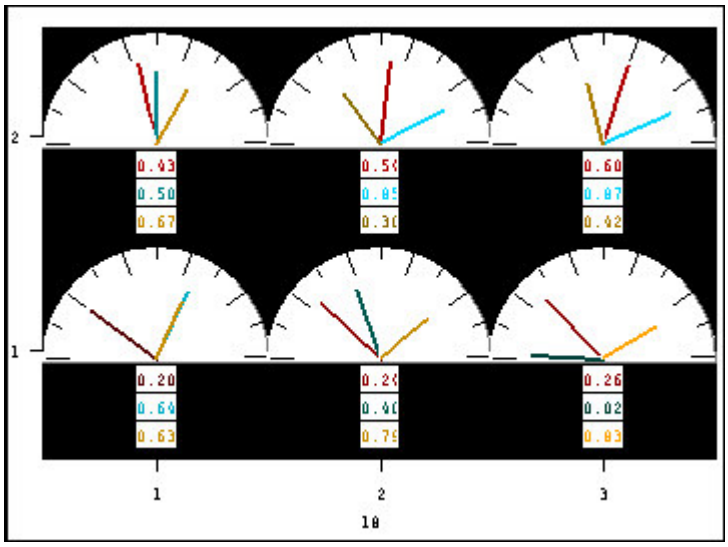


Figure 46: Dial encompassing 180 degrees

Draws a dial encompassing 180 degrees, with the lowest value at the left and the highest value at the right. The variable is represented by a needle pointing to the corresponding value.

Multiple variables are represented by different colors and different length needles. The difference in colors is more distinctive than the difference in lengths.

Tick marks and tick labels are drawn in the foreground color of the graph.

Dials with history

The maximum number of variables for dials with history is 5.

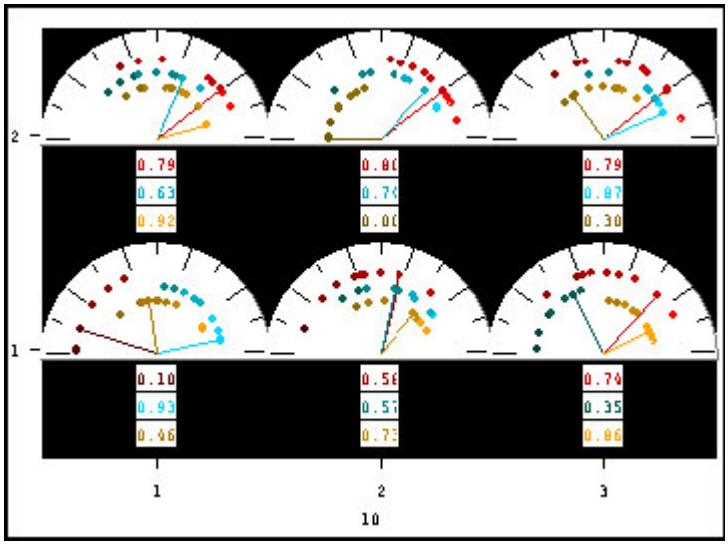


Figure 47: Dials with history

Dials with history draws a dial with the lowest value at the left and the highest value at the right. The variable is represented by a needle pointing to the corresponding value. A dot appears at the tip of each dial needle. As the value changes, the graph leaves the dot of each value as a history of the variable values. The number of samples specifies the number of dots displayed.

If the number of samples is 1, no history is shown.

Tick marks and tick labels are drawn in the foreground color of the graph.

The range of the variable maps to the dial edge. To display ticks around the edge of the dial, turn on the **Ticks** option on the **Y Axis** tab of the **Graph Properties** dialog.

Digits graph

The maximum number of variables for the digits graph is 5.

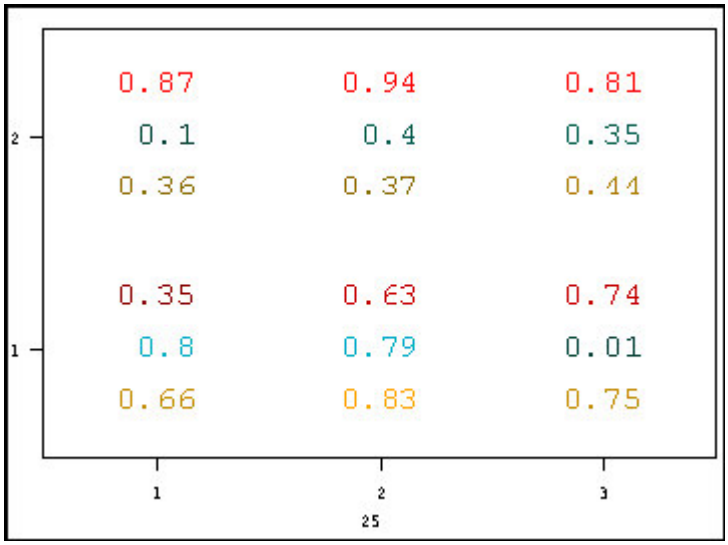


Figure 48: The digits graph

The digits graph displays data as an array of numbers that displays the actual value of the data in the variable. The digits are displayed in the largest text size that fits into the graph. Adding text dynamics to text objects can produce the similar results.

This graph type uses the data variable range to determine the number of significant digits displayed using the following criteria:

- three (sometimes four) digits
- the number of significant digits in the variable's minimum value
- the number of significant digits in the variable's maximum value

For example:

If the range is:	It must allow at least:	If the range is:	It must allow at least:
[0,1]	4 digits	[0,1001]	4 digits
[0,10]	4 digits	[0,10001]	5 digits
[0,100]	3 digits	[0,100001]	6 digits

If the digits graph shares a variable with an input object, the range of the digits graph must match the range of the input variable.

The **C Format** option on the **Basic** tab of the **Graph Properties** dialog lets you specify the C format for displaying your data. The conversion character must be preceded by a % sign. Valid conversion characters and the type of data they indicate are:

- s: character string
- c: single character
- f: float, double, decimal notation
- e, E: float or double converted to scientific notation
- g, G: converts to e, E or f, depending on whether the graph allows for the number of decimals specified
- d, i: integer converted to decimal
- o: unsigned octal
- u: unsigned decimal
- x, X: unsigned hexadecimal
- p: address

There can only be one conversion character per format string.

When a g, G format is specified in the form x.y, y specifies the number of decimal places, not the total width of the field.

Other characters in the string appear as the user enters them. These include \n for a newline, \t for a tab, and \octal_digits for special characters.

If the data is changed, make sure the conversion character matches the format of the new data. If the conversion character does not match the format of the data, Display Builder displays an error message when the view drawing is run and uses a default format.

The **Position** option lets the user specify the text justification within the graph.

Knob

The maximum number of variables for the knob is 1.

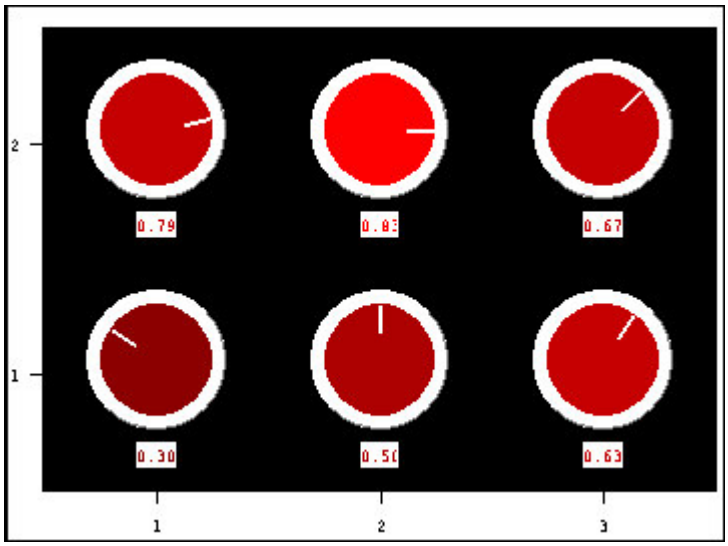


Figure 49: The knob graph

Draws a knob. Only one data sample is displayed at a time.

The variable is represented by a line pointing to the corresponding value. The current value appears as a white line. If the user changes the color of the knob, the line may change colors.

If the value tick labels are on, the data values are displayed digitally below each knob.

Meter
The maximum number of variables for the meter is 1.

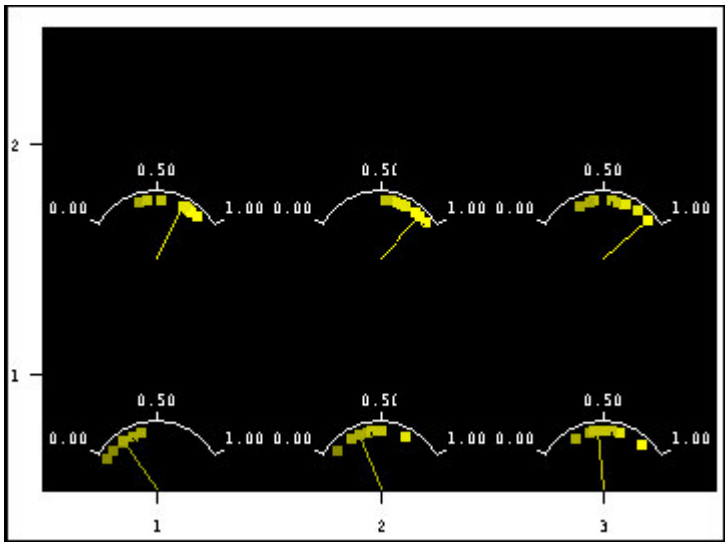


Figure 50: Meter graph with the lowest value on the left and the highest on the right

Draws a meter with the lowest value at the left and the highest at the right. The variable is represented by a needle pointing to the corresponding value.

When the number of samples is greater than one, a dot appears at the tip of the meter needle. As the value changes, the graph leaves the dot at each value as a history of the values. The number of samples is unlimited. The number of samples specifies the number of dots displayed.

5.11.2.10 Line graphs

Line graphs display data values using lines. Additional variables are displayed as lines, bars, points, or additional graphs.

The color of each line or other symbol is determined by the color or color threshold table associated with the variable. Each line is divided into sections of different colors depending on the variable's value and the colors assigned in the threshold table.

Line graphs display scalar, vector, or matrix data. A vector variable whose length equals the number of samples updates most efficiently. A matrix whose total number of elements equals the number of samples also updates efficiently.

Line graphs can display an unlimited number of data samples.

Most line graphs scroll from right to left, or wrap around to the left edge of the graph. The exceptions are the waterfall and raster waterfall graphs, which scroll down, or wrap around to the bottom of the graph.

Different line types can be used to make it easier to distinguish between different variables. To change variable line types, select the variable on the Variables tab of the **Graph Properties** dialog and use the **Line Type** option.

For graphs that use markers, different markers can be used to make it easier to distinguish between different variables. To change markers, select the variable on the **Variables** tab of the **Graph Properties** dialog and use the **Marker** option.

Filled line
The maximum number of variables for the filled line is 10.

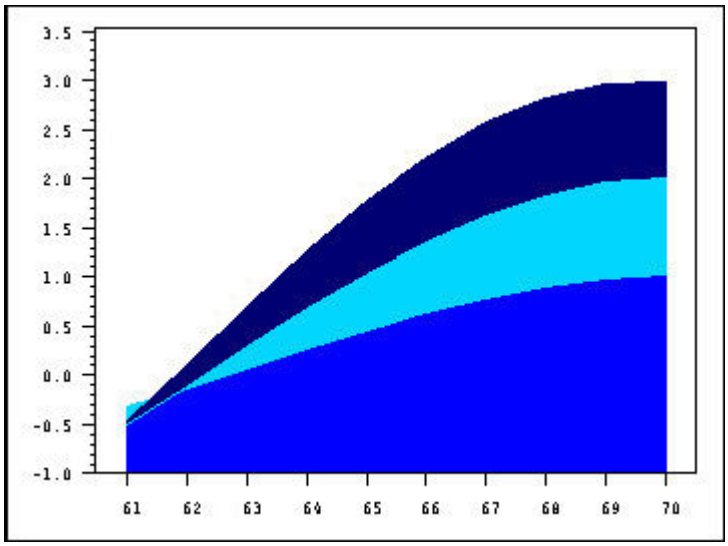


Figure 51: Line graph with filled lines

The filled line draws a line graph for each variable and fills below the line with the variable color. The line graphs are stacked vertically, adding each variable value to the sum of the values beneath it. The first variable is on the bottom, the last variable on the top. The height of the line graph is proportional to the sum of the values of all the variables.

The range of the graph equals the sum of the ranges of the attached variables. The variables should be either all logarithmic or all linear.

For the filled line graph to be meaningful, all variables should have the same range.

The color of the area below each line is determined by the color associated with the variable. If the variable has a color threshold table, the area is divided into sections of different colors to match the threshold table.

Filled line distribution
The maximum number of variables for the filled line distribution is 10.

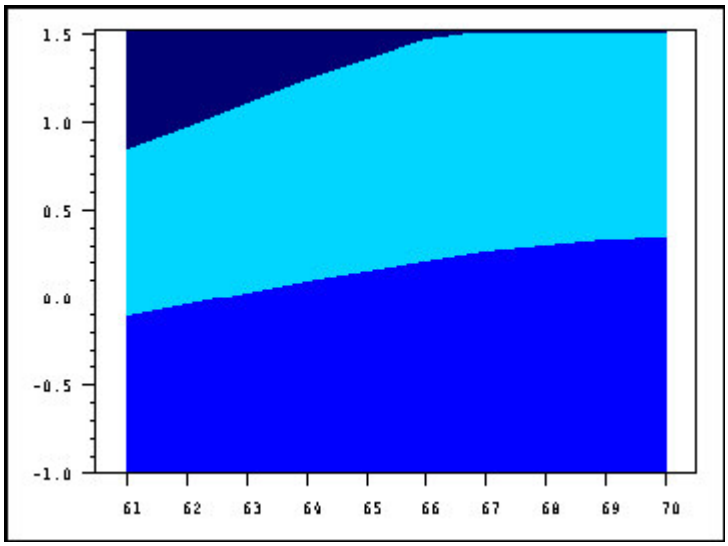


Figure 52: Filled line distribution

The filled line distribution draws a filled line graph in which the range of the graph equals the range of the attached variables.

The filled line distribution graph is useful for displaying statistical distribution.

All variables must have the same range.

For this graph type to be meaningful, the sum of the variable values for each sample should not exceed the maximum range value. For example, if a graph has three variables with a range of 0 to 10, the range of the graph is 0 to 10. A given sample of the three variables might have values of 2, 3, and 5 or 2, 3, and 4 but not 2, 3, and 6.

See also the filled line graph.

Line graph
The maximum number of variables for the line graph is 10.

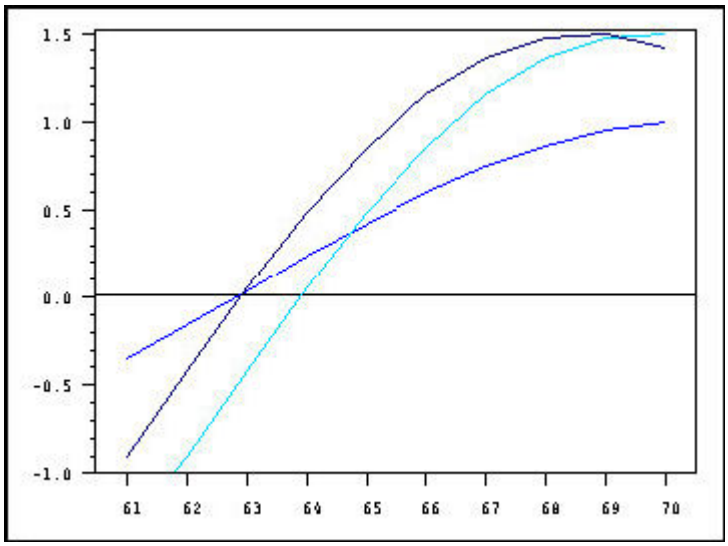


Figure 53: Line graph

The line graph draws a line graph for each variable, starting at the left edge of the graph.

Point-line
The maximum number of variables for the point-line is 11.

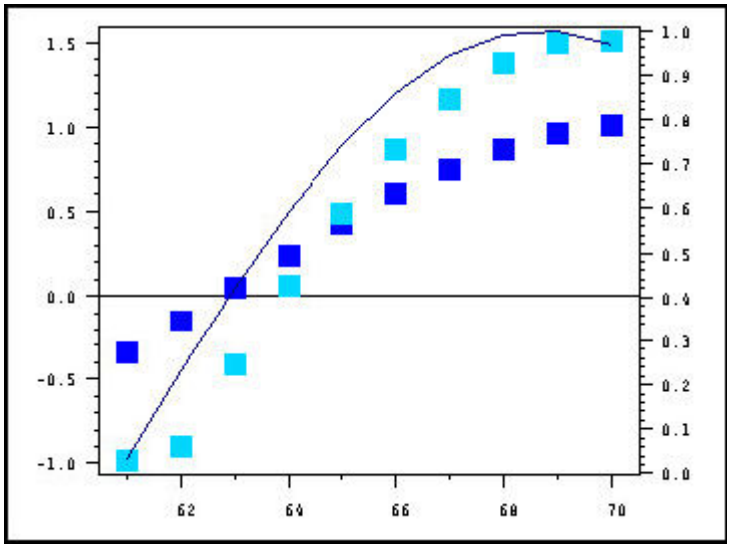


Figure 54: Point-line chart

The point-line graph displays all variables as points except for the last variable, which displays as an independent line graph.

If there is only one variable, both the line and the points use the same variable and the line is superimposed on the points.

For the variable displayed as a line graph, if a value is outside the given range, the line is drawn at the correct slope, but is clipped to the bounds of the data area. For variables displayed as points, values outside the given range are displayed at the closest value within the range.

If all the variables have the same range, only the left value axis is displayed. If the last variable (displayed as the line graph) has a different range from the first variable, a second value axis is displayed on the right.

All point variables appear in the legend. The line variable does not appear in the legend. To list additional variables in the legend, turn the legend off in the point-line graph and use the legend graph to display the variables.

The minimum number of samples allowed is 2.

Raster strip chart
The maximum number of variables for the raster strip chart is 10.

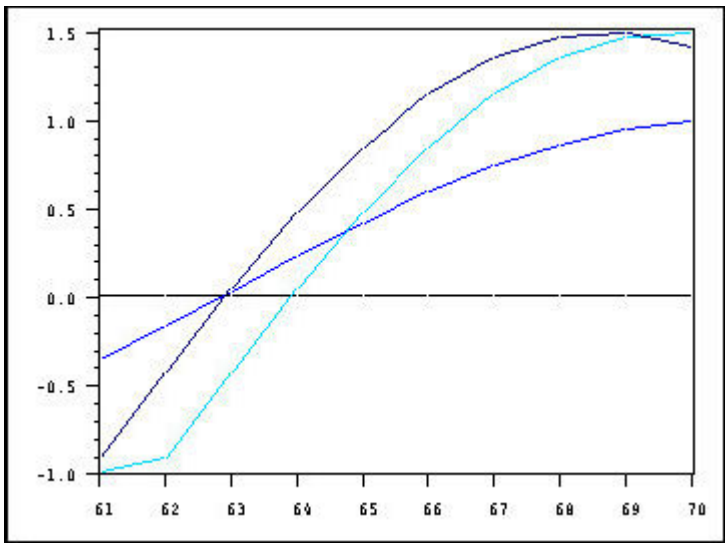


Figure 55: Raster strip chart

The Raster strip chart plots a line graph that begins at the right edge of the graph and scrolls toward the left of the graph. The most recent value appears at the right edge of the graph and the history shifts continually to the left.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The strip chart is slower than the line graph because it redraws the entire plot and time axis for each sample. One way to increase the speed is to use the raster strip chart. The raster strip chart is faster than a strip chart if you are plotting a lot of samples and numerous variables. The raster strip chart takes a raster image of the current data and shifts the image before plotting each new sample.

The entire raster strip chart must be visible in the display area to draw properly in the **Run** and **Prototype** menus.

If your display device does not support raster operations, the raster strip chart behaves like the strip chart.

Plotting the strip chart overloads the plotter. Before sending the strip chart to the plotter, convert to a line graph.

Raster waterfall
The maximum number of variables for the raster waterfall is 10.

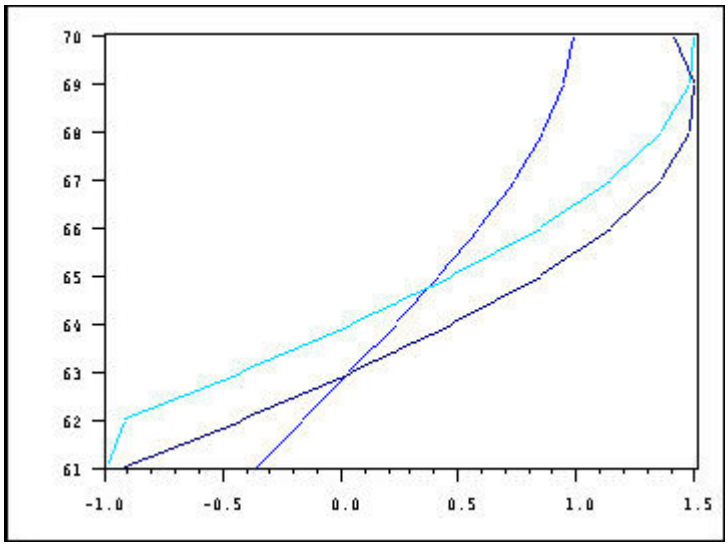


Figure 56: Raster waterfall chart

The raster waterfall plots a strip chart that begins at the top edge of the graph and scrolls down. The most recent value appears at the top edge of the graph and the history shifts continually down.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The time axis is displayed on the left side of the graph and the value axis is displayed at the bottom.

The raster waterfall graph is faster than the waterfall graph if the user is plotting a lot of samples and numerous variables. The raster waterfall takes a raster image of the current data and shifts the image before plotting each new sample.

The entire raster waterfall must be visible in the display area to draw properly in the **Run** and **Prototype** menus.

If the user's display device does not support raster operations, the raster waterfall behaves like the waterfall.

Plotting the raster waterfall overloads the plotter. If the data must be plotted, use a line graph.

Stacked filled line graph
The maximum number of variables for the stacked filled line graph is 16.

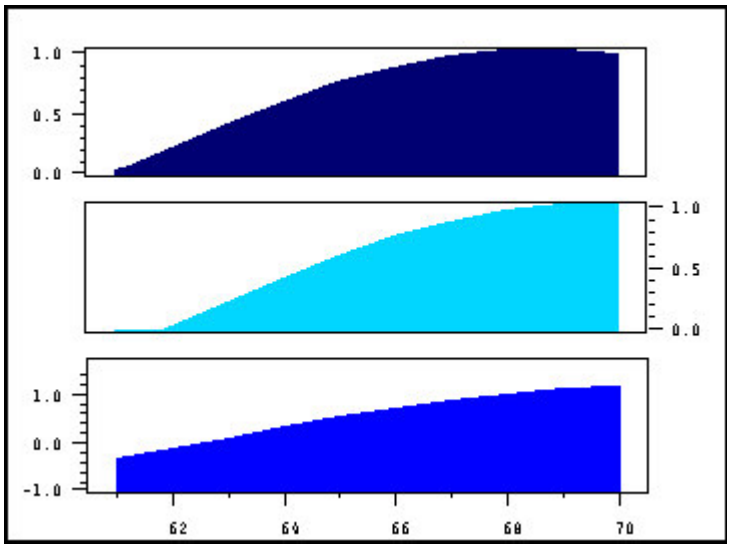


Figure 57: Stacked filled line graph

The stacked filled line graph displays each variable as a filled line graph, stacking each graph above the previous one.

The value axis of each graph is displayed on alternate sides of the graphs, starting at the left side of the bottom graph.

This graph type displays a single title and a single legend for the stack of graphs. The legend lists all of the variables in the stack of graphs.

Stacked line graph
The maximum number of variables for the stacked line graph is 16.

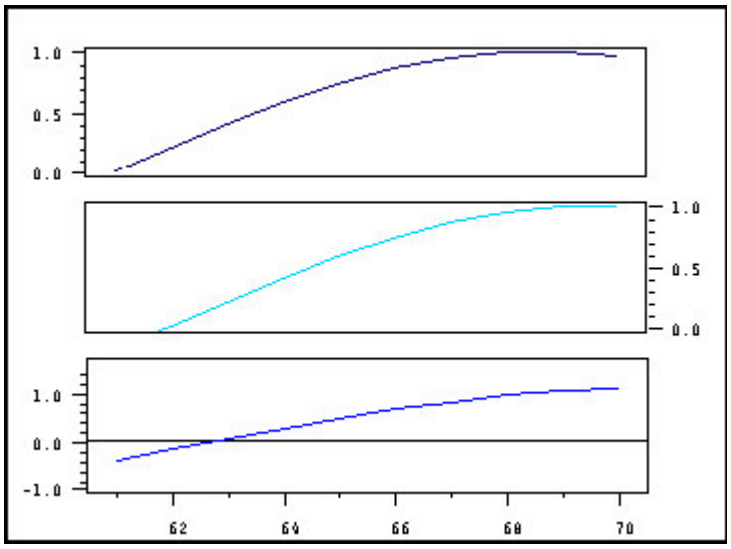


Figure 58: Stacked line graph

The stacked line graph displays each variable as a line graph, stacking each graph above the previous one.

The value axis of each graph is displayed on alternate sides of the graphs, starting at the left side of the bottom graph.

This graph type displays a single title and a single legend for the stack of graphs. The legend lists all of the variables in the stack of graphs.

Stacked strip chart

The maximum number of variables for the stacked strip chart is 16.

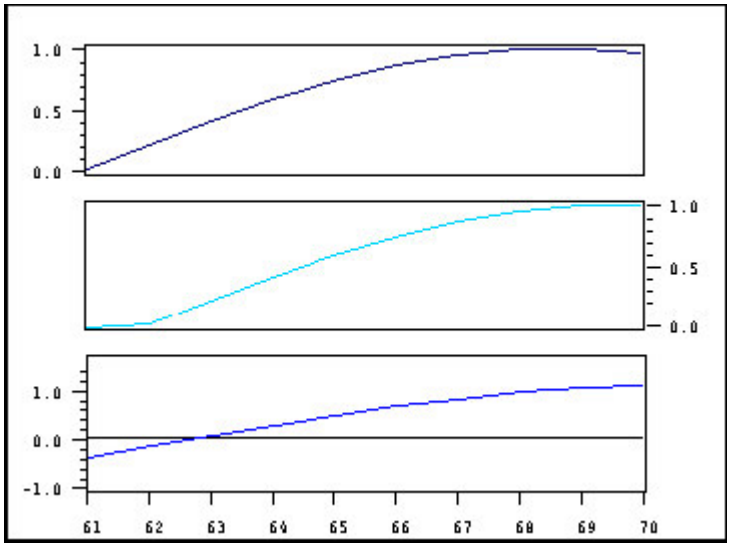


Figure 59: Stacked strip chart

The stacked strip chart displays each variable as a strip chart, stacking each graph above the previous one. Each graph plots a line graph that begins at the right edge of the graph and scrolls toward the left of the graph. The most recent value appears at the right edge of the graph and the history shifts continually to the left.

The value axis of each graph is displayed on alternate sides of the graphs, starting at the left side of the bottom graph.

This graph type displays a single title and a single legend for the stack of graphs. The legend lists all of the variables in the stack of graphs.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The stacked strip chart is slower than a stacked line graph because it redraws the entire plot and time axis for each sample. To increase the speed, turn off the **Tick Labels** option on the **X Axis** tab of the **Graph Properties** dialog.

Plotting strip charts overloads the plotter. Before sending a stacked strip chart to the plotter, convert it to a stacked line graph.

Step graph

The maximum number of variables for the step graph is 10.

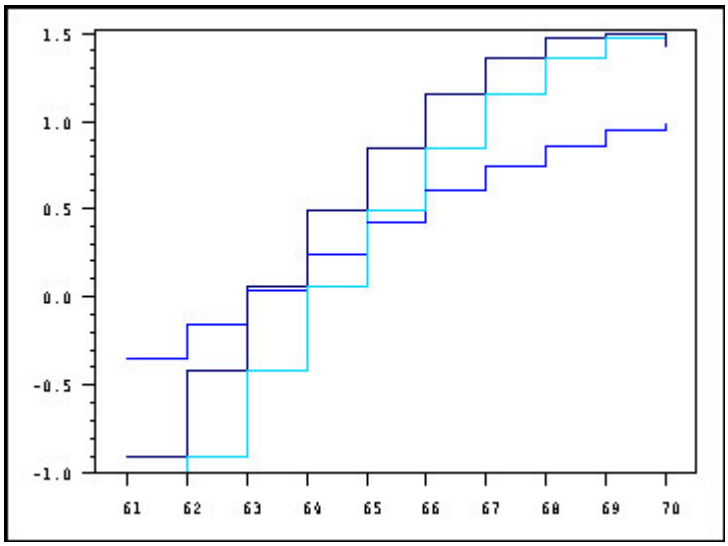


Figure 60: Step graph

The step graph displays each variable element as a horizontal line connected to the adjacent values by vertical lines.

Each horizontal line is plotted together with the following vertical line. Since the vertical line cannot be plotted until the next value is known, values are plotted with a delay of one time slot.

Strip chart
The maximum number of variables for the strip chart is 10.

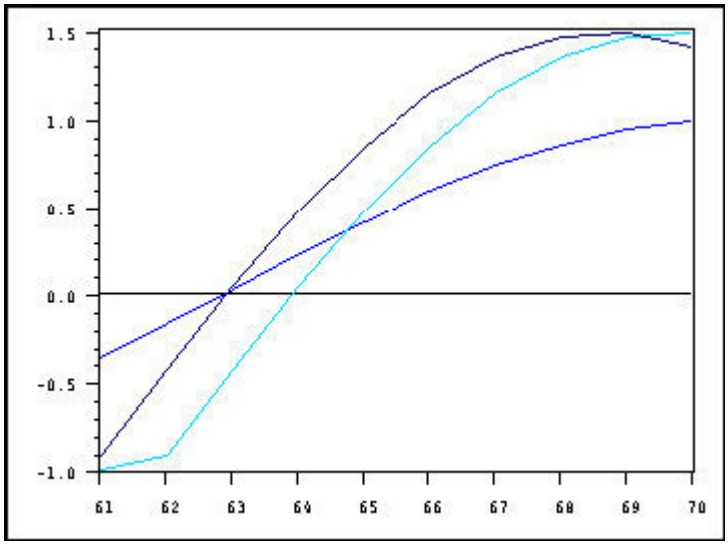


Figure 61: Strip chart

The strip chart plots a line graph that begins at the right edge of the graph and scrolls toward the left of the graph. The most recent value appears at the right edge of the graph and the history shifts continually to the left.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

This graph type is slower than a line graph because it redraws the entire plot and time axis for each sample. To increase the speed, turn off the **Tick Labels** option on the **X Axis** tab of the **Graph Properties** dialog. Another way to increase speed is to use the raster strip chart. A strip

chart may be faster than a raster strip chart when only plotting a small number of samples and variables. The user should compare the speeds using their own data.

Plotting a strip chart overloads the plotter. Before sending a strip chart to the plotter, convert it to a line graph.

Vertical raster strip chart
The maximum number of variables for the vertical raster strip chart is 10.

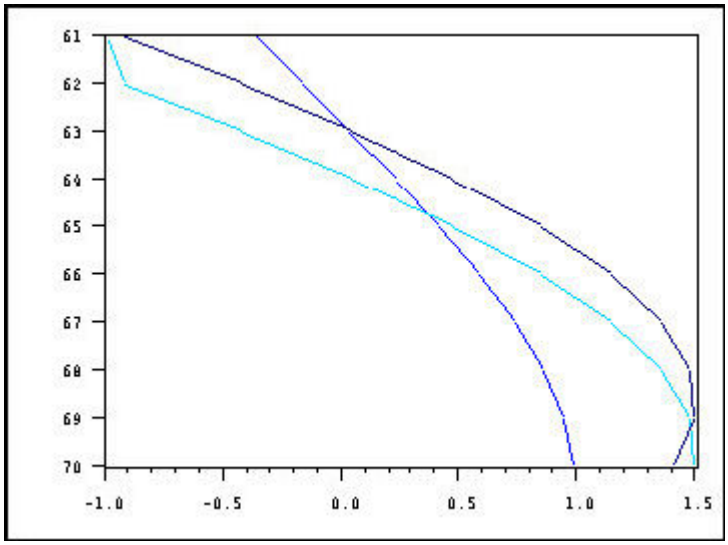


Figure 62: Vertical raster strip chart

The vertical raster strip chart plots a strip chart that begins at the bottom edge of the graph and scrolls up. The most recent value appears at the bottom edge of the graph and the history shifts continually up.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The time axis is displayed on the left side of the graph and the value axis is displayed at the bottom.

The vertical raster strip chart is faster than a vertical strip chart when plotting a lot of samples and numerous variables. The vertical raster strip chart takes a raster image of the current data and shifts the image before plotting each new sample.

The entire vertical raster strip Chart must be visible in the display area to draw properly in the **Run** and **Prototype** menus.

If the user's display device does not support raster operations, the vertical raster strip chart behaves like the vertical strip chart.

Plotting the vertical raster strip chart overloads the plotter. If the data must be plotted, use a line graph.

Vertical strip chart
The maximum number of variables for the vertical strip chart is 10.

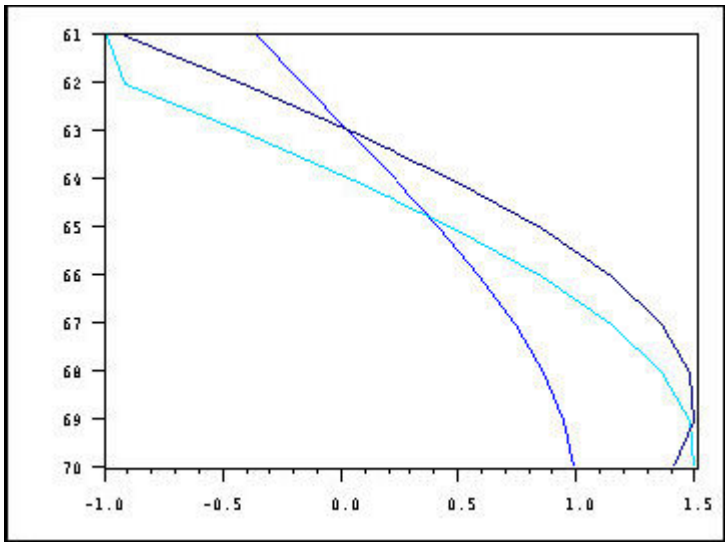


Figure 63: Vertical strip chart

The vertical strip chart plots a strip chart that begins at the bottom edge of the graph and scrolls up. The most recent value appears at the bottom edge of the graph and the history shifts continually up.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The time axis is displayed on the left side of the graph and the value axis is displayed at the bottom.

This graph type redraws the entire plot and time axis for each sample. To increase the speed, turn off the **Tick Labels** option on the **X Axis** tab of the **Graph Properties** dialog. Another way to increase speed is to use the vertical raster strip chart. The vertical strip chart may be faster than the vertical raster strip chart when only plotting a small number of samples and variables. The user should compare the speeds using their own data.

Plotting the vertical strip chart overloads the plotter. If the data must be plotted, use a line graph.

Waterfall
The maximum number of variables for the waterfall is 10.

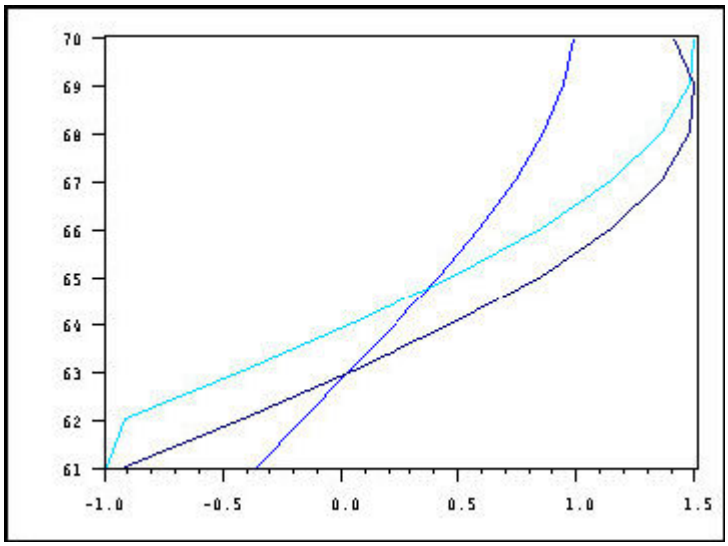


Figure 64: Waterfall graph

The waterfall graph plots a strip chart that begins at the top edge of the graph and scrolls down. The most recent value appears at the top edge of the graph and the history shifts continually down.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The time axis is displayed on the left side of the graph and the value axis is displayed at the bottom.

The waterfall graph redraws the entire plot and time axis for each sample. To increase the speed, turn off the **Tick Labels** option on the **X Axis** tab of the **Graph Properties** dialog. Another way to increase speed is to use the raster waterfall graph. The waterfall graph may be faster than the raster waterfall graph when only plotting a small number of samples and variables. The user should compare the speeds using their own data.

Plotting the waterfall overloads the plotter. If the data must be plotted, use a line graph.

5.11.2.11 Point chart

The maximum number of variables for the point chart is 10.

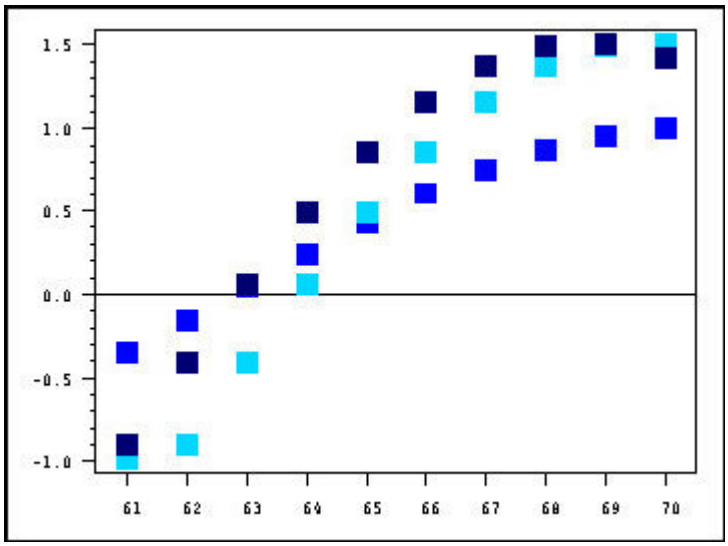


Figure 65: Point chart

The point chart displays the current value of a variable as a marker. The height of the marker is proportional to the variable value.

The point chart can display scalar, vector, or matrix data. A vector variable with a length equal to the number of samples updates most efficiently. A matrix whose total number of elements equals the number of samples also updates efficiently.

The point chart can display an unlimited number of data samples.

Different markers can be used to make it easier to distinguish between different variables. To change markers, select the variable on the **Variables** tab of the **Graph Properties** dialog and use the **Marker** option.

The color of each marker is determined by the color or color threshold table associated with its variable.

The point chart scrolls from right to left or wraps around to the left edge of the graph.

The horizontal position of each marker represents a new time sample, not a spatial (x,y) value. To plot spatial (x,y) data, use the impulse graph.

A vector variable whose length equals the number of samples updates most efficiently. A matrix whose total number of elements equals the number of samples also updates efficiently.

5.11.2.12 Primitive graphs

Primitive graphs display an array of geometric shapes.

Primitive graphs can display scalar, vector, or matrix data. The shape of the variable determines the number of shapes displayed.

Only one data sample is displayed at a time.

These graph types work well with color thresholds. The color of the shape(s) is determined by the color or color threshold table of the first variable.

Primitive graphs have no context except for the outline.

Box
The maximum number of variables for the box graph is 3.

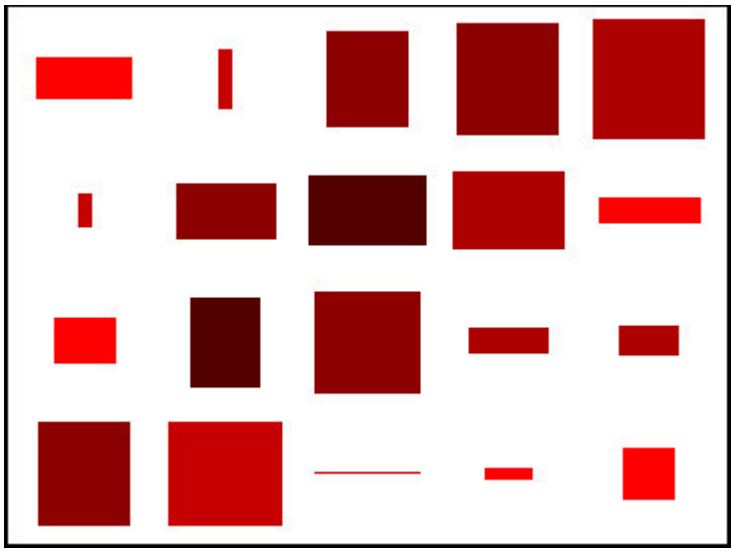


Figure 66: Box graph

The box graph draws a rectangle using up to three variables. The first variable determines the color of the rectangle, the second variable determines the width and the third variable determines the height.

Circle
The maximum number of variables for the circle graph is 2.

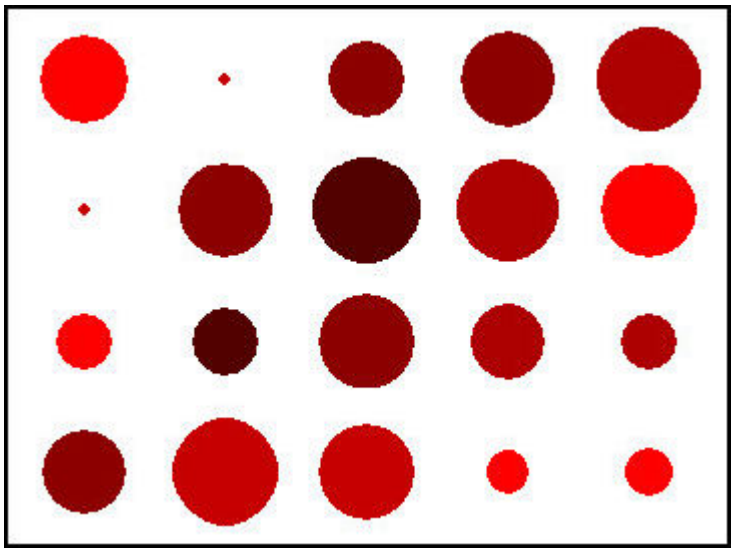


Figure 67: Circle graph

The circle graph draws a circle using one or two variables. The first variable determines the color of the circle and the second variable determines the radius of the circle. As the values change, the color and size of the circles change.

If one variable is used, the graph displays the largest circle possible.

Triangle graph
The maximum number of variables for the triangle graph is 3.

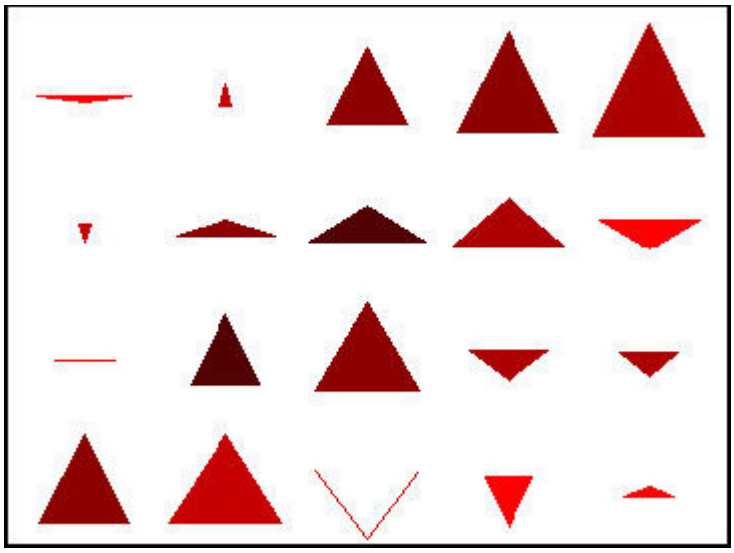


Figure 68: Triangle graph

The triangle graph draws a triangle using up to three variables. The first variable determines the color of the triangle, the second variable determines the width of the base of the triangle and the third variable determines the height of the triangle.

If one variable is used, the figure is the largest triangle possible. If two variables are used, the width of the triangle base is set to the maximum value.

5.11.2.13 Real-time graphs

Real-time graphs display data values with corresponding time stamps. The time axis displays the day counter and time stamp.

The first variable displays as a day counter and the second variable displays as a time stamp representing the time elapsed since the beginning of the day in tenths of milliseconds. The first two variables must be in binary ULONG format. Remaining variables are plotted as lines and can be in any data format. Up to ten variables can be displayed as data.

Typically, the data used in these graphs has already been collected, and the time stamp data represents the times when data was taken rather than the current system time.

The first two variables must both have values that only increase or only decrease. Values for the time stamp (the second variable) do not have to represent regular intervals. The time axis is labeled in regular intervals regardless of the time stamp intervals.

The real-time graphs display data differently from other graphs that display time series data such as bar charts and line graphs. Instead of displaying one data value per slot, the real-time graphs plot the data at the proper place along the time axis based on the value of the time stamp (the second variable). Since each sample of a data variable is paired with the corresponding time stamp, the horizontal gap between data values can vary. Multiple data points can even be plotted at the same point in time if the same time stamp value occurs more than once.

Because of this different approach to plotting data, some features of the real-time graphs are controlled differently from those of other graph types:

- The Time span displayed along the time axis is controlled by a variable range, not by the **Samples** value.
  Time span. The range of the second variable controls the span of time displayed along the time axis. The basic unit is a tenth of a millisecond. For example, a range of [0,100]

displays 100 tenths of milliseconds in 10 intervals of 10 milliseconds each. A range of [0,50] displays 50 tenths of milliseconds in 5 intervals of 10 milliseconds each.

- The Scroll by amount is controlled by the **Samples** and **Scroll By** values.
  Scroll by amount. The graph scrolls only when it must make room for new time stamp data. The **Samples** and **Scroll By** values determine the amount scrolled. For example, if **Samples** is 20 and **Scroll By** is 4, the graph scrolls 20% of the time axis. To eliminate scrolling, make the **Scroll By** value greater than the **Samples** value. In this case, all old data is erased at once and the new data is drawn starting at the left. The **Samples** value can be thought of as an estimate of the number of data points that will be displayed in the time span. Then the **Scroll By** value specifies the estimated number of data points to scroll by.
- The format for the Time axis tick labels is controlled by a variable range.
  Time axis tick labels. The range of the second variable also controls the format for the time axis tick labels. For example, a range of [0,100] displays time axis labels in the format SS.TTT.T (seconds.milliseconds.tenths of milliseconds). A range of [0,10000] displays time axis labels in the format MM:SS.TTT (minutes:seconds.milliseconds). A range of [0,1000000] displays time axis labels in the format HH:MM:SS.
- The number of Data points redrawn after an expose event is controlled by the **Samples** value.
  Data points plotted on an expose. The **Samples** value controls the number of data points that can be redisplayed on an expose event. However, some data may be lost on the redisplay if the graph was displaying more data points than estimated in the **Samples** value.
- The Display direction is controlled by the sign of the day counter (the first variable).
  Display direction. The real-time graphs let the user reverse the direction of the data display. Note that the time stamps must correspond to the graph direction, so if the direction of the graph is changed, the time stamps must reverse direction at the same time. Time stamps must be increasing whenever the graph is going forward, and must be decreasing whenever the graph is going backward.

To reverse the display direction, change the sign of the day counter (the first variable). Changing direction resets the graph and all history is lost.

Real-time line graph

The maximum number of variables for the real-time line graph is 12.

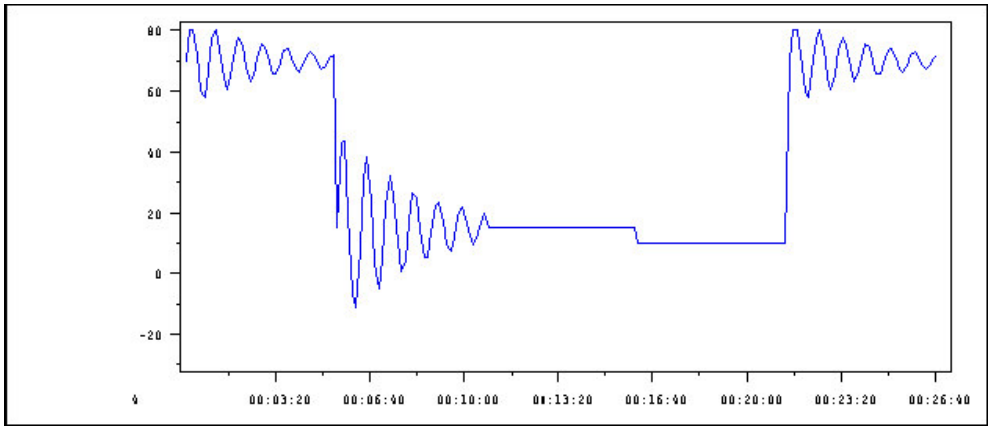


Figure 69: Real-time line graph

The real-time line graph draws a line graph for each variable, starting at the left edge of the graph.

Real-time step graph

The maximum number of variables for the real-time step graph is 12.

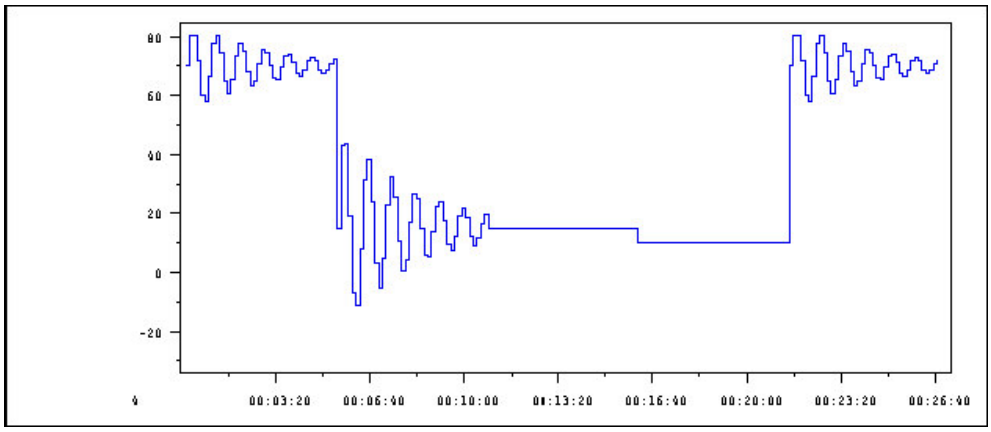


Figure 70: Real-time step graph

Displays each variable element as a Step Graph, stacking each graph above the previous one.

Each horizontal line is plotted together with the following vertical line. Since the vertical line cannot be plotted until the next value is known, values are plotted with a delay.

The real-time step graph displays a single title and a single legend for the stack of graphs. The legend lists all of the variables in the stack of graphs.

5.11.2.14 Scatter graphs

For each pair of variables, scatter graphs display a marker whose x coordinate is the value of the first variable and whose y coordinate is the value of the second variable.

These graph types use an even number of variables, unpaired variables are ignored.

Scatter graphs can display scalar, vector, or matrix data. A vector variable with a length equal to the number of samples updates most efficiently. A matrix whose total number of elements equals the number of samples also updates efficiently.

Scatter graphs can display an unlimited number of data samples, with the exception of the multiple-Y web chart, which can display up to 10 samples.

The number of samples specifies the number of history values displayed. The minimum number of samples allowed is 2.

The marker type is determined by the second variable of each pair. The color of the marker is determined by the color or color threshold table associated with the second variable of each pair.

Different markers can be assigned to different variables to make it easier to distinguish between them.

To change markers, select the variable on the **Variables** tab of the **Graph Properties** dialog and use the **Marker** option.

If lines are used, different line types can be used to make it easier to distinguish between different variables. To change variable line types, select the variable on the **Variables** tab of the **Graph Properties** dialog and use the **Line Type** option.

These graph types display the x and y value axes. The time axis tick label displays the iteration number centered below the value (x) axis.

Labels are supported for both value axes and the time axis. The label for the vertical value axis cannot be entered until a second variable is attached to the graph.

Impulse graph
The maximum number of variables for the impulse graph is 20.

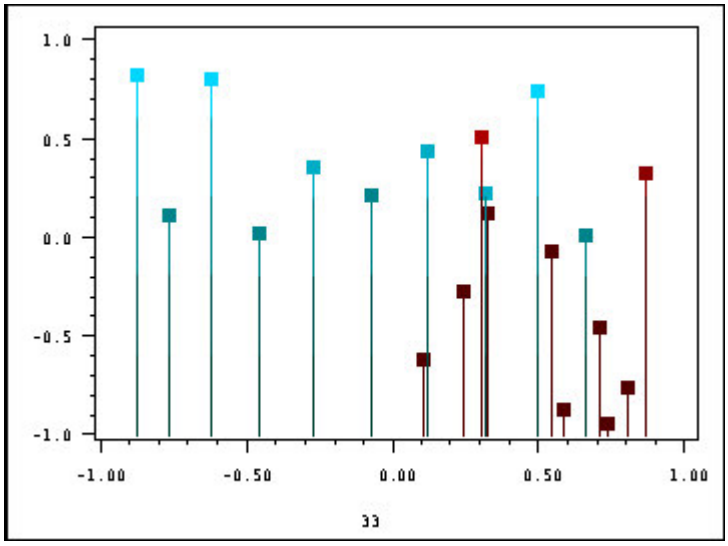


Figure 71: Impulse graph

The impulse graph draws a scatter plot in which a vertical line of the variable color is drawn from the marker to the X axis.

If either value in a variable pair is out of range, the marker falls outside the data area and is not drawn.

If a point is above the range, the marker is not drawn, but the vertical line is drawn from the horizontal axis to the top of the data area.

Impulse to zero graph
The maximum number of variables for the impulse to zero graph is 20.

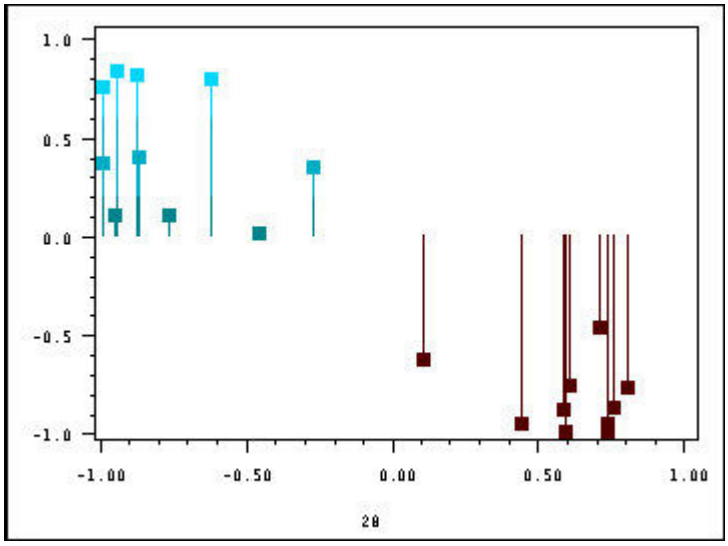


Figure 72: Impulse to zero graph

The impulse to zero graph draws a scatter plot in which a vertical line is drawn from the marker to the zero line in the variable color.

If either value in a variable pair is out of range, the marker is outside the data area and is not drawn. If a point is below the given range, the vertical line is drawn from the horizontal axis to

the bottom of the data area. If a point is above the range, the vertical line is drawn from the horizontal axis to the top of the data area.

Multiple-Y web chart
The maximum number of variables for the multiple-Y web chart is 20.

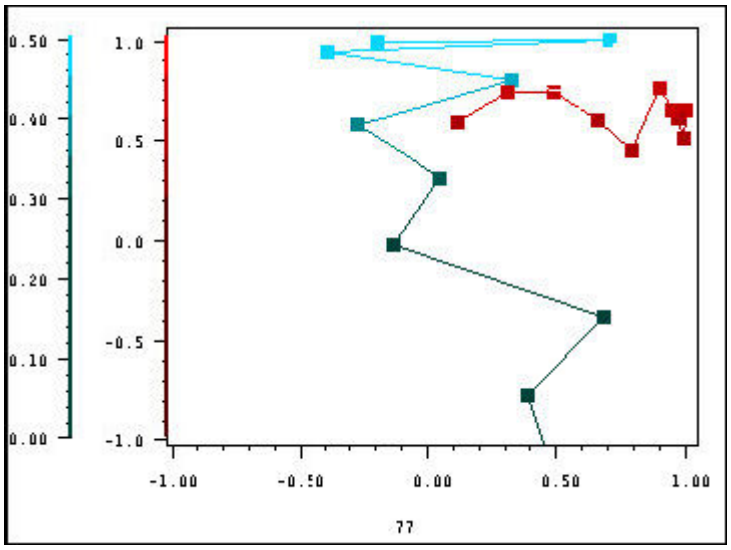


Figure 73: Multiple-Y web chart

The multiple-Y web chart draws multiple scatter plots connecting each point to the adjacent points and multiple vertical value axes.

The Y axis is displayed for each variable pair. The values are determined by the second variable of each pair. The color of the axis matches the color of the variable. The axis is displayed for every variable pair even if the variable range is not unique.

Each Y axis is labeled with the name of the second variable in the pair. If the second variable in a pair has no name, the axis label is used if one was assigned.

Markers are not drawn for values outside the given range. Lines are drawn with the correct slope as if they were connected to points outside the range, but are clipped to the bounds of the data area.

The horizontal axis option is not supported for this graph type.

Scatter plot
The maximum number of variables for the scatter plot chart is 20.

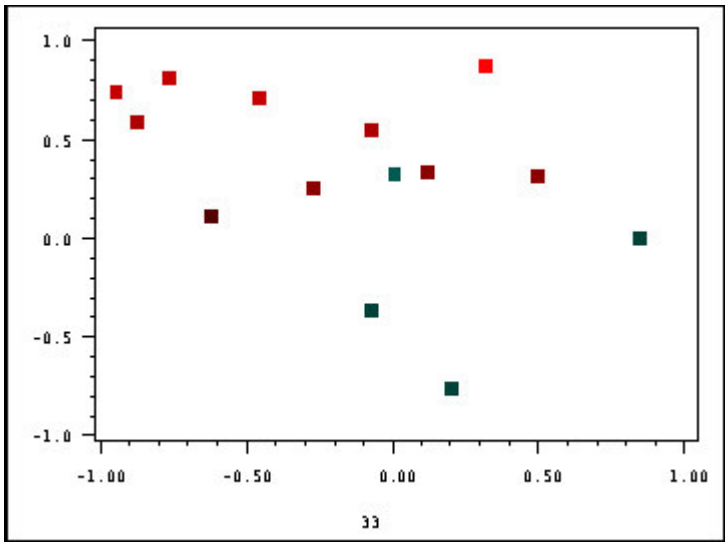


Figure 74: Scatter plot chart

The scatter plot chart draws a scatter plot. If either value in a variable pair is out of range, the marker is outside the data area and is not drawn. The value axis displays the range of the second variable.

Web chart
The maximum number of variables for the web chart is 20.

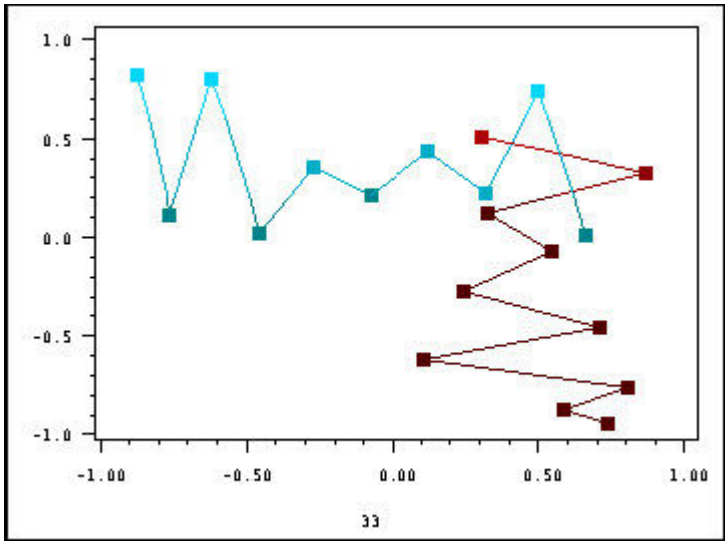


Figure 75: Web chart

The web chart draws a scatter plot with lines connecting each point to the adjacent points.

Markers are not drawn for values outside the given range. Lines are drawn with the correct slope as if they were connected to points outside the range, but are clipped to the bounds of the data area.

5.11.2.15 Section graphs

Fan
The maximum number of variables for the fan chart is 2.

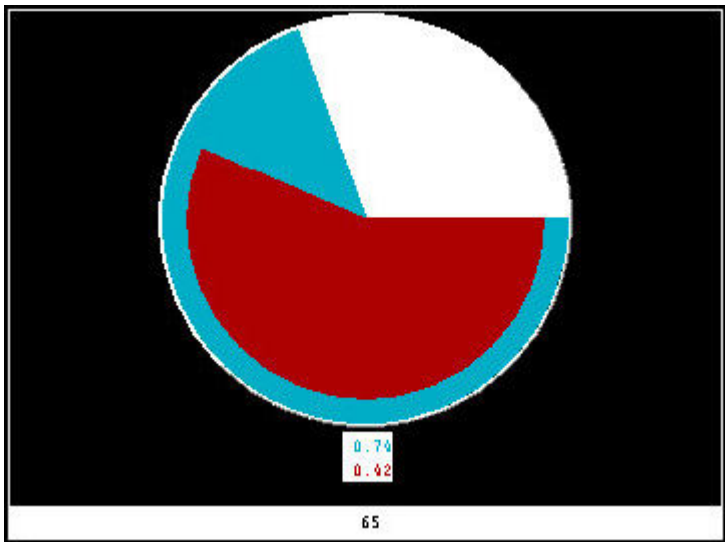


Figure 76: Fan chart

The fan chart draws fans that open in a clockwise direction. A fan is a filled arc resembling a pie slice. The greater the value, the larger the fan. The lowest value is an empty circle. The highest value shows a full circle in the colors of the variable.

The fan graph can display scalar, vector, or matrix data. Only one data sample is displayed at a time.

The shape of the variable determines the number of fans displayed. Multiple variables display as fans superimposed on each other with decreasing radii.

The color of the fan is determined by the color or color threshold table associated with the variable.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. For matrix data, both the horizontal and vertical axis ticks and tick labels apply. For vector data, only the horizontal ticks and tick labels apply. For scalar data, neither the horizontal nor vertical axis ticks and tick labels apply.

The time axis tick label displays the iteration number centered below the horizontal axis. The value axis tick label displays the variable value centered below each fan.

Pie chart
The maximum number of variables for the pie chart is 10.

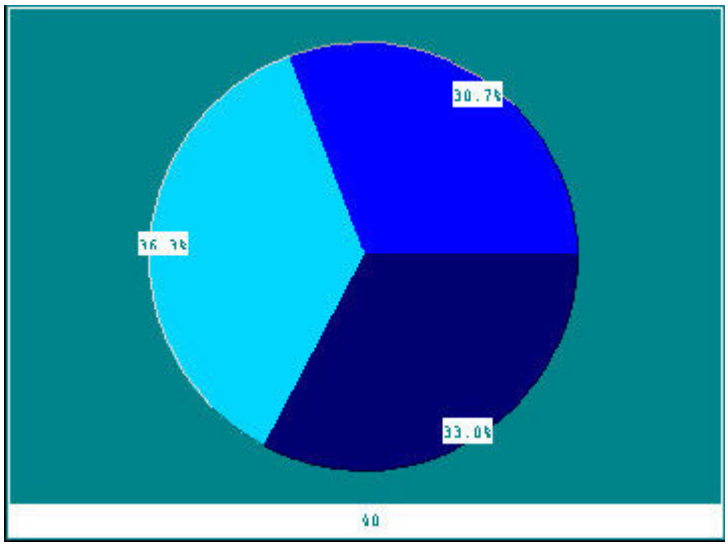


Figure 77: Pie chart

The pie chart draws a standard pie chart.

Only one data sample is displayed at a time. This graph type displays a single pie chart, regardless of the shape of the variables.

This graph type is most legible if each variable uses a different color. If using color thresholds, each variable should have a different set of colors.

For this graph type to be meaningful, all variables should have the same range.

The color of each pie slice is determined by the color or color threshold table associated with the variable.

The time axis tick label displays the iteration number centered below the horizontal axis. The value axis tick label displays the variable value centered at the edge of each pie section.

Labels use the current foreground color of the graph.

5.11.2.16 Size graph

The maximum number of variables for the size graph is 3.

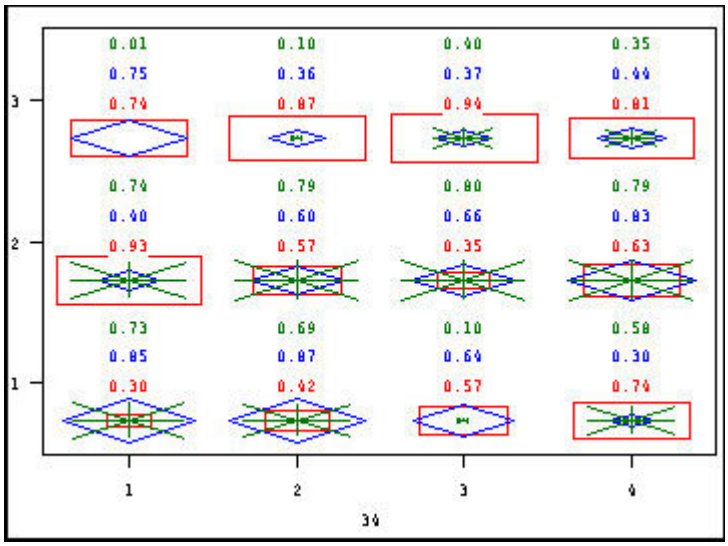


Figure 78: Size graph

The size graph displays up to three variables as geometric shapes whose sizes change as the variable values change. The first variable appears as an unfilled rectangle, the second as an unfilled diamond superimposed on the rectangle, and the third as an unfilled star superimposed on the diamond and rectangle. If there is only one variable, the geometric shape is a filled rectangle.

The size graph can display scalar, vector, or matrix data. Only one data sample is displayed at a time.

The default shapes can be replaced by associating markers with the variables.

If the value tick labels are on, the data values are displayed digitally directly above the shape sets.

The shape of the variable determines the number of shape sets displayed.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. The time axis tick label displays the iteration number centered below the horizontal axis. The value axis tick label displays the variable value centered below each shape. If the variable is a matrix, both the horizontal and vertical axis ticks and tick labels appear. If the variable is a vector, only the horizontal ticks and tick labels appear. If the variable is a scalar, neither the horizontal nor vertical axis ticks and tick labels appear.

The color of each shape is determined by the color or color threshold table associated with the variable.

5.11.2.17 Spectro graphs

Spectro graphs display a colored bar for each sample of a vector variable. The bar is divided vertically into the number of elements in the vector variable and each region of the bar is colored to reflect the value of the element according to the color threshold table. If the variable is scalar, each bar is a single solid color.

Spectro graphs can display scalar, vector, or matrix data. This graph type works best with a vector variable that has a color threshold table. Matrix data is not meaningful with this graph type.

Spectro graphs can display an unlimited number of data samples.

The data for the first sample appears in the left slot of the graph and subsequent samples fill in the slots from left to right. Scrolls from right to left or wraps around to the left edge of the graph.

The maximum length of a vector variable is 250.

The legend is a color bar that shows the colors corresponding to the threshold values. In the case of multiple variables, the color bar only displays the colors of the last variable. The variable values are mapped uniformly to the axis of the color bar, not to the threshold values. To turn the color bar off, turn off the **Legend** option on the **Basic** tab of the **Graph Properties** dialog.

These graph types are effective on color machines using a gray scale, but not on monochrome machines.

Smoothed spectro graph
The maximum number of variables for the smoothed spectro graph is 1.

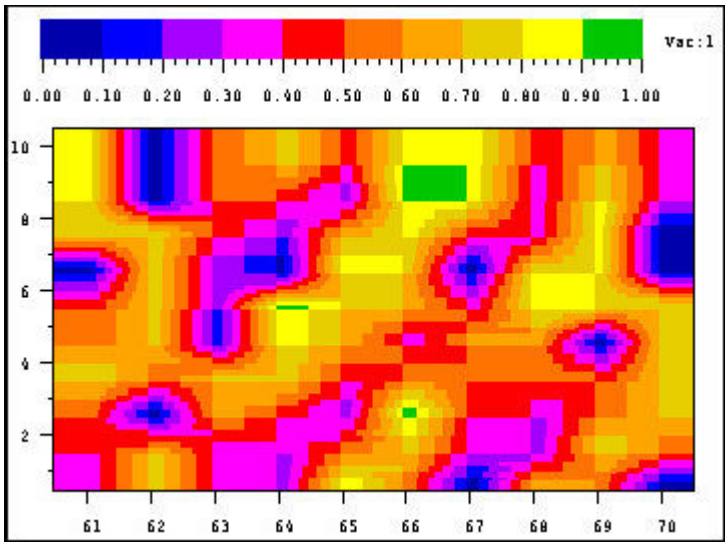


Figure 79: Smoothed spectro graph

The smoothed spectro graph displays a spectro graph with smoothed color transitions between the elements in a vector variable and between the samples. The color transitions are drawn between neighboring values using the color thresholds.

The column axis displays the numbers of the elements in a sample. For example, the column axis of a vector variable with a length of 8 has values from 1 to 8. The column tick marks appear at the center of each element's height. The value of each element is indicated by the color of the rectangle, not by its vertical position. The legend at the top of the graph maps the colors to the variable range.

The entire smoothed spectro graph must be visible in the display area to draw properly in the **Run** and **Prototype** menus.

If the user's display device does not support raster operations, the smoothed spectro graph behaves like the spectro graph.

To display more than one variable, use a stacked smoothed spectro graph.

Spectro graph
The maximum number of variables for the spectro graph is 1.

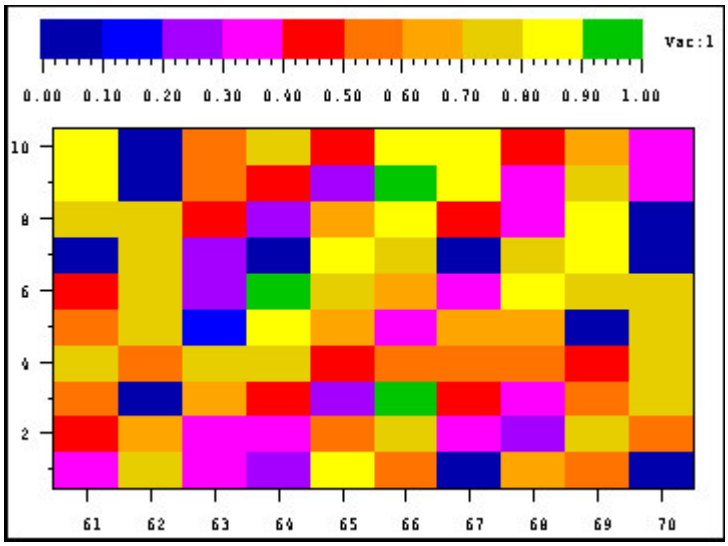


Figure 80: Spectro graph

The spectro graph displays a spectro graph.

The column axis displays the numbers of the elements in a sample. For example, the column axis of a vector variable with a length of 8 has values from 1 to 8. The column tick marks appear at the center of each element's height. The value of each element is indicated by the color of the rectangle, not by its vertical position. The legend at the top of the graph maps the colors to the variable range.

To display more than one variable, use a stacked spectro graph.

Stacked smoothed spectro graph
The maximum number of variables for the stacked smoothed spectro graph is 16.

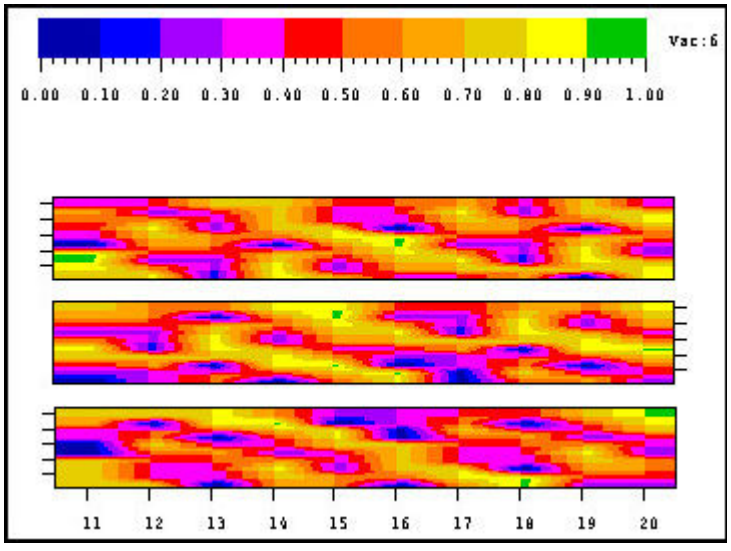


Figure 81: Stacked smoothed spectro graph

Displays each vector variable as a smoothed spectro graph, stacking each graph above the previous one.

The value axis of each graph is displayed on alternate sides of the graphs, starting at the left side of the bottom graph.

This graph type displays a single title for the stack of graphs.

The value axis displays the numbers of the elements in a sample. For example, the value axis of a vector variable with a length of 8 has values from 1 to 8. The value tick marks appear at the center of each element's height. The value of each element is indicated by the color of the rectangle, not by its vertical position.

For this graph to be meaningful, all variables should use the same color threshold table.

The legend only displays the colors of the last variable.

The entire stacked smoothed spectro graph must be visible in the display area to draw properly in the **Run** and **Prototype** menus.

If the user's display device does not support raster operations, the stacked smoothed spectro graph behaves like the stacked spectro graph.

Stacked spectro graph
The maximum number of variables for the stacked spectro graph is 16.

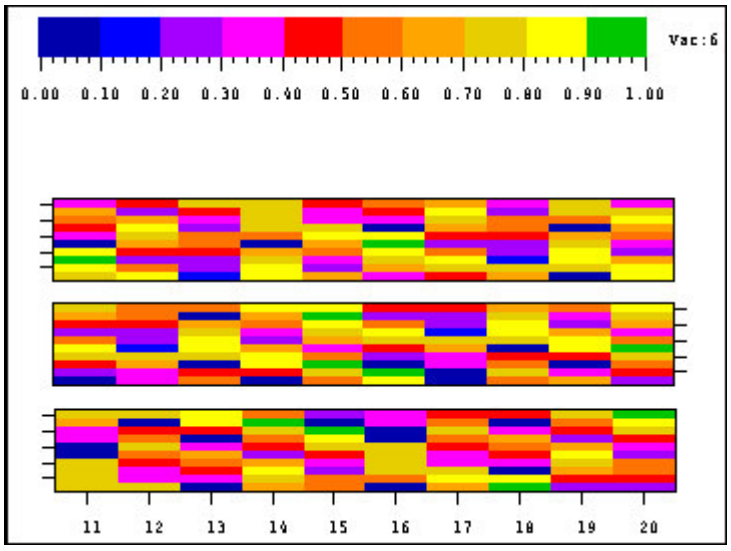


Figure 82: Stacked spectro graph

Displays each vector variable as a spectro graph, stacking each graph above the previous one.

The value axis of each graph is displayed on alternate sides of the graphs, starting at the left side of the bottom graph.

This graph type displays a single title for the stack of graphs.

The value axis displays the numbers of the elements in a sample. For example, the value axis of a vector variable with a length of 8 has values from 1 to 8. The value tick marks appear at the center of each element’s height. The value of each element is indicated by the color of the rectangle, not by its vertical position.

For this graph to be meaningful, all variables should use the same color threshold table.

The legend only displays the colors of the last variable.

5.11.2.18 Surface graph

The maximum number of variables for the surface graph is 1.

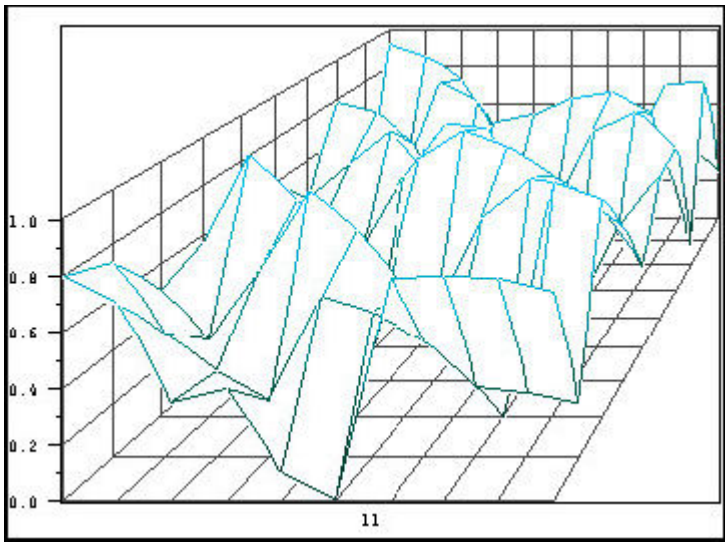


Figure 83: Surface graph

The graph plots the data as a three dimensional surface with the hidden lines removed.

The grid represents the data array positions. The position of each surface point above the grid corresponds to the element's location in the data array. The height of a point on the surface is proportional to the data value. The origin is in the lower right corner.

The color of the surface lines is determined by the color or color threshold table associated with the variable.

The time axis tick label displays the iteration number centered below the graph.

This graph type can display scalar, vector, or matrix data, but works best with matrix data. A scalar variable plots as a plane.

5.11.2.19 Text graphs

Text graphs display the contents of text variables.

Message graph
The maximum number of variables for the message graph is 18.

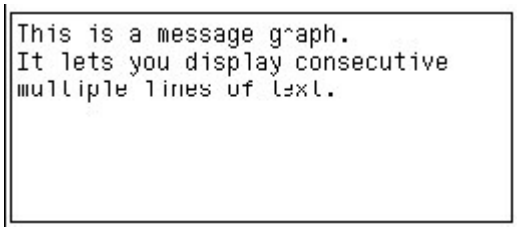


Figure 84: Message graph

The graph displays the contents of one or more text variables, adding each successive iteration of strings below the previous strings.

The message graph can display an unlimited number of data samples.

Only text variables can be displayed in the message graph. To display numerical data, the data must be in text format. Non-text variables can be used to control aspects of the message display.

Multiple text variables display side by side. The first iteration of all the variables appears on the first line, the second iteration on the second line, etc. To separate entries on the same line, space must be included in the text files.

The number of samples specifies the number of text values in the current sampling.

The first scalar variable specifies which iteration of text values to display at the top of the graph from among the current sampling. For example, if the scalar value is 1 and the number of samples is 10, the first text value from among the current sampling appears at the top of the graph, with the next nine values below it. When the eleventh value is displayed, the values scroll up by one, and the second text value becomes the first in the current sampling.

If the scalar value is more than 1, each text value appears at the top of the graph, then scrolls off the top until the specified iteration is displayed. That iteration remains at the top of the graph and the remaining values in the current sampling appear below it. For example, if the scalar value is 3 and the number of samples is 10, the first text value appears at the top of the graph. The second text value then replaces the first. Then the third text value replaces the second, with the remaining seven values appearing below it.

The first scalar variable can be used to scroll backward through earlier values in the current sampling, especially if an input object is used to control the iteration number. For example, an input object can be connected to the message graph's first scalar variable. The range of the input object should be equivalent to the number of samples specified. For example, if the number of samples is 50, the range of the input object can be from 1 to 50. If the user then selects a value of 30 with the input object, the message graph restarts its display at the 30th text item of the current sampling.

If the graph is not large enough to display all the samples, it only displays enough samples to fill the graph. To make the graph scroll upward to display the latest iterations, use a scalar value of -1. In this case, the scalar value does not control which text value appears at the top of the graph, but only makes the text values scroll up with every iteration after the specified number of samples is displayed.

The second scalar variable controls the text size. The text size variable should be a constant. If it is not a constant, the graph uses only the first value to determine text size. A value of 1 corresponds with the smallest text size, a value of 4 corresponds with the largest. This graph type can use a maximum number of 16 text variables and 2 numerical variables.

Text graph
The maximum number of variables for the text graph is 2.

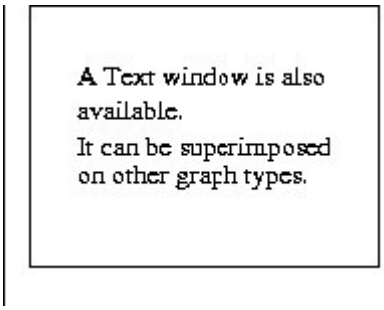


Figure 85: Text graph

The graph displays text variables. Text is left justified in the vertical center of the graph, and can be static or dynamic. Adding text dynamics to text objects can produce similar results.

Only one data sample is displayed at a time.

To display dynamic text, the first variable must be a text variable. A second variable of any type can be added to determine the text color. If only a text variable is used, the text appears in an arbitrary color.

If the first variable is not a text type variable, only the graph title appears, centered in the area.

If the first variable is a text variable, the title is justified in the upper left corner of the area, and the text from the text variable is vertically centered along the left edge.

The graph title uses the graph foreground color.

Both static and dynamic text may be displayed at the same time. If a text variable is specified as the first variable, its data is displayed in the center of the area in the color of the second (non-text) variable.

The graph title is displayed at the top of the area in the foreground color.

The text graph displays the last string if it runs out of data.

The text graph uses machine text which does not shrink or zoom with a drawing. To make text change size with the drawing, use vector text.

5.11.2.20 Vector graphs

Vector graphs display data in the form of vectors. These graphs are not meaningful with fewer than three variables. Any variables not specified are set to zero.

Vector graphs display scalar, vector, or matrix data. Only one data sample is displayed at a time.

For the angles to be meaningful, the variable ranges should be symmetrical around zero.

The color of the vectors is determined by the color or color threshold table associated with the last variable descriptor.

Flowfield
The maximum number of variables for the flowfield chart is 5.

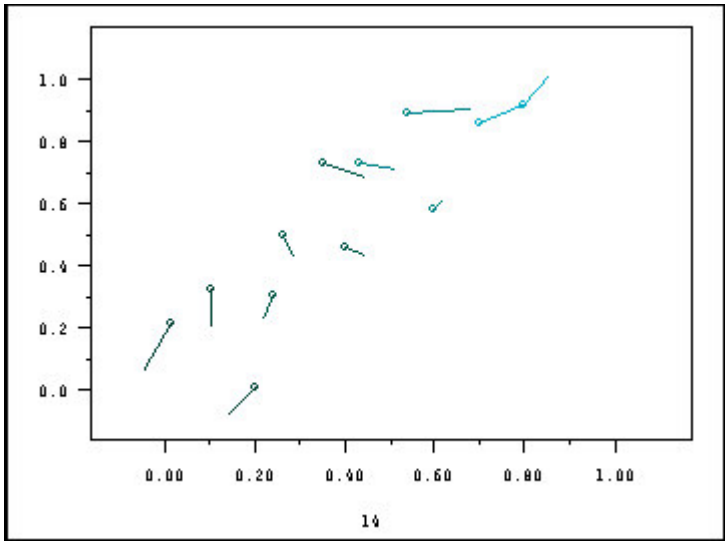


Figure 86: Flowfield chart

The chart displays up to five variables as points, each with a vector attached. A minimum of three variables required to supply the x and y coordinates of the points and the length of the

vectors. The first variable provides the x coordinate of each point, the second provides the y value, the third variable provides the x component of the vector, and the fourth variable, if used, provides the y component of the vector.

Each vector is drawn with its corresponding plotted point as its origin. The fifth variable, if used, provides the z component of the vector. The z component, if used, is displayed as color changes using the color threshold table of the fifth variable.

The third, fourth, and fifth variables should all have the same range.

The number of elements in the variable determines the number of point-vector sets displayed. A scalar variable displays only one data set at a time. The number of data sets displayed by a vector equals the length of the vector, and the number of data sets displayed by a matrix equals the number of elements in the matrix.

The value axis tick and tick label settings control display of the ticks and tick labels of both the horizontal and vertical axes. The time axis tick label displays the iteration number centered below the horizontal axis.

Vector graph
The maximum number of variables for the vector graph is 3.

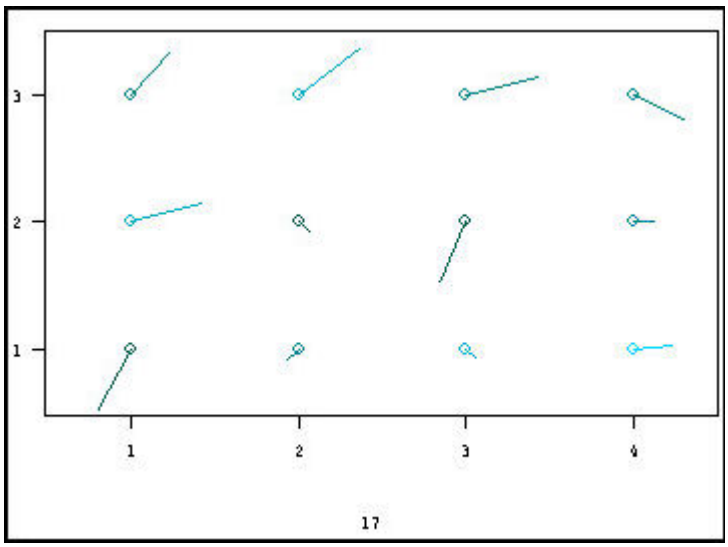


Figure 87: Vector graph

The vector graph plots a three dimensional vector field. The origin of each vector is constant. For each vector, the first variable provides the x component, the second variable provides the y component, and the third variable provides the z component. The z component is represented by the color of the line.

The shape of the variable determines the number of vectors displayed.

The horizontal and vertical axis tick labels serve to number the rows and columns displayed. The time axis tick label displays the iteration number centered below the horizontal axis. If the variable is a matrix, both the horizontal and vertical axis ticks and tick labels appear. If the variable is a vector, only the horizontal ticks and tick labels appear. If the variable is a scalar, neither the horizontal nor vertical axis ticks and tick labels appear.

5.11.3 Input Objects

Input objects gather data directly from the user. Input objects are dynamic objects that interpret input from the user, also called a user event, by converting it into a value, record the

selection in a data variable, and change their appearance on the screen as they interpret input from the user. Examples of input objects are checklists, palettes, menus, and scrollbars.

An input object is inherently dynamic since it collects input from the user in an interactive manner and changes to reflect that input. Input objects become active when the user runs the view or prototype.

Input objects are defined by their type and graphical layout. The user interacts with an input object by pressing specified keys to invoke defined actions. The result of the action is a value.

Input objects are defined by:

- Type: Each type (checklist, scrollbar, etc.) has a set of defaults that lets the user use an input object without making any changes, although the user can change the defaults to customize the input object for their application. For example, the user may want to add more options to a menu, define a new object as a check mark for your checklist, add more detail to a slider, or change the objects used in a toggle.
- Graphical layout: The graphical layout is defined by the layout template. The template passes graphical information, along with flags that control its behavior, to the input object.
- Actions: The actions are a set of special operations that include **Select**, **Done**, **Restore**, **Cancel**, **Clear**, and **Toggle Poll**. Defined actions can also be associated with specified keys.
- Value: The resulting value is stored by the input object and can be used by rules.

The user can customize the input object's appearance by editing it like any other graphical object. For example, input objects can be moved, resized, copied, mirrored, reordered, scaled, and deleted. In addition, the appearance or function of an input object can also be customized.

There are two basic ways to customize an input object. To edit the general characteristics of the input object, use the **Input Object Properties** dialog, as discussed in Section 5.11.4. To edit characteristics that are specific to that type of input object, the input object's template must be edited as discussed in Section 5.11.4.4.

5.11.3.1 Input object types

The input object types are

- Button
- Checklist: Object or Text
- Menu: Object or Text
- Palette
- Slider or Scrollbar
- Slider2D
- Text Editor
- Text Entry
- Toggle: Object or Text

5.11.3.2 Button

The button input object presents a single selectable item. The item can be any graphical object except a graph or another input object. A button can echo the selection in one of two ways: while the button is being pressed or until the button is pressed again to deselect it. These different kinds of buttons are called push buttons and toggle buttons, respectively. Both kinds of buttons can highlight to indicate when the cursor is within its boundary. The appearance of the item in its selected, unselected, and highlighted states is controlled by graphical objects in the template. The button input object has one attached variable, whose value is 0 when unselected, and 32K when selected.

Button input objects can be embedded in the templates of other input objects. They can be used as buttons offering choices in menus, checklists, or toggles. They can also be used as action buttons or increment buttons in templates for any other type of input object.

5.11.3.3 Checklist: Object or Text

The checklist input object presents a list of checkable items. Items can be text, button input objects, or non-dynamic objects such as circles, subdrawings, or icons. Selected items are marked in a user-defined area with a user-defined check. The **Select** key toggles items between selected and deselected. More than one item can be selected at a time. Each item in the checklist has an attached variable. The select value is controlled by the user, and the deselect value is zero.

5.11.3.4 Menu: Object or Text

The menu input object presents a menu of selectable object items, text items, or button input objects. Only one item can be selected at a time. The selected item can be echoed by a filled pickable area around it, by a border, or not at all. This echo area can be any object shape. This input object has one attached variable. A text menu uses text items and an object menu uses object items. Object items can be any object except graphs or other input objects.

5.11.3.5 Palette

The palette input object presents a palette of selectable colors or patterns. Only one color or pattern can be selected at a time. The current selection can be echoed in a separate area within the input object. This input object has one attached variable that is set to the color index of the selected color item. Selecting a color from the palette sets the value of the variable to that color index.

5.11.3.6 Slider or Scrollbar

The slider and scrollbar input objects present a valuator which lets the user select a value within a predetermined range. The slider and scrollbar differ only in the way they echo the current value. The value stored in the variable is determined by the current slider position.

In a slider, the current value is indicated by the leading edge of a sliding rectangular area anchored at one end.

In a scrollbar, the current value is indicated by a moveable rectangle. The moveable rectangle occupies a specified percentage of the slider range. The leading edge, center, or trailing edge of the moveable rectangle can be used to indicate the current value.

The minimum and maximum values of the range are at either end of the Slider.area. The range and current values can be echoed numerically within the input object. These input object types have one attached variable.

The value can be changed in three ways depending on the template definition. A value can be selected directly by picking a position inside the Slider.area with the **Select** key, continuously by moving the cursor inside the Slider.area, or incrementally by selecting inside the increment areas or buttons.

If the value is less than the slider increment, it is rounded to zero.

5.11.3.7 Slider2D

The slider2D input object presents a two dimensional valuator in which the current value is indicated by the position of a marker within a rectangular plane according to x,y coordinates. The user can select a marker from the available options or design their own.

The x and y values are based on the range of the two variables associated with the slider. The values stored by the variables are determined by the marker position. The x value range is the width of the Slider2D.area and the y value range is its height. The range and current x and y values can be echoed numerically within the input object.

The x and y values can be changed in three ways depending on the user's template definition. A value can be selected directly by picking a position inside the Slider2D.area with the **Select** key, continuously by moving the cursor inside the Slider2D.area, or incrementally by selecting a direction area or button.

5.11.3.8 Text Editor

The text editor input object presents a text input box for multiple-line text entry and editing. If the entry does not fit in the box, the text scrolls up and to the left. Text can be entered whenever the cursor is inside the text echo area. Text can be inserted, overwritten, or marked for cutting, copying, and pasting. The input is echoed by the text block in the box. This input object lets the user specify a bell to indicate a text entry error and a Help area to display a list of the editing commands. The **Help** list can be scrolled. This input object type has an attached fixed length text variable that controls the number of characters you can enter. The default length is 10.

5.11.3.9 Text Entry

The text entry input object presents a text input box for single-line text entry. The attached variable controls the number of characters the user can enter. If the entry does not fit in the box, the text scrolls to the left. This input object lets the user specify either a bell or a flashing text area to indicate a text entry error. Text can be entered whenever the cursor is inside the input object, not only when it is in the text echo area. The input is echoed by the text string in the box. This input object type has an attached fixed length text variable with a default length of 10.

5.11.3.10 Toggle: Object or Text

The toggle input object presents a list of items in a predefined order, showing one option at a time. The current item echoes the current value. This input object has one attached variable. A text toggle uses text items or button input objects, and an object toggle uses object items. Object items can be any object except graphs or other input objects.

5.11.4 Customizing input objects

Use the **Input Object Properties** dialog to edit the general characteristics of an input object.

- Input object type: Determines the form of interaction such as slider, menu, or text entry.
- Variable: Controls which data variable stores the selected values of the input object and the range of values to be used for sliders and scrollbars.
- Display options: Determine how the input object is erased, whether the obscuring area raster is saved before drawing, whether the outline of the input object is drawn, and whether the outline of the echo area is drawn.
- Template: Specifies the template file that contains the graphical information which controls the input object's appearance. Indicates whether a template is required or

optional for this input object type. Indicates the number of pickable items allowed in the template.
- Pickable Items: Specifies the number of items to appear in the input object, together with their labels and associated values.
- Action keys: Associates specified key strokes and mouse buttons to various actions.

5.11.4.1 Guidelines for creating input objects

Input objects are designed to follow a particular model. Ignoring their design requirements can make them behave inconsistently with the model. Some important guidelines are listed below.

- All required areas, objects, and flags must be included.
- Specially named items and areas must follow the correct naming conventions. Anything named incorrectly appears as an ornament. If an object or flag is not functioning correctly, check the spelling of its name.
- Objects must have unique names. Additional objects with the same name are ignored. This does not apply to ornaments.
- All control points of an object must be completely inside the layout area or objects area.
- All ornament control points must be inside the layout area, or the ornament does not appear.
- Pickable areas should not overlap each other.
- The layout, flags, and objects areas must not intersect.
- Flag control points must be inside the flags area.
- In menus and checklists, the items should have corresponding pickable areas.
- Variables should not be shared among input objects or their items.
- Checklist item values should not be set to zero.
- Menu, toggle, and checklist items should have unique values.
- A key binding must be defined for the **Select** action key for all input object types.
- In a menu, the **Done** and **Select** action keys must have the same key binding.

For creating input objects, see Section 4.9.10.

5.11.4.2 How input objects use templates

Input object type and template are the two major features that control the form of user interaction and the physical appearance of an input object. The template is a view, created in Display Builder, that contains the graphical and behavioral information. The input object type determines what objects and text are required and what objects and text are optional for an input object. Each type has an associated set of valid objects and flags.

When an input object is created, the graphical and text objects contained in its template file are copied and included in the input object's internal structure. The input object refers to these copies for drawing and behavior information.

When creating several input objects of the same type, if they should look and behave the same, use the same template, and if they should look and behave differently, use different templates.

5.11.4.3 Specifying an input object template

The input object's layout template determines the physical appearance of the input object, such as the shape and location of pickable areas, the maximum number of pickable items displayed, and ornamentation. When using an input object, the template is not drawn, but it provides information about how the input object looks and performs.

Each input object type has a default template provided in the Templates directory. This tab indicates whether a template is required or optional for the current input object type. If a

template is not used, the input object uses internal defaults for required information on layout and performance.

Default template: A different default template is used for each input object type.

To change the template used by the input object:

1. Double-click the input object to display the **Input Object Properties** dialog.
2. Select the **Template** tab.
3. Click **Change**.
   This displays the **Open File** dialog with the file type filter set to Template Files.
4. Select the desired template file.
5. Click **OK**.

The template can be edited to provide additional information including labels, outline, and number of strings appearing in a menu. To edit a template, load the template view file into Display Builder. After editing the template, reload the template into the input object to see the effect of the edits.

5.11.4.4 Input object template editing

Editing the template lets the user:

- Change the appearance of elements within the input object.
- Add items to checklists, menus, and toggles.
- Add ornamentation to the input object.
- Create action areas or buttons.
- Change where picks register.
- Set aspects of the input object behavior.

Templates are views created using Display Builder. New template views can be created from scratch, but the easiest way is to start with an existing template file and customize it. Default and example templates are available in the Templates subdirectory of the lib directory.

To edit a template, load the template view file into Display Builder. After editing, be sure to save the edited template with a different file name. The edited template file is saved in the user's working directory unless a full path name is specified.

If the template is saved using a new name, the new template must be loaded into the input object to see the effect of the edits. To load a different template, use the **Template** tab of the **Input Object Properties** dialog. If the template is edited again and is saved using the same name, click on the **Reload** button to reflect the edits in the input object.

Naming conventions
The user must follow strict naming conventions for the objects in the template. Objects that are not named correctly do not function as intended in the input object. To name a template object, double-click the object and enter a name in the text box that appears. Names are case sensitive and spelling is critical: for example, Layout.area cannot be spelled as layout.area or Layout.Area.

The templates are composed of two main areas comprising two rectangle objects that must be named Layout.area and Flags.area. A third main area, named Objects.area, is useful for buttons, toggles, menus with scrolling, and checklists with scrolling. Within these areas, certain named objects are required. Others are optional but helpful in tailoring the input object. Any of Display Builder's objects can be used in templates, but dynamics are usually stripped when using dynamic objects. Graphs are not useful in templates, and input objects are useful only in limited cases. Vector text or hardware text can be used for any text object unless otherwise noted.

The input object type determines what objects are valid in the template and what names should be given to them. These names, object types, and brief descriptions of the function of the objects are listed in tables in Section 5.11.3.

Layout area
The layout area is a graphical object whose bounding box defines the boundary of the visible portion of the input object. This object must be named Layout.area. The layout area is mapped to the two control points selected when an input object is created.

All objects within the layout area, except for hardware text labels, are stretched to fit the input object when drawn. Hardware text labels scale to fit when a corresponding named area exists. When the input object's aspect ratio differs from the layout area's aspect ratio, the objects in the layout area may become distorted. Usually, the input object should have an aspect ratio that is similar to the aspect ratio of the layout area in the template.

Certain text objects that occur in the layout area serve only to define text attributes and location, not to provide textual information. These are called attribute labels. Examples of attribute labels are Min.text, Max.text, and Varname.text in the slider input object. Attribute labels are replaced with the corresponding value when the interaction is run, and therefore the user does not need to enter the value in the template. Vector text attribute labels are scaled to fit the input object when drawn. Hardware text attribute labels change font size to fit the input object when drawn. When the content of the label does not fit even after scaling, it is cropped to fit. Vector text labels are not cropped, but instead their character width decreases.

Pickable areas are designated areas where user input can take place, such as the slider echo area. These areas are named in the layout area according to conventions specific to each input object type. The named objects for pickable areas can be any shape, but the object's bounding rectangle defines the area where picks are registered, unless otherwise set using the PostType.flag in the flags area. Therefore, picks outside of a circular or polygonal pickable area may register as picks.

Action areas, labels, and buttons
All input object types, except for the button type, lets the user define special pickable areas with text labels or buttons associated with the actions Done, Restore, and Cancel. The results of these actions, other than normal echoing, may not be visible in the input object, but are useful with rules and in applications.

- Done: Signals that the interaction is complete. The area must be named Done.area, the label must be named Done.text, and the button must be named Done.button.
- Restore: Signals that the interaction is to be restored. Restores the state of the input object and returns the input value or text entry to the value it had when it was drawn. The input object is updated to reflect this action. The area must be named Restore.area, the label must be named Restore.text, and the button must be named Restore.button.
- Cancel: Signals that the interaction is to be aborted and restores the state of the input object and returns the input value or text entry to the value it had when it was drawn. The area must be named Cancel.area, the label must be named Cancel.text, and the button must be named Cancel.button.
- Ornaments. Objects in the layout area that are not named according to the naming convention serve as context or decoration and are called ornaments. They are drawn in the input object as they appear in the layout area.

Action areas are defined using geometric objects, such as rectangles or polygons. Labels that identify these areas can be placed in or near them using vector or hardware text. Labels within the areas should be named and are never drawn beyond the boundaries of the areas. Labels outside the areas should be unnamed. The text attributes of the labels (style, size, etc.) can be edited in the template. Vector text scales with the input object so distortions in size and aspect ratio may occur. Hardware text uses a best fit size.

Action buttons are defined using button input objects. Action areas are optional when action buttons are defined, but when both buttons and areas are used, the button is scaled to fill the

corresponding area. Action text labels and action buttons are mutually exclusive. For any single action, the user should not use both a text label and a button. Instead, the label for a button can be specified using the Label option on the **Template** tab of the **Input Object Properties** dialog.

Multiple sets of template objects can be created in a single view, then the layout area can be used to indicate the set the user wants to use. For example, the user could have two layouts that differ in their arrangement, ornamentation, or button styles. To change from one to the other, simply move the layout area rectangle from one group to the other. Note that while objects in one group can have the same names as objects in the other group, there can only be one object named Layout.area in a template view.

Object area
The objects area is a rectangle whose bounding box defines where items for buttons, checklists, menus, and toggles are placed. This area must be named Objects.area. The objects area is required for buttons and object toggles. It is optional for menus and checklists, and is used to contain the objects or attribute labels for checklists and menus that provide scrolling.

Flags area
The flags area is a rectangle whose bounding box defines where the input object control flags are placed. This area must be named Flags.area. The control flags are specially named text objects which determine certain dynamic details such as echoing and polling for the interaction. The flags area and the text objects in it are not visible in the input object.

The names of the control flags follow strict naming conventions and have the format ControlDetail.flag, where ControlDetail identifies what the flag controls. Names are case sensitive and must be spelled correctly for the flag to function.

The text strings of the flags follow this the format Flag:VALUE. The Flag describes what ControlDetail the flag controls and must be to the left of the colon. The VALUE determines the behavior of the flag. The Flag to the left of the colon is not interpreted, the VALUE to the right of the colon is. The text string of the flag is not case sensitive and can contain extra spaces after the colon. For example, POLL:YES, Polling: YES, PollFlag:YES and Poll:yes are all interpreted the same. For any of these flags to control the polling function, it must be named Poll.flag.

Section 6 Power process functionality

This section provides information on power process specific actions.

6.1 Object Browser

Object Browser shows the process object hierarchy for the specified application.

To open **Object Browser** and instantiate a symbol:

1. Select **Actions/Object Browser**. The Object Browser task pane is shown.
2. Select the application from the drop-down menu.
3. Click **Select**. The object hierarchy of the application is shown in the dialog.
4. Drag and drop the process object from the Object Browser to the drawing area of the Display Builder, or select the object and double-click it. A corresponding subdrawing and data variables of MicroSCADA data source are created.

The subdrawing data variables are automatically mapped to the data variables of the data source. For certain objects, different presentations, for example, horizontal or vertical symbols, can be selected. It is also possible to create a group of symbols for an object, for example when a circuit breaker symbol is created, a label is also created beside the breaker symbol.

To configure symbols created, or to customize the Object Browser, see the Application Design Manual. To select a presentation for an object, select the presentation from the drop-down menu at the bottom of the Object Browser.

The Object Browser task pane can be dragged to the right side of the drawing area so that the dialog merges to the edge of the Display Builder. This way the user can drag and drop the objects without having to move the Object Browser task pane all the time.

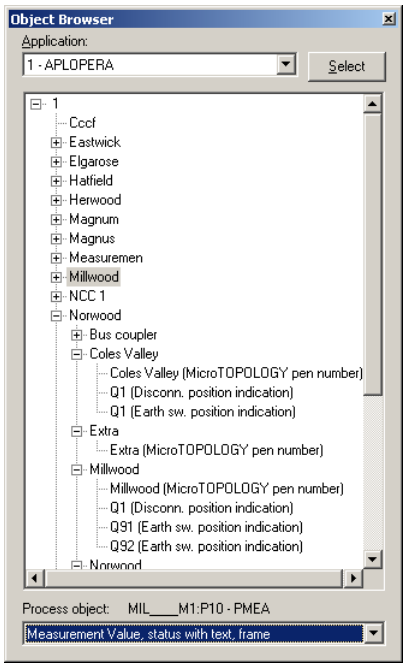


Figure 88: Object Browser

The **Object Browser** creates the variables automatically. It identifies the type of control dialogs to open so you do not need to configure them manually.

6.2 Object connections

The MicroSCADA Process Database Object Browser of Display Builder is used to create process displays. Drag and drop, for example, switch device symbols into the display and align them as needed. Finalize the single line diagram by adding busbar symbols.

Another way to create a process display is to add the appropriate symbols from the symbol palette. With object connections the created process display can be connected to the process database. If PCM600 is installed to the computer, symbols for launching PCM600 tools can be connected to the IEDs of the PCM projects.

It is also possible to detect connected and unconnected symbols on the display.

6.2.1 Connecting objects

To connect the symbols on a process display to the process database:

1. Select the required objects on the display.
2. Select **Actions/Object Connections/Connect Selected Objects**. A dialog opens.
3. Select the application from the drop-down menu.
4. Click **Select**.
5. Select a process object in the object tree.
6. Click **Connect**.

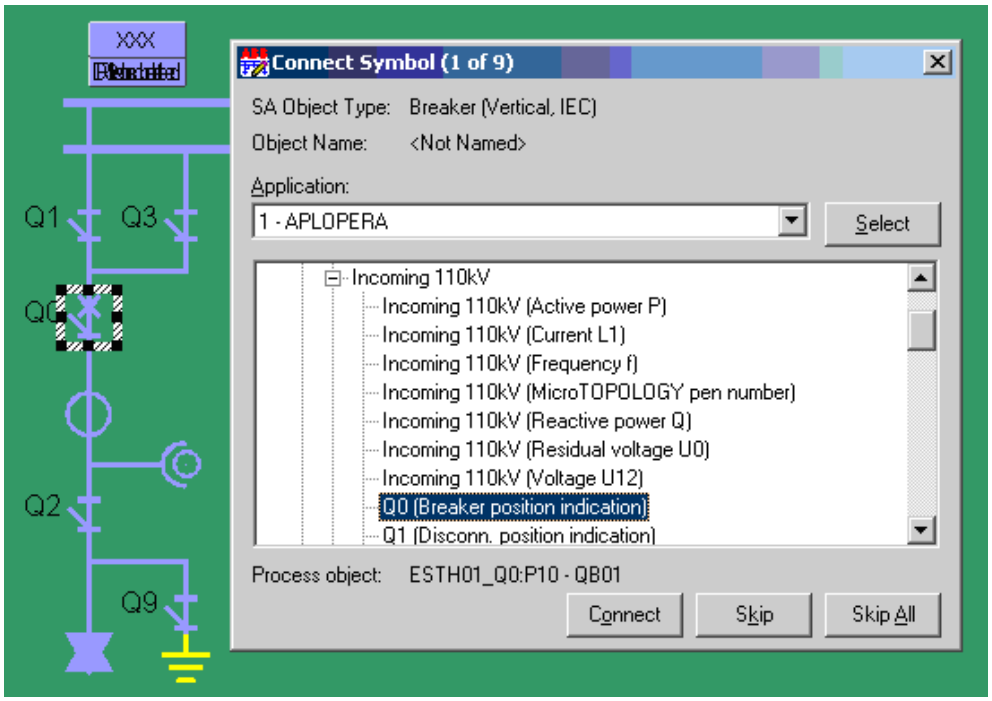


Figure 89: Dialog for selecting a process object

The following information is shown in the dialog:

- Total number of connectable objects in the selection
- Index of the object that is currently handled
- SA Object Type of the object
- SA Orientation of the object (if available)
- SA Representation of the object (if available)
- Name of the object (if already given)
- Selected process object information

The display is zoomed to the handled object and the object is selected while the connection dialog is open. If the object cannot be connected to the selected process object, the **Connect** button is disabled. To leave the current object unmodified, click **Skip**. The next connectable object is selected for modification. To leave all connectable objects unmodified, click **Skip All**.

If **Re-connect Connected Objects from Selection** is selected, only previously connected objects will be handled. By clicking **Connect Unconnected Objects from Selection**, only unconnected objects will be handled. It is also possible to select **Disconnect Selected Objects**.

If an action cannot be done, for example, if the selection of objects does not contain any connectable objects, an information message will be shown. A warning message is displayed before objects are disconnected.

Connect Selected Objects and **Connect Unconnected Objects from Selection** functions are also available in the context menu opened by right clicking an object in the process display.

6.2.2 Selecting objects

The Select Objects functionality can be used to find connected or unconnected symbols from the active process display. The found symbols can be, for example, connected with the connecting functionality.

To select connected symbols, click **Actions/Object Connections/Select Connected Objects**. To select unconnected symbols, click **Actions/Object Connections/Select Unconnected Symbols**.

When the selection is completed, the total number of selected objects is shown in the status bar. An information message is displayed if required objects cannot be found.

6.3 Network topology coloring

Network topology coloring can be used to calculate and visualize the electrical state of the network. The state is calculated for each component in the system.

Figure 90 and Figure 91 present the station with and without the network topology coloring. For more information on using network topology coloring in the process displays, see the SYS600 Operation Manual.

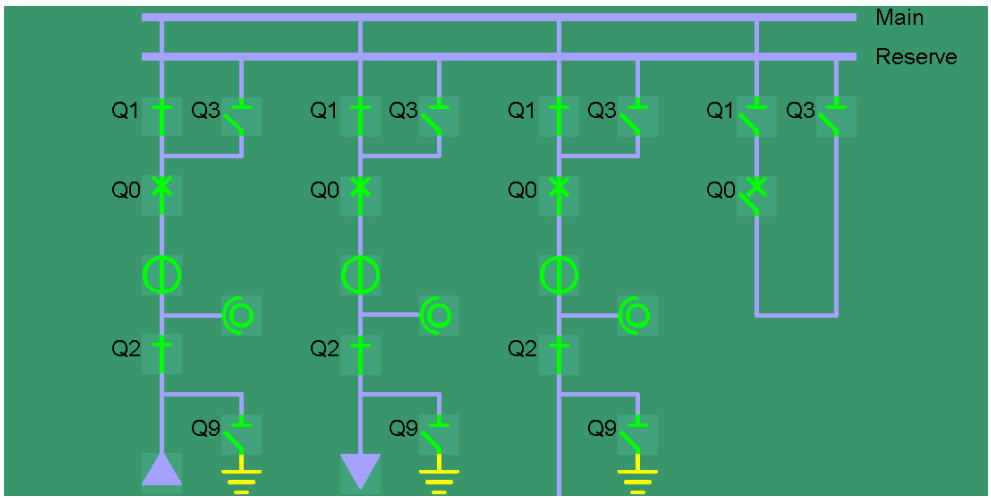


Figure 90: Station without network topology coloring

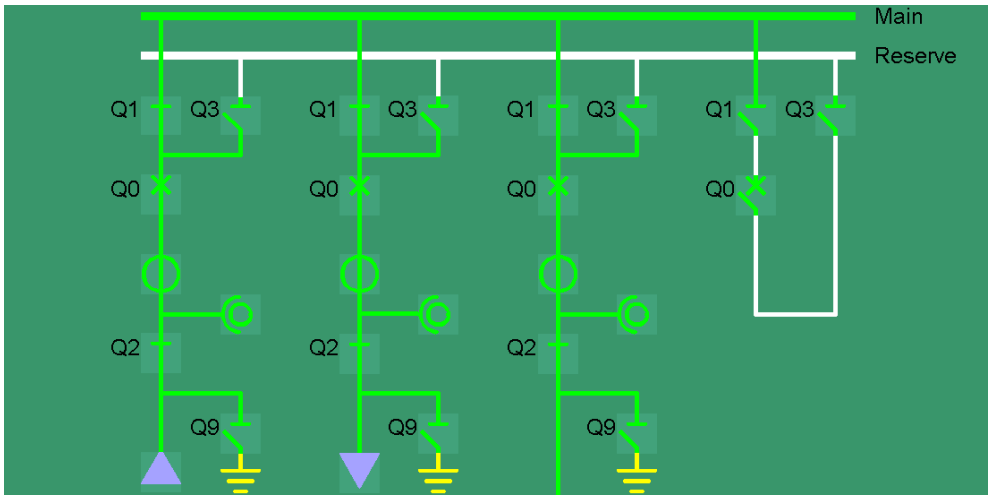


Figure 91: Station with network topology coloring

To enable network topology coloring:

- build a network topology.
- define voltage levels for all voltage sources.
- add display to a network topology model.

The same object, for example, certain disconnector, can only be in either Monitor Pro or Workplace X topology model. If same object is configured in both, export of Workplace X topology model from the View Builder will cause the related Monitor Pro topology model to be deleted.
When upgrading from Monitor Pro to Workplace X the following scenarios are possible:

1. Migrate all topologies at once from Monitor Pro to Workplace X
2. Migrate step-wise from Monitor Pro to Workplace X by, for example, separating one substation at a time from Monitor Pro topology model and creating a corresponding Workplace X topology model for it in View Builder.

For more information about building topology for Workplace X please refer to SYS600 10.2 Workplace X Process Picture Design manual.

6.3.1 Building the topology

Objects can be included to the topology with the **Add Selected Objects to Topology** function and excluded with the **Remove Selected Objects from Topology** function.

If **Actions/Options/Add New Objects to Topology by Default** is selected, all the objects that can be a part of the topology are included when they are created.

The objects that are in the topology should be connected to each other to build a network. An object can be connected to the nearest object by selecting the **Snap to Symbol** function. With busbars, the **Snap and Strech to Symbol** function can also be used. The function snaps the other end of a busbar to a nearby object and stretches or shrinks the other end of a symbol so that it is connected to some other object.

View the topology by selecting **View/Highlight Topology**. When the topology is highlighted the objects are colored according to the following rules:

- if the object is not included in the topology, it is white.
- the selected object is light blue.
- all the objects that are connected to the selected object are dark blue.
- if the object expects more connected objects, it is orange.

It is not necessarily an error if an object is colored orange.

The display can be edited while the topology highlight is active. Note that changing the color properties of an object may not work, if **Highlight Topology** is active.

Connections of topology objects are based on invisible connection points called hotspots. Both ends of the Display Builder primitive lines contain a hotspot. Each corner in the polyline contains a hotspot and the center point of each line segment contains a hotspot. There is also a hotspot in the center of the starting point and the end point of the polyline. In Figure 92 the arrows point at the hotspots of a breaker symbol.

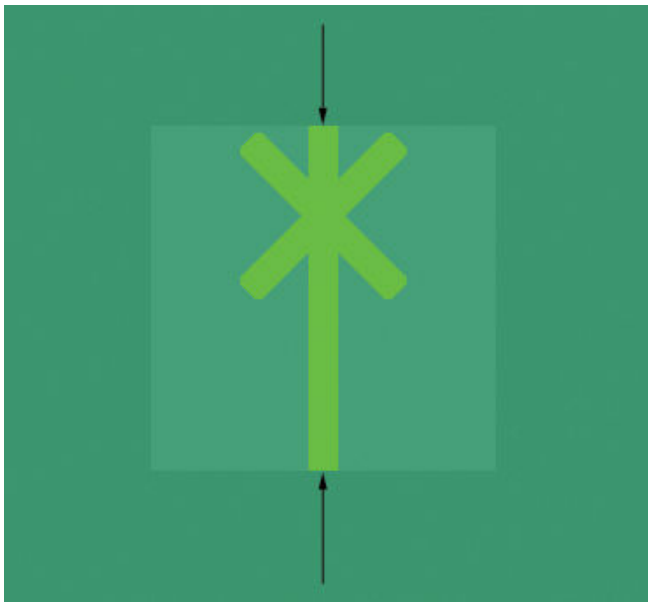


Figure 92: Hotspots of breaker symbol

6.3.1.1 Alternatives for busbar subdrawings

The primitive line objects or polyline objects with can be used instead of the busbar subdrawings. Use filled four point polylines for representing the 45 degree or other free angle busbars.

The horizontal and vertical lines behave similarly with the horizontal and vertical busbar subdrawings. Snap and Stretch to Symbol only supports the horizontal and vertical lines. It does not support the 45 degree lines, for example.

6.3.2 Specifying voltage levels

Voltage level information must be specified for all objects that act as voltage sources. Such objects are line indicators, transformers and generators. Default voltage levels are listed below. If other than one of the configured values is entered, the value is rounded to the nearest configured value. For example in the default configuration, value 120 kV is treated as 110 kV.

Table 46 lists the default voltage levels. The voltage levels can be configured with the NT Manager tool. For more information on the NT Manager tool, see the SYS600 Application Design manual.

Table 46: Default voltage levels

0 kV	15 kV	66 kV	220 kV
3 kV	20 kV	110 kV	400 kV
6 kV	30 kV	132 kV	800 kV
10 kV	45 kV	150 kV	1600 kV

The levels are given by mapping the following data variables to constant kilovolt values:

- Line indicators: Voltage Level (kV)
- 2-Winding transformers: Primary Voltage Level (kV) and Secondary Voltage Level (kV)
- 3-Winding transformers: Primary Voltage Level (kV), Secondary Voltage Level (kV) and Tertiary Voltage Level (kV)
- Generators: Voltage Level (kV)

To map the data variables to constant kilovolt values:

1. Select the object and select **Object/Properties**.
2. Select the **Subdrawing** tab.
3. Set the **Mapping Type** to **Map to Constant**.
4. Enter the value to the **Is Mapped To** column.

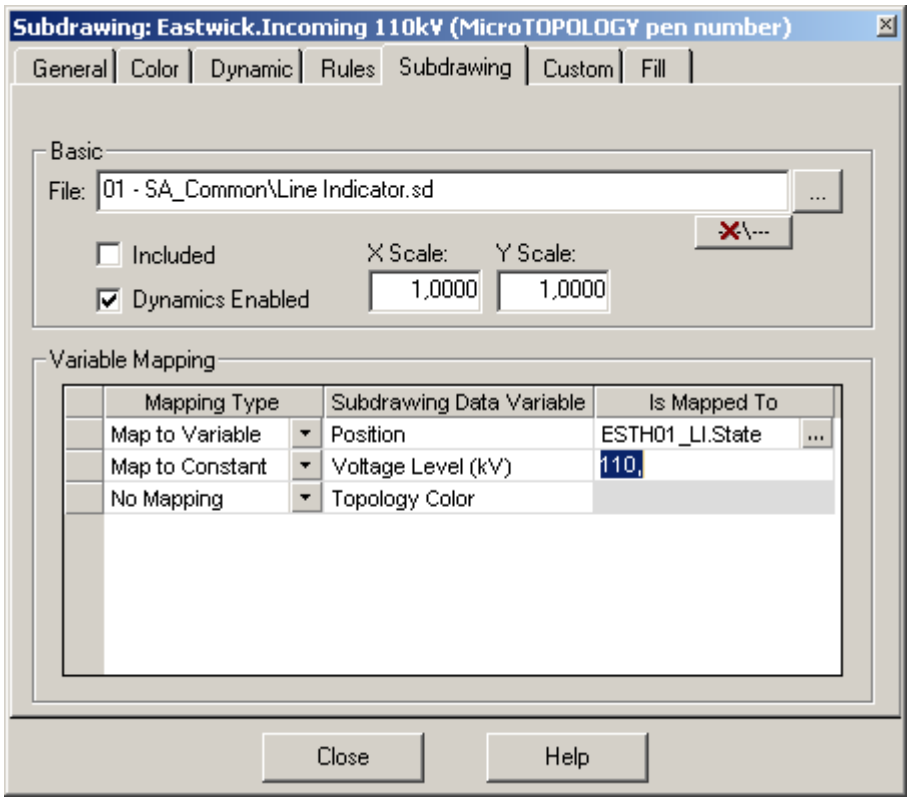


Figure 93: Mapping voltage level of line indicator to 110 kilovolts

Check that all the objects have voltage levels defined by selecting **Actions/Check Voltage Levels**. When the **Check Voltage Levels on Save** option Section 6.12) has been selected and the display has been saved, the dialog below is shown for every object that still does not have voltage levels defined. The mapping can be skipped by clicking **Skip** or **Skip All**.

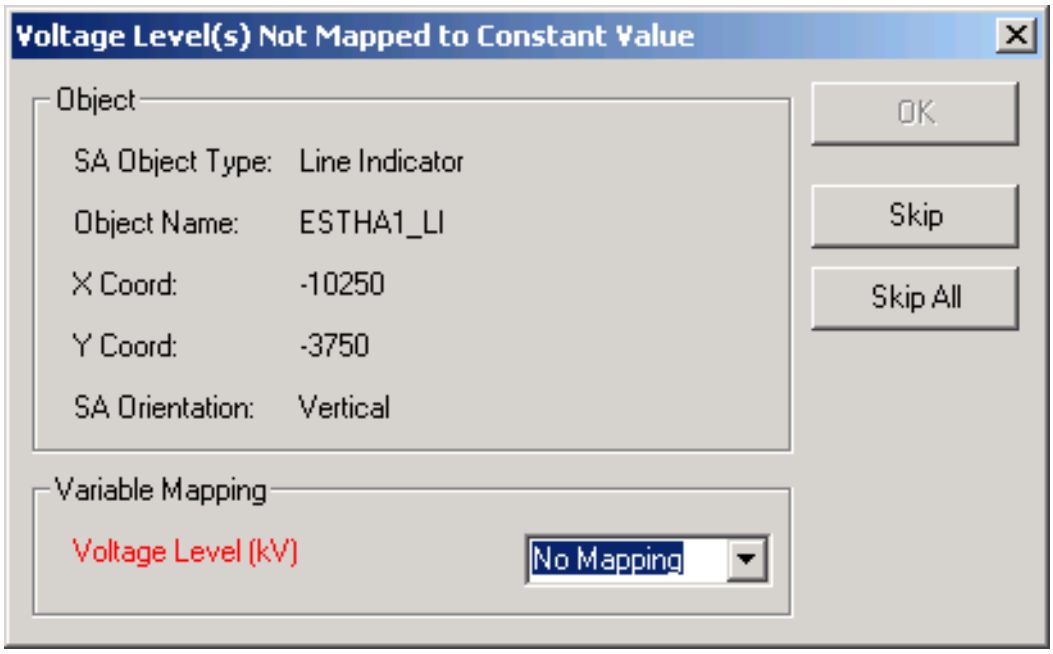


Figure 94: Defining voltage levels

Define the correct voltage levels to the drop-down combo boxes under the Variable Mapping field. The number of the variables to be mapped depends on the object type, for example 3-Winding Transformer has three variables to define.

If the value is illegal, the data variable's name is shown with red color. The color changes to black when legal voltage level is defined. The **OK** button is not enabled until all the data variables have a legal value.

6.3.3 Topology models

To manage topology models, select **Actions/Manage Topology Models...**. The dialog is used to create, update and delete topology models, see Figure 95.

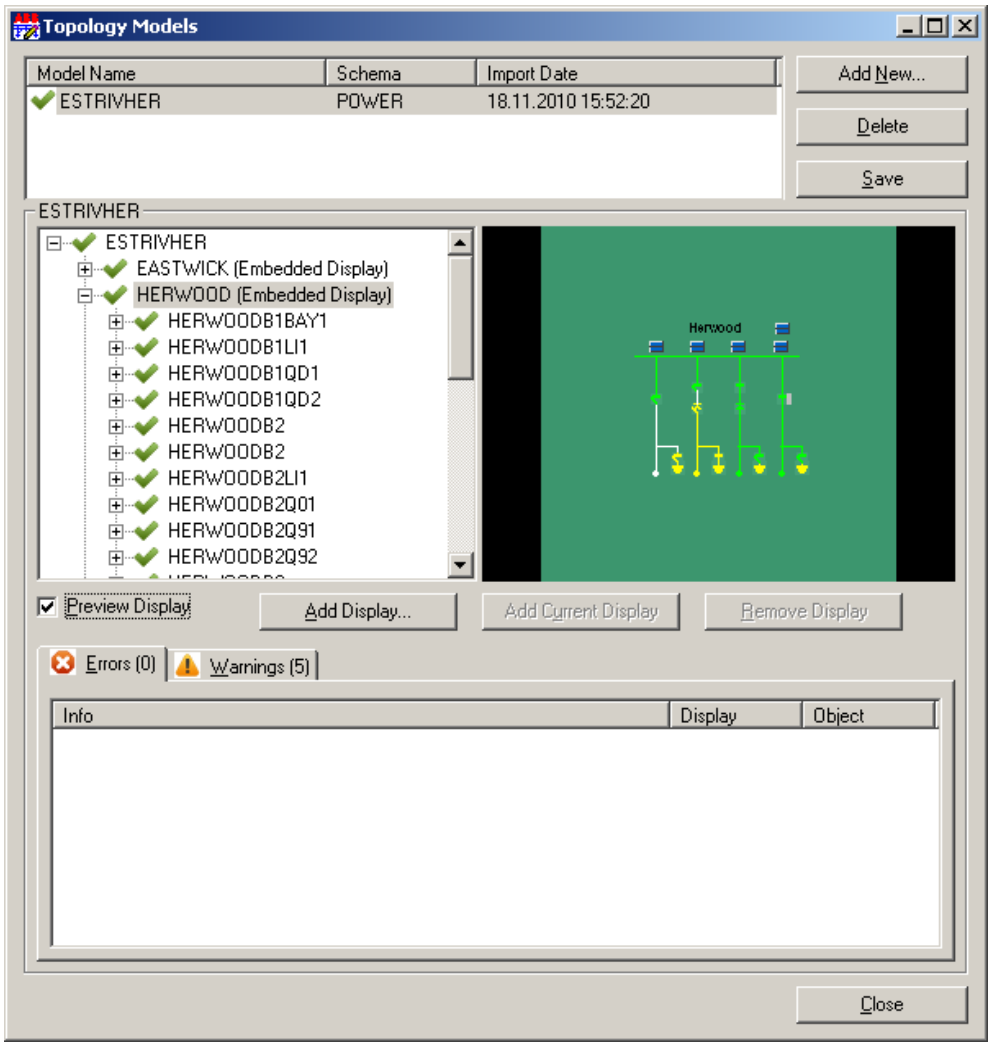


Figure 95: Showing topology information

The upper part of the dialog lists all topology models in the application. Models can be added, deleted and saved. A model can be selected by clicking the corresponding row.

The lower part of the dialog shows information about the selected model. Displays can be added to the model. Adding displays to the model adds all the process objects that are connected to items in the display to the model.

Some elements in a display can prevent the display from being added to the model. Some errors are detected when the display is added to the model and some are detected when the user clicks **Save**. For some errors, there is a quick help that provides some shortcuts for resolving the error. The quick help is displayed when the error row is clicked.

6.4 Deleting unmapped data variables

Unmapped data variables, sources and empty data sources are usually unnecessary and can increase the display size and loading time.

Select **Actions/Batch Operations/Delete Unmapped Data Variables** to delete all unmapped data variables and empty or unmapped data sources of the active process display. A confirmation dialog is shown before the deletion. It is not possible to restore the deleted items.

6.5 Creating names for objects

Display Builder can give names for objects that have been mapped to process database. The object names are formulated from the process object attributes by selecting the following menu items in **Actions/Batch Operations/Create Names for Selected Objects** submenu:

- **From LN:** Create names for selected objects from LN, for example, ESTH01_Q0.
- **From LN (TX)**: Create names for selected objects from LN and TX, for example, ESTH01_Q0 (Breaker position indication).
- **From OI:** Create names for selected objects from OI, for example, Eastwick.Incoming 110 kV.Q0.
- **From OI (TX):** Create names for selected objects from OI and TX, for example, Eastwick.Incoming 110 kV.Q0 (Breaker position indication).
- **Clear All**: Clear names of selected objects.

6.6 Highlight Clickable Objects

Display Builder can define custom attribute SAHighlight to all clickable objects. Monitor Pro uses highlighting for all such objects that have the specific custom attribute defined.

The default highlight functionality can be configured in \sc\prog\graphicsengine\etc \dvconfig.dat.

- **DVMOUSEOVERFILTER**: To use a custom highlight color set the value to the desired color index added by number 256. The default value for this setting is 272 which is the color index 272-256=16 (Frame Highlight color) in the loaded color scheme (clut file). Disable the highlight functionality by using value 2. In this case the custom attribute SAHighlight is ignored for all clickable objects.
- **HIGHLIGHT_SHADOW**: Shadow color for the highlight rectangle. The default value for this setting is 17 (Frame Shadow) in the loaded color scheme (clut file).
- **HIGHLIGHT_OFFSET**: Offset for the highlight rectangle. Negative value means absolute, positive means percentage of the box size (100 max). The default value for this setting is 0.
- **HIGHLIGHT_THICKNESS**: Thickness for the highlight rectangle. Positive value means absolute thickness. The default value for this setting is 2.

The DVMOUSEOVERFILTER needs to be enabled and the menu item **Actions/Options/On Save/Highlight Clickable Objects** toggled in order to have highlight functionality available. When changing the values in dvconfig.dat, Monitor Pro needs to be reopened.

6.7 Building DMS import file

When a process display is being built, the information about used busbars, switching devices, measurements, line indicators and other objects, the connectivity between them as well as the signal mapping information is saved into a display file (.v). The saved process display information is transferable into DMS600. For that purpose, Display Builder creates a transfer file, which can be directly imported into the DMS600.

The import file is saved in Import SYS600 Graphics.is6g format into the same directory as the display file.

To build a DMS import file:

1. Select **Actions/Build DMS Import File** or
2. Select **Actions/Options/Build DMS Import File on Save** and save the display file.

Display Builder can also send the build import file to DMS600 Network Editor for importing. This is done automatically after the file has been built, if option **Actions/Options/Send DMS Import File to Network Editor after Build** is selected.

The Send option can be selected if DMS600 4.0 or later is installed. Likewise the .is6g format files can be used for importing graphics in DMS600 4.0 or later. However, the import files can always be created regardless of the DMS600 installation. For more information on importing displays into the DMS600 and the .is6g file format, see the DMS600 System Administration manual.

6.8 Logical colors

Monitor Pro process displays use a system specific color palette of 256 color indexes that can be chosen from 16 million different RGB values. The RGB values are stored in Color LookUp Tables (.clut files) in \sc\apl\[appl]\PAR\APL\ColorSchemes. The color schemes are application specific. When a color is assigned to a graphical object, the Display Builder actually assigns only a color index. The RGB values for the color indexes are refreshed each time a process display is opened or Color Setting Tool is closed.

Color indexes from:

- 16 to 31 generally affect the displays and objects
- 32 to 55 are generally used to indicate the status of devices
- 56 to 99 are used for network topology coloring
- 110 to 119 are used to indicate the different tagout priorities

Network topology colors from:

- 56 to 63 are common network topology colors
- 64 to 71 are voltage source specific colors
- 72 to 79 are custom voltage source specific colors
- 80 to 99 are voltage level specific colors.

Colors 0 to 15, 120 to 223 and 240 to 255 are basic colors. Colors 224 to 239 are recommended for customer purposes. Detailed logical color information is listed in Table 47 below.

Table 47: Color indexes

Index	Name	Usage
Miscellaneous colors		
1	Tooltip	Background color of the tooltips
2-15	Reserved for Hitachi ABB Power Grids	Do not use
16	Frame Highlight	The light frame around some Power Process symbols
17	Frame Shadow	The dark frame around some Power Process symbols
19	Symbol Bounding Box	Rectangle on the background of most Power Process symbols
20	Unmapped Primary Winding	Transformer symbols' primary windings that are not mapped
21	Unmapped Secondary Winding	Transformer symbols' secondary windings that are not mapped
22	Unmapped Tertiary Winding	Transformer symbols' tertiary windings that are not mapped
23	Process Display Background	Default background for new displays
Table continues on next page		

24	Measurement Low Alarm	Measurement bar when value is below low alarm limit
25	Measurement Low Warning	Measurement bar when value is below low warning limit
27	Measurement High Warning	Measurement bar when value is above high warning limit
28	Measurement High Alarm	Measurement bar when value is above high alarm limit
29	Unmapped Device	Devices that are not mapped
30	Text	Default for new text objects
31	New object	General default for new objects
Status colors		
32	Normal	
34	Blocked	
35	Faulty or invalid Value	
37	Manually Entered	
38	Not Sampled or OPC Quality Bad	
39	Obsolete or Not Updating	
41	Selected	
42	Selected, Under Command (Blinking)	
43	Substituted	
44	Warning	
45	Alarm	
46	Alarm, Unacknowled (Blinking)	
47	Control Blocked	
48	Alarm, Unacknowledged BG	
49	Selected, Under Command BG	
50	Faulty Time	
55	Transparent color	
Network topology colors 56-99		
Common network topology colors		
56	Uncertain	One or more devices connected to sectionare in uncertain
57	Unpowered	No voltage in section
58	Earthed	Section is connected to ground
59	Looped	Two or more voltage sources form a loop
60	Powered	Default for powered sections
61	Error	Error situations, for example, powered section is earthed
62	Line Segments (when coloring disabled)	
63	Reserved for Hitachi ABB Power Grids	Do not use
Voltage source colors		

Table continues on next page

64	Generator	Section connected to generator
65	Incoming Line Indicator	Section connected to incoming line indicator
66	Reserved for Hitachi ABB Power Grids	Do not use
67	Primary Transformer Winding	Section connected to transformer's primary winding
68	Secondary Transformer Winding	Section connected to transformer's secondary winding
69	Tertiary Transformer Winding	Section connected to transformer's tertiary winding
70	Quarternary Transformer Winding	Section connected to transformer's quarternary winding
Custom voltage source colors		
72	Powered 1 (external color handling)	
73	Powered 2 (external color handling)	
74	Powered 3 (external color handling)	
75	Powered 4 (external color handling)	
76	Powered 5 (external color handling)	
Voltage level colors		
80	1st level (0 kV on default configuration)	
81	2nd level (3 kV on default configuration)	
82	3rd level (6 kV on default configuration)	
83	4th level (10 kV on default configuration)	
84	5th level (15 kV on default configuration)	
85	6th level (20 kV on default configuration)	
86	7th level (30 kV on default configuration)	
87	8th level (45 kV on default configuration)	
88	9th level (66 kV on default configuration)	
89	10th level (110 kV on default configuration)	
90	11th level (132 kV on default configuration)	
91	12th level (150 kV on default configuration)	
92	13th level (220 kV on default configuration)	

Table continues on next page

93	14th level (400 kV on default configuration)	
94	15th level (800 kV on default configuration)	
95	16th level (1600 kV on default configuration)	
96	17th level	
97	18th level	
98	19th level	
99	20th level	
Pipeline colors		
40	Drink Water	
51	Filtered Water	
52	Other Pipe	
53	Rain Water	
54	Raw Water	
63	Recycled Water	
71	Rinsed Air	
77	Rinsed Water	
78	Sludge Water	
79	Vented Water	
106	Pump state ON / enabled	
107	Pump state OFF / disabled	
State colors		
18	Local (Local / Remote)	
26	Remote (Local / Remote)	
33	Auto (Auto / Manual)	
36	Manual (Auto / Manual)	
100	Switching device/valve, Open	
101	Switching device/valve, Opening	
102	Switching device/valve, Close	
103	Switching device/valve, Closing	
104	Switching device/valve, Intermediate	
105	Switching device/valve, Faulty	
108	Generic state ON	
109	Generic state OFF	
Tagout priority colors		
110	Highest priority	Emergency
111	Very High	Alert

Table continues on next page

112	High	Critical
113	Medium	Error
114	Low	Warning
115	Very Low	Notice
116	Lowest priority	Information
117..119	Reserved for Hitachi ABB Power Grids	Do not use
147...159	Reserved for Hitachi ABB Power Grids	Do not use

6.9 Applying color schemes

Color scheme represents the colors for each elements in all of the displays. In SYS600, the available color schemes are black, blue, brown, dark grey and grey.

6.9.1 Color Setting Tool

Color Setting Tool is used for changing the RGB values of logical colors. The tool contains tabs for status colors, network topology colors and miscellaneous colors, which are related to the Process display. There are also dedicated tab sheets for rest of the displays (**Alarm**/**Event**/**Blocking**/**Trends**/**Measurement Reports** display).

All logical colors are shown in the tabs as a color box and as RGB value. Clicking the color box opens a **Color Selection** dialog for specifying new RGB value. Clicking **OK** saves changes and exits the tool. Clicking **Cancel** exits the tool without saving the changes.

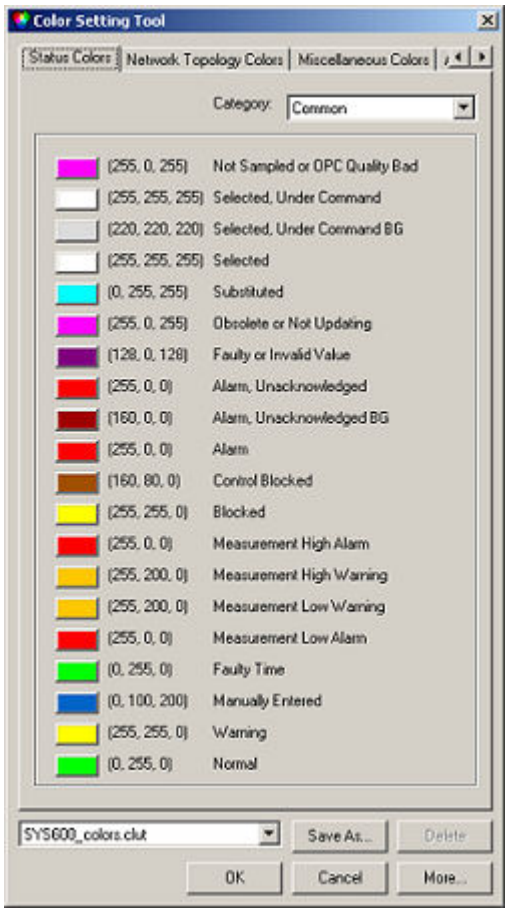


Figure 96: Color Setting Tool

6.9.2 Modifying the color settings with the Color Setting Tool

The user must have at least authorization level 2 in authorization group in TOOLS in order to make modifications to color schemes.

Only one Color Setting Tool instance at the time is allowed to edit a color scheme. If the used color scheme is being edited by someone else when the Color Setting Tool is started, the user will be informed that the file is locked, and the Color Setting Tool can be opened either in read only mode or not opened at all.

6.9.2.1 Modifying the color settings in Process display

To modify the color settings in the Process display:

1. Open **Color Setting Tool** by selecting **Tools/Engineering Tools/Color Setting Tool** in Monitor Pro.
2. Select either the **Status Colors**, **Network Topology Colors** or **Miscellaneous Colors** tab, and the category.

The categories for status colors are ***common*** and ***tagout***, and for network topology colors the categories are Common, Voltage source and Voltage level. The color scheme also includes miscellaneous colors, for example, for the background and text.

3. Click the color field.
4. Specify the color values in the **Color** dialog, and click **OK**.

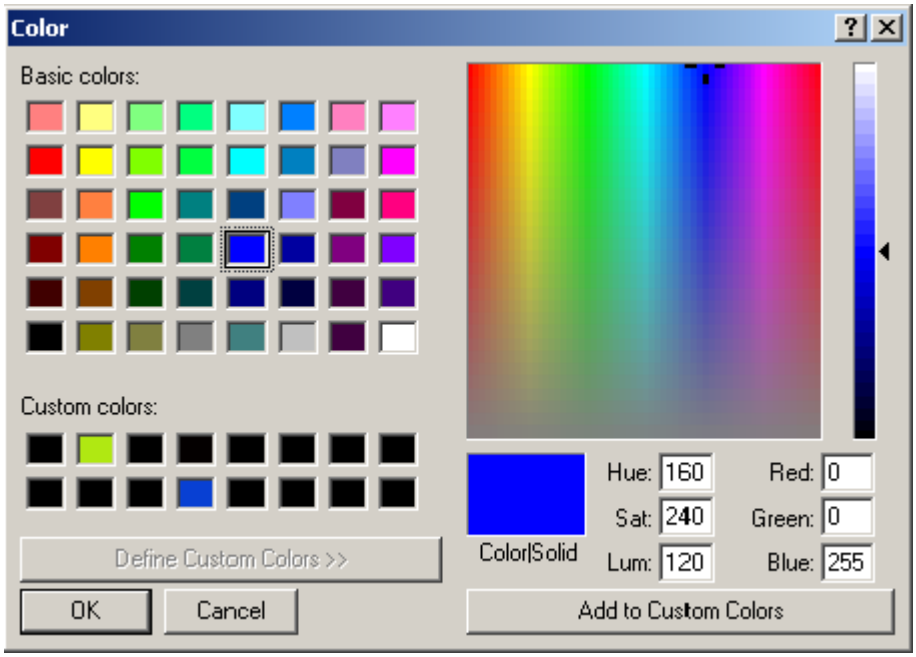


Figure 97: Color dialog

6.9.2.2 Modifying the color settings in Alarm, Event and Blocking displays

To modify the color settings in the Alarm, Event or Blocking display:

1. Select **Settings/Display Settings.../Color Settings...**.
2. In the **Color Settings** dialog, click **More**. Select the **Use color scheme** check box if the color should be used from the color scheme.
3. Click **Edit...**.
4. In the **Color Setting Tool**, click the color field for any of the available colors.
5. Specify the color values in the **Color** dialog and click **OK**.
6. Click **OK** to save the changes.

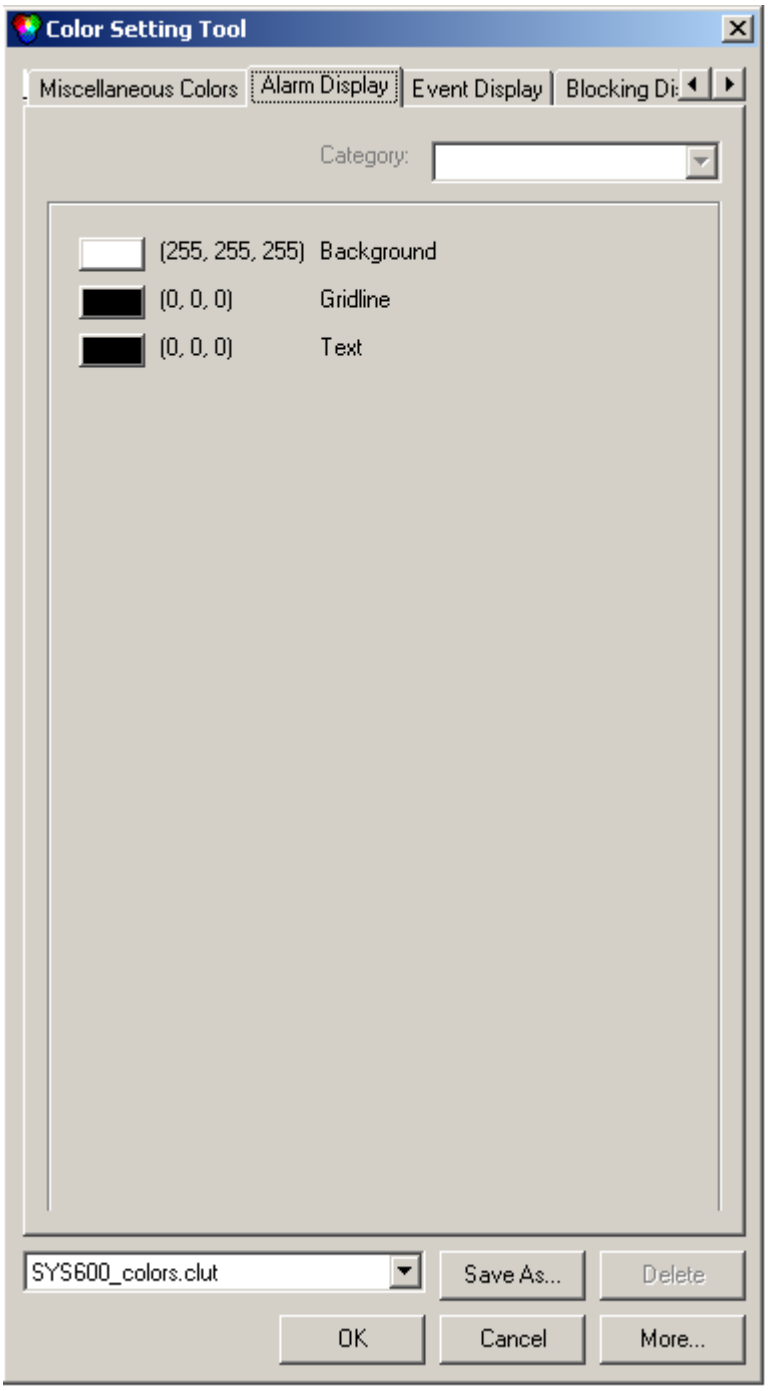


Figure 98: Color Setting Tool for specifying background, gridline and text colors

6.1. In the **Color Setting Tool** dialog, click the color field to edit background, gridline or text colors.
6.2. Specify the color values in the **Color** dialog, and click **OK**.

7. Modify the colors for Filter in use color, Frozen mode color and Day Break color.

Filter in use color and Frozen mode color are not applicable in the Blocking display. Also, Day Break color is only available in the Event display.

7.1. Click **Edit**.
7.2. Specify the color values in the **Color** dialog, and click **OK**.

8. In the **Color Settings** dialog, accept the changes by clicking **OK**.

6.9.2.3 Modifying the color settings in Trends and Measurement Reports displays

To modify the color settings in the Trends and Measurement Reports displays:

1. Select **Settings/Display Settings...**.
2. In the **Properties** dialog, on the **Common settings** tab, select the **Use color scheme** check box if the color should be included in the Color Scheme.

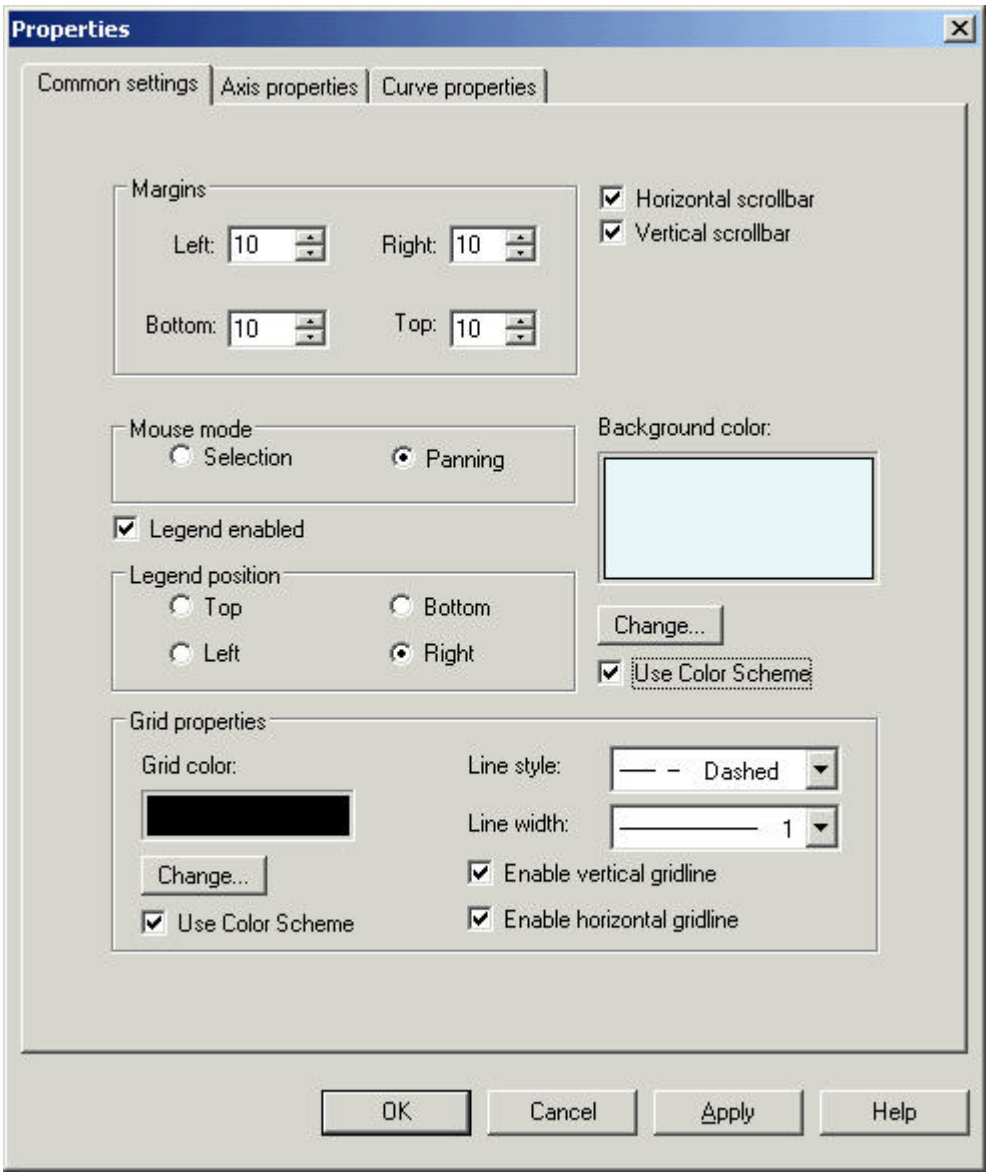


Figure 99: Properties dialog

3. Click **Change**.
4. In the **Color Setting Tool** dialog, select **Graphical View** in the **Category** list.
5. Click the color field for any of the available colors.
6. Specify the color values in the **Color** dialog, and click **OK**.

7. In the **Category** list, select **Tabular View** and change the colors in the same way.
8. Edit the settings on the **Axis properties** and **Curve Properties** tabs if necessary.
9. Click **OK** to save the changes.

6.9.3 Saving a new color scheme

A modified color scheme can be saved as a new one if the user wants to, for example, leave the current color scheme untouched and create a new color scheme based on the modifications.

To save a new color scheme:

1. In the **Color Setting Tool** dialog, select **Save As...**.
2. In the **Color Setting Tool/Save Color Scheme** dialog, specify the name for the color scheme.
3. Click **OK**.

New color schemes are available for all users of a specific SYS600 application.

6.9.4 Loading of a new color scheme

The effect of changing color scheme applies to all displays in SYS600 Monitor Pro in a specific SYS600 application. A new color scheme can be loaded by changing the selection in the **Color Setting Tool**.

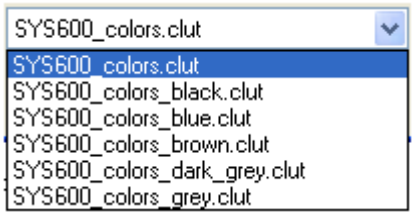


Figure 100: Changing a color scheme

To load a new color scheme, select the color scheme file, for example SYS600_colors.clut, in the list at the bottom of the **Color Setting Tool** dialog.

6.9.5 Exporting and importing color scheme colors

Scheme colors can be exported and imported with the **Color Setting Tool**.

Exporting saves all colors in the color scheme to an .ini file. Importing replaces the appropriate colors in the color scheme with .ini file definitions. It is possible to import only certain colors to a currently loaded color scheme.

To export or import color scheme colors:

1. In the **Color Setting Tool** dialog, select **More**.
2. To import a color, select **Import**.
   Select the .ini file and click **Open**.
3. To export a color, select **Export**.
   Give a name for the .ini file and click **Save**.

6.9.6 Restoring default color scheme settings

The installation defaults can be restored from the corresponding default color scheme file to the current one.

To restore default color scheme settings:

1. In the **Color Setting Tool** dialog, select **More**.
2. Select **Restore Defaults**.
3. Click **Yes**.

6.10 Migration log

Process displays and symbols available in Monitor Pro must be migrated before they can be used in Monitor Pro+. The migration procedure is handled by a Migration Tool started by SYS600. Existing displays and symbols that have not been converted earlier will be handled automatically. When a display file or subdrawing is modified in Display Builder, it will be converted to an xml view available in Monitor Pro+ as well.

To display logged information from the conversion, select **Actions/Migration Log...**.

6.10.1 Migration notifications

A summary of error and warning messages from the application and system-wide display and symbol conversion is displayed in the notifications view. A detailed conversion log can be opened by double-clicking or right-clicking the chosen file and selecting **Open File**. A full migration log is also accessible from the notifications view.

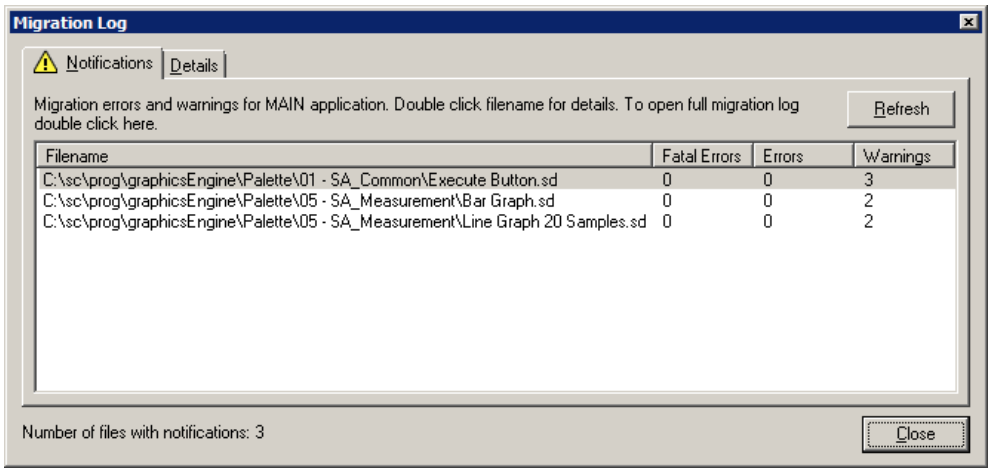


Figure 101: Migration notifications

6.10.2 Migration details

Detailed error and warning messages from files opened into Display Builder are shown here divided to separate tabs based on message severity. The active file open in Display Builder can be switched by double-clicking or right-clicking the chosen file and selecting **Switch File**.

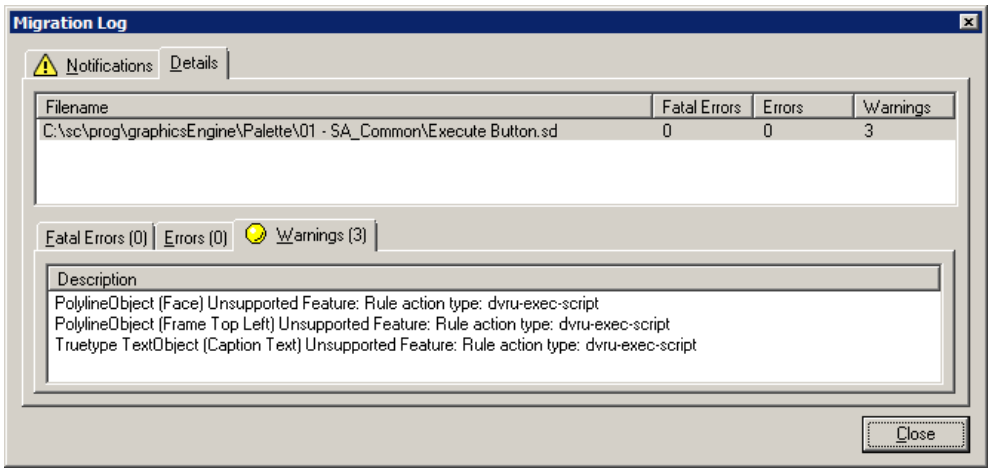


Figure 102: Migration details

6.11 Monitor Pro informative data variables

There are some data variables related to Monitor Pro that have a special meaning. These data variables should be created to the MicroSCADA data source named DisplayInfo, and they can be used for revealing runtime information related to the current user and Monitor Pro application window properties.

- UserName: User logged to Monitor Pro.
- UserLanguage: Primary language ID of the user.
- LowerLeftX, LowerLeftY, UpperRightX, UpperRightY: Dimensions of the current Process Display in world coordinates (same coordinate system as with Zoom Areas).

6.12 Default options

The **Actions** menu contains selectable options that Display Builder makes automatically. These options can be marked by selecting **Actions/Options** The selection is on until it is deselected.

- **Open Monitor Pro Active File on Startup**: When this option is selected, the Monitor Pro Active File is opened at the Display Builder startup.
- **Add New Objects to Topology by Default**: The user can define Display Builder to add new objects to the topology by default.
- **Check Voltage Levels on Save**: When this option is selected, Display Builder checks the voltage levels when saving the display. If some of the objects do not have voltage level or correct mapping defined, Display Builder asks the user to define the voltage level. To define voltage levels, see Section 6.3.2.
- **Delete Unmapped Data Variables on Save**: When this option is selected, Display Builder deletes unmapped data variables and sources and empty data sources on the display on save. If the automatic deletion has not been selected and the display contains unmapped data variables or sources, a warning message is shown.
- Highlight Clickable Objects: When this option is selected, the Monitor Pro uses highlighting for all clickable objects, see Section 6.6.
- **Build DMS Import File on Save**: When this option is selected, Display Builder generates on display file save an import file that can be used for importing the process display into DMS600. The import file is saved into the same directory as the display file.

- **Migration Log on Save**: When this option is selected, Display Builder opens the Migration Log dialog and selects the details view after saving the display.
- **Send DMS Import File to Network Editor after Build**: When this option is selected, Display Builder sends the built import file automatically to DMS600 Network Editor for importing. This option can be selected if DMS600 4.0 or later is installed.
- **Double Click Action**: The user can define the action that will occur when an object is double-clicked. The double-click can open the Object Properties dialog or snap the symbol to the nearest object.

Section 7 Dynamic data properties

Display Builder lets the user create and control dynamics, which are graphical components that change in response to changes in the user's data. These dynamic components include graphs, subdrawings, graphical objects whose shape, position, attributes, and size change, and input objects that gather data directly from the user and topology actions. While dynamic components show changing information, static portions of the Display Builder display provide a background or additional context within the display.

To show changes in data using dynamic objects, Display Builder must first gather the data from an existing source, store it temporarily, and then show it using dynamic objects. Gathering the data and storing it is handled using the data sources and data variables. The data source describes where the data can be obtained, such as from a file. The data variables describe the information that the data represents in detail.

7.1 Data sources

A data source is a single source of incoming data. Display Builder can use input data from process database or data that is stored in files, input from a user interaction, or input that is a set of one or more predefined, constant values. These inputs are called data sources. Once the user has defined their data source, the data is read in as a stream of numbers. This stream can represent one or more variables.

Display Builder supports the following data source types:

- MicroSCADA
- File
- Memory
- Constant

The user can create, modify, and delete data sources if necessary. However, a data source whose variables are attached to an object cannot be deleted. To do this, the objects must first be deleted or the variables must be deleted from the objects, and then the data source can be deleted.

In Display Builder, the user controls what data sources are used in the display, just as they control what graphical objects are in the display. A major difference between them is that to add a graphical object to a display, the graphical object is created in the interface. To add a data source to a display, the user describes the data source, which may already exist.

7.1.1 Creating data source

The Data Selection dialog is shown when the user explicitly displays it, whenever Display Builder requires the user to select a data variable, or whenever the user chooses to add or change a data variable attached to a Display Builder component, such as a dynamic object, dynamic feature, or rule. Depending on the context in which the data selection is shown, it shows either the Data Source List alone, or the Data Source List together with the data configuration information on the right.

To show the Data Selection dialog, select **Edit/Data Variables**. The data selection is shown next to the data source options.

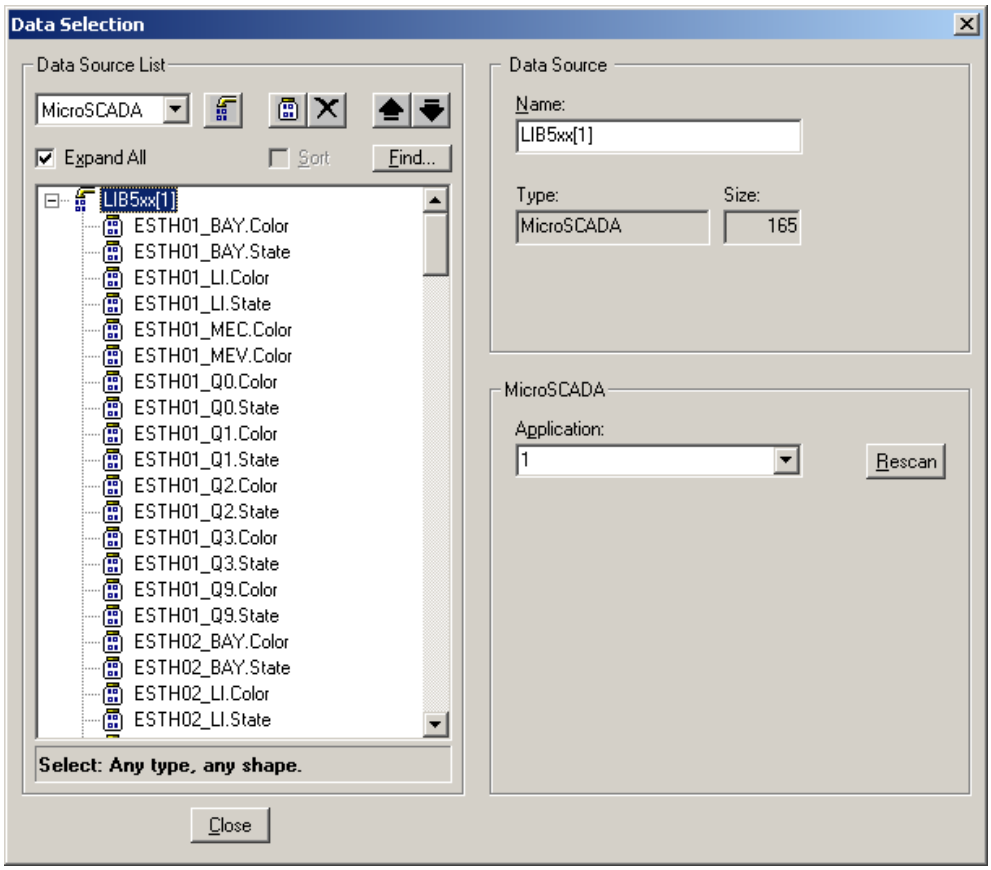


Figure 103: Data Selection dialog

Many dialogs show data variable assignments. For example, graph variables are shown in the **Graph Properties** dialog, and variable assignments for object dynamics are shown in the **Object Properties** dialog. If the user adds or changes variables in these dialogs, the **Data Selection** dialog shows only the information under the Data Source List when it is shown. The configuration information area can be opened and closed by using the **Edit** button.

7.1.2 Data source editing

The Data selection shows a hierarchy of the user's data sources and their variables. Both the data sources and their data variables can be shown in the hierarchy, or the hierarchy can be collapsed to show only the data sources by checking the Expand All check box. The hierarchy of any particular data source can be collapsed or expanded by double-clicking its name in the hierarchy.

To add a data source:

1. Open the **Data Selection** dialog.
2. Click the **New Data Source** drop-down menu and select a data source type from the list. For descriptions of the data types, see Section 7.1.3.
3. Click on the **Add New Data Source** button next to the New Data Source drop-down menu.
4. Click **Rescan** in the MicroSCADA field both to scan the configuration files and to update and save the index configurations.

A new data source is shown in the data source list. If the user creates a Constant data source, the Constant data source is shown in the hierarchy.

7.1.3 Data source types

The data source type refers to the source of the data used. The data source types are file, memory, constant and MicroSCADA. The type of a data source cannot be changed. To use a different type of data source, one of the correct type must be created.

The information on provided data source types:

- **MicroSCADA**: The MicroSCADA data source is used to establish a connection to MicroSCADA OPC server. The MicroSCADA data source enables the use of the Power Process functionality described in Section 6. To use this functionality with data variables, see Section 7.2.
- **File**: The user can show the data contained in any file they have permission to read. Display Builder uses files that are either ASCII numbers separated by spaces, tabs, newlines, or binary data. Text variables are of fixed length.
- **Memory**: The user can define data sources that receive their data from other sources within Display Builder. These include memory data sources used by the Display Builder input objects or control objects. In the display, the user can programmatically access a wider range of sources, including sources not available in the editor.
- **Constant**: The user can define data sources that contain only constant values. The value is set when the user defines the constant and does not change while the display runs. One use for a constant in a graph is as a reference point with which to compare other changing values.
  The dialog options at the right side of the palette let the user describe the following characteristics of the data source:
  - **Format**: The data is either in ASCII or binary format. This applies only to file data sources. See Section 7.1.3.3.
  - **File**: Location where the data is stored. This applies to file data sources. See Section 7.1.3.4.

    Data from the file data source is only read when the display is opened. Further updates in the file data source are not updated to the display until it is opened again.
  - **Name**: The name by which the data source is known within the graphics engine. Applies to MicroSCADA, Memory, and Constant data sources. See Section 7.1.3.3.

7.1.3.1 Specifying MicroSCADA data source

It is possible to customize the process object indexes that graphical symbols use for indicating status color. For more information about customizing process object indexes, see the Application Design manual.

7.1.3.2 Specifying file data source format

The Format option indicates whether the data is in ASCII or binary format. File data sources can contain either ASCII or binary data, but the two cannot be mixed in a single data source. However, data from both ASCII and binary data sources can be shown in the same graph.

ASCII data is stored as float data or text strings. Binary data can be read and stored as float or integer data of various sizes, or as text strings. Thus, binary format gives the user more control over how the data is read, ASCII data is converted to internal float numbers. Because conversion to floating point numbers is time consuming, it is recommend to produce binary data when using very large files or when speed is important.

To specify the format of a data source:

1. Open the **Data Selection** dialog.
2. Select the appropriate radio button under the **Format** field.

7.1.3.3 Specifying file data source name

The name of a data source is used to identify it within Display Builder. When the user edits the name in the Data Source field, the user's changes are reflected in the data source name in the hierarchy on the left. Names can be assigned for function, memory, and constant data sources.

For file data sources, Display Builder constructs a data source name by stripping all directory information from the origin. This means that the name of a file data source is its file name. The name of a file data source changes automatically when a different origin is specified. If two files with the same name are specified and they are saved to different directories, Display Builder keeps track of their original names, even though the names that are shown on the screen are identical.

Default data source names:

- MicroSCADA data source: MicroSCADA
- File data source: default.dat
- Memory data source: default.mem
- Constant data source: constant

To change the name of a data source:

1. Show the Data Selection dialog.
2. Edit the name in the data source hierarchy.

7.1.3.4 Specifying file data source origin

The origin of a File data source is the file specification in the user's system from which the data is taken. This can be the full path and file name or a path and file name relative to the user's current directory. When the name is changed in the text box, the changes are reflected in the data source name in the hierarchy on the left.

The user can use data from files that reside on their system or are accessed through their network, as long as the user has either read or execute permission for them. When a File data source is created, the editor uses a default origin with system supplied data. The default origin can be changed with the path and file name of the user's own data. The search path in the run script can be modified to access a different file.

To change the origin of a File data source:

1. Show the **Data Selection** dialog.
2. Edit the name in the File field or click on the ellipsis button to show the **File Datasource Origin** dialog.
3. Select a data file.
4. Click **Open**.

7.1.3.5 Data in file data sources

When the graphics engine reads a file, it assigns the first value(s) read to the first variable in that data source. If there are more variables, the next value(s) are assigned to the second variable, and so forth, until it has read values into each variable.

The number of values assigned to each variable corresponds to the shape of the variable. If a variable is a vector or a matrix, the engine reads one value for each element of the vector or the matrix. The same data in a data source could be defined as six scalar variables, a 3 x 2 matrix, or a vector of length 6. The engine reads the data values one number after another, places each

value in the next variable or variable element, and stops when it has filled each variable once. The engine then displays the first set of data as specified in your view.

When the engine reads the next set of data, it starts from the next value in the file. If it runs out of data, for example by reaching the end of the file, the engine uses the last value read for all subsequent values in that particular run.

The following table shows how a stream of ascending integers (1 2 3 4 ...) is divided to fill the assigned variables.

Table 48: Values of assigned variables

Variables	Shape	1st read	2nd read
Var:1	Vector[3]	1 2 3	10 11 12
Var:2	Scalar	4	13
Var:3	Matrix[2,2]	5 6 7 8	14 15 16 17
Var:4	Scalar	9	18

If the variable is a text variable, characters are read in without conversion to numbers. These text strings can then be displayed using the text graph type.

When the data stream is broken up, each datum is assigned to only one variable. The same data can be displayed by more than one dynamic object. The data from the same stream can be interpreted differently by specifying multiple data sources with the same origin and letting each data source define the stream differently. For example, if a data stream contains a vector for each variable, one graph can display all elements of the vector while another displays only the first element. Alternately, a single data stream can be displayed differently by multiple dynamic objects by defining different dynamic features for each object.

7.2 Data variables

While the data source characteristics describe the source of the data, the variables describe the data itself: its organization, type, name, and in the case of a constant, its value.

7.2.1 Creating data variables

To add a data variable to a data source:

1. Show the **Data Selection** dialog by selecting **Edit/Data Variables**.
2. Select the data source name from the hierarchy.
3. Click **New Variable** button. This adds a variable and displays settings specific to the variable.

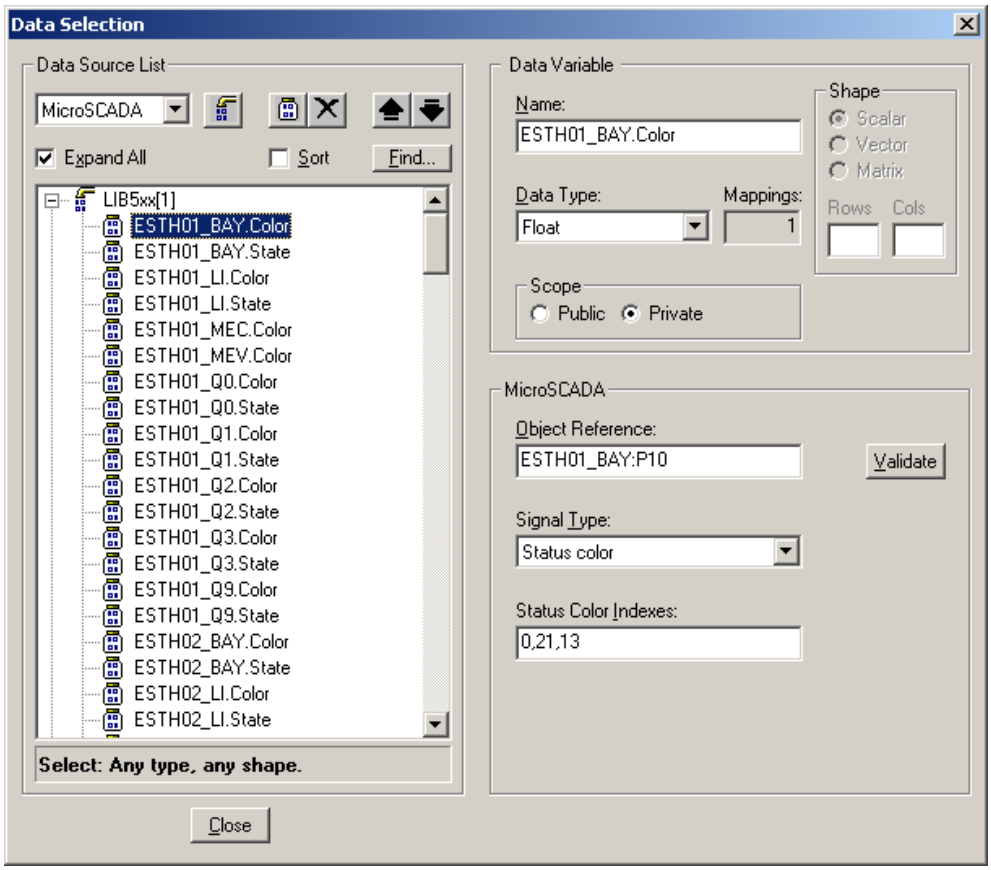


Figure 104: Adding a new data variable

7.2.2 Data variable editing

Data variables sort the data stream supplied by the data source into separate buffers in preparation for display. Display Builder lets the user describe the organization of the data by specifying the variables the user wants to sort it into.

7.2.2.1 Specifying characteristics

Display Builder adds the data variable using default characteristics. The user can change the defaults and add additional variables to match the characteristics and organization of the data from their data source. When using data variables, the user can describe the following characteristics of the data:

- Scope: Whether a variable is private or public. Mapping is only possible for variables that are defined as public in the original display, so it is important to set the scope correctly for data variables in displays to be used as active subdrawings.
- Name: The name of the data variable or constant.
- Data type: The classification of the data, such as text or an integer value.
- Value: The value of a constant in a constant data source.
- Shape: The number of elements in the variable or constant.
- Length: Define the length of the text type data variable. By default, the maximum length of the data variable is 255 characters.

7.2.2.2 Specifying name

The variable name is used to identify the variable in the Display Builder menus and in graph legends. Display Builder assigns default names of Var:1, Var:2, Var:3 and so on in sequence according to the total number of variables created in the drawing.

After creating a variable, the user can change its default name to something more descriptive without affecting the data flow. When the name is edited in the text box, the changes are reflected in the data source name in the hierarchy on the left. The name changes in every place where it is referenced on the display.

To specify the name of a data variable:

1. Show the **Data Selection** dialog by selecting **Edit/Data Variables**.
2. Edit the name in the Name field.

7.2.2.3 Specifying type

The data in the data source is stored in one of the formats listed in Section 7.2.3. The options in the **Type** drop-down menu depends on the format of the data source where the variable belongs to. File data sources supply either ASCII or binary data.

If the data is ASCII, all data is interpreted either as text strings or as numbers which are converted into floating point numbers for display. If the data is binary, the data can be interpreted as any of the data types, see Section 7.2.3. Note that all data in one file must be of the same type. The default variable data type is Float for both ASCII and binary data variables.

To specify the type of a data variable:

1. Show the **Data Selection** dialog.
2. Select the **Type** drop-down menu.
3. Click the right type.

7.2.2.4 Specifying scope

The scope of a variable indicates whether the variable is private or public. This is useful if the display will be used as an active subdrawing. Only the public data variables of a subdrawing can be mapped to constant values or data variables in the display containing the subdrawing.

To specify the scope of a data variable:

1. Show the **Data Selection** dialog by selecting **Edit/Data Variables**.
2. Select the appropriate radio button under Scope.

The default scope is Private.

7.2.2.5 Specifying value of constant data variable

Constants are assigned values that do not change when the user runs a display. When the value of a constant is changed, its value changes every place where it is referenced in the display. For example, if the user creates a constant variable with a value of 2.59, uses it in several graphs, and changes its value to 157, its value changes to 157 in every graph that uses that constant.

7.2.2.6 Specifying shape of data variable

The shape of a data variable or a constant is composed of the number of data values or elements. If the user selects the Vector or Matrix option, the number of rows and columns can be specified to the text boxes under the Shape field.

Display Builder can use the following shapes:

- Scalar: A single value.
- Vector: A one-dimensional array of numbers.
- Matrix: A two-dimensional array of numbers default variable shape is Scalar.

Different types of dynamic objects are useful for displaying different kinds of data. The following table suggests some types of graphs and object dynamics to display the user's data.

Table 49: Data variable shapes

Data shape	Display as
Vector or matrix	Use a graph. Object dynamics use only scalar variables.
Scalar	Use a graph or a graphical object with dynamics. The user may want to use a dial graph, a line graph, or a graphical object with color dynamics.

7.2.2.7 Specifying MicroSCADA data variable settings

The data variables created under the MicroSCADA data source can have specified functions. The user can define the type of the variable to be State indication, Status color or Generic signal when mapping different symbols to the process objects. For more information on mapping variables, see Section 7.2.4.

The State indication variable type enables the user to switch values for state signal. The values can be changed by clicking one of the option buttons under Value swapping for state indication field, see (Figure 104).

7.2.3 Data variable types

Display Builder supports the following data variable types:

- Float: Floating point number. Numbers are interpreted correctly whether or not they have a decimal point.
- Double: A double precision floating point number.
- Long: Long integer, usually 4 bytes (32 bits), but system dependent.
- Ulong: Unsigned long, a long integer whose range begins at zero and is always positive.
- Short: Short integer, usually 2 bytes (16 bits), but system dependent.
- Ushort: Unsigned short integer, a short integer whose range begins at zero and is always positive.
- Byte: An integer the size of a byte (8 bits) whose range is from -128 to 127.
- Ubyte: An unsigned byte, an integer the size of a byte (8 bits) whose range is from 0 to 255.
- Text: Fixed length string of characters. The length can be specified.

7.2.4 Mapping subdrawing variables

When creating a subdrawing, all data sources and dynamics in the subdrawing's original display are included in the subdrawing object. When the subdrawing's dynamics are enabled, all dynamic activity in the subdrawing behaves as in the original display when the user runs the display. Nothing has to be done to its internal data variables.

The user can, however, change the data used by a subdrawing by mapping its variables to constant values or to data variables in the current display. When a variable in a subdrawing is mapped, it uses the specified constant value or gets data from a data source in the current display.

Mapping provides several benefits:

- The user can specify what values or data sources in the current display should drive dynamics in the active subdrawing.
- The user can use multiple instances of the same subdrawing to display different data.
- In the user's application, it is easy to get the user's program data into the active subdrawings by programmatically stuffing the buffers of the data variables.

Mapping is only possible for variables that are defined as public in the original display, so it is important to set the scope correctly for data variables in displays to be used as active subdrawings.

To map the public variables of a subdrawing:

1. Select the object.
2. In the **Object Properties** dialog, select the **Subdrawing** tab.

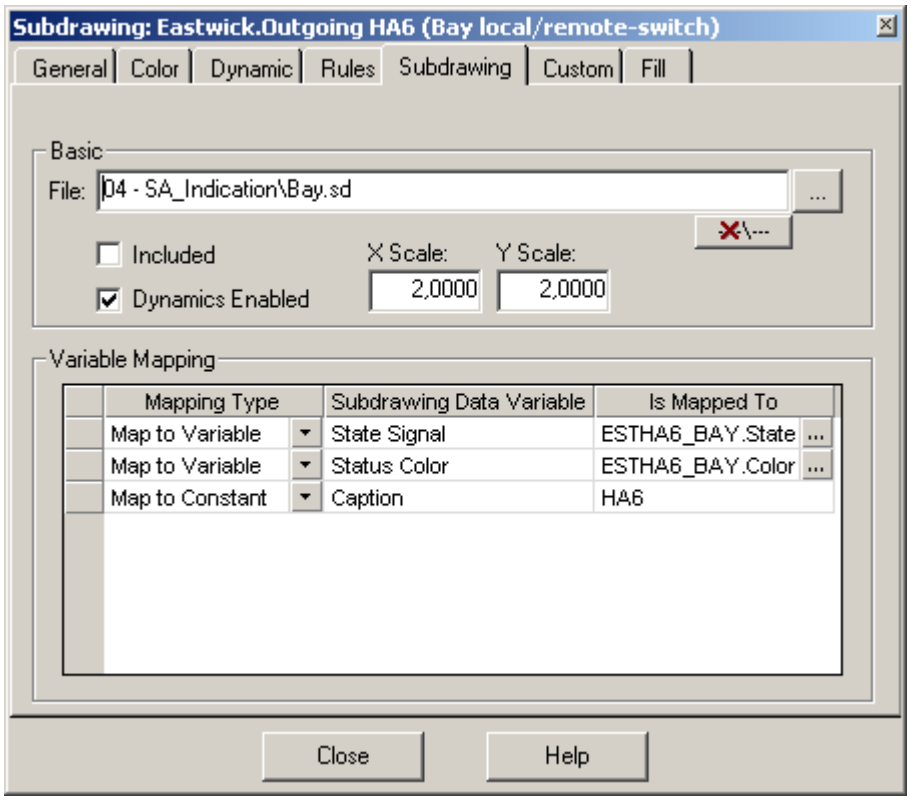


Figure 105: Mapping public variables

The public variables used in the subdrawing display are shown under the Variable Mapping field depicting to which display variable each subdrawing variable is mapped. The Mapping Type column shows whether or not the variable is currently mapped, and provides choices for the type of mapping to do. The Subdrawing Data Variable column for each variable indicates the type of data for that variable.

The mapping types are No Mapping, Map to Constant and Map to Variable. Map to Constant is convenient, because it lets the user map a subdrawing variable to a constant value without having to create a constant data source variable to map it to.

If there are no variables in the subdrawing, the message that subdrawing has no global variables is shown in the Variable Mapping list. If the subdrawing has variables but none of them are public, the message is not shown but the list is empty.

To map a variable in the subdrawing display to a constant value:

1. Select the variable name in the **Subdrawing Data Variable** column.
2. Click **Mapping Type** from the drop-down menu and select **Map to Constant**.
3. Enter the constant value in the **Is Mapped To** column.

To map a variable in the subdrawing display to a variable in the current display:

1. Select the variable name in the **Subdrawing Data Variable** column.
2. Click Mapping Type drop-down menu and select **Map to Variable**. This shows the data selection.
3. Select a variable. If the desired variable is not shown in the **Data Source List**, the user can expand the dialog and edit the list.
4. Click **OK**. The selected variable is shown in the **Is Mapped To** column.

The subdrawing variables must be mapped to display variables with the same data type, although the type of data source can be different.

Some display objects can be mapped only to a certain type of variable. For example, Tripping Tag has to be mapped to Status Color, not for instance to Status State.

The display variable must have at least as many data elements as the subdrawing variable. When the user selects a display variable for any subdrawing variable, the subdrawing displays the data of the current display variable instead of the data of the original subdrawing variable.

If the user maps a subdrawing variable to a constant, the data type is less restrictive. If the subdrawing variable is a text variable, the constant value is interpreted as a text string. If the subdrawing variable is numeric, the text box for the constant value only allows numeric entry.

To change the variable's mapping destination:

1. Click the ellipsis button to the right of the variable name. The **Data Selection** dialog opens.
2. Select a different display variable.

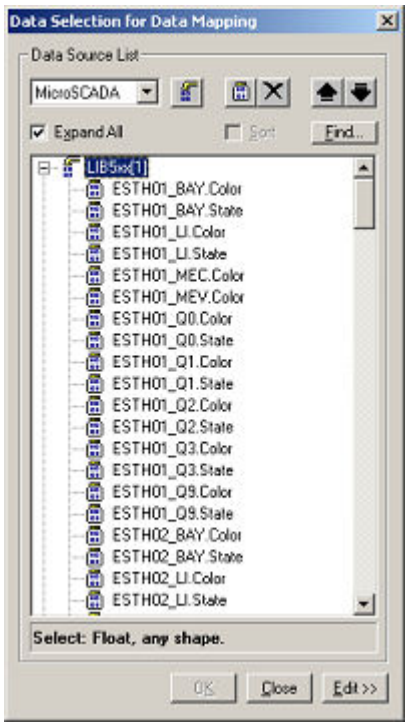


Figure 106: Changing variable mapping

To unmap a mapped variable, double-click the display variable name. This clears the display variable so that the subdrawing displays the data from its internal variable.

If the user is building a display that uses nested layers of active subdrawings, the variables must be made public at the lowest layer. At each subsequent layer, the user must map the variables to other public variables if they should be available for mapping at the top layer.

7.3 Object dynamics

Dynamic components can be added to a graphical object so that changes occur to the object when the data changes.

Display Builder provides the following dynamic components with dynamic features:

- Attribute dynamics: Changes properties such as color, line type, line width, curve type, and arc direction.
- Motion dynamics: Changes position, rotation, and scale.
- Subdrawing dynamics: Displays different subdrawings depending on a data value.
- Visibility dynamics: Changes whether or not the object is visible.
- Text dynamics: Changes the content of a text object to show the data value.
- Blinking dynamics: Alternates the colors of an object between two sets of colors to make the object blink. An object can blink or not blink, or blink in different colors, depending on the data value.

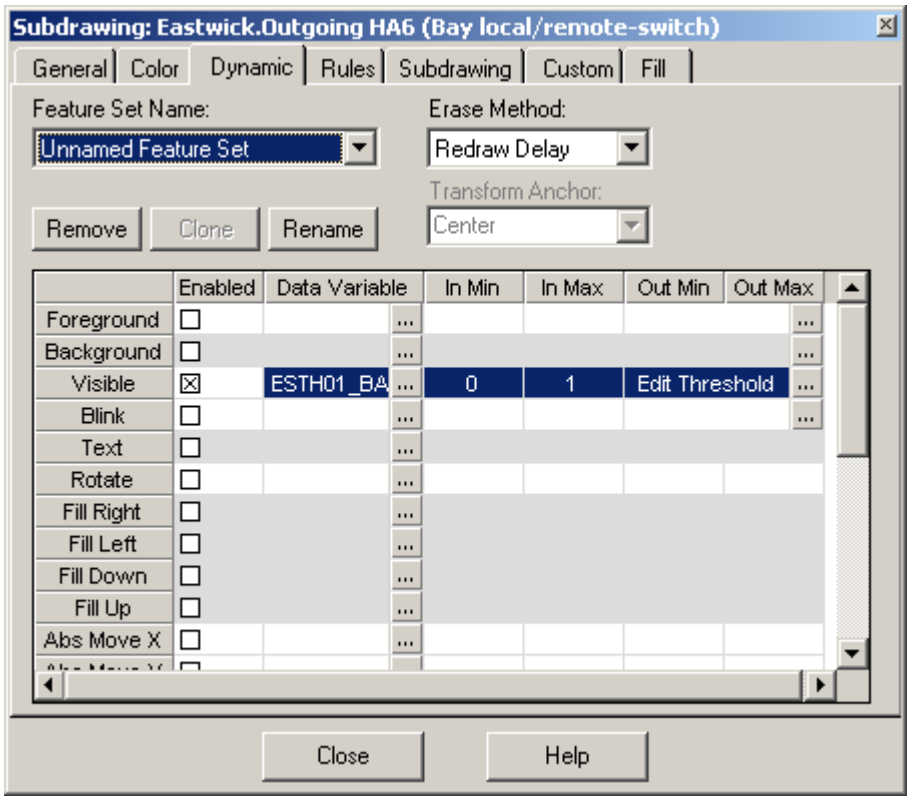


Figure 107: Object dynamics

Dynamic features are connected to a graphical object as a group. This group is called a dynamic feature set. The user can name a dynamic feature set and attach it to multiple graphical objects, giving these multiple objects the same dynamic characteristics.

A dynamic feature set contains one of each kind of dynamic feature. Each feature is enabled or disabled in a specific dynamic feature set. The only objects that cannot have a dynamic features are graphs and input objects, which are inherently dynamic.

A dynamic feature is connected to a data variable and produces dynamic changes in any graphical object attached to it as the data changes. For example, the user can use the color dynamics by connecting a circle object to a source of temperature data, and assign different colors to a series of thresholds within the range of the temperature. As the temperature rises, the circle changes color as each threshold is crossed.

The subdrawing dynamics can be made to use different displays to describe various operating and broken states. Each drawing must be exported as a subdrawing display file (.sd file). As transmission success rate changes, different images of the equipment are shown.

If a dynamic feature set that is attached to multiple objects is changed, the changes affect all graphical objects that this dynamic feature set is attached to.

The dynamics can be embedded in a subdrawing object by applying them from the subdrawing display file. If a subdrawing object contains embedded dynamics, those are shown in run time in Monitor Pro. Embedded data variables can also be mapped to variables in the current display, see Section 7.2.4.

7.3.1 Dynamic feature set

The following sections briefly describe how to use a dynamic feature set.

7.3.1.1 Naming dynamic feature set

To name a dynamic feature set or change the current name:

1. Select the object.
2. In the **Object Properties** dialog, select the **Dynamic** tab.
3. Click **Rename**.
4. Enter a name in the **Rename Feature Set** field.
5. Press Enter.

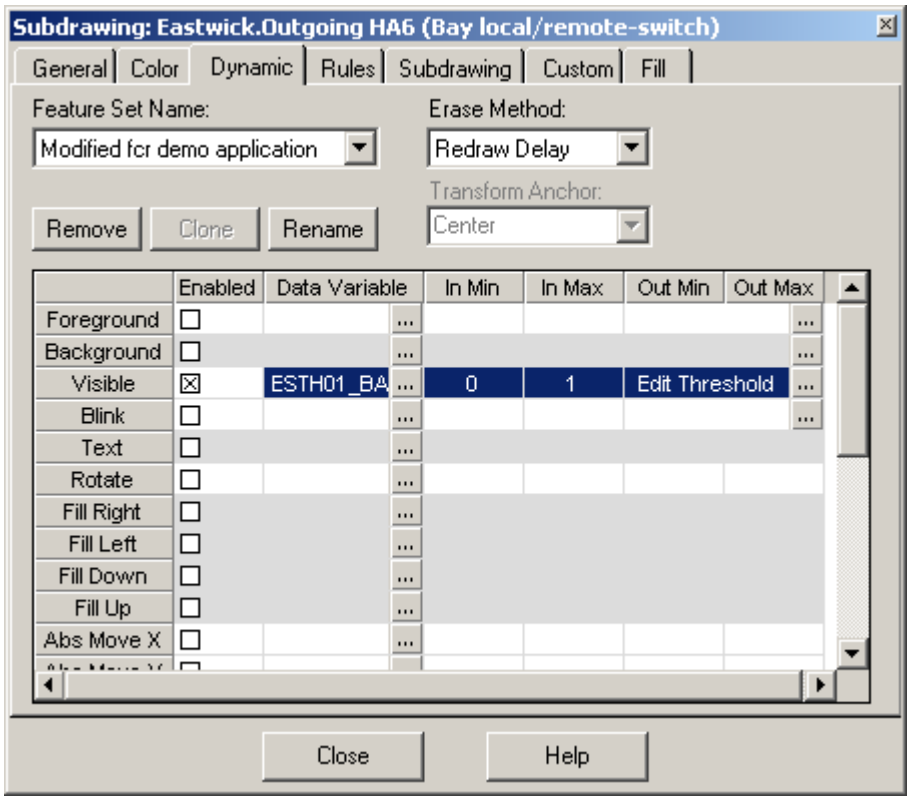


Figure 108: Naming dynamic feature set

7.3.1.2 Using named dynamic feature sets

It is not necessary to name a dynamic feature set. However, naming a dynamic feature set is useful if the user wants to attach it to multiple objects that are not copies of each other, or if the user wants to use multiple complex dynamic feature sets that only slightly differ from each other.

To select a named dynamic feature set for a new object:

1. Select the object.
2. In the **Object Properties** dialog, select the **Dynamic** tab.
3. Click **Feature Set Name** drop-down menu.
4. Select a named dynamic feature set.

A graphical object can only have one dynamic feature set attached to it at a time. To change which dynamic feature set is attached to a graphical object, select a different named dynamic feature set from the **Feature Set Name** drop-down menu.

7.3.1.3 Guidelines for sharing dynamic feature sets

Sharing dynamic feature sets among different kinds of objects can be convenient while creating displays to apply identical dynamic features. Features that do not apply are ignored, so they do not cause problems while editing. However, any features that do not apply should eventually be deleted to prevent flickering when objects are updated. To keep a named dynamic feature set available for sharing or copying, it must belong to at least one object.

A dynamic feature set is destroyed when it no longer belongs to any object. Make sure not to use the same name for different dynamic feature sets in different displays. If these displays are merged at a later time, or both of them are brought into the same display as subdrawings, the dynamics may not behave as expected.

7.3.2 Changing data variable for dynamic feature

Every enabled dynamic feature must have data associated with it. By default, all dynamic features use default.dat/Var:1, but the user can replace this data variable with any other. Dynamic features can have only one data variable each.

To change the data variable used by a dynamic feature:

1. Select the object.
2. In the **Object Properties** dialog, select the **Dynamic** tab. The tab includes the data variable name for each dynamic feature.
3. Click the ellipsis button for the data variable that should be changed. The **Data Selection** dialog opens.
4. Select a data variable.
5. Click **OK** to close the data selection.

The name of the new data variable is shown in the **Dynamic** tab.

7.3.3 Enabling and disabling dynamic features

To enable or disable a dynamic feature:

1. Select the object.
2. In the **Object Properties** dialog, select the **Dynamic** tab. The tab lists all of the dynamic features. Dynamic features that do not apply to the type of the currently selected object are disabled.
3. Toggle the checkbox for any dynamic feature. When the checkbox is checked, the feature is enabled, and when not, the feature is disabled.

Not all features apply to all kinds of graphical objects. If the user enables dynamic features that do not apply to the selected graphical object, they have no effect on that object. However, if the same dynamic feature set is attached to a different graphical object, different dynamic features may apply.

7.3.4 Removing dynamic feature set

A dynamic feature set can be removed from an object. This removes all dynamic settings from the object. It does not matter whether the feature set is named.

To remove a dynamic feature set from an object:

1. Select the object.
2. In the **Object Properties** dialog, select the **Dynamic** tab.
3. Click on **Remove**.

7.3.5 Dynamic feature types

The **Dynamic** tab of the **Object Properties** dialog lists all of the dynamic features. Dynamic features that do not apply to the type of the currently selected object are the dynamic features that can be divided into groups. The major groups are attribute dynamics, which affect object properties, and motion dynamics.

Subdrawing dynamics are a special form of attribute dynamics in which the whole subdrawing object image changes instead of a graphical attribute, such as line type or color. Visibility dynamics and text dynamics are in groups by themselves.

	Arc	Circle	Ellipse	Icon	Image	Line	Polygon	Rectangle	Sub-drawing	Hardware Text	Vector Text	Scalable Text	Windows Control
Arc Direction	•												
Background Color	•	•	•				•	•		•			
Blink	•	•	•			•	•	•	•	•	•	•	•
Curve Type							•						
Fill Down	•	•	•				•	•					
Fill Left	•	•	•				•	•					
Fill Right	•	•	•				•	•					
Fill Up	•	•	•				•	•					
Foreground Color	•	•	•			•	•	•	•	•	•	•	
Line Type	•	•	•			•	•	•					
Line Width	•	•	•			•	•	•					
Subdrawing									•				
Absolute Move X	•	•	•	•	•	•	•	•	•	•	•	•	•
Absolute Move Y	•	•	•	•	•	•	•	•	•	•	•	•	•
Path Move	•	•	•	•	•	•	•	•	•	•	•	•	•
Relative Move X	•	•	•	•	•	•	•	•	•	•	•	•	•
Relative Move Y	•	•	•	•	•	•	•	•	•	•	•	•	•
Rotate	•	•	•			•	•	•	•		•	•	•
Scale	•	•	•	•	•	•	•	•	•		•	•	•
Scale X	•	•	•	•	•	•	•	•	•		•	•	•
Scale Y	•	•	•	•	•	•	•	•	•		•	•	•
Text										•	•	•	
Visibility	•	•	•	•	•	•	•	•	•	•	•	•	•
Fill Effect	•	•	•				•	•					

Figure 109: Dynamic feature types

The types of dynamic features are briefly described in the following sections.

7.3.5.1 Attribute dynamics

Background:

Changes the background color of the object. If the fill status is **Fill with Edge**, the edge color of the object changes. If the fill status is **Edge with Fill**, the inside color changes. The background attribute does not apply to line objects and has no effect on bitmap objects, icon objects, subdrawing objects, vector text, true type text, or controls. Background dynamics and blinking dynamics are mutually exclusive.

Foreground:

Changes the foreground color of the object. If the fill status is **Fill with Edge**, the inside color of the object changes. If the fill status is **Edge with Fill**, the edge color changes. The foreground attribute has no effect on bitmap objects, icon objects, or controls. Foreground dynamics and blinking dynamics are mutually exclusive.

Blink:

Alternates the foreground and/or background colors of an object between two sets of colors to make the object blink. An object can blink or not blink, or blink in different colors, depending on a data value. Blinking dynamics and foreground or background dynamics are mutually exclusive. The blink attribute has no effect on bitmap objects.

Line Type:

Changes the line type of the object's outline. The line type attribute applies to line, arc, circle, ellipse, polygon, and spline objects.

Line Width:

Changes the thickness of the object's outline. The line width attribute applies to line, arc, circle, ellipse, polygon, and spline objects.

Arc Direction:

Changes the arc direction of the arc. Applies only to arcs.

Curve Type:

Changes the curve type of a polygon, polyline, open spline, or closed spline object. The attribute applies only to these object types.

7.3.5.2 Subdrawing dynamics

Changes the display of a subdrawing in the run time. Applies only to subdrawings that do not have enabled dynamics.

When the user wants the contents of a subdrawing to change, these are two methods:

- Create a series of displays and use subdrawing dynamics to change the display when being played.
- Create one display with dynamics in it, use the display as a subdrawing, and enable the dynamics in the subdrawing.

The dynamics applied to a subdrawing, such as subdrawing dynamics or motion dynamics, should not be confused with dynamics included within a subdrawing.

7.3.5.3 Visibility dynamics

Visibility:

Changes whether or not the object is visible depending on the data values and threshold settings. The visibility attribute applies to any type of object. By default, all objects are visible in Display Builder. This attribute is especially useful for path polygons, as well as for making objects, such as alarms or warning messages, appear and disappear. Objects cannot be picked when they are not visible. The visibility can only be controlled using object dynamics. Note that the Redraw Immediate Erase Method does not work with visibility dynamics in nested subdrawings.

7.3.5.4 Text Dynamics

Text:

Changes the content of a text object to show its formatted variable value. Applies to true type text, vector text, and hardware text. This lets the user control the text attributes and justification, as well as combine variable and static content in a single text object. Using text dynamics is more efficient than using a Text Graph or Digits Graph and offers more flexibility.

7.3.5.5 Motion dynamics

Fill:

The fill options apply to circles, arcs, ellipses, rectangles, polygons, polyline, and splines, and are effective only when the object has a fill status of Fill, Edge with Fill, or Fill with Edge. They are mutually exclusive. The in range values are mapped to the out range, and out range values are expressed as a percentage of the object's area, with 1=100%. The out range values should always be between 0 and 1, values over 1 are meaningless, see Section 7.3.6.

Fill Left: Changes the filled area of the object, filling from right to left. The fill percentage is measured from 0 at the right edge to 1 at the left edge.

Fill Right: Changes the filled area of the object, filling from left to right. The fill percentage is measured from 0 at the left edge to 1 at the right edge.

Fill Up: Changes the filled area of the object, filling from bottom to top. The fill percentage is measured from 0 at the bottom edge to 1 at the top edge.

Fill Down: Changes the filled area of the object, filling from top to bottom. The fill percentage is measured from 0 at the top edge to 1 at the bottom edge.

Rotate:

Rotates the object around the transform anchor point. The transform anchor point can be specified in the Transform Anchor drop-down menu. This is not useful for any object with only one or two control points.

Scale:

Changes the scale of the object using the transform anchor point as a reference point. The transform anchor point can be specified in the Transform Anchor drop-down menu. The out range units are shown per cent, in which 1=100%. For example, a range of [0,1] makes the object equal to or smaller than its original size. A negative variable value flips the image horizontally and vertically. This feature is not useful for hardware text.

Scale X:

Changes the horizontal scale of the object using the transform anchor point as a reference point. The out range units are shown per cent, in which 1=100%. A negative variable value flips the image horizontally, from right to left. This is not useful for hardware text.

Scale Y:

Changes the vertical scale of the object using the transform anchor point as a reference point. The out range units are shown per cent, in which 1=100%. A negative variable value flips the image vertically, from up to down. This is not useful for hardware text.

Absolute Move X:

Moves the object's transform anchor point horizontally with respect to the drawing area origin. Applies to all object types. The out range units are shown in world coordinates.

Absolute Move Y:

Moves the object's transform anchor point vertically with respect to the drawing area origin. Applies to all object types. The out range units are shown in world coordinates.

Relative Move X:

Moves the object horizontally with respect to the object’s original (untransformed) position. Applies to all object types. The out range units are shown in world coordinates.

Relative Move Y:

Moves the object vertically with respect to the object’s original (untransformed) position. Applies to all object types. The out range units are shown in world coordinates.

Path Move:

Moves the object's transform anchor point along the edge of the specified polygon. Applies to all object types. The out range units are shown in range mapped to the path polygon.

Special usage notes
Path objects can also have dynamics. This lets the user add complex motion to objects using the Path Move feature, such as applying rotation to the path polygon. A path polygon can use the Path Move feature, but cannot use itself as the path.

Note that path polygons should be polygon objects with a polygon type setting of Polygon, not Closed, Open, or Floating. If a different kind of graphical object is used as a path polygon, the dynamic object may not move as expected. This is because the dynamic object moves only from one control point to another and not necessarily along the perimeter of the path object.

Also note that when path motion dynamics are assigned to a graphical object in association with a path polygon object, the graphical object retains the path motion dynamics even when the path polygon is removed. To properly disable the dynamic behavior, first remove the path motion dynamic associated with the graphical object, then delete the path polygon object.

7.3.6 Setting data range

The data range is shown in the **Object Properties** dialog, on the **Dynamic** tab.

7.3.6.1 Incoming data range

The In Min and In Max fields specify the minimum and maximum values to be considered from the data variable. If the value of a variable exceeds the maximum value, Display Builder indicates the maximum value. Likewise, if a value goes below the minimum value, Display

Builder indicates the minimum value. The incoming data range may specify the actual range of data, or it can be used as a filter.

To set the incoming data range:

1. Select the object.
2. In the **Object Properties** dialog, select the **Dynamic** tab. The tab includes the data variable name for each dynamic feature.
3. Enter new values in the In Min and/or In Max fields. The lower limit must be a smaller value than the upper limit.

Incoming data range does not apply to text dynamics. The default incoming data range is 0.0 to 1.0.

7.3.6.2 Outgoing data range

The OutMin and OutMax fields specify the minimum and maximum values to be shown. The incoming data range is mapped to the outgoing data range. For example, for the Rotate dynamic feature, an incoming data range of [0,1] might map to an outgoing range of [0,360] degrees.

To change the outgoing data range:

1. Select the object.
2. Go to the **Dynamic** tab of the **Object Properties** dialog. The **Dynamic** tab includes the data variable name for each dynamic feature.
3. Enter the new value(s) or click the ellipsis button to show the **Data Selection for Dynamics** dialog.

For outgoing ranges that are expressed using two values, the lower limit can be a larger value than the upper limit. Entering a smaller number for the lower limit effectively swaps the high and low values of your display. For example, swapping the out range values of the Move Path option reverses the direction of movement along the path.

7.3.6.3 Setting of outgoing data ranges for motion and text dynamics

Motion and text dynamics use continuous outgoing data ranges to allow continuous motion.

Move Path dynamic feature
To set the Move Path dynamic feature, pull down the **out range option** menu, and select a named polygon.

The incoming data range maps to the distance from the start point of the polygon to its end point. When the view is run, the transform anchor point of the graphical object moves from one place to another along the path polygon according to the current value of the variable.

If the user changes the name of the path object at a later time, any path dynamics that use that object are automatically updated.

Other motion dynamic features
To set other motion dynamic features, enter new values for the upper and lower range limits.

The units vary depending on the dynamic feature type. The Rotate outgoing range is expressed in degrees with [0,360] expressing full range of rotation. Rotation values are not limited to the outgoing range of [0,360], but all values are mapped into that range, wrapping around if necessary (like an altimeter).

The outgoing ranges of the Scale options are expressed as a percentage with 1=100%.

The outgoing ranges of the Move options are expressed in world coordinates.

Text dynamics
To set text dynamics, enter the conversion character together with any additional text to be displayed.

The conversion character must be preceded by a % symbol. The conversion character conforms to the ANSI C standard for format conversion, except for g, G.

The valid conversion characters and the type of data they indicate are:

- s: text string
- c: single character
- f: float, double, decimal notation
- e, E: float or double converted to scientific notation
- g, G: converts to e, E or f, depending on the number of decimals allowed
- d, i: integer converted to decimal
- o: unsigned octal
- u: unsigned decimal
- x, X: unsigned hexadecimal
- p: address

Only one conversion character can be specified, but it can be embedded in a string. Other characters in the user's string appear as they are entered. These include \n for a new line, \t for a tab, and \octal_digits for special characters, for example, volume=%6.2f, score=%d%%, account name: %s.

When a g, G format is specified in the form x.y, y specifies the number of decimal places, not the total width of the field.

If the user enters a conversion character that does not match the format of the data, Display Builder displays an error message when the view is run and uses a default format. If the data is changed, make sure the conversion character matches the format of the new data.

7.3.6.4 Setting of outgoing data ranges for visibility and attribute dynamics

Visibility and attribute dynamics, including subdrawing dynamics, use threshold data ranges to allow discrete changes.

A threshold table maps incoming data into ranges, each corresponding to a single attribute setting.

To set the outgoing data range, click **Edit Threshold** to display a threshold table dialog. For instructions on setting thresholds, Section 7.3.6.7.

7.3.6.5 Default outgoing data ranges

The default outgoing data ranges are listed in the following.

Motion dynamics:	Absolute Move X -16383 to 16383 (world coordinates) Absolute Move Y Relative Move X Relative Move Y Rotate 0 to 360 (degrees) Scale .5 to 1 (1 = 100%) Scale X .5 to 2 Scale Y Fill Left 0.0 to 1.0 (1 = 100%) Fill Right Fill Up Fill Down Fill Effect
Attribute dynamics:	Arc Direction Current attribute setting (none where not applicable) Background Color Foreground Color Curve Type Line Type Line Width Other dynamics: Move Path No Path Polygon Visibility Visible Text %s %f or %d

7.3.6.6 Specifying a transform anchor point

When applying rotation or scale dynamics, the user can designate a transform anchor point around which the action takes place. When path dynamics are applied, the designated transform anchor point is the point that moves along the path.

This feature is only active when the dynamics feature set includes Rotation, Scale, Scale X, Scale Y, Absolute Move X, Absolute Move Y, or Path Move dynamics.

The setting for this feature applies to the dynamic feature set as a whole, not to an individual dynamic action, so if more than one of these dynamic types is applied, they all use the same transform anchor point.

To specify a transform anchor point:

1. Select the object.
2. Select the **Dynamic** tab in the **Object Properties** dialog.
3. Pull down the **Transform Anchor Point** option menu.
4. Select a transform anchor point.

Default transform anchor: Center

7.3.6.7 Setting of threshold values for object dynamics

Thresholds are used to specify the behavior of dynamic attribute features, including object dynamics applied to subdrawings. A threshold is a value at which the attribute changes, or at which a different subdrawing is displayed. A threshold value can be an unchanging number, or a data variable can be used to create a threshold value that changes. The user can set as many thresholds as they want by using the slider or the text box to set the values.

The dynamic features that use thresholds are:

- Foreground Color
- Background Color
- Line Width
- Line Type
- Arc Direction

- Curve Type
- Visibility
- Subdrawing

Fill Effect (for more information, see "Setting pattern fill effects for thresholds" and "Setting gradient fill effects for thresholds").

The colors associated with graph variables are also controlled by threshold tables.

The options that appear inside the threshold areas differ for each dynamic feature type.

When a threshold is added, the new attribute selection or subdrawing is assigned to the area above the threshold.

When setting line width or line type thresholds, the user can toggle through a series of values by clicking in the threshold area. To reverse direction, hold down the Shift key and select the threshold area with the mouse.

Adding a threshold
To add a threshold:

1. Use the slider or enter a value within the allowed range.
2. Click **Add**.
   For example, if the range of a variable is 0 to 100, a threshold can be added at any intermediate value, such as 25, 50, or 99. Numbers outside the range are ignored.
3. Indicate a new attribute setting.

Some attribute selections are made the way they are set for ordinary graphical objects. For example, colors are set by selecting a new color from the color palette, and subdrawing filenames are selected from the **Open File** dialog that appears when the **Add** button is clicked. Other attribute settings are made by toggling the current setting in the threshold area, such as curve type, arc direction, line type, and line width.

Before adding subdrawing thresholds to a subdrawing, the desired subdrawings must exist as saved views. To change a subdrawing name in a threshold, double-click the subdrawing filename to display the **Open File** dialog.

Using a data variable to control a threshold value
A data variable can be used to control a threshold value. This means that the value at which the user's object changes will change according to the specified data variable as the application runs. This can be useful when using subdrawings that have embedded dynamics. For example, the user might create a subdrawing file of a gauge and use multiple instances of that gauge in a new view. The user might want each gauge to turn red at a different threshold value. This can be accomplished by using a data variable to control the threshold value in the subdrawing file, making the data variable public, and mapping it to a different data variable for each gauge in the parent view.

Note that a data variable cannot be used to control the bottom threshold value, since that value is always the minimum value of the data range.

To select a data variable to control a threshold value:

1. Select in the area immediately above the threshold.
2. Click **Use Data Variable**. The Data Browser appears.
3. Select a variable.
4. Click **OK**.

The text box shows the name of the currently selected variable. The variable name cannot be edited in this text box. To change the current variable, click the variable button and make a new selection in the **Data Browser**.

Deleting a threshold
To delete a threshold:

1. Select the setting immediately above the threshold.
2. Click **Delete**.

The attribute setting below the deleted threshold is assigned to the area above. If the bottom setting is deleted, the attribute setting immediately above it is assigned to the bottom area.

Saving a threshold table to a file
The threshold table assigned to a particular dynamic feature can be saved to a file. This file can then be used with another dynamic feature, providing it with the same attribute thresholds. This capability lets the user use the same set of thresholds with a variety of dynamic features or with different dynamic feature sets.

To save a threshold table:

1. Click **Save**.
   The **Open File** dialog appears with the file type filter set to Dynamic Threshold Table Files. Pull down the file type menu to change the filter to All Files if necessary.
2. Select or enter the desired threshold table filename.
3. Click **Save**.

When the user saves a set of thresholds, they are saved as normalized values. For example, suppose the user has a variable with a range of 0 to 50 and thresholds at 20, 25, and 36. When the current set of thresholds is saved, they are saved as percentages in the range of 0 to 1, in this case, .40, .50, and .72. This lets the user create thresholds using one range and to keep the proportions the same when using these thresholds in variables with different ranges. This means that different ranges can be used with different dynamic variables and have the changes occur at the same time.

Some precision may be lost when changing ranges.

When changing the incoming range of a variable with thresholds, the thresholds remain at their actual values, and their relative position within the range changes. If the same relative position should be maintained in a new range, save the thresholds first, then change the range and retrieve the saved thresholds.

Loading a threshold table from a file
To load a threshold table:

1. Click **Load**.
   The **Open File** dialog appears with the file type filter set to ***Dynamic Threshold Table Files*** . Pull down the file type menu to change the filter to ***All Files*** if necessary.
2. Select or enter the desired threshold table filename.
   Color threshold tables saved for foreground or background dynamics can be used for either. Color threshold tables saved for graphs can also be used with either foreground or background dynamics.
3. Click **Open**.

Clearing all thresholds
To clear all thresholds, click **Clear**.

Setting blink colors

To set up Blink dynamics, set thresholds as described above. Blink colors must also be specified.

To set blink color, click **Edit Threshold** for the Blink dynamics feature. The Blink Threshold Editor is displayed.

Each threshold has a **Blink** or **No Blink** setting. The default setting for each new threshold is **No Blink**.

To make the object blink when the data value is within the threshold range, click inside the threshold to toggle the setting to **Blink**.

For threshold regions that have the **Blink** setting, set the foreground and background colors of both the normal and blink state, as well as the blink rate. For threshold regions that have the **No Blink** setting, set the foreground and background colors of the normal state.

Note that an object in the blinking state can have different normal foreground and background colors from the normal foreground and background colors of the same object in the non-blinking state.

To set a color:

1. Select one of the squares in the **Normal** or **Blink** areas.
   The upper left square sets the foreground color and the lower right square sets the background color.
2. Use the color palette to specify the color.
   If the selected object is a Windows control, the normal colors appear as hatched tones. These hatched colors indicate that the object uses system colors. If these squares are changed to another color using the color palette, it is possible to go back to the system color specification by checking the **System Default** option.

The blink period is the interval between one state and the next, expressed in milliseconds. To clarify, normal colors and blink colors are the two states. Whether the object is currently blinking at all is determined by the threshold table and the current data value.

To set the blink period:

1. Click a blinking threshold area.
2. Enter a blink period.
3. Press Enter.

Experiment with different rates. 1000 is fairly slow, 100 is fast.

Setting pattern fill effects for thresholds

To set up fill effect dynamics, set thresholds as described above. The user must also set two colors, as well as fill effect pattern styles.

Fill effects only apply to filled rectangle, circle, arc, ellipse, polygon, polyline, open spline, or closed spline objects.

Pattern fill effects can also be combined with gradient fill effects in the user's fill effect dynamics. For more information on gradient fill effects, see "Setting gradient fill effects for thresholds".

To set pattern fill effects for thresholds:

1. Click **Edit Threshold**. The **Fill Effect Threshold Editor** is displayed.
2. Select the **Has Fill Effect** check box.
3. Make sure **Pattern** is selected in the combo box.

The pattern portion of the Fill Effects Threshold Editor allows the user to choose new foreground or background colors, elect to keep the original object's inherited colors, apply color transparency, as well as choose from a variety of pattern styles.

Setting gradient fill effects for thresholds
To set up fill effect dynamics, you set thresholds as described above. You also must set two colors, as well as fill effect gradient styles.

Fill effects only apply to filled rectangle, circle, arc, ellipse, polygon, polyline, open spline, or closed spline objects.

Gradient fill effects can also be combined with pattern fill effects in the user's fill effect dynamics. For more information on pattern fill effects, see "Setting pattern fill effects for thresholds".

To set gradient fill effects for thresholds:

1. Click **Edit Threshold**. The **Fill Effect Threshold** dialog is displayed.
2. Select the **Has Fill Effect** check box.
3. Select **Gradient** in the combo box.

The gradient portion of the Fill Effects Threshold Editor allows the user to choose new foreground or background colors, elect to keep the original object's inherited colors, select the coarseness of shading, as well as choose from a variety of shading styles.

7.4 Rules

Rules let the user transform a collection of individual displays into a functioning user interface for the application, right in Display Builder. Rules let the user interface perform many kinds of operations in response to user events like mouse picks and events in control objects, and application events such as the loading of a display.

The actions the user interface can perform include switching between displays, adding objects from other displays to the current display, manipulating data values, and changing the update rate. Data can be written back to a writeable custom data source item. The user can also write their own actions and conditions using the **Execute Function in Script** condition or action.

For example, the execute button (see Section 5) has an internal rule that executes a VBA script function when the subdrawing is clicked.

Using rules lets the user work on a much broader scale than that of a single display. It helps to think not only in terms of individual displays, but in terms of how all the displays work together.

The basic steps for creating a user interface in Display Builder are:

- Planning the user interface and its behavior, see Section 7.4.1.
- Organizing an application directory, see Section 7.4.2.
- Creating displays, using Display Builder to add rules, see Section 7.4.4.

To enable internal rules that execute a function in script, the script file must be the same in the subdrawing and in the process display that contains the subdrawing. It is recommended to always use the default script file \sc\prog \graphicsEngine\etc\SYS600_scripts.bas.

Do not use the Go To View and Go To Previous View rule actions. These rules are obsolete mechanisms and, for example, the network topology coloring does not work when these rules are used. The rules may be removed in the future releases.
Use the operation **Open/Display** in the **Tool Launcher Settings** dialog instead.

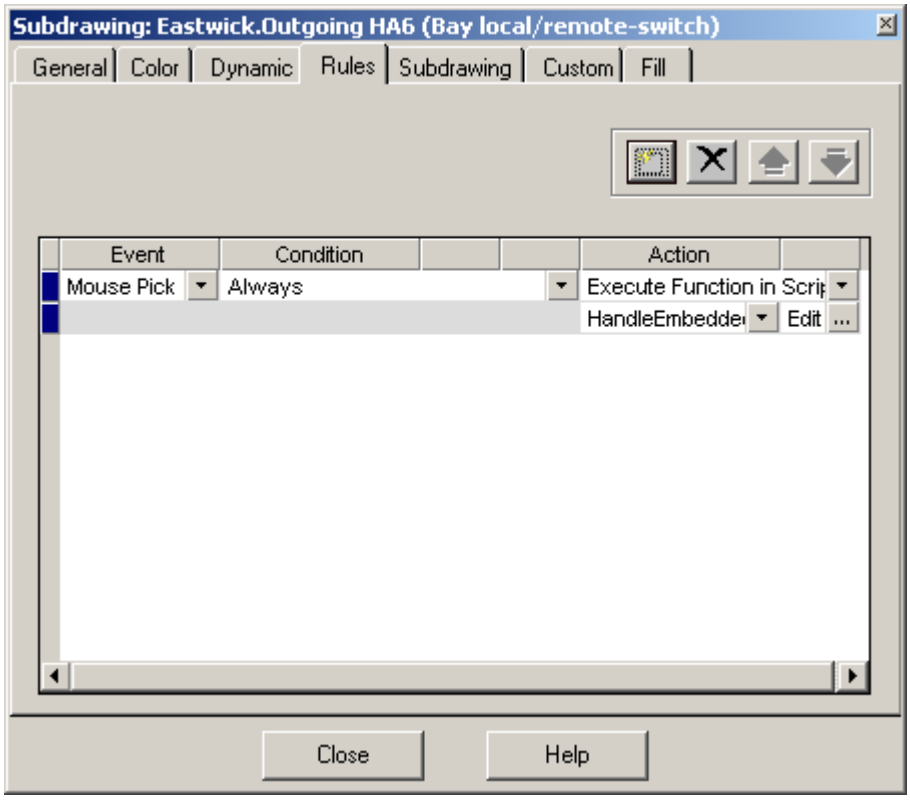


Figure 110: Adding a new rule

7.4.1 Planning of user interface

Planning the user interface and its behavior is fundamental to application development. Behavior can include the display sequence of displays, the addition of overlays and popups, and changes to data that result from user interaction. The user must also consider the events and conditions which cause the interface to change.

In the conceptual plan of a graphical user interface, information needs to be organized in a logical hierarchical structure. The displays and the transitions between them should parallel this structure, making information access as intuitive as possible. Consider how much information can be displayed and absorbed in a single screen. Each display should contain only the elements needed to make the information clear and easy to understand, and how to indicate what additional information is available at other levels.

To associate additional information with visible objects, use rules. For each rule, determine what event triggers each action that displays new information, and the condition under which the action occurs

7.4.2 Planning of application directory

Before creating the displays for the application, the user should consider the structure of its working directory.

Each application should have its own working directory to avoid conflict with other projects. Use subdirectories and file names with consistent extensions. The user's project may include any or all of the following subdirectories:

```
<application name>\PICT
```

Displays to be displayed by the application. The standard extension for a display files is .v.

```
<application name>\AplMod4\Palette\*
```

Subdrawings. Typical extensions for subdrawing files are .sd, .d, or .v.

```
sc\prog\graphicsEngine\lib\Templates
```

Display files used for the graphical and functional layout of input objects

7.4.3 Selecting of objects for rules

Rules can be attached to objects or to the displays. Most rules are attached to objects. Sometimes a rule is attached to a single object, such as a primitive shape or an input object.

However, sometimes the user wants to attach rules to a complex image. If an image is composed of several objects and rules are attached to only one, the user might not pick the object with the rules. Some suggested solutions are:

- Store the complex image as a subdisplay and use it to create a subdrawing. Then attach rules to the subdrawing object. This lets the user pick anywhere on the subdrawing object to access the rules without worrying about which component object is selected. Also, to pop up or overlay a subdrawing, only one rule is needed instead of a separate rule to display each object in the image.
- Store the complex image as a bitmap object. Then, import the bitmap and attach rules to it.
- Place a single transparent object in front of the complex image and attach rules to the transparent object.
- If one object in the complex image extends to all edges of the image, make that object transparent, place it in front of the other objects, and attach rules to it. This only works if one object covers the whole image area, and it can be conveniently made transparent.
- If the complex image serves as a button, use a button control or input object instead.

7.4.4 Adding of a rule

The rules in the user's display specify interface behavior by describing actions, or changes in the interface, which occur depending on specified events and conditions.

An action can be changing to another display, overlaying a display, overlaying an object, popping up an object at a cursor location, or erasing displays and overlays. The user can also manipulate data and write their own actions using the **Execute Function in Script** option.

An event can be a pick or an input object event.

A condition can be, for example, which mouse button was clicked, which keyboard key was pressed, or whether or not data values meet specified criteria. The user can also write their own conditions using the **Execute Function in Script** option.

Rules can be attached to objects or to the display.

Before adding rules, note the following guidelines:

- Any object can have rules.
- An object can have up to 100 rules.
- Any object required by an action must be named.

For a rule to function, the named displays and objects must exist. However, the user can create rules that refer to the names of displays and objects that do not yet exist. This causes an error, such as Cannot open file name when running the display, but does not prevent the display and other associated displays from running.

If the **Execute Function in Script** option is used, the display can use only one VBA script for rules. All conditions and actions that the user wants to write in a script must be contained in that single script file. If the display also uses VBA scripting data sources, rule and data source scripts cannot be in the same file.

Do not use directory names when specifying subdrawings or displays in rules. Edit the search path setting in the **View Preferences** dialog to add the appropriate directories and subdirectories to the search path.

7.4.4.1 Adding rules to objects

To add a rule to an object:

1. Select the object.
2. Select the **Rules** tab in the **Object Properties** dialog.
3. Click **Create a New Rule** to add a default rule.
4. Specify the event. This needs to be done only if the user wants an event other than Mouse Pick, see Section 7.4.6.
5. Specify the condition. This needs to be done only if the user wants an event other than Always, see Section 7.4.7.
6. Specify the resulting action.

For more information, see Section 7.4.8.

7.4.4.2 Adding rules to display

To add a rule to the display:

1. Display the **View Properties** dialog.
2. Select the **Rules** tab.
3. To add a default rule, click **Create a New Rule**.
4. Specify the event. This needs to be done only if the user wants an event other than Mouse Pick, see Section 7.4.6.
5. Specify the condition. This needs to be done only if the user wants an event other than Always, see Section 7.4.7.
6. Specify the resulting action.

For more information, see Section 7.4.8.

7.4.5 Managing rules

Both objects and displays can have multiple rules. All rules with true conditions are processed in the order in which they appear in the list, so organization is important. When multiple rules are used, the **Rules** tab lets the user add, delete, and organize rules.

To delete a rule:

1. Select the rule in the list.
2. Click on the **Delete** button.

To move a rule up in the list:

1. Select the rule in the list.
2. Click on the Up arrow button.

To move a rule down in the list:

1. Select the rule in the list.
2. Click on the Down arrow button.

7.4.6 Specifying event

An event must occur before the action of a rule can take place.

To change the event selection, pull down the event option menu and select one of the events listed. The events available for the current object depend on the object type. A different list of events appears for a display.

Event types for the display, input objects, and most other primitive objects are listed in the following sections.

Events are processed in the following order:

1. Input object events: Pick, Done, Cancel, and Event Used.
2. All true-condition rules, in the order in which they appear in the list.
3. Dynamic updates.

7.4.6.1 Object events

Mouse Pick or Key Pick:

Occurs when the cursor is located on an object or display and any key or mouse button is pressed. A pick registers first on an object if there is one at the cursor location. If there is no object, the pick registers on an overlay display. If there is no overlay display, the pick registers on the base display. If an object with visibility dynamics is not visible when the user picks on it, it does not register the pick event. These two events are available for all kinds of objects and for the display.

7.4.6.2 Input object events

- Mouse Pick or Key Pick:
  Occurs when the cursor is located on an object or display and any key or the mouse button is pressed. A pick registers first on an object if there is one at the cursor location. If there is no object, the pick registers on an overlay display. If there is no overlay display, the pick registers on the base display. If an object with visibility dynamics is not visible when the user picks on it, it does not register the pick event. These two events are available for all kinds of objects and for the display.
  An input object Pick event must occur in an active area of the input object. Picks that occur in inactive areas of the input object do not trigger rules of the input object. If the input object is part of a subdrawing, the pick triggers execution of the subdrawing's rules.
- Done:
  Occurs when a Done action of an input object is selected. To use Done events, the input objects must define a Done areas or keys. The input object updates accordingly before it processes the rule.
- Cancel:

Occurs when a Cancel action of an input object is selected. To use Cancel events, the input objects must define a Cancel area or keys. The input object updates accordingly before it processes the rule.

- Event Used:
  Occurs when any meaningful event takes place in an input object. Meaningful events include motion that changes the variable value, picks in active areas, done events, and cancel events.

7.4.6.3 Display events

- Mouse Pick or Key Pick:
  Occurs when the cursor is located on an object or display and any key or the mouse button is pressed. A pick registers first on an object if there is one at the cursor location. If there is no object, the pick registers on an overlay display. If there is no overlay display, the pick registers on the base display. If an object with visibility dynamics is not visible when the user picks on it, it does not register the pick event. These two events are available for all kinds of objects and for the display.
- Draw:
  Occurs when the display is drawn, overlaid, or redrawn.
- After Draw Next:
  Occurs immediately after a display is attached to a window.
- Event Used:
  Occurs when any meaningful event takes place in an input object. Meaningful events include motion that changes the variable value, picks in active areas, done events, and cancel events.
- After Attach:
  Occurs immediately after all objects in the display have been updated.
- After Detach:
  Occurs immediately after a display is detached from a window.
- After Initial Draw:
  Occurs immediately after the display is drawn on the screen for the first time.
- After Read Data:
  Occurs immediately after the data sources are read, so the data variable buffers contain the new data.
- After Resize:
  Occurs immediately after the window has been resized.
- After Display Loaded:
  Occurs immediately after the display has finished loading, before it is displayed.
- Before Attach:
  Occurs before a display is attached to a window.
- Before Detach:
  Occurs before a display is detached from a window.
- Before Draw Next:
  Occurs immediately before all objects in the display are updated.
- Before Read Data:
  Occurs immediately before the data sources are read.
- Click and Click on Object:
  These events may occur when the user clicks a mouse button over any object. Click on Object can be used to build hot spots in the user's display. The object must be named, and the user must click the mouse button both down and up within the same object in order for a Click On Object event to occur on that object.
- Double Click:
  Occurs when the user double-clicks a mouse button over any part of the window.
- Horizontal Scroll:
  Occurs when a user interacts with a horizontally oriented Windows scroll bar, Slider, or Spin Control in the display.
- Key Down:
  Occurs when the user presses a keyboard key down. The key must be a lower case letter or numeric key.

- Key Press:
  Occurs when the user presses any keyboard character key.
- Key Up:
  Occurs when the user releases a keyboard key. The key must be a lower case letter or numeric key.
- Mouse Down:
  Occurs when the user presses any mouse button.
- Mouse Move:
  Occurs when the user moves the mouse.
- Mouse Over Changed:
  Occurs when the user moves the mouse over an object boundary, when it moves from one object to over another object, or when it leaves the boundary of the object and is over the background of the display. This event can be used to build hot spots in the user's display that display information when the mouse moves over them.
- Mouse Up:
  Occurs when the user releases a pressed mouse button.
- Vertical Scroll:
  Occurs when a user interacts with a vertically oriented Windows scroll bar, Slider, or Spin Control in the display.

7.4.7 Specifying a condition

7.4.7.1 Condition types

Display Builder provides the following rule condition types:

- **Always**:
  Makes the action take place whenever the specified event occurs.
- **Button Pressed is**:
  Makes the action take place if the pick was a specified mouse button. The options are Left and Right.
- **Key Pressed is in**:
  Makes the action take place if the pick was one of the specified keyboard keys. For example, the setting "Key Pressed is Qq" lets the user press either the upper or lower case Q. Control and function keys are not allowed.
- **Data Var Compares to Value**:
  Makes the action take place if the specified data variable has the specified relationship to a value.
- **Data Var Compares to Data Var**:
  Makes the action take place if the specified data variable has the specified relationship to another data variable.

7.4.7.2 Object's Var Compares to value

Makes the action take place if the object's variable has the specified relationship to a value. This condition is only useful with dynamic objects. If a graph, input object, or control object has more than one variable, only the first variable is used in the comparison. To compare variables other that the one attached to the object, use one of the Data Var Compares condition types.

7.4.7.3 Execute Function in Script

This option lets the user specify any condition in a separate VBA script file. For more information about using a script, see Section 7.4.9.

The rule's condition determines whether or not the rule's action takes place when the specified event occurs.

To edit the rule's condition, pull down the action option menu and select one of the conditions listed.

All condition types are available both for objects, including input objects, and for displays. For a description of each condition type, see Section 7.4.7.1.

7.4.7.4 Button Pressed is

If the condition type is **Button Pressed is**:

1. Pull down the bottom option menu.
2. Select the correct mouse button for the action.

7.4.7.5 Key Pressed is in

If the condition type is Key Pressed is in, enter one or more keyboard keys in the field below the condition name.

Keyboard keys do not need to be separated by other characters. For example, entering Qq makes the event happen when the user selects the object by pressing an upper or lower case Q. Teh user can enter multiple keyboard keys, but the condition is true when any of them is pressed.

If the event is Key Down or Key Up, the keys must be lower case letter or numeric keys.

7.4.7.6 Data Variable Compares to Value

If the condition type is **Data Variable Compares to Value**, the dialog displays an operator and a comparison value.

1. Select one data variable from the Data Browser.
   The Data Browser appears when this condition type is selected.
2. Specify the mathematical relationship.
3. Pull down the option menu next to the equal sign (=) and select the correct operator.
4. Specify a value by entering a numerical value or text string.

A variable can only be tested for one relationship in a single rule. Variables can be numerical or text variables. A variable cannot be tested for a value between two other values.

To change the selected data variable at a later time, click on the ellipsis button to display the Data Browser again.

7.4.7.7 Data Variable Compares to Data Variable

If the condition type is **Data Variable Compares to Data Variable**:

1. Select one data variable from the Data Browser and click **OK**.
   The Data Browser appears when this condition type is selected.
2. Select another data variable from the Data Browser and click **OK**.
3. Specify the mathematical relationship.
4. Pull down the option menu next to the equal sign (=) and select the correct operator.

Note that a variable can only be tested for one relationship in a single rule. Variables can be either numerical or text variables.

To change either selected data variable at a later time, click on the ellipsis button to display the Data Browser again.

7.4.7.8 Object's Variable Compares to Value

If the condition type is **Object's Variable Compares to Value**, the dialog displays the variable, an operator, and a comparison value. The user cannot select or change the object's variable in the rule.

7.4.7.9 Execute Function in Script

See Section 7.4.9.

If the condition type is **Execute Function in Script**, the dialog displays a combo box of existing scripting condition names from the script and provides options for editing the script.

No scripted condition name appears until a VBA script file has been specified for the display. A condition name can be entered directly in the combo box, which adds a stub for the condition to the script file. To edit the script file, click on the ellipsis button next to Edit Script.

7.4.7.10 Guidelines for using conditions

Before assigning rules using graph, input object, or control object values, check the variable names to make sure which ones to specify in the rules.

The Button Pressed is condition lets the user perform different actions by selecting the same object using different mouse buttons. This is useful when the action choices are limited in number and easy to remember. However, sometimes it is easier to remember which condition controls each action if the condition is a letter associated with the action, such as Q for quit.

The Object's Variable Compares to Value condition is useful when an object has only one data variable. However, if an object has more than one data variable, this condition only recognizes the first one. To compare an object's other data variables to a value, use the Data Variable Compares to Value condition. To compare an object's data variables to each other, use the Data Variable Compares to Data Variable condition.

If a rule applies to all but one variable value of an input object, a condition such as Object's Variable != 0 can be used instead of repeating the rule for both variable conditions, such as Object's Variable = 1 and Object's Variable = 2.

7.4.8 Specifying the resulting action

To describe the resulting action of a rule, pull down the action option menu and select one of the actions listed.

All action types are available both for objects, including input objects and control objects, and for displays.

If the user selects an action type involving an object, display, or system command, the user must complete the description by supplying the appropriate information in the fields that appear in the action area. For example, the Overlay Object action provides spaces for the name of the object and the name of the display where the object occurs.

Any object required by an action must be named. There are no naming conventions. When calling an object from another display, the rule uses the first object created with the specified name.

7.4.8.1 Execute Function in Script

See Section 7.4.9.

If the action type is **Execute Function in Script**, the dialog displays a combo box of existing scripting action names from the script and provides options for editing the script.

No scripted action name appears until a VBA script file has been specified for the display. An action name can be entered directly in the combo box, which adds a stub for the action to the script file. To edit the script file, click on the ellipsis button next to Edit Script.

7.4.8.2 Guidelines for using actions

Redraw Screen should be available in every display of every prototype. It is recommended to add it at the beginning. One way to accomplish this is to assign this actions to a visible object or button in each base display. Another way is to attach this rule to each base display itself with a mnemonic key press, such as R, as the event.

Use Go to Display to go from the current base display to any other display. To provide a choice of destination displays, the same rule (on Mouse Pick, Always, Go to Display name) can be given to different objects. Furthermore, it is also possible to give different rules to the same object. The rules can differ in their conditions or events. For example, when the object is picked, if the key pressed is P, go to the power display, or if the key pressed is T, go to the temperature display.

Only different events can be used if the object is an input object or a control object.

Do not use the Go To Display and Go To Previous Display rule actions. These rules are obsolete mechanisms and, for example, the network topology coloring does not work when these rules are used. The rules may be removed in the future releases. Use the operation **Open/Display** in the **Tool Launcher Settings** dialog instead.

Use Erase Overlay Object to erase a specified overlaid display. To erase more than one overlaid display, multiple rules specifying each display in turn, or Erase Visible Overlays can be used. However, the Erase Visible Overlays option erases both displays and objects.

To pop up the object against the background of the display, give the Popup Object rule to the display itself.

Any number of displays or objects can be overlaid. The only limits are readability and the memory capacity of the user's system.

Be careful about overlaying dynamic objects. Dynamics may bleed through other objects.

Avoid letting objects overlap graphs because graphs bleed through other objects. If an object might overlap a graph, the Popup Object action should be preceded by a Pause action. Use Resume when erasing the overlaid object.

Set Value can be used to set a variable for comparison to other variables, to reset a variable, or to act as a one item menu. The value can be changed using Increment and Decrement.

Increment and Decrement can be used to change a memory data variable for comparison to other variables, to create increment buttons for values that control dynamics such as rotation, movement or scale, or to implement a counter.

Set Value, Increment, and Decrement act on the variable value as it currently appears in the display, not on the value in the actual data source. For example, if the user creates a constant data variable with a value of .5 and uses rules to change the value, the data display changes, but if the value is checked in the data browser, the value is still .5.

If the Execute Function in Script option is used, the display can use only one VBA script for rules. All conditions and actions that the user wants to write in a script must be contained in that single script file. If the user's display also uses VBA scripting data sources, rule and data source scripts cannot be in the same file.

7.4.9 Using a VBA script for rule actions and conditions

Hitachi ABB Power Grids does not recommend using VBA Script for rules since it may result in performance problems as well as in strange behavior, such as error messages when opening, changing and using process displays.

7.4.10 Rules in subdrawings

All rules included in a subdrawing are active when the subdrawing's dynamics are enabled. Therefore, if rules are assigned in a display, then use that display as a subdrawing, all of its rules can be active in the interface.

Display Builder lets the user build deep hierarchies of displays and subdrawings. Rules can be attached to objects and subdrawings in displays at all levels of the hierarchy. If dynamics are enabled at all levels, all rules are active. Rules are executed starting at the bottom of the hierarchy, then proceeding upward.

Rules can affect other objects at any level in the hierarchy. However, if a rule erases an object higher in the hierarchy, no rules attached to the erased object are executed. If an object is erased by a rule attached to that object, the remaining rules in its list are executed.

7.5 Fill patterns

7.5.1 Applying fill patterns to an object

Display Builder lets thw user assign pattern effects to filled graphical objects using the **Fill** tab on the **Object Properties** dialog.

To apply pattern style fill effects:

1. Select the object.
2. Select foreground and background colors.
3. Select a fill style option.
   Plausible fill style options to use with fill effects are:
   - Fill: FG, Edge: None: The object has no linear edge, and the interior has the current foreground color.
   - Fill: FG, Edge: BG: The object has an edge of the current background color and its interior has the current foreground color.
   - Fill: BG, Edge: FG: The object has an edge of the current foreground color and its interior has the current background color.

4. In the **Object Properties** dialog, select the **Fill** tab.
5. Select the **Has Fill Effect** check box.
6. Make sure **Pattern** is selected in the combo box.

The pattern portion of the **Fill** tab allows the user to choose new pattern colors, elect to keep the original object's inherited colors, apply color transparency, as well as choose from a variety of pattern styles

7.5.2 Selecting colors for pattern fill effects

When using pattern fill effects for a selected filled object in the drawing region, the user can choose to keep the object's original colors, use color transparency, or choose new colors.

7.5.3 Inheriting colors for pattern fill effect

To retain the original object's background color settings for the pattern:

1. Locate the Color region of the tab.
2. Select the **Inherit object BG color** option.

To retain the original object's foreground color settings for the pattern:

1. Locate the Pattern Color region of the tab.
2. Select the **Inherit object FG color** option.

When combining pattern fill effect dynamics with either blink attributes or blink dynamics, select these inherit options in order to retain the blink color settings.

7.5.4 Using transparency in pattern fill effect

To apply a transparent background color for the pattern:

1. Locate the Color region of the tab.
2. Select the **Transparent** option.

To apply a transparent foreground color for the pattern:

1. Locate the Pattern Color region of the tab.
2. Select the **Transparent** option.

7.5.5 Applying different colors to the pattern fill effect

To apply a different background color for the pattern:

1. Select the rectangle below Color.
2. Choose a color from the color table.

To apply a different foreground color for the pattern:

1. Select the rectangle below Pattern Color.
2. Choose a color from the color table.

7.5.6 Applying pattern styles for pattern fill effects

When using pattern fill effects for a selected filled object in the drawing region, the appearance of the pattern fill effect can be changed by changing the pattern style.

To choose a pattern style, select an appropriate style from the palette of pattern styles.

The user can choose from various styles of patterns. Click the desired style with the mouse to apply the pattern.

Section 8 Terminology

The following table presents a list of terms associated with Display Builder that the user should be familiar with. The list contains terms that are unique to MicroSCADA X Control System SYS600 or have a usage or definition that is different from standard industry usage.

Term	Description
Clut file	Color LookUp Table, file where Display Builder saves the RGB (Red, Green, Blue) values for each color in color palette.
custom attributes	Display specific keys with text value.
data source	Source of incoming data that controls the behavior of process displays. In SYS600, primary data source is the process database. Other possibilities are, for example, files, display specific memory and constant values.
data variable	Single item in a data source. Each variable has value, name, type, scope and shape. Different types of data sources can contain different types of variables, for example, the type of a variable in a file data source can be float or text. Private scope variables can be accessed only within the display, public scope variables can be accessed also if the display is used as a subdrawing in other displays. Shape can be scalar, vector or matrix.
display	Drawing that contains all objects, properties, data sources and data variable mappings that have been created or edited with Display Builder. Can be saved in Display file format .v or subdrawing format .sd.
drawing area	In Display Builder, the drawing area contains the space where you edit the objects.
object	Graphical item in a process display or a logical entity in the process database. The graphical objects can be classified into primitive objects (lines, rectangles and so on), text objects, graphs, subdrawings, bitmaps, and input objects.
Power Process symbols	Set of standard symbols for substation automation, power transmission and distribution, and other electrical applications.
subdrawing	Subdrawings are displays that are made in Display Builder and are saved as a subdrawing file format .sd.

Section 9 Abbreviations

The following is a list of abbreviations associated with Display Builder that the user should be familiar with. See also Section 8.

Abbreviation	Description
CPU	Central Processing Unit
DMS	Distribution Management System
ASCII	American Standard Code for Information Interchange
SA	Substation Automation
VBA	Visual Basic for Applications
WMF	Windows Meta File

Appendix A Display configuration conversion to SYS600 9.4 FP2 or later

Display configuration files made for Monitor Pro are automatically converted to be used in Monitor Pro+. Changes in the display configuration files are continuously monitored and an automatic conversion takes place whenever the display configuration is changed. The automatic conversion has been made as complete as possible, but due to different graphics technologies, the converted display configurations can have the following limitations.

1.1 Process Display locations

The supported Process Display locations in the conversion are PICT and its subfolders although it has been possible to save the display configuration files to any location from Display Builder.

Regarding the symbols, both the application and system palettes are supported. Possible references outside PICT will be handled in addition to the palette locations mentioned earlier.

1.2 Ellipses and splines

The conversion regarding ellipses and splines is not exact. Monitor Pro defines ellipses using three points and splines using a collection of points. Monitor Pro+ is based on SVG graphics, and native drawing methods are used, which are based on different point calculations.

1.3 Fill effects

The fill patterns and line style conversions may look different in Monitor Pro+ than in Monitor Pro. In addition, fill patterns do not scale in Monitor Pro but in Monitor Pro+ they do.

1.4 Dynamic features

All dynamics except Arc Direction, Curve Type and Path Move are converted. The initial status of the process object is seen in the display, but dynamics are not applied for such process objects.

In Display Builder, it is possible to group dynamics to feature sets, and the change in the feature sets will affect the objects using it. These feature sets are not converted as is, but the instantiated dynamics are supported.

1.5 Rules

All rule conditions except Object’s Var Compares to Value and Execute Function in Script are converted.

Rule action types regarding object handling and calling functions in a script are not converted.

The non-converted action types are as follows:

- Overlay Object
- Erase Overlay Object
- Popup Object at Cursor
- Erase Popup Object
- Erase Visible Popups
- Redraw Screen
- Disable Rules
- Do Nothing
- Execute Command
- Resume
- Pause
- Faster
- Slower
- Call Generated Handler
- Execute Function in Script

1.6 Input objects

Layout files (.lay), which define the appearance of an input object in Monitor Pro, are not converted.

The Text Entry input object is converted to a HTML5 input element with a transparent background (the appearance is not exactly the same as in Monitor Pro). In Monitor Pro+, the text was cleared instantly by right-clicking the mouse button, but in Monitor Pro, the text is not cleared by right-clicking.

The button is converted to Monitor Pro+, but the graphical appearance differs from Monitor Pro, for example, the toggle effect is not implemented.

1.7 Graphs

Only basic graphs are converted.

Currently supported graphs:

- Meter graph
- Line graph 20 samples.sd – Limits for the symbol must be given.
- Bar graph.sd – Limits for the symbol must be given.

1.8 VB scripts

VB scripts are not converted.

1.9 MFC components

MFC components are not converted.

1.10 Bitmap mask

Monitor Pro+ supports transparency as defined by supported image formats (for example, GIF, JPG, and PNG).

1.11 Bitmap color table

Monitor Pro+ supports true color images. A palette-based format is used, and the palette defined in the bitmap file is used.

1.12 Symbols (9*, 10) conversion

Symbols from the palette folder "\sc\prog\graphicsEngine\Palette\10 - Scales" and "\sc\prog \graphicsEngine\Palette\9*\" are not automatically converted.

1.13 File Data Source

File Data Source is not supported. Data variables in File Data Source type are converted to memory variables. The conversion will issue a warning if this happens.

1.14 Vector texts

The .vf file definitions in folder \sc\prog\graphicsEngine\lib\fonts are not honored since custom applications are using TrueType fonts. A hard-coded font will be used for vector texts.

1.15 Vector text attributes

Vector text C-Space, L-Space and Slant definitions are supported.

1.16 Text direction

In Monitor Pro, the vertical direction shows the text in a horizontal orientation. Monitor Pro+ will change the text direction completely.

1.17 Transparent pattern fill effect

When transparency in the pattern fill effect is being used, it will be converted to the actual object foreground or background color depending on the configuration.

1.18 Blink with fore/background color dynamics

It is possible to define the objects' fore/background color via fore/background dynamics or with blink dynamic. Depending on the order in which the colors were defined, Monitor Pro has decided the coloring. Monitor Pro+ graphics will always use the colors defined in the blink dynamic overriding the fore/background color dynamics.

When one color dynamic is prioritized, it is clearer and simpler for the user to see which color is used. The conversion will issue a warning if a conflicting dynamics combination is used.

1.19 SetViewResForPriority feature

Sets a view resolution value to a given visibility priority. Determines which objects should be hidden when the view is zoomed to different view resolutions. The view resolution for a given priority can be obtained using the GetViewResForPriority method.

Monitor Pro+ supports these values with the following Custom Attributes: SAPriorityResolution1, SAPriorityResolution2, SAPriorityResolution3, SAPriorityResolution4, etc.

1.20 Removing ToolLauncher Actions from Symbol

Unchecking the “From subdrawing” checkbox in ToolLauncher does not have any effect. The ToolLauncher actions defined in the symbol are always executed and cannot be disabled separately from the main display.

A workaround solution is to copy the original symbol and remove the actions from it. Then use the modified symbol in the process display.

1.21 Custom Busbars

Customized busbars are in most cases converted to regular busbars and the extra functionality is discarded, if the symbol name ends with a text “busbar [horizontal |vertical | L].sd” (these are default product busbar names). If the custom busbar is not downgraded to regular, it might have a negative effect on the display performance and/or have issues with the topology coloring.

1.22 Bidirectional data variables in Symbols

Symbols where a data variable is used so that the value in the subdrawing is updated and the updated value is used in the containing sub-drawing. This functionality is no longer supported.

One example of such mapping is Button.sd with Toggle/Sunken data variables.

1.23 Blink with Fill dynamics

Combining the blink with any of the fill dynamics/effects will not function reliably. In most cases, both dynamics will fail to function correctly (for example, the item will not blink and is always filled to max/not at all). The conversion will issue a warning if such combination exists in the view.

1.24 Custom Process Display Notes are not supported

Custom Process Display Notes conversion from Monitor Pro to Monitor Pro+ is not supported.

1.25 Visibility layers and Z-order of the objects

The Z-order of the objects at the same visibility layer will be sequential in the converted version. Therefore, there may be a deviation in the relative Z-order of the objects in different visibility layers. This means that the different object can be on top of the other object in the converted version, if they overlap.

1.26 Functional differences between Monitor Pro and Monitor Pro+

1.26.1 Binary format conversion

View and symbol files, which are stored in the binary format, are not compatible between Monitor Pro and Monitor Pro+. If there exists a view or symbol file, which has been developed in a different environment, the display conversion for the file may fail with the error message "GIPSY core error - ...". To successfully convert the file, save the file in the ASCII format using Display Builder.

1.26.2 Drag-and-drop of Process Display files

Drag-and-drop of Process Display files is not supported in Monitor Pro+.

1.26.3 Scroll bars

Scroll bars are not supported in Monitor Pro+.

1.26.4 Middle mouse button pan

Middle mouse button panning is not supported in Monitor Pro+.

1.26.5 Find Process Objects functionality

The Find functionality in Monitor Pro+ differs from that of Monitor Pro, for example, Busbars are not found in Monitor Pro+.

1.26.6 Touch behavior in Monitor Pro+

Panning and (pinch) zooming with touch gestures will depend on the pan toggle in Monitor Pro +. Previously Monitor Pro allowed panning and zooming with touch gestures even with special gestures (pinch and two-finger pan) and when the pan toggle was not enabled. The touch support can still be disabled with command line switches. There are minor changes in the behavior if only touch zooming is disabled: it is possible that the disabled pinch gesture is interpreted as a two-finger pan.

Navigation swipes are not supported (back and forward flick) in Monitor Pro+.

1.26.7 ANSI-encoded Process Display Notes

A Note Marker file encoded in ANSI is not supported in Monitor Pro+. The encoding was changed from ANSI to UCS-2 LE in order to support Cyrillic characters in the note text (in the Note Marker dialog). As a workaround, file encoding can be set manually.

1.26.8 Process Display Notes colors

When the color of the Process Display Notes is changed in Monitor Pro+, the color is not backward compatible with Monitor Pro.

1.26.9 System resource consumption

Monitor Pro+ may consume more system memory than Monitor Pro. The actual amount depends on the application.

Index

Hitachi ABB Power Grids
Grid Automation Products
PL 688
65101 Vaasa, Finland

https://hitachiabb-powergrids.com/microscadax

Scan this QR code to visit our website

1MRK 511 478-UEN