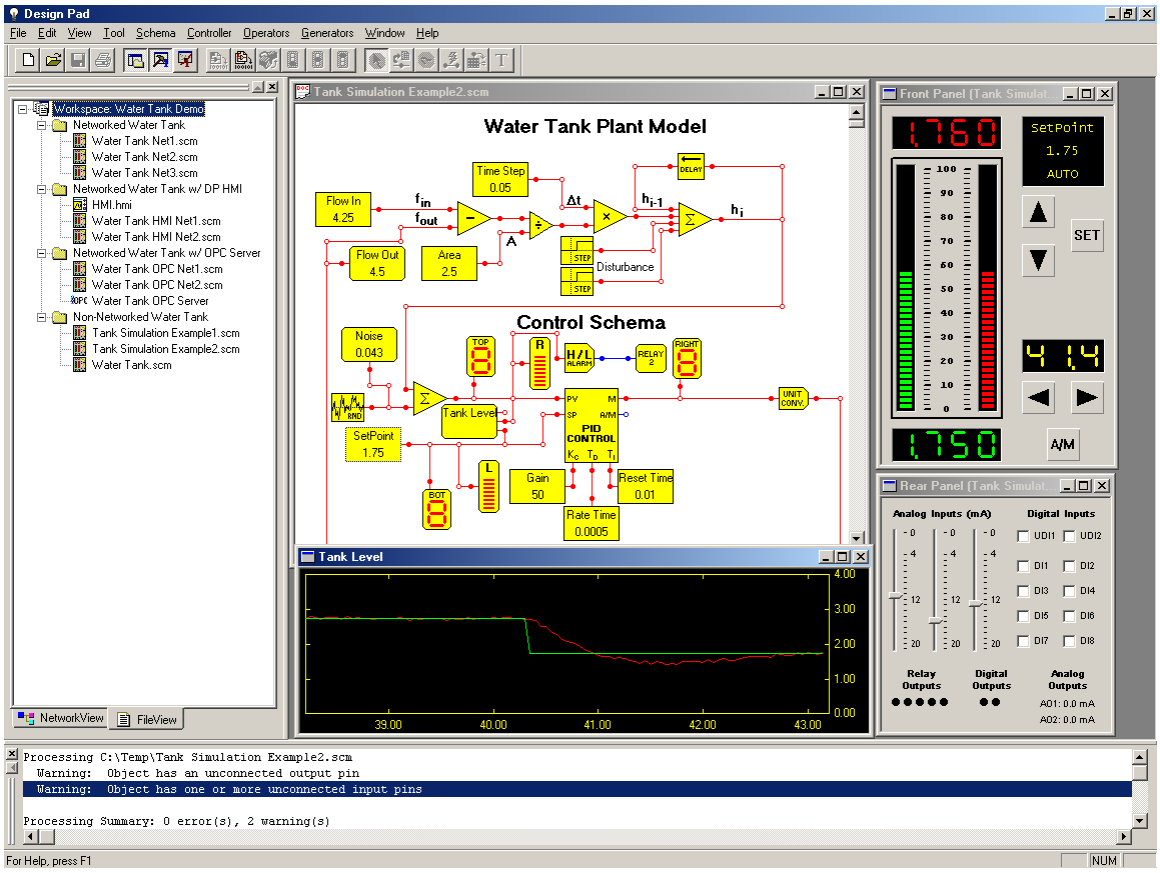
FAIRMOUNT AUTOMATION

Design Pad 2000 Professional Instruction Bulletin

A programming guide for the FAC-2000 controller



Fairmount Automation Controller Series, Model 2000

Version 2.1
May 2001

Technical Bulletin 9110-0002

NOTICE

This document contains proprietary information of Fairmount Automation, Inc. This document and the software described herein, are owned by Fairmount Automation, Inc. and are protected by United States copyright laws and international treaty provisions. The document may not be used (or reproduced for use) to reverse-engineer, develop, or manufacture the software described herein. No part of this document may be photocopied or reproduced without the prior written consent of Fairmount Automation, Inc.

The information contained in this document is subject to change without notice.

Fairmount Automation, Inc. is not obligated to provide updates to this document or the software described herein.

Fairmount Automation, Inc. and its employees specifically disclaim all liabilities and warranties, express and implied (including warranties of merchantability and fitness for a particular purpose), for the accuracy, currency, completeness, and/or reliability of the information contained herein. Fairmount Automation, Inc. and its employees further disclaim all liabilities and warranties, express and implied, for the fitness for any particular use and/or for the performance of any material and/or equipment selected in whole or part with the user of/or in reliance upon information contained herein. Selection of materials and/or equipment is at the sole risk of the user of the publication.

The software described herein is provided without warranty of any kind. Any use by you of this software is at your own risk. To the maximum extent permitted by law, Fairmount Automation, Inc. disclaims all warranties of any kind, either express or implied, including, without limitation, implied warranties of merchantability and fitness for a particular purpose.

1. CONTENTS

The Fairmount Automation Controller Series, Model 2000 *(FAC-2000)* is a general-purpose, highly configurable, multi-loop process controller. This instruction bulletin describes Fairmount Automation's *Design Pad*® —a software package used to program the *FAC-2000* controller.

1.1 What's new in *Design Pad 2000*?

This bulletin is devoted to *Design Pad 2000* and *Design Pad 2000 Professional*, the second generation of Fairmount Automation's *Design Pad* software package. This new version contains a number of features that were not available in prior versions. Whenever a new feature is described in the documentation, the *new feature graphic* (shown to the right) will appear in the right margin of the bulletin.

Some of the new features found in *Design Pad 2000* include:

- A new *workspace* environment to organize the multiple controller stations (and their associated control schemas) used throughout a plant.
- Support for *FAC-2000* networking including:
  1. New networking operators (signal broadcasters, signal receivers, remote constants, and remote A/M buttons) to support inter-controller communication over RS-485 and/or RS-232 data lines
  2. An OPC Server configuration tool to interface a controller network to a PC network
- New schema editing features (*e.g.,* copy/paste, selection box to select multiple operators)
- Multi-schema simulations (to test a group of networked schemas)
- *FAC-2000* rear-panel simulation
- Multi-input probe operators with data recording features
- Automatic file recovery for unsaved documents during a system crash
- Automatic creation of back-up files for schema documents
- Integrated e-mail support for schema documents
- Recent file and recent workspace lists on the File menu
- Window docking support
- Operator property printing support
- Print preview support

TABLE OF CONTENTS

LIST OF FIGURES

2. INTRODUCTION TO *Design Pad 2000™*

Design Pad 2000 is a graphical programming environment for the *FAC-2000* Controller. It makes programming control schemes for the *FAC-2000* as easy as connecting functional operator blocks together. Over 60 such *operators* are provided with *Design Pad 2000*, including controller blocks, signal conditioning functions, comparison blocks, math operators, logic functions, networking operators, and hardware components. The hardware blocks provide access to the *FAC-2000* hardware, including its analog inputs and outputs, digital inputs and outputs, relays, LED bargraph displays, numeric displays, alphanumeric displays, and keypad buttons.

Design Pad 2000 provides the control engineer with complete design flexibility. There are virtually no restrictions on how the functional operators can be interconnected. Moreover, all the functional operators have user-configurable properties. Want a design parameter to be adjustable from the *FAC-2000*'s front display panel? No problem—just enable the *Constant* operator's '*Front Panel Access*' property and specify minimum, maximum, and increment magnitude values. Want to specify the brightness of a bargraph display? It's simple—just modify the 'Brightness' property of the *Bargraph Display* operator. Want to change the number of inputs for an *Addition* operator? Want to repeat a message on the lower alphanumeric display six times? Want to specify the dead-band of a *Limiter* operator? Done—just a few keystrokes and mouse-clicks is all it takes.

Perhaps the best way to illustrate the power and flexibility of *Design Pad 2000* is by example. A typical *Design Pad* "program" implementing a PID control strategy to regulate the water level in a tank is shown in Figure 1. In this control *schema*, a *PID* operator (the large block at the center of the figure) acts to maintain a process variable measured by analog input 1 at a set point of 1.75 meters. (In the diagram, an *Analog Input* operator is connected to the Process Variable—PV—input of the *PID* operator. And, a *Constant* operator—labeled "Set-Point"—is connected to the Set Point—SP—input of the *PID* operator.) Also connected to the *PID* operator are three *Constant* operators setting the PID operator's proportional gain (unitless), derivative time (in minutes), and integral time (in minutes) to 1.25, 0.005, and 0.1, respectively.

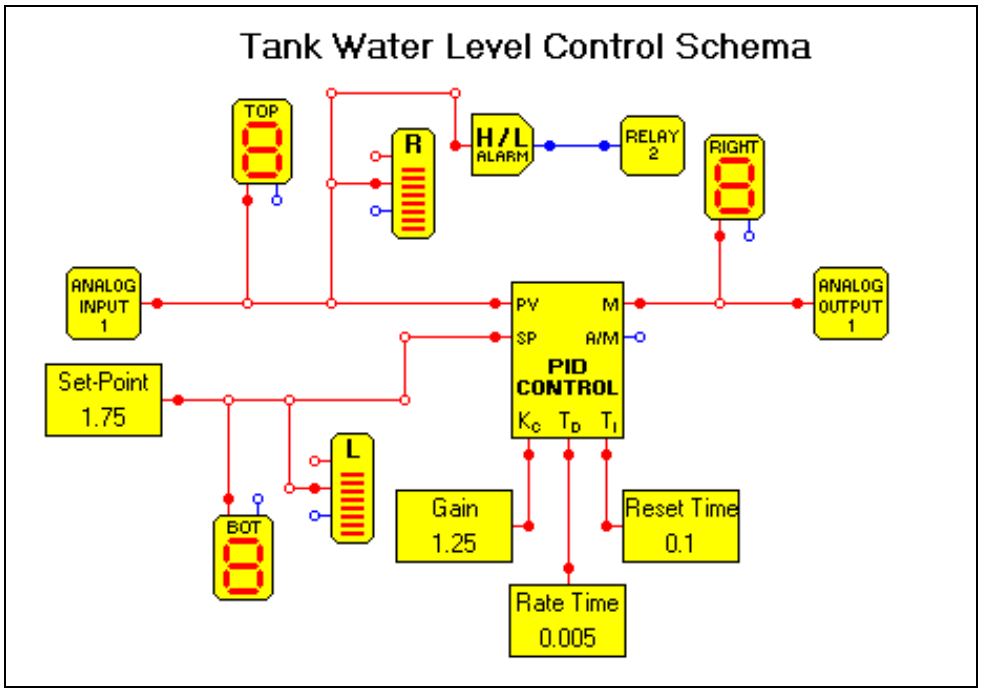


Figure 1. A typical *Design Pad* "program": a schema diagram of a simple PID controller.

The schema diagram of Figure 1 further specifies that the set-point signal should be displayed on the *FAC-2000* left bargraph display, as indicated by the connection between the "Set-Point" *Constant* operator and the left *Bargraph Display* operator. The set-point signal is also to be indicated on the bottom-left numeric display. And, the process variable is to be shown on the right bargraph display and on the top-left numeric display.

The output *M* of the *PID* block (the control signal) is fed to analog output 1 of the *FAC-2000* controller to regulate the physical process. The control signal value is also displayed on the right numeric display of the *FAC-2000* front panel, and is passed to a *High/Low Alarm* operator. The *High/Low Alarm* operator compares the output signal to upper and lower limits. It activates Relay 2 of the *FAC-2000* unit if the output signal exceeds these limits. (The relay could be tied to an audible alarm that would alert plant personnel that unsafe operating conditions have been reached.)

All of the operators in the schema diagram of Figure 1 have user-definable properties. Figure 2 shows the property sheet for the "Set-Point" *Constant* operator. Since we have set the '*Front Panel Access*' property of this operator to 'Operator level', plant personnel will be free to adjust the controller set point directly from the *FAC-2000* front panel. When plant personnel push the SET button on the *FAC-2000* faceplate, they will be presented with the set-point value and allowed to adjust it by pressing the UP/DOWN arrow keys. Each UP/DOWN arrow key-press will adjust the set point by 0.05 meters (as specified by the '*Increment Magnitude*' property—see Figure 2). Also, as shown in Figure 2, we have limited the set-point range: it may not exceed 3.00 meters and must be at least 1.00 meter (as defined by the *'Maximum'* and *'Minimum'* properties).

The control schema shown in Figure 1 represents the first step in the *FAC-2000* programming process. The next step is to download the schema into the device. *Design Pad* has built-in functionality that provides two-way communication between a personal computer and the *FAC-2000* unit. Once you have completed your schema design, simply connect your computer to the controller via serial cable, choose the Export Schema item on the Controller menu, and voilà—the *FAC-2000* is running your control scheme!

Other *Design Pad* communication features allow you to retrieve a schema from a programmed controller and to calibrate the device. (The *FAC-2000* controller is initially calibrated at the factory. The calibration facility mentioned here might be useful after long periods of sustained device operation. For instance, it could be used to make minor adjustments on an annual basis.)

Design Pad also offers powerful simulation capabilities. You're not sure if a particular control scheme will work? Rather than trying it on the actual device, why not run a simulation first? *Design Pad* provides a number of signal generators to simulate input signals. And, with *Design Pad's Probe* operator, you can view a dynamic plot of any schema signal. In addition, with *Design Pad's Front Panel Window* emulation, you can verify that the *FAC-2000* front panel displays and keypad buttons behave as you planned. *Design Pad 2000* also includes *Rear Panel* emulation, which allows you to adjust *FAC-2000* input signals and monitor its output signals. In order to test a control strategy, you simply connect probe operators to signals of interest and choose the Start Simulation item on the Schema menu. *Design Pad* will simulate the process in near real-time and display schema

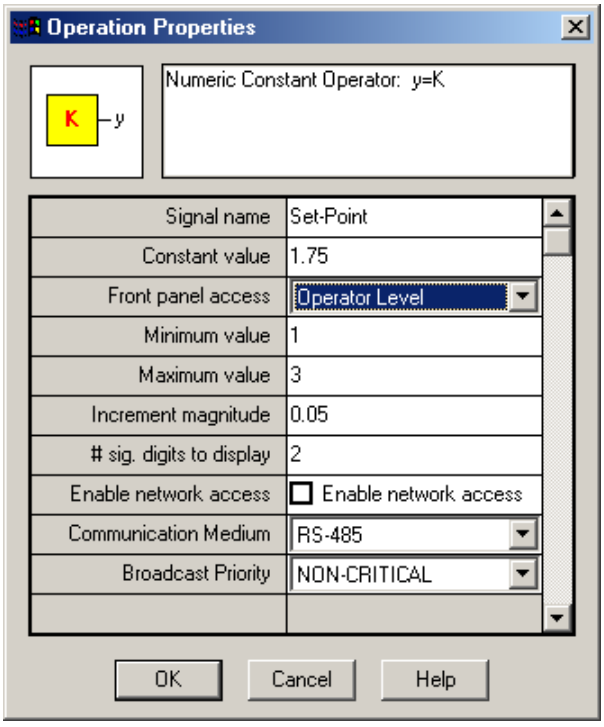


Figure 2. Property sheet for the "Set-Point" *Constant* operator of Figure 1.

signals as they vary with time. Feedback from simulation experiments may make the difference between a truly robust control scheme and one that is susceptible to occasional disturbances.

3. HARDWARE AND SOFTWARE REQUIREMENTS

To use *Design Pad*, your computer must have:

- An 80486 (or compatible) CPU or greater
- 8MB (or more) of RAM
- A hard disk with 8MB of available disk space
- A serial communications port (to interface with the *FAC-2000* controller)
- A 1.44 floppy drive (for installation)
- Windows 95 or higher
- A Windows-compatible mouse

4. PROGRAMMING THE *FAC-2000* WITH *DESIGN PAD*

The *FAC-2000* controller is programmed with Fairmount Automation's graphical programming software package: *Design Pad™ 2000*. All "programming" is done graphically—by "drawing" connections between functional operator blocks on a *Design Pad* window (see Figure 1 for a sample "program").

Design Pad supports well over 60 operators, including

- Controller Blocks (*e.g.,* PID controller, lead-lag controller)
- Signal Conditioning Functions (*e.g.,* characterizer, rate limiter, track & hold)
- Signal Comparator Blocks (*e.g.,* high/low alarm, equality, less or equal)
- Mathematical Operators (*e.g.,* addition, multiplication, absolute value)
- Logic Functions (*e.g.*, NAND gate, XOR gate, RS flip flop)
- General Purpose Operators (*e.g.,* timer, delay element, multiplexer)
- Hardware Operators (*e.g.,* analog input, bargraph display, A/M button)
- Network Operators (*e.g.*, analog broadcast, digital receiver, remote A/M button)

A detailed description of all the operators available in *Design Pad* is provided in section 11 of this instruction bulletin.

The collection of operators and their associated interconnections is called a control *schema*. The schema diagram represents the actual program that will be run on the *FAC-2000* controller. The sections that follow describe how to create a schema, how to adjust schema properties and properties of individual schema operators, and how to compile and download a schema into the *FAC-2000* controller.

4.1 Creating a Control Schema

A *schema* is a collection of interconnected functional operator blocks that form a control strategy. It is a graphical block-diagram that represents the algorithm that is to be run on the *FAC-2000* controller. A sample schema is shown in Figure 1. (A digital copy of this schema file is located in the `.\Fairmount Automation\Examples` directory. The file name is `H2O_TANK.SCM`.). This section provides a step-by-step description of how you could synthesize that schema.

Step 1. Create and Name a New Schema Document

New Document

To begin the design process, choose the New Document item in the File menu (or press the New Document button on the speed-bar menu). When *Design Pad* prompts you for the document type, choose the schema option. *Design Pad* will respond with a blank

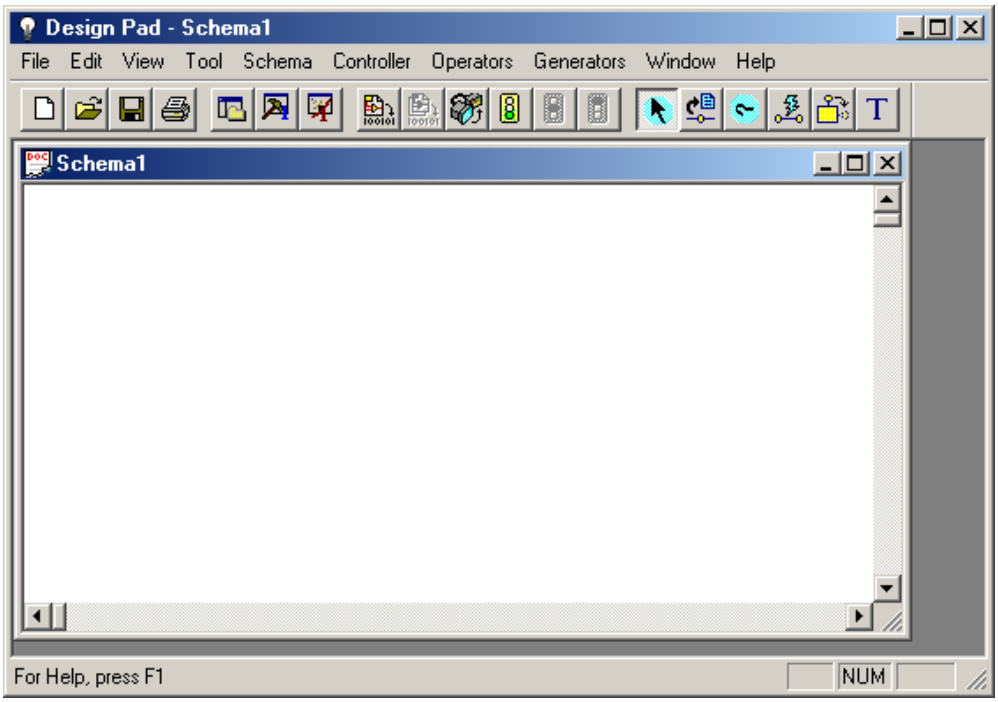


Figure 3. *Design Pad* environment containing a blank schema document.

schema document window, as shown in Figure 3. (Each window in the *Design Pad* client area holds a distinct schema diagram, allowing you to work on multiple schemas simultaneously).

Save Document

Open Document

Before you begin creating the schema of Figure 1, first use the Save Document As item in the File menu to name your schema and save it to disk. In the File Open/Save dialog shown Figure 4, name your blank schema document '`pidtest.scm`' and press the Save button. Be sure to regularly save your documents to disk as you develop and make changes to your designs. To save schema documents, use the Save Document item in the File menu; to retrieve them, use the Open Document item in the File menu. (As an alternative to using the menu items, you may press the speed-bar buttons shown to the left.)

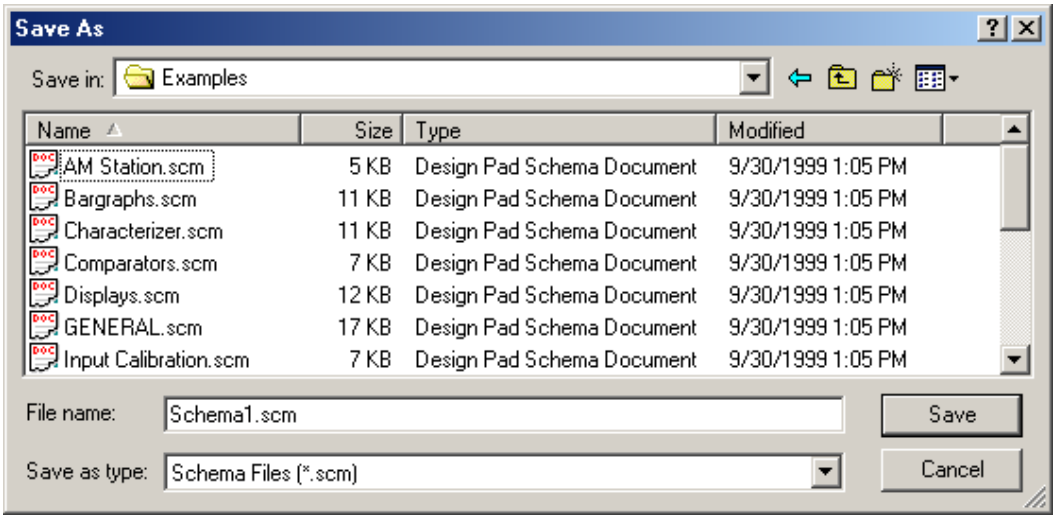


Figure 4. Save As dialog box

Design Pad 2000 maintains a list of recently opened schema documents. To open a schema document that you have used recently, select the Recent Files item in the File menu. A sub-menu will appear listing the four most recently open schema documents. If the document you wish to open is on the list, select it. (You can change the number of entries in the recent-file list in the Preferences dialog—see section 10.1.)

Design Pad 2000 has an auto-recovery feature designed to recover unsaved changes to schema documents after an abnormal program termination. When the auto-recovery feature is enabled (see section 10.2), *Design Pad* saves copies of open schema documents at regular intervals. When you execute *Design Pad*, it checks to see if it terminated abnormally during a prior session (*e.g.*, due to power interruption). If it detects an abnormal termination, it prompts you if you would like to attempt to recover unsaved changes to schema documents.

Design Pad 2000 automatically creates a back-up copy of a schema document before it saves any changes to the document. For example, when you save changes to the `pidtest.scm` schema file, *Design Pad* first makes a copy of the existing file (and saves the copy as `pidtest.bak`) before overwriting `pidtest.scm` with the changes that you made to it. This feature could be useful if you have saved changes to a schema but later decide you want to discard those changes and revert to the back-up version of the schema. To disable the auto-backup feature, see section 10.2.

Step 2. Insert Functional Operator Blocks

You are now ready to begin creating a control schema. To begin, you will want to insert functional operator blocks into a blank schema document. The Operators menu contains all the operator blocks available in *Design Pad*, grouped by functionality (*e.g.*, controller blocks, signal conditioning functions, logic operators). Each operator is described in detail in the operator reference section of this instruction bulletin (section 11).

Under the Controller Blocks sub-menu of the Operators menu, choose the PID Controller item. *Design Pad* will attach a *PID operator* object to your cursor, and will "drop" it wherever you click the left mouse button. Thus, to position the PID block, move your

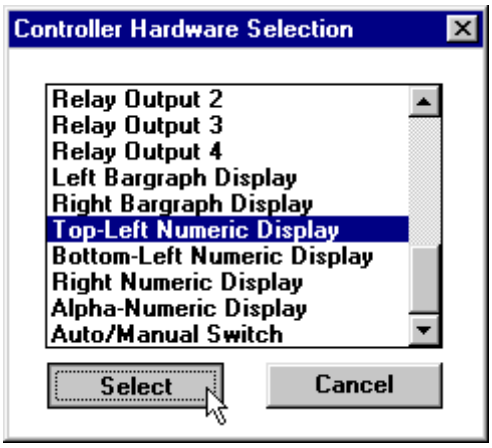


Figure 5. The Controller Hardware Selection dialog.

cursor to the center of the document window and click the left mouse button.

Next, choose the Controller Hardware item from the Operators menu. *Design Pad* will respond with the Controller Hardware Selection dialog box that contains all of the *FAC-2000* hardware operators—the device's input/output channels and its faceplate buttons and displays. As shown in Figure 5, highlight the *Top-Left Numeric Display* listbox item, and press the Select button. As before, the operator block will be attached to your cursor until you press the left mouse button. Position the *Numeric Display* operator above and to left of PID operator block.

Next, use the Controller Hardware Selection dialog box again, select the *Analog Input 1* operator, and position it below, and to the left of, the *Top-Left Numeric Display* operator. While the Controller Hardware Selection dialog is active, you should notice that listbox item *Top-Left Numeric Display* has been "dimmed" (it may no longer be selected). Hardware operators are limited resources—the number of input/output channels and faceplate displays are finite physical characteristics of the controller. Each hardware operator can only be used once in each schema file.

There are two hardware operators that are exempt from the single-use rule, since they are shared resources: *Alphanumeric Display* and *A/M Button* operators. The *Alphanumeric Display* operator provides access to the three alphanumeric displays on the *FAC-2000* faceplate. You can associate a text string with each operator, and write unlimited messages on the alphanumeric displays. (The messages will be placed in a queue, and cycled on the text displays.) The *A/M Button* operator provides access to the A/M button on the *FAC-2000* faceplate. It can be used to switch a control loop between automatic and manual modes. You may use several *A/M Button* operators in the same schema, associating each with a subsection of the control system.

Selection Tool

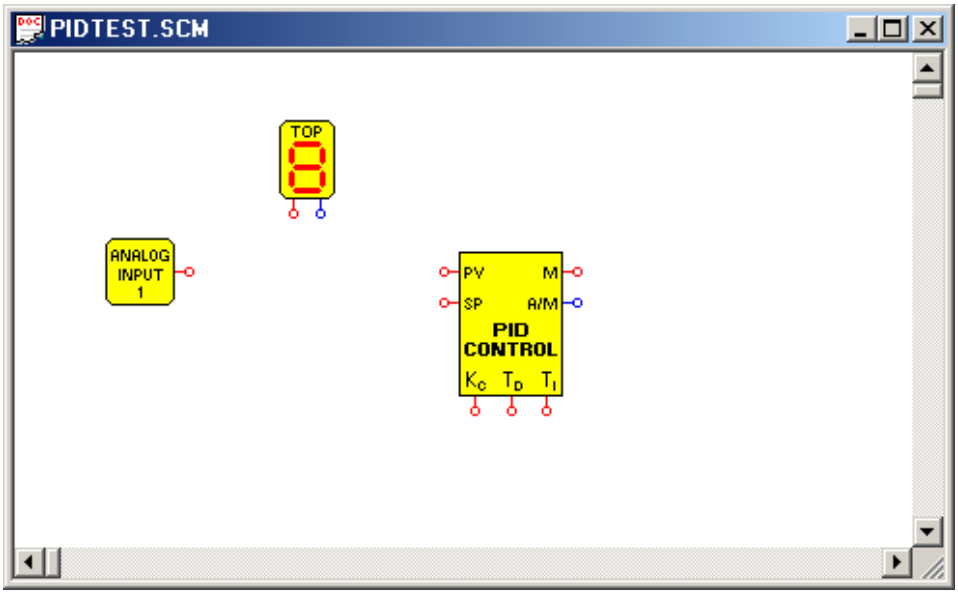


Figure 6. Initial stage of schema development.

At this point, your schema diagram should resemble the schema shown in Figure 6. Note that you can change the position of any operator by "dragging" it. (With Selection tool active, position the cursor within the operator and click the left mouse button. Hold the mouse button down while moving the cursor to the desired location, then release the mouse button.) You can also change the position of several operators and signals at the same time. First, select the operators you wish to reposition (hold the SHIFT key as you click the left mouse button on each object). Then, drag the objects to their new location—simultaneously hold down the left mouse button and the SHIFT key on your keyboard while moving the cursor to the desired screen position.

NEW

Design Pad 2000 provides a selection box feature that is useful for selecting multiple operator objects at the same time. To activate the selection box, click the left mouse button over schema *whitespace*—a location in the schema that does not contain an operator block, a signal wire, a text block, or any other object. Then, as you move the mouse (while holding the left button down), *Design Pad* displays a rectangular box—the selection box. When you release the left mouse button, *Design Pad* will select all the objects that are inside the selection box.

By default, schema documents have a blank white background. However, when you are creating a schema, you may find it convenient to display a grid in the background. The grid can be a useful alignment tool when you are arranging objects. To activate the background grid, select the Grid item from the View menu. You can remove the grid by selecting the Grid item again.

As illustrated in Figure 6, your schema should have three operators *(PID controller, Analog Input 1, and Left-Top Bargraph Display*) with no connections between them. You could now continue adding the remaining operators in Figure 1, or you could begin making connections between the existing blocks. The next subsection describes the process of connecting operators.

Step 3. Connect Operator Blocks

The next step in the schema design process is to make connections between the operator blocks. That is, to feed the output of one operator to the input of another. *Design Pad* supports two signal types: analog signals (floating point values) drawn in red, and digital signals (boolean value, *i.e.,* true/false or high/low) drawn in blue. These signal types cannot be directly mixed—you may not connect an analog signal to a digital one. (However, *Design Pad* does provide operators to convert from one type to the other.)

Operator blocks have input and/or output pins (I/O) that serve as connection points or "hot-spots". An I/O pin can be an operator input, or an operator output, but not both. It is drawn as a short line attached to a small circle:  . The unfilled square indicates that the pin is not connected. When the pin is connected, *Design Pad* fills in the circle:  . (*Design Pad* will not draw the circle—filled or unfilled—if the Operator "Hot-Spots" item in the View menu is not selected. This can be useful when printing a schema.)

While every operator has at least one I/O pin, most have both input and output pins. For instance, the *PID* operator has 5 analog inputs (SP, PV, Kc, Ti, and Td); an analog output (M); and, a digital output (A/M), as shown in Figure 6. The *Analog Input* operator has a single analog output, and the *Numeric Display* operator has two input pins (one analog and one digital).

The PID control schema of Figure 1 was designed under the assumption that sensor measurements from an external process are sampled by analog input 1 of the *FAC-2000* controller. Therefore, the *Analog Input 1* operator signal was fed to the process variable (PV) input of the *PID controller* block. To make this connection in your schema, select the New Signal tool from the Tools menu (or press the corresponding button on the toolbar menu). Position the cursor over the output pin of the *Analog Input* block, click the left mouse button, and move the cursor away from the operator. You should notice that a signal "wire" now connects the *Analog Input* block to the cursor—*Design Pad* has switched into *wiring* mode. You can control the configuration of the signal wire with the mouse as follows:

New Signal Tool 

- If you left-click on white space, a signal joint  will be created, and you will remain in wiring mode. Signal joints, or "hot-spots" are useful when reshaping a signal. (*Design Pad* will not draw the signal joints if the Signal "Hot-Spots" item in the View menu is not selected. This can be useful when printing a schema.)

- If you click the right mouse button while in wiring mode, the last signal joint will be deleted. If no signal joints remain, the wire will be removed, and *Design Pad* will switch out of wiring mode and into normal mode.

- If you double-click the right mouse button, all signal joints will be deleted, the wire will be removed, and *Design Pad* will switch into normal mode (out of wiring mode)

- If you click the left mouse button while the cursor is positioned over an operator I/O pin, the wire will be connected to that pin, and *Design Pad* will switch into normal mode (out of wiring mode).

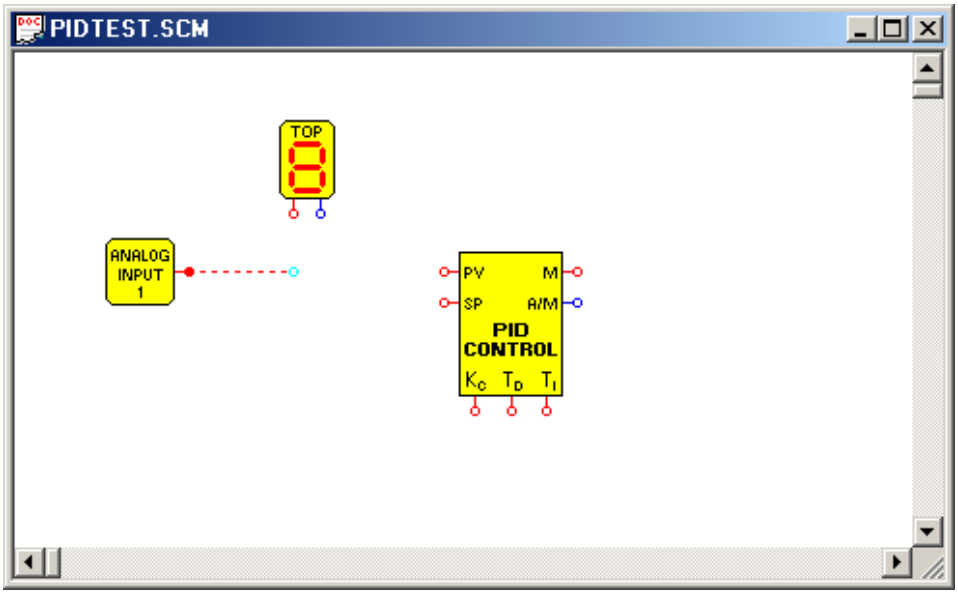


Figure 7. The Analog Input 1 operator is about to be connected to the PID operator.

- If you click the left mouse button while the cursor is positioned over another signal, the two signals will be connected, and *Design Pad* will switch into normal mode (out of wiring mode).

Figure 7 illustrates the connection process. A signal wire has been connected to the output of the *Analog Input 1* object. The cursor holds the active signal wire and is moving toward the PV input of the PID block. When the cursor is positioned over the PV input pin, the left mouse button will be clicked, thereby connecting the two blocks.

When a new signal is being created, *Design Pad* displays the signal wire as linear dashed segments, with signal joints between segments. The dashed line indicates that the signal is *selected*. Signals that are not selected are drawn as solid lines. (A similar convention is used for operators as well—the outline of an operator is solid when not selected and dotted when selected.)

The *PID* and *Analog Input* operators of your schema should now have a wire connecting them. The next step is to connect the same signal to the *Numeric Display* operator. Position the cursor on the analog input pin of *the Numeric Display* operator and click the left mouse button. While in wiring mode, move the cursor down until it is directly over the existing signal wire, and click the left mouse button again to make the connection.

The next section takes up the topic of operator properties and completes the process of creating the sample schema of Figure 1.

4.2 Modifying Operator Properties

Every operator block has a list of user-definable properties. These properties determine the characteristics and behavior of each operator. For instance, the 'Initial State' property of the *PID* operator determines the startup mode—automatic or manual. The *Moving Average* operator has a property that sets the number of signal samples to consider in the computation of the mean. And, a property of the *Multiplication* operator establishes the number of input signals that the operator can process. The properties of every operator are described in detail in the operator reference section of this instruction bulletin (section 11).

Design Pad organizes the properties of each operator object with individual property sheets (see Figure 2 for an example). The property sheet contains a description of the operator and a two-column table of object parameters. Each row in the table contains the name of the property (left column) and the corresponding text, numeric, or boolean value assigned to that property (right column). To view the property sheet of an object, activate the Selection Tool, position the cursor on the object, and double-click the left mouse button.

With this brief introduction to operator properties, you are ready to resume the process of creating the schema of Figure 1. Bring up the property sheet for the *Analog Input 1* object (with the Selection Tool active, double-click the left mouse button on the object). Recalling that this schema was designed to regulate the water level of a tank, name the object 'Water Level' and assign 0.00 meter to the 0% value and 4.00 meter to the 100% value, as shown in Figure 8. *Analog Input 1* will now map a 4-20mA signal on channel 1 of the *FAC-2000* controller, to a 0-4 meter signal. (Note that if the input signal is outside of the 4-20mA range, the output of the *Analog Input 1* operator will be outside of the 0-4 meter range. See the operator reference section of this instruction bulletin for more details.)

Continuing with the sample schema design, insert an *Analog Constant* operator from the Operators menu, and position it directly below the *Analog Input 1* object. This constant object will serve as the set-point signal for the *PID controller* block. Connect the constant object to the SP (set-point) input of the PID block. Next, retrieve the property sheet for this

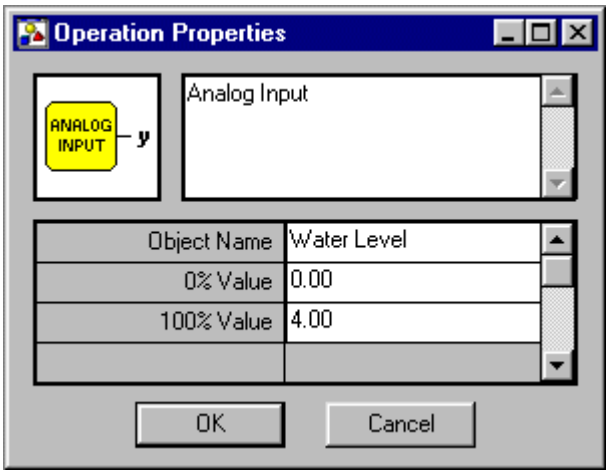


Figure 8. Properties of *Analog Input 1* object.

object and copy the values shown in Figure 2. Namely, set the object identifier to 'Set-Point' and the constant value to 1.75 meters; set the 'Front Panel Access' property to 'Operator level'; and set the minimum, maximum, and increment magnitude values to 1.00 meter, 3.00 meters, and 0.05 meter. With this arrangement, plant personnel will be able to vary the Set-Point value directly from the *FAC-2000* faceplate (using the SET button and UP/DOWN arrows). They will be able to change the set-point value in 5-cm increments, within a range of 1-3 meters.

You are now prepared to complete the schema illustrated in Figure 1. From the Controller Hardware Selection dialog, insert analog output 1, relay 2, left and right bargraph displays, and two additional numeric displays. Add three constant operators to serve as the proportional gain, derivative time, and integral time inputs of the PID object. And, select a *High/Low Alarm* operator from the Signal Comparator sub-menu of the Operators menu. Connect objects as shown in Figure 1 and inspect the property sheet of each operator (refer to section 11 for a full description of each parameter). Make assumptions about the water-level process you are controlling and adjust the operator properties accordingly.

4.3 Correcting Signal Wiring Mistakes

Connection Tool

Design Pad provides two signal wiring tools to correct connection errors: a Connection tool and a Signal Segment Zap tool. The Connection tool provides a means to connect and disconnect signals from operator I/O pins. To disconnect a signal from a pin, position the cursor over the pin, press (and hold down) the left mouse button, and drag the wire off the pin. (Of course, the Connection tool should be active as you do this.) You should notice that as you remove the wire from the pin, the pin joint changes to an unfilled circle: ⊸. To reconnect the signal to another I/O pin, continue to drag the wire until the cursor is positioned over the pin, the release the left mouse button. You should notice that the pin joint of the newly connected pin has changed to a filled-in circle: ●.

Signal Segment Zap Tool

The Signal Segment Zap tool provides a means to remove signal wire segments. To "zap" part of a signal wire, activate the Signal Segment Zap tool, position the cursor over the segment you would like to remove, and click the left mouse button. If you are interested in removing an entire signal, rather than a single segment, you can select the signal (use the Selection tool) and press the `DELETE` key on your keyboard. (Alternatively, you could choose the Delete item from the Edit menu.)

4.4 *Design Pad*'s Edit Menu

In addition to the signal editing tools described in section 4.3, *Design Pad* provides menu-driven commands to edit a schema. For instance, *Design Pad* can automatically remove any signals (or signal segments) that are not connected to anything. If you have several extraneous signals, choose the Remove Disconnected Signals item from the Edit menu, and *Design Pad* will eliminate all the unnecessary signal segments.

Deleting Operators and Signals

The Delete command mentioned in section 4.3 works with operators as well as signals. If you wish to remove an operator from your schema, select the operator with the Selection Tool and choose the Delete item from the Edit menu. (If an operator or a signal is selected, it can also be removed from the schema using the DELETE key on your keyboard.)

Selecting Multiple Schema Objects

Design Pad 2000 provides a selection box feature that is useful for selecting multiple operator objects at the same time. To activate the selection box, click the left mouse button over schema *whitespace*—a location in the schema that does not contain an operator block, a signal wire, a text block, or any other object. Then, as you move the mouse (while holding the left button down), *Design Pad* displays a rectangular box—the selection box. When you release the left mouse button, *Design Pad* will select all the objects that are inside the selection box.

Selecting All Schema Objects

Occasionally, it may be useful to select all the operators and signals in a schema. To do so, activate the Select All item from the Edit menu. *Design Pad* will then draw all the schema objects in their selected state: Operators will be drawn with a dotted border and signals will be drawn as dashed linear segments.

Activating Perpendicular Signal Mode

By default, *Design Pad* does not restrict how schema signals are laid out—signal segments can be drawn at any angle. However, schemas often "look" better if signals are drawn as horizontal and vertical line segments perpendicular to each other. (The example schema of Figure 1 is drawn this way.) *Design Pad* offers a drawing mode in which signals are restricted to horizontal and vertical planes. You can activate this *Perpendicular Signal* mode by selecting the Perpendicular Signals item from the Edit menu. *Design Pad* will place a check mark next to the Perpendicular Signals item in the Edit menu to indicate that *Perpendicular Signal* mode is active. You can return to normal drawing mode by selecting the Perpendicular Signals item from the Edit menu again.

Activating Orthogonal Mode

By default, you can select a schema object and "drag" it to any new position in a schema document window. *Design Pad* offers an *Orthogonal* mode that restricts object movement to horizontal and vertical planes. You can activate this mode by selecting the Orthogonal item from the Edit menu. *Design Pad* will place a check mark next to the Orthogonal item in the Edit menu to indicate that *Orthogonal* mode is active. To return to normal mode, select the Orthogonal item from the Edit menu again. Note that Orthogonal Mode may not be active while Perpendicular Signal Mode is active (and vice-versa).

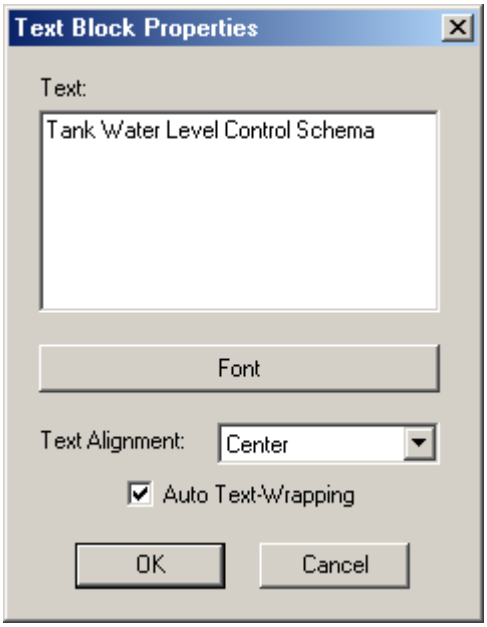


Figure 9. Text block properties dialog.

4.5 Arranging Operator I/O Pins

Pin Rotate Tool

Occasionally, the default location of an operator's I/O pin proves to be inconvenient for a desired schema layout. For instance, in the schema shown in Figure 1, the input pins of the *Bottom Numeric Display* object were placed above the object (rather than below the object—their default location.) In addition, the output pins of the *Rate Time Constant* and *Reset Time Constant* objects were shifted to the left of the objects, as opposed to their default left-of-object positions. These changes were made with *Design Pad's* Pin Rotate tool. The tool is so named because it rotates the I/O pin locations from one side of an object to an adjacent side (in the clockwise direction). To change the position of an operator's I/O pin, activate the Pin Rotate tool, position the mouse cursor over the object of interest, and click the left mouse button.

4.6 Adding Text to a Schema

Text Tool

You can add descriptive text to a *Design Pad* schema using the Text Tool. For example, in Figure 1, we added the title block "Tank Water Level Control Schema." To insert text into a schema, select the Text Tool; then, click and hold down the left mouse button. As you move the mouse, a rectangle outline will follow the Text Tool pointer. When you release the mouse button, the *Text Block Properties* dialog will appear on the screen (see Figure 9).

In the *Text Block Properties* dialog, type the text you wish to display in the *Text* edit box. Select the desired font and text alignment with the *Font* button and *Text Alignment* combo, respectively. And, if you would like the text to automatically wrap at the edges of the text

rectangle outline, check the *Auto Text-Wrapping* checkbox. When you press the OK button, the text you entered in the Text edit box will be displayed within the text rectangle outline.

If you wish to edit a text block, activate the Selection Tool, position the pointer on the text, and double-click the left mouse button. *Design Pad* will display the *Text Block Properties* dialog and allow you to adjust the text and its properties.

If you wish to reposition the text, activate the Selection Tool, position the cursor on the text, and drag the block to a new position with the left mouse button. If you wish to resize the text rectangle, activate the Selection Tool and position the pointer on one of the rectangle corners. Click the left mouse button and drag the rectangle corner to a new position. You should notice that the rectangle outline changes as you drag the mouse.

4.7 The *FAC-2000 Set Menu* System

Constant operators (analog or digital) often serve as fixed values that feed other operators. However, they can also function as adjustable parameter values accessible from the *FAC-2000* front panel via the *Set Menu* system. For instance, in section 4.2 we described the use of an adjustable constant as the process set point.

The behavior of a constant operator (fixed or adjustable) is specified by its 'Front Panel Access' property. (See sections 11.22 and 11.23 for a complete description of the constant operator properties.) The 'Front Panel Access' property has three settings: (i) None; (ii) Operator level; and (iii) Engineer level. When this property is set to 'None', the constant operator output value is fixed. When set to 'Operator level' the constant operator will behave as an adjustable parameter accessible from the *FAC-2000* front panel. And, when the 'Front Panel Access' property is set to 'Engineer level', the constant operator is adjustable but access to it is password protected. This password-protection feature of engineering constants is useful in situations where the control engineer wishes to restrict access to a design constant (*e.g.*, a tuning parameter). To specify a password for access to 'Engineer level' constants, select the Controller Options item from the Schema menu and click the Password Tab (see section 0 for more details).

The *FAC-2000 Set Menu* system is accessed and managed with the SET, UP, and DOWN buttons on the controller front panel. To operate the *Set Menu,* follow these steps:

1) Press the SET button to enter the *Set Menu*.

2) If there is only one 'Operator Level' constant in the schema then go to step 5.

3) The top two lines[1] of the alphanumeric display will present the user with a list of 'Operator Level' parameter names (such as 'PID gain' or 'set point').

4) Use the UP and DOWN arrow buttons to scroll through the list of adjustable parameter names. If you wish to adjust a parameter, press the SET button again—

[1] All user-defined alphanumeric text messages on the top two display lines are paused while the controller is in the *Set Menu* system.

the current parameter value will then be displayed—and go to step 5. Otherwise, if you wish to exit the *Set Menu*, scroll down the list until the 'Press SET to exit set menu' message is displayed. Press the SET button and the device will exit the *Set Menu*.

5) Press the UP arrow button to increase the parameter value; press the down arrow button to decrease the parameter value. If you hold an arrow button down, the parameter value will begin to change at a rapid rate. (To learn how to adjust this rate of change, refer to section 4.2, Modifying Operator Properties.)

6) When the desired value has been reached, press the SET button to record the change. If there is only one adjustable constant operator, the controller will exit the *Set Menu* mode. Otherwise, the controller will remain in *Set Menu* mode and will again display the list of adjustable parameters. That is, the controller will return to step 3.

You may have noticed that the preceding instructions do not mention how to access the password-protected 'Engineer Level' constants. How do you access these constant operators in the *Set Menu*? To access these parameters, enter the *Set Menu* system and push the UP and DOWN arrow buttons simultaneously. The controller will then prompt you for a password—it presents the character 'A' on the top alphanumeric display. You enter the password one character at a time, using the UP or DOWN arrow buttons to scroll through the character set. When the correct character is displayed, press the SET key to record it and move on to the next character. Repeat this process until you enter all six characters. After you enter the last character, the controller will compare the password with the one in its schema file. (This is the password you specified using the Controller Password command in *Design Pad's* Schema menu.) If the password is correct, the 'Engineer Level' constant operators will be accessible. These parameters will continue to be accessible even if you exit the *Set Menu* and later re-enter it. To return the controller into its normal restricted access mode, enter the *Set Menu* and push the UP and DOWN arrow buttons simultaneously.

4.8 The *FAC-2000* A/M Menu System

A typical process controller may operate in automatic mode (where the device computes a controller output) or in manual mode (where a plant operator specifies the controller output). The *FAC-2000* unit is a versatile multi-loop controller that provides this functionality and more. It can be configured to operate in a partially-manual, partially-automatic mode called *mixed* mode. While in mixed mode, a subset (*e.g.,* an individual loop) of the control scheme executed by the device may operate in automatic mode, while another subset operates in manual mode, while yet another subset operates in automatic mode. (For instance, in a three-loop control scheme, the plant operator may maintain two loops in automatic modes and the third in manual mode.)

Design Pad provides four operators with built-in auto/manual functionality: *A/M Button*, *PID Controller*, *PI Controller*, and *PD Controller*. There is no limit on how many of these operators you can use in a schema. The state of each operator is adjusted via the *FAC-2000 A/M Menu* system. This system presents the user with a list of A/M operators; it allows the

user to toggle each operator between automatic and manual modes; and it provides the user with a means to adjust an operator's output signal value when in manual mode.

The *FAC-2000 A/M Menu* system is accessed and managed with the A/M, LEFT, and RIGHT buttons on the controller front panel. To use the *A/M Menu* when your schema contains a single A/M operator, follow these steps:

1) Press the A/M button to enter the *A/M Menu* system.
2) The single A/M operator will switch into manual mode.
3) Edit the manual output by pressing the LEFT and RIGHT arrow buttons. If an arrow button is held down, the output value will begin to change at a rapid rate. (To learn how to adjust this rate, refer to section 4.2, Modifying Operator Properties.)
4) To return the system to automatic mode, press the A/M button again. The *FAC-2000* will exit the A/M Menu system.

Typically, the *FAC-2000* controller shows the A/M state on the top alphanumeric display and the manual output value (if applicable) on the middle alphanumeric display. (To learn how to override this default behavior, refer to section 4.2, Modifying Operator Properties.) However, when the controller is in *Set Menu* mode, both the A/M state and the manual output value (if applicable) are indicated on the bottom alphanumeric display.

To use the *A/M Menu* when your schema contains multiple A/M operators, follow these steps:

1) Press the A/M button to enter the *A/M Menu* system.
2) The top two[2] alphanumeric displays will present the user with a list of A/M operator names (such as 'water level' or 'temperature'). Use the LEFT and RIGHT arrow buttons to scroll through the list until the controller displays the operator you wish to change.
3) If you wish to exit the *A/M Menu* system, scroll through the list until the 'Press A/M to exit A/M menu' message is displayed, and press the A/M button.
4) To switch a selected A/M operator from automatic mode to manual mode, press the A/M button and go to step 7.
5) To switch a selected A/M operator from manual mode to automatic mode, press and hold the A/M button for more than two seconds and go to step 2.
6) To adjust the manual output of a selected A/M operator that is already in manual mode, press and hold the A/M button for less then two seconds and go to step 7.
7) Adjust the manual output by pressing the LEFT and RIGHT arrow buttons. If an arrow button is held down, the parameter value will begin to change at a rapid rate

[2] All user-defined alphanumeric text messages on the top two display lines are paused while the controller is in the auto-manual menu system.

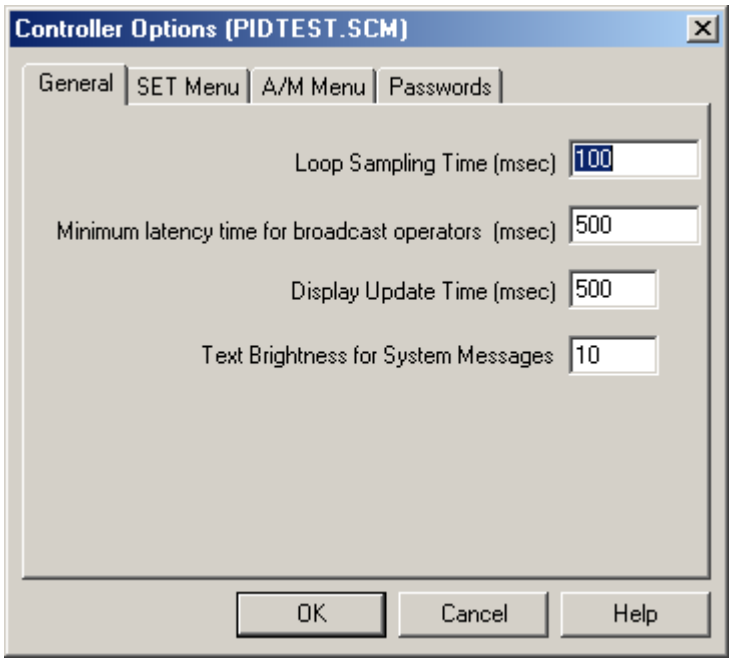


Figure 10. Controller Options dialog.

(To learn how to adjust this rate, refer to section 4.2, Modifying Operator Properties.)

8) When the desired value has been reached, press the A/M button to record the change. Note that the A/M circuit will remain in manual mode. Go to step 3.

If the control schema has multiple A/M operators, you may not access the *Set Menu* and *A/M Menu* simultaneously. You must exit out of one menu before you can enter the other.

4.9 Modifying Controller Properties

You have now nearly completed the process of programming the *FAC-2000* controller. You have created a schema diagram that resembles the schema of Figure 1 and have adjusted various properties of several operator blocks. In this section, we consider properties that determine how the *FAC-2000* controller should operate while running your schema. These properties include the schema execution frequency, the behavior of faceplate keys, the presentation of adjustable constants and A/M operators, and password protection for engineering constants.

The controller properties are set in the Controller Options dialog, reproduced here in Figure 10. To display this dialog, select the Controller Options item from the Schema menu. The following parameters are specified in the General Tab of the Controller Options menu.

Loop sampling time: This parameter specifies the time in milliseconds between consecutive iterations of the schema. This parameter, labeled Δt, is often cited in

the operator reference of this instruction bulletin. The minimum loop sampling time is 10 milliseconds; the maximum loop time is 28,800,000 milliseconds (8 hours).

Latency time for non-critical broadcast operators: This parameter specifies the time in milliseconds between network broadcasts (for non-critical broadcast messages). The default broadcast latency time is 500 milliseconds. With the broadcast latency time set to the default value, all non-critical broadcast operators that are part of this schema will transmit their state (signal value) over the network every 500 milliseconds.

Display update time: This parameter controls the refresh rate for the numeric and bargraph displays on the *FAC-2000* faceplate. It corresponds to the amount of time (in milliseconds) between display updates. The minimum update time is 100 milliseconds; the maximum update time is 28,800,000 milliseconds (8 hours).

Text brightness for system messages: The alphanumeric displays have a brightness setting between 0-15. This parameter controls the brightness setting for system messages. System messages include the *Set Menu* information, *A/M Menu* information, and any controller error conditions that may arise.

The following parameters are available in the SET MENU tab of the Controller Options dialog:

SET activation time: When the SET button is pressed (and held down for the *SET activation time*) on the *FAC-2000* faceplate, the alphanumeric displays present a list of constants that can be adjusted using the up/down arrow keys. This *SET activation time* parameter is the amount of time (in milliseconds) the SET button must be held down before the controller enters the *Set Menu*. A long activation time may lower the risk of unauthorized *Set Menu* access by personnel that are experimenting with the device—pushing buttons to "see what happens".

SET operation timeout: When the SET button is pressed on the *FAC-2000* faceplate, the alphanumeric displays present a list of constants that can be adjusted using the up/down arrow keys. After the constant values are adjusted, the user should press the SET button again to exit the *Set Menu*. If the operator leaves the unit without exiting SET mode, the *FAC-2000* will automatically exit *Set Mode* after this *SET operation timeout* time elapses. This *timeout* parameter is expressed in seconds. (The *FAC-2000* controller measures the elapsed time between keypad input. If the device is in *Set Mode* and the time of keypad inactivity exceeds this *timeout* value, the controller will automatically exit *Set Mode*.)

SET delay before key repeat: This parameter determines the behavior of the UP/DOWN arrow keys on the *FAC-2000* front panel. The UP/DOWN arrow keys are used to adjust a parameter value in SET mode—when a key is pressed, the parameter is increased or decreased by a pre-specified increment. If the arrow key is held down, the parameter value will be continually adjusted. This property specifies the time (in milliseconds) before automatic repeated increments begin.

SET key repeat time: As explained above, a parameter value will be continually changed (by a pre-specified increment) if the UP or DOWN arrow key is held down. This parameter specifies the time between parameter increments.

Adjustable constant ordering: The *Analog Constant* and *Digital Constant* operators have a property called 'Front Panel Access'. When this property is set to either 'Operator level' or 'Engineer level', the constant value can be adjusted from the *FAC-2000* faceplate, using the SET and arrow keys. When the SET key is pressed, a list of adjustable constants will be presented on the alphanumeric displays. ('Engineer level' constants are password-protected—they are not accessible until the correct password is entered.) This property specifies the ordering of this list. When the Controller Options dialog is first invoked, the eligible constants are placed in the list-boxes with a default ordering. To rearrange the list, simply click on the constant name and drag it to the desired location in the list.

The following parameters are available in the A/M MENU tab of the Controller Options dialog:

A/M delay before key repeat: This parameter determines the behavior of the LEFT/RIGHT arrow keys on the *FAC-2000* front panel. The LEFT/RIGHT arrow keys are used to adjust the manual output value of an A/M operator or Controller block. When a LEFT/RIGHT key is pressed, the parameter is increased or decreased by a pre-specified increment. If the arrow key is held down, the parameter value will be continually adjusted. This property specifies the time (in milliseconds) before automatic repeated increments begin.

A/M key repeat time: As explained above, a manual output value will be continually changed (by a pre-specified increment) if the LEFT or RIGHT arrow key is held down. This parameter specifies the time between output increments.

A/M state in alphanumeric display: If this property is checked, the top-line of the alphanumeric displays will indicate if the controller is functioning in automatic, manual, or mixed mode. Otherwise, if this property is not checked, the *FAC-2000* controller will not display the operational mode. An alternative method to indicate the Auto/Manual state is to force the bargraph and/or numeric displays to flash on and off when the controller is in manual mode. (To implement flashing displays, connect the A/M signal—from a *PID* or *A/M Button* operator—to the boolean input of the *Bargraph Display* and/or *Numeric Display* operators. See the operator reference section for more details on these operators.)

A/M operator ordering: The *PID Controller, PI Controller, PD Controller* and *A/M Button* operators may function in automatic or manual modes. In general, a schema will contain only one of these objects. However, complex control schemes may require more than one. In the single A/M operator case, pressing the A/M key on the *FAC-2000* front panel will toggle between automatic and manual modes. In the multiple A/M operator case, a list of A/M objects will be presented on the alphanumeric displays when the A/M key is pressed. (The arrow keys are then used to select the desired A/M object.) This property specifies the ordering of this list.

When the Controller Options dialog is first invoked, the eligible A/M operators are placed in the listbox with a default ordering. To rearrange the list, simply click on the operator name and drag to the desired location.

The following parameters are available in the Passwords tab of the Controller Options dialog:

Engineering Password. The engineering password restricts access to the 'Engineering Level' constants on the SET MENU. The password must have six alpha-numeric characters. You must enter the password two times in order to prevent inadvertent typing errors (once in the "Enter engineering password" edit box and a second time in the "Re-enter engineering password" edit box).

Communication Password. The communication password restricts access to some of the *FAC-2000* communication features, namely importing schemas from a controller to a PC, erasing a controller's schema, and replacing a controller's schema with another schema. You choose the communication functions that are password protected by selecting the appropriate check box. For instance, if you provide a password and select the "importing controller schema" check box, *Design Pad* will prompt you for the communication password when you attempt to import that schema into *Design Pad* from a controller. The communication password must have six alpha-numeric characters. You must enter the password two times in order to prevent inadvertent typing errors (once in the "Enter communication password" edit box and a second time in the "Re-enter communication password" edit box).

4.10 Schema Properties

The last step in the design process is to record general information about the schema, such as a title, the designer's name (author), the revision number, and notes about the design. This information is entered in the schema properties dialog box shown in Figure 11. (To view this dialog box, select the Schema Properties item from the Schema menu.) *Design Pad* automatically records the date the schema was first created, the date it was last modified, and the date it was exported into the controller. It also records the *Design Pad* version used to create the schema.

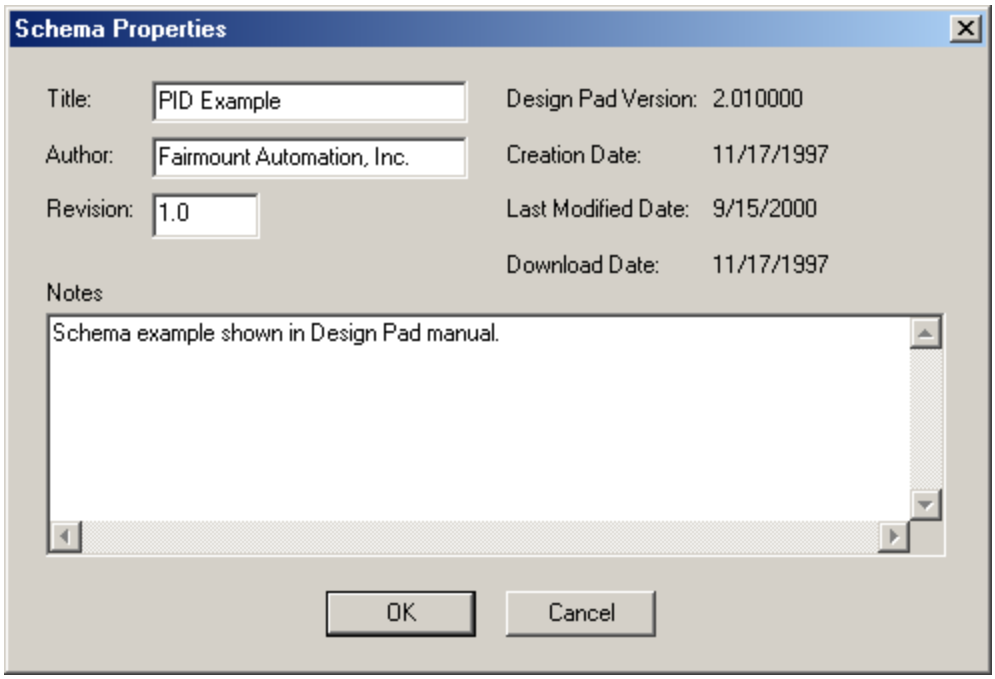


Figure 11. The schema properties dialog box.

4.11 Schema Compiling

A schema diagram is a representation of a control algorithm to be executed by the *FAC-2000* controller. In order for the algorithm to be executed, it must be translated or compiled into a form that the central processing unit (CPU) of the device can understand. *Design Pad* has a built-in compiler that does just that.

Process Schema

To compile a schema diagram, select the Process Schema item in the Schema menu. Alternatively, you can press the corresponding button on the speed-bar menu (shown to the right.) *Design Pad* will then convert your schema into executable form.

During the compilation process, *Design Pad* checks for errors or anomalies in your schema. For instance, *Design Pad* will alert you if you have left an input pin disconnected or if you have combined an analog signal with a digital one. All of the errors and warnings that *Design Pad* may issue during compilation are described in the following section.

View Message Window

Design Pad displays compilation errors and warnings in its Message window. Each line in the message window refers to an individual error or warning. When you select an error or warning in the message window, *Design Pad* will highlight the corresponding offending object(s) and/or signal(s). *Design Pad* will also re-center the schema so that the highlighted object(s) appear at the center of the document window. You toggle the visible state of the message window (shown or hidden) from the view menu. When you make the message window visible, *Design Pad* displays a check mark next to the Message Window item in the View menu; it removes the check mark when the window is not visible. You can also toggle

the visible state of the message window by pressing the corresponding button on the toolbar menu (shown to the right).

The message window is always displayed on top of other windows in the *Design Pad* application environment. By default, it is docked on the bottom-left side of the screen. But it can be positioned (and resized) anywhere on the screen—it can be docked at the top, bottom, left side or right side of the client area or it can be floating anywhere on your computer screen (even outside of the application's client message). If the message window is docked and you wish to position it elsewhere, click on the window handle (two raised lines adjacent to the close button) and drag it to the desired location. To resize the message window while it is docked, click the right mouse button on the inner edge of the window and drag it to the desired location. When the message window is floating, you can resize it as you would any other window.

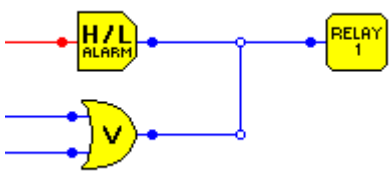
4.12 Schema Errors and Warnings

While processing a schema diagram, *Design Pad* checks for error conditions. The errors and warnings that *Design Pad* may issue are described below.

Error: Signal wire contains loop. Signal wires should be used to connect operator I/O pins. This error is issued if a schema signal is connected to itself.

Error: Object has one or more unconnected input pins. Certain operators may only be processed if all of their input pins are connected. For instance, the *PID Controller* operator must have all of its five inputs connected in order to function properly. This error is issued if an input pin that must have a connection is left unconnected. (This error is not issued if the object is not used at all. That is, if none of its I/O pins are connected.)

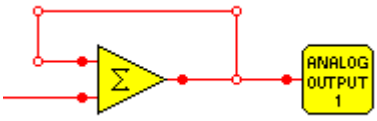
Error: Signal is connected to multiple output pins. A signal wire represents data flow from one operator to another—the output signal of one operator being fed to the input of another. The same output signal can be fed to multiple input pins. However, multiple outputs may not be combined—they may not feed the same input. The partial schema shown below exhibits this error condition. The outputs of the *High/Low Alarm* operator and of the *Logical OR* operator are both fed to Relay 1. Since *Design Pad* cannot discern which output should drive the relay, it issues an error.



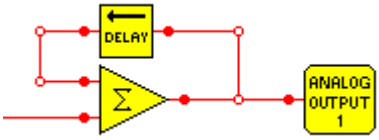
Error: Signal connects input pins only. A signal wire represents data flow from one operator to another—the output signal of one operator being fed to the input of another. An input pin of one operator may not feed the input of another operator. This error is issued when an output pin is not part of a wired connection.

Error: Cannot combine an analog signal with a digital (boolean) signal. *Design Pad* supports two signal types: analog signal (floating point values) drawn in red and digital signals (0 or 1) drawn in blue. *Design Pad* issues an error if the signal types are combined. (*Design Pad* provides operators to convert from one signal type to another. See the operator reference section of this instruction bulletin for more details.)

Error: Objects form an algebraic loop (use delay elements to implement feedback). Algebraic loops are circular references that cannot be resolved by the *Design Pad* compiler. For instance, consider the schema shown below, where the output of an *Addition* operator is fed back to one of its inputs:



The operator output at time *t* is equal to the sum of its inputs, $y(t) = x_1(t) + x_2(t)$. And, we have $x_1(t) = y(t)$, so the diagram above attempts to compute $y(t) = y(t) + x_2(t)$. But, the computation of the output value cannot be a function of its present value, resulting in an error condition. The correct way to implement feedback is to include a *Delay* operator in the feedback path, as shown below.



With this arrangement, the output value is a function of its previous value, $y(t) = y(t - \Delta t) + x_2(t)$.

Error: Schema contains an RS-232/RS-485 receiver operator but is not assigned to an RS-232/RS-485 network. Receiver operators capture signal broadcasts from other controller stations on a network. When a schema contains a receiver operator it must be assigned to a network. (If it contains an RS-485 receiver, it must be assigned to an RS-485 network; if it contains an RS-232 receiver, it must be assigned to an RS-232 network.) Refer to section 6 for a detailed description of controller networking (*e.g.*, creating a network, assigning a network to a schema, *etc.*).

Error: Receiver references a broadcast signal that does not exist. Receiver operators capture signal broadcasts from other controller stations on a network. To specify which broadcast signal the receiver is to capture, you must specify the name of the broadcast signal in the properties dialog box of the receiver operator—the signal name the receiver is to capture must match the broadcast signal name exactly (signal names are case-sensitive). *Design Pad* issues this error when the receiver's signal name does not match any broadcast signal on the network. (If the operator is an RS-232 receiver, the schema must be assigned to an RS-232 network and another schema assigned to that network must have a matching broadcast operator. If the operator is an RS-485 receiver, the schema must be assigned to an RS-485 network and another schema assigned to that network must have a matching

broadcast operator.) Refer to section 6 for a detailed description of controller networking (*e.g.*, creating a network, assigning a network to a schema, *etc.*).

Error: Network contains two or more broadcast operators with duplicate signal names. All broadcast operators within a communications network must have unique signal names. This error is issued if the schemas assigned to a network contain two or more broadcast operators with the same signal name. Signal names are case-sensitive, meaning that *Design Pad* considers *OilPressure*, *oilPressure*, and *OILPRESSURE* to be unique signal names.

Error: Receiver references a broadcast operator with a different signal type (analog/digital/mixed signal mismatch). Receiver operators capture signal broadcasts from other controller stations on a network. Analog receivers capture analog broadcasts; digital receiver capture digital broadcasts; and mixed (both analog and digital) receivers capture mixed broadcasts. *Design Pad* will issue this error if a receiver/broadcast pair is not the same signal type.

Warning: Object is not used. Objects that have no I/O pin connections have no effect during schema program execution. *Design Pad* issues a warning if a schema contains any unused objects.

Warning: Object has one or more unconnected input pins. Certain operators can be processed even if one or more of their input pins are not connected. For instance, the digital input of a *Bargraph Display* operator need not be connected for a signal to be displayed on a *FAC-2000* LED bargraph display. *Design Pad* issues a warning if a pin is not connected, in case the open connection was unintended. (This warning is not issued if the object is not used at all. That is, if none of its I/O pins are connected.)

Warning: Object has an unconnected output pin. In general, operator objects generate an output signal based on their inputs. *Design Pad* will issue a warning if an object has an unconnected output pin and one or more of its input pins are connected. While this condition does not represent an error—the schema can still be executed—it may not have been intended by the schema designer.

Warning: Signal wire does not connect output pin to input pin. Signal wires should connect the output pin of one operator to the input pin of another. Otherwise, the wire has no effect. *Design Pad* issues a warning if a signal wire does not make an input/output connection.

Warning: Schema contains an RS-232/RS-485 broadcast operator but is not assigned to an RS-232/RS-485 network. Broadcast operators transmit signals to other controller stations on a network. When a schema contains a broadcast operator, the operator will not be active (will not broadcast) unless the schema is assigned to a network. (If it contains an RS-485 broadcaster, it should be assigned to an RS-485 network; if it contains an RS-232 broadcaster, it should be assigned to an RS-232 network.) Refer to section 6 for a detailed description of controller networking (*e.g.*, creating a network, assigning a network to a schema, *etc.*).

4.13 Protecting your Work

Design Pad 2000 offers a mechanism for password protecting documents to prevent unauthorized access to your intellectual work. To protect a document, select the Password Lock item from Schema menu. *Design Pad* will display the Password Lock dialog shown in Figure 12. The dialog box contains three text boxes: you enter the old password (if applicable) in the first, and the new password in the second two. If the document has not been protected yet, leave the "Old Password" edit box blank. Otherwise, you need to provide the old password before *Design Pad* will accept a new one. In either case, you must enter the new password twice in order to lower the probability that a typing error occurred.

The Password Lock dialog box (Figure 12) also allows you to specify the access privileges while a document is locked. At minimum, unauthorized users will be unable to view or edit a locked document. But you can also prevent unauthorized users from running a simulation of the locked document, or exporting the locked document into a controller (by selecting the appropriate check boxes in Figure 12).

When *Design Pad* opens a password-protected file, it displays a blank document and disables all editing functions (Figure 13 shows a locked schema document). In order to unlock the document, a user will have to click on the document window. *Design Pad* will then prompt the user to enter a password and will unlock the document if the correct one is provided.

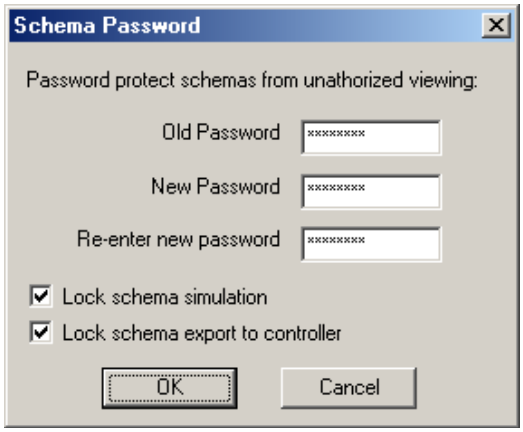


Figure 12. The Password Lock dialog.

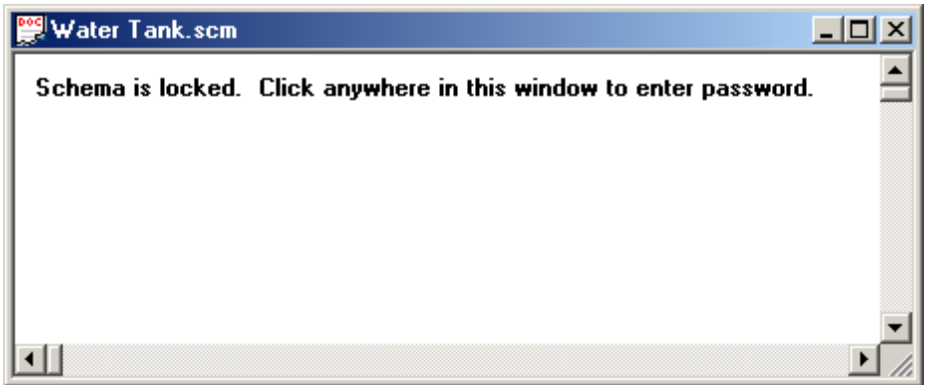


Figure 13. When Design Pad opens a locked schema document it does not display its contents.

While a document is locked, you may not add operators, generators, or text labels. And you may not view or edit operator properties, schema properties, or controller options. However, if the locked document allows simulation execution, a user that has not provided a password will still be able to run a simulation of the schema and interact with the virtual front panel and rear panels (section 8 describes *Design Pad* simulations). And, if the locked document allows exporting to a control station, a user could still program a *FAC-2000* controller even if they lacked the password to unlock the schema.

Enabling simulations in locked documents may be useful if the schema is used for training or as part of a human-machine interface (see section 9).

Enabling schema exporting in locked documents may be useful if you would like field personnel to be able to program a *FAC-2000* controller but would prefer that they not be able to view the actual program "code".

If you wish to remove password protection from a schema that is protected, select the Password Lock item from the Schema menu. Then, enter the old password in the appropriate edit box and leave the other two edit boxes blank (see Figure 13Figure 12).

5. Managing Schemas and Networks in a Workspace

Prior versions of *Design Pad* focused on the design of a schema for a single control station. *Design Pad 2000* is also intended for the design of distributed control strategies requiring multiple controller stations to regulate many inter-related processes. *Design Pad 2000* uses *workspaces* to organize the collection of schemas that form a multi-station control strategy. But workspaces can be used to organize any collection of schemas—even schemas that are loosely related or not related at all.

In addition to organizing schemas, workspaces provide a means to configure communication networks, to configure an OPC server node, to create human-machine interfaces (HMIs), and to access (communicate with) stations on a controller network. (An OPC server is used to link a controller network to a PC network.)

This section describes schema management in workspaces. Section 6 describes network configuration and OPC server configuration, and section 7 describes PC-to-controller communication via *Design Pad*.

5.1 The Workspace Window

View Workspace Window

The workspace document exists within the workspace window—a window normally docked on the left side of the application's client area (see Figure 14). To view the workspace window, select the Workspace item from the View menu. It may sometimes be useful to hide the workspace window to provide more room on the screen to view a schema document. To hide the workspace window, select the Workspace item from the View menu (or click on the window's close button). Notice that the Workspace item on the View menu toggles the visible state of the workspace window (when the window is visible, the menu item is checked, otherwise it is unchecked). The visible state of the workspace window can also be toggled with the toolbar menu button shown to the left.

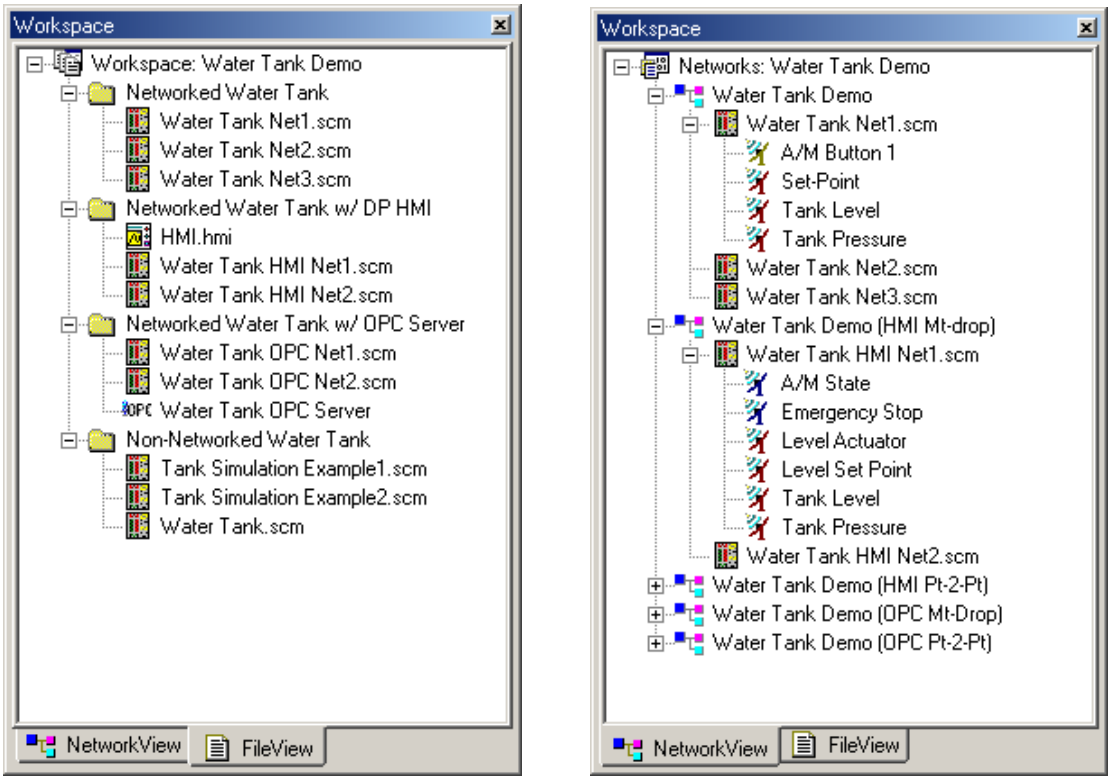


Figure 14. The workspace window, including (a) the *File-View* tab and (b) the *Network-View* tab.

Note: Hiding the workspace window does not close a *workspace* document—a *workspace* is still active while the workspace window is hidden.

The workspace window is always displayed on top of other windows in the *Design Pad* application. By default, it is docked on the left side of the screen. But it can be positioned (and resized) anywhere on the screen—it can be docked at the top, bottom, left side or right side of the client area or it can be floating anywhere on your computer screen (even outside of the application's client area). If the workspace window is docked and you wish to position it elsewhere, click on the window handle (two raised lines adjacent to the close button) and drag it to the desired location. To resize the workspace window while it is docked, click the right mouse button on the inner edge of the window and drag it to the desired location. When the workspace window is floating, you can resize it as you would any other window.

The workspace window is used to manage a collection of controller schemas and controller communication networks. The workspace window has two sections: the *Files* section and the *Networks* section. A sample workspace window is shown in Figure 14—Figure 14a displays the workspace with the *Files* section active, and Figure 14b displays the workspace with the *Networks* section active. The files section lists all the schema documents, HMI documents, and OPC server configuration files that are part of the *workspace*. To activate the files section, click on the *Files* tab. The networks section of the workspace window lists

all the communication networks that the workspace manages. To activate the networks section, click on the *Networks* tab.

5.2 Organizing Documents in the Workspace

The sample workspace shown in Figure 15 organizes schema documents for control stations regulating several processes in multiple plants of the same facility (in this case, the steam propulsion plants in an aircraft carrier, the USS JOHN F. KENNEDY). The workspace—named CV-67—maintains all the schemas for the eight boilers (four spaces) that power the ship. It uses folders to organize the schemas into meaningful groups—much like an operating system uses folders to group related files. For instance, the sample workspace of Figure 15 contains four top-level folders, one for each main machinery space in the ship (named *MMR1*, *MMR2*, *MMR3*, and *MMR4*). Each top-level folder contains two sub-folders named *Control Room*, and *Feed Pumps*. The *Feed Pumps* sub-folder contains the three schemas for the three control stations that regulate the feed pumps in the machinery space. The *Control Room* sub-folder contains the schemas for the control stations housed in the main machinery control room. It also contains two sub-folders, one for each boiler in the space.

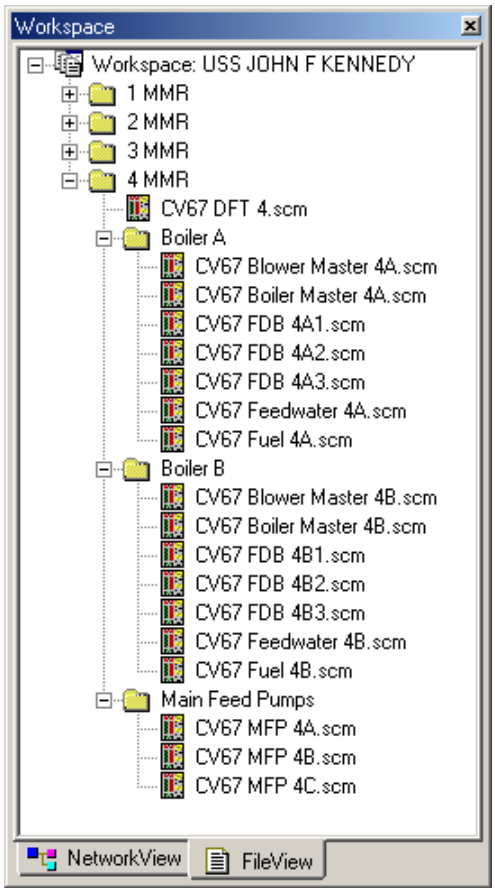


Figure 15. A sample workspace.

Clearly, the grouping of schemas within a workspace is arbitrary. In the sample workspace, we could have chosen to eliminate all sub-folders, and maintained only the top-level folders corresponding to each main machinery room. For that matter, we could have eliminated using folders altogether and maintained all the schemas at the workspace root. Another approach would be to maintain four separate workspace documents—one for each machinery space. Or we could have combined the sample workspace with workspaces for other steam-powered aircraft carriers, or with workspace for controls systems on all surface ships.

The benefit of maintaining several systems within the same workspace is that you can easily access all the schema files in the workspace. To open a schema file, you simply double-click on the schema name in the workspace window. You don't need to search your drive directories for a schema file—the workspace knows where each file is. The drawback of maintaining a workspace with many schemas is that it may become needlessly complex.

5.3 Working with Workspaces

5.3.1 Creating a new Workspace

To create a new workspace, select the New Workspace item from the File menu. *Design Pad* prompts you for the workspace name and the directory where you would like to store the workspace document. The new workspace document will be saved with a `.dpw` file extension. *Design Pad* will also create a corresponding workspace environment file—with a `.wen` file extension. It will use the environment file to restore the *Design Pad* state the next time you use the workspace. Among other things, it will use the environment file to automatically open any documents that are open when you close the workspace.

5.3.2 Opening and Closing Workspaces

Design Pad can only have a single workspace document open at any given time. If you would like to open a workspace document, you must first close the workspace document you are presently working with. To close a workspace, select the Close Workspace item from the File menu. If necessary, *Design Pad* will prompt you to save changes to the workspace before closing it. Occasionally, you may be surprised that *Design Pad* prompts you to save changes to the workspace. While you may not have made any changes to the workspace explicitly, any changes you made to the documents referenced in the workspace may alter its own settings. In general, when *Design Pad* prompts you to save changes to the workspace, you should respond affirmatively, unless you are certain that you want to discard your recent work.

When *Design Pad* closes a workspace, it also closes all other open documents and their associated windows. If necessary, *Design Pad* will prompt you to save any changes you may have made to a document before closing it.

To open an existing workspace, select the Open Workspace item from the File menu. *Design Pad* will provide you with the familiar Windows Open File dialog where you can browse for the desired workspace. If a workspace is open when you attempt to open another one, *Design Pad* will first close the open workspace (and, if necessary, prompt you to save any changes) and then prompt for the new workspace to open.

Design Pad maintains a list of recently opened workspace documents in the Recent Workspaces item of the File menu. To open a recently-accessed workspace document, select the Recent Workspaces item in the File menu. A sub-menu will appear listing the four most recently opened workspace documents. If the workspace you wish to open is on the list, select it. (You can change the number of entries in the recent-file list—see section 10.1.)

5.3.3 Saving the Workspace

As you modify the workspace, you should periodically save your changes to permanent storage. To save the workspace, select the Save Workspace item in the File menu. When you save the workspace, *Design Pad* will only record any changes made to the workspace itself; it will not save any modifications you may have made to other documents (schema files, HMI files, *etc.*).

Design Pad 2000 automatically creates a back-up copy of a workspace document before it saves any changes. For example, when you save changes to the workspace named `workspace.dpw`, *Design Pad* first makes a copy of the existing file (and saves the copy as `workspace.bak`) before overwriting `workspace.dpw` with the changes that you made to it. This feature could be useful if you have saved changes to a workspace but later decide you want to discard those changes and revert to the back-up version. To disable the auto-backup feature, see section 10.2.

5.3.4 Copying a Workspace Document

The workspace (`.dpw`) file records links to the documents it contains; it does not store the actual documents. The documents it references are stored separately, as individual document files. Therefore, if you would like to share your work with others (*e.g.*, via e-mail or disk copies), you need to provide them with the workspace (`.dpw`) file as well as all the schema (`.scm`) documents and HMI (`.hmi`) documents that the workspace references.

If you would like to copy the workspace and all related documents to a new location (*e.g.*, a floppy disk), you can do it manually, or you can select the Copy Workspace As item from the File menu. *Design Pad* will automatically copy all the necessary files.

The document(s) that a workspace file references need not be located in the same disk directory (as the workspace file). The workspace file records the absolute path between its location and the location of the documents it references. When you execute the Copy Workspace As command, *Design Pad* can automatically create the same relative directory structure in the target location.

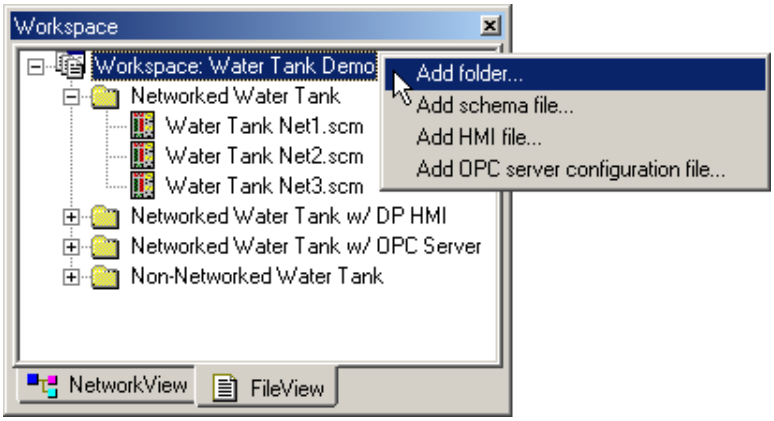
5.4 Workspace File View

The workspace window has two views: a *File View* (Figure 14a) and a *Network View* (Figure 14b). The file view displays all the schemas, HMI documents, and OPC server configuration files that are referenced in the workspace. It organizes these documents in a graphical hierarchical tree format, using folders to group related files. The file view displays the name of each document alongside a small graphic that indicates the type of document it is (for a schema file, for an HMI file, or for an OPC-server configuration file).

Once you have created a workspace document (section 5.3.1), you can begin adding schema documents and folders to the workspace.

5.4.1 Using Workspace Folders

Workspace folders serve to group related documents together, much like directories serve to group related files on a disk drive.

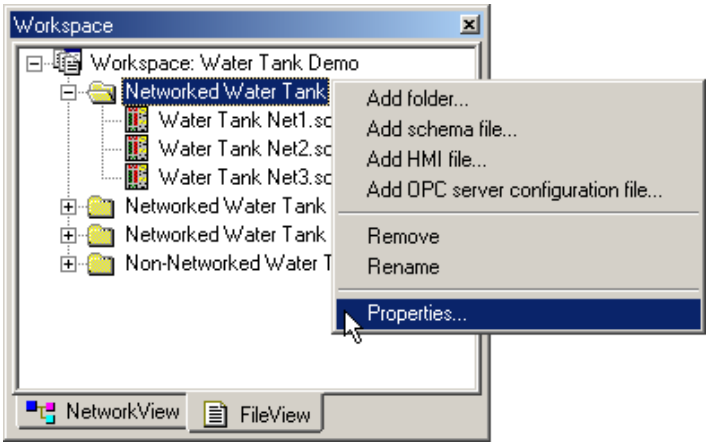


To add a folder to the workspace, first make the *File View* active (click on the *Files* tab in the workspace window). Next, click the right mouse button on the workspace label (the root node) and select the Add Folder item from the drop-down menu. Then, after *Design Pad* inserts a new folder into the workspace, right-click the folder, select the Rename item from the drop-down menu, and type in the folder name.

You can also add a folder to another folder (*i.e.,* as a sub-folder). To do so, right-click on the folder name and select the Add Folder item from the drop-down menu. As before, you need to rename the folder after *Design Pad* inserts it into the workspace—use the Rename item from the folder drop-down menu.

After you have added folders to the workspace, you can re-arrange the workspace folder structure using drag and drop. That is, you can select a folder, drag it to a new location, and release it there. For instance, you can move a sub-folder to the workspace root node, or you can move a folder into another folder. Of course, when you drag a folder to a new location, the folder contents move with it.

You can remove a folder from the workspace altogether by right-clicking on the folder and selecting the Remove item from the drop-down menu. When you remove a folder from the workspace, the folder and all of its contents (including sub-folders and their contents) are also removed from the workspace. To ensure that you did not inadvertently choose to remove a folder, *Design Pad* will prompt you to confirm you selection before proceeding.



You can add information about a folder in the Folder Properties dialog box. To view the folder properties, right-click on the folder and select Folder Properties item from the drop-down menu. The Folder Properties dialog box (Figure 16) displays the folder name along with notes about the folder. You can use the *Notes* field to add information about the folder contents. For example, you could explain the relationships between the schema documents in the folder (*e.g.*, how they work together, *etc.*)

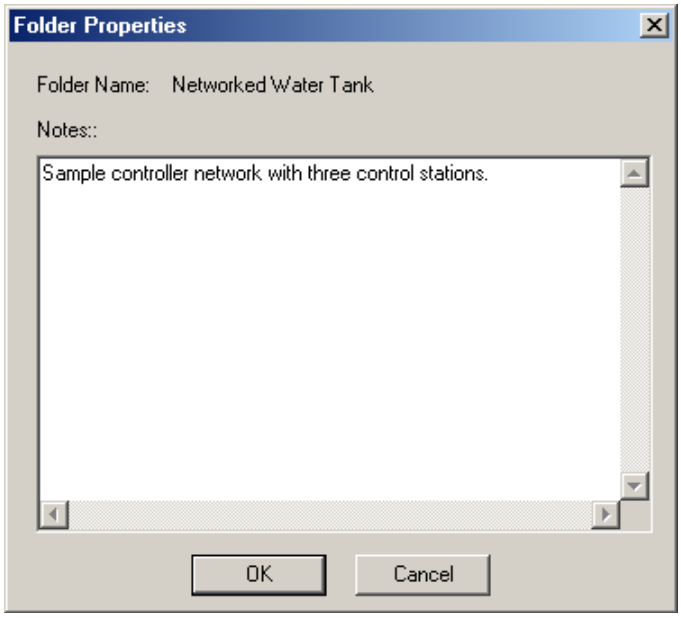


Figure 16. Folder properties dialog used to record notes about the contents of the folder.

5.4.2 Adding Documents to the Workspace

In *Design Pad 2000*, you can add three types of document files to the workspace: schema files, human-machine interface (HMI) files, and OPC-server configuration files: Schema

files were described in section 4; they represent the control algorithm that a *FAC-2000* control station is to execute. HMI files are synthesized in the same manner that schema files are created (see section 9.1). But HMI files offer additional graphical user-interface (GUI) resources to convey control system operation on a PC platform. They are not intended for use in a *FAC-2000* controller. OPC server configuration files (described in section 9.2) contain all the requisite information to establish a gateway between a *FAC-2000* controller network and a third-party PC-based HMI package (*e.g.*, Wonderware, ICAS, *etc.*).

To begin adding documents to the workspace, you must first activate the *File View* tab on the workspace window. Then, right-click on the root workspace node (or a folder in the workspace) and select the appropriate item from the drop down menu. If you choose the Add Schema Files item or Add HMI Files item, *Design Pad* will display the familiar Windows Open File dialog, where you can browse for and select the document file(s) you wish to add to the workspace. If you choose the Add OPC-Server Configuration File item, *Design Pad* will insert a blank OPC server item into the workspace with a default name. To change the default name, right-click on the OPC server item, select Rename from the drop-down menu, and type a new name.

Note: When adding schema files or HMI files, you can select multiple documents in the Windows Open File dialog using the SHIFT and CTRL keys. All the selected documents will be added to the workspace root or workspace folder.

Note: Use drag-and-drop if you need to re-organize the workspace documents (*e.g.*, move a document from one workspace folder to another).

Schema (`.scm`) files and HMI (`.hmi`) files referenced in the workspace do not need to reside in the same disk directory as the workspace (`.dpw`). They can be located in different directories, in different drives, and/or in different computers on a local area network. The workspace file records a link to each document referenced in the workspace, rather than incorporating the document files within the workspace file. Since only a link is recorded, a problem could arise if the document referenced in the workspace no longer exists or has been moved to another location. If *Design Pad* cannot locate a file referenced in the workspace, it will prompt you for its new location. (You can remove a document from the workspace by right-clicking the document name in the workspace window and selecting the Remove item from the drop-down menu.)

The workspace *File View* offers a convenient location to access schema files and HMI files. You can open a document referenced in the workspace simply by double-clicking on the document name in the workspace window (or you can right-click on the workspace item and select Open File from the drop down menu). You can also perform a number of functions with the workspace documents without opening them at all. For instance, you can process (compile) a schema or export a schema to a control station. The available commands are displayed in a drop-down menu when you right-click on the document name.

5.5 Workspace Network View

The *Network View* displays all the communication networks that are part of the workspace (see Figure 14b). It organizes the communication networks in a graphical hierarchical tree format. The workspace is the root node of this tree and the networks it contains are its first-level sub-nodes. Each network can have sub-nodes representing the schema files for the control stations on that network or the HMI files and OPC server files for PC stations on that network. Each control station or PC on the network in turn can have sub-nodes representing the signals that the station is to broadcast on the associated network.

Each item in the *Network View* tree has descriptive text alongside a small graphic that indicates the type of entry it is. For network items, the tree displays the network name next to a graphic. For schema items, the tree displays the schema file name next to a graphic. For HMI items, the tree displays the HMI file next to a graphic. For OPC-server items, the tree displays the name of the configuration file next to a graphic. And for broadcast signal items, the tree displays the name of the signal next to a graphic (for analog broadcasts, including network-enabled *Constant* operators), graphic (for digital broadcasts), graphic (for mixed broadcasts, including network-enabled A/M Button operators).

The next section in this bulletin describes how to add and configure networks to the workspace and how to associate schemas, HMI documents, and OPC servers with each network.

6. Networking *FAC-2000* Controllers with *FAIRNET*

Design Pad 2000 was developed to support the networking capabilities of Fairmount Automation's *FAC-2000* multi-loop control stations. *FAC-2000* controllers can share process information over a communication network using the *FAIRNET* protocol.

Each control station on a network executes a *schema* created in *Design Pad*. *FAC-2000* controllers share information with each other by broadcasting (and receiving) signals over the network. A schema signal in one controller is broadcasted over the network and processed by schema receivers in other control stations connected to the network.

The *FAC-2000* controller has two communication channels: one is accessible over a RS-232 port and the other over a RS-485 port. The RS-232 channel supports point-to-point networking, where two devices communicate with each other (generally used to link a controller to a PC). The RS-485 channel supports multi-drop networking, where multiple controllers (and PCs) can communicate with one another. Since the *FAC-2000* has two communication channels, it can be attached to two distinct communication networks.

In a *FAIRNET* network, one controller serves as the master station, coordinating all network traffic. All other stations on the network are slaves, responding to information requests from the master station. If the master station goes offline, one of the remaining slave stations on the network will take on the master role.

6.1 Networking Operators

This section describes *Design Pad's* new networking operators that enable *FAC-2000* controllers to communicate with one another (and with personal computers). The new operators fall into two main classes: broadcasters and receivers. Broadcast operators transmit their information (signal values) over the network and receiver operators capture this information.

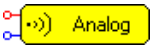
Design Pad includes broadcast (and receiver) operators to transmit (and receive) analog, digital, and mixed (combination analog/digital) signals. Analog broadcast/receiver operators use 16 bits to encode signal values for transmission over the network. Digital broadcast/receiver operators need only a single bit to encode their state (ON or OFF). Mixed broadcast/receiver operators combine an analog signal and a digital signal by using 15 bits to encode the analog signal and 1 bit for the digital signal.

In *Design Pad 2000,* constant operators and A/M button operators can also be configured to broadcast their information over the network. Three new operators—remote analog constant, remote digital constant, and remote A/M button—are the complementing receiver operators for the analog constant, digital constant, and A/M button broadcast-enabled operators. A broadcast-enabled constant paired with remote constant receivers, allow the same parameter value to be adjusted from the faceplate of multiple controllers. Similarly, a broadcast-enabled A/M button paired with remote A/M button receivers, allow the auto/manual state

(and manual output value when in manual mode) to be adjusted from the faceplate of multiple controllers. These same operators enable the adjustment of a controller's parameter value (*e.g.*, a process set-point) and the switching of a controller's automatic and manual operating modes from a remote computer.

6.1.1 Broadcast Operators

Broadcast operators encode their input signal(s) into a communication message that they broadcast over a controller network. All controllers that are part of the network will receive the broadcasted message. But, only those controllers with a receiver operator "tuned" to that broadcast will actually decode the message.



Analog broadcast operators transmit analog signals. They have two inputs—one analog and one digital—as shown in the figure in the left margin. The analog input (the red pin) is the signal that the operator is to broadcast. The digital input (the blue pin) enables (*high*) or disables broadcasting (*low*). If the digital input is not connected, broadcasting is always enabled.

Digital broadcast operators transmit digital signals. They have two digital inputs, as shown in the figure in the left margin. The top digital input is the broadcast signal and bottom digital input is the enable pin.

Mixed broadcast operators transmit communication messages consisting of both analog and digital signals. They have three inputs—one analog and two digital—as shown in the figure in the left margin. The bottom digital input enables/disables broadcasting for the operator. The analog input and middle digital input are combined into a single communication message that the operator broadcasts over the network.

The advantage of mixed broadcast operators is that they may reduce network traffic, since two signals are combined into one. The disadvantage is that the analog signal resolution is lower for mixed broadcast operators (15 bits) as compared to analog broadcast operators (16 bits).

Note: If a schema contains multiple digital broadcasts, *Design Pad* automatically encodes them into a single communication message, thereby reducing network traffic.

Broadcast operators have a number of properties that you must configure for proper network operation. The properties dialog for a broadcast operator is shown in Figure 17 (to view the dialog, double-click on the operator graphic). You must first specify a signal name for the broadcast to distinguish it from other broadcast signals. The name you assign the operator must be unique—no other broadcast operator in any schema on the same network may have the same signal name. *Design Pad* uses the broadcast signal names to associate receiver operators to the broadcast operators.

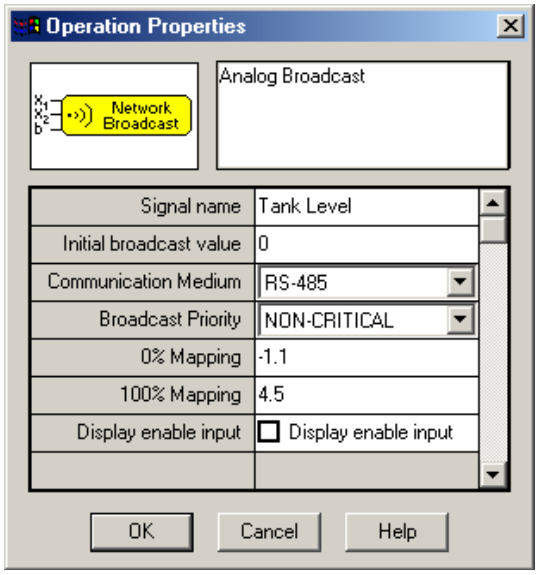


Figure 17. Broadcast operator properties.

Next, you must define which *FAC-2000* communications channel—RS-232 or RS-485—the broadcast operator should transmit its signal on. An individual broadcast operator can only be associated with a single communications network. If you need to transmit the same signal over both networks, you will need to create two distinct broadcast operators.

Each *FAC-2000* controller attached to a communication network will periodically broadcast its information over the network. Some of the broadcasted messages—those that are deemed most critical—can be configured to occur at user-defined intervals. The FAIRNET protocol guarantees that these critical messages are broadcasted within the specified interval. FAIRNET attempts to broadcast the remaining signals—those that are deemed non-critical—at regular user-defined intervals, but it does not guarantee that it will do so. In other words, critical signals are deterministic but non-critical messages are not.

You specify if a broadcast signal is critical or non-critical in the operator's property menu. Since critical messages are deterministic, you may be tempted to setup every broadcast operator as a critical broadcaster. When deciding between the critical and non-critical designation for broadcast messages, it is important to consider how the FAIRNET protocol schedules and prioritizes network traffic. All critical messages in a controller network are assigned top priority—they are broadcasted first. After all critical broadcasts from all stations have been sent, some time remains before the process must be repeated and all critical broadcasts must be sent again. It is during this time window, that the other communication messages are processed. These other communication messages (including non-critical broadcasts and network status/control messages) are placed in a first-in first-out (FIFO) queue maintained by the master controller. During the time window between critical broadcasts, the master station will process as many messages in the queue as possible. If most or all broadcasts are set up as critical broadcasts, then there may be little time left to handle all other network traffic.

6.1.2 Receiver Operators

Receiver operators decode messages that were broadcasted over a communication network.

Analog receiver operators decode analog signals broadcasted over a communications network. They have two outputs—one analog and one digital—as shown in the figure in the left margin. The analog output (the red pin) is the broadcast signal that the operator receives. The digital output (the blue pin) indicates if the operator is actively receiving the broadcasted signal: the output is *low* (0) while transmissions are regularly received; the output is *high* (0) if the operator has not received a broadcast in a user-specified time interval (called the *Maximum Broadcast Dead-Time*). You specify the *Maximum Broadcast Dead-Time* parameter in the operator properties dialog of the receiver.

Digital receiver operators decode digital signals broadcasted over a communications network. They have two digital outputs, as shown in the figure in the left margin. The top digital output is the decoded broadcast signal and bottom digital output is the broadcast status.

Mixed receiver operators decode communication messages consisting of both analog and digital signals. They have three outputs—one analog and two digital—as shown in the figure in the left margin. The bottom digital output indicates if the operator is actively receiving broadcasts. The analog output and the middle digital output are the analog and digital components of decoded communication message that the operator receives.

In order to associate a receiver operator to a particular network broadcast, you must set the signal name property of the receiver to the broadcast signal name. The signal name in the receiver must match the broadcast signal name exactly. (Signal names are case sensitive—*Design Pad* considers the names *OilPressure*, *oilPressure*, and *OILPRESSURE* to refer to different signals.*)*

6.1.3 *Network-Enabled* Constant *and* Remote Constant *Operators*

The *Constant* operator and the *Remote Constant* operator are special networking operators that work together to allow synchronized parameter adjustment over multiple control stations attached to a *FAIRNET* network. Suppose a design effort requires that the same process set-point parameter be adjustable from four control stations. To satisfy the design requirements, you could include an *Constant* operator in the schema for one control station (the "local" station) and associate a *Remote Constant* operator in the schemas of each remaining control station (the "remote" stations). *FAIRNET* takes care of synchronizing them: if a user adjusts the set-point value in any one of the four stations, the value will be automatically updated at the other three stations.

In order to associate a *Remote Constant* operator in one schema to a *Constant* operator in another, you must match the signal name property of the operators—just as you would to associate a receiver operator with a broadcast operator. Both the *Constant* and the *Remote Constant* operators have built-in network broadcasting and receiving capabilities. The *Constant* operator arbitrates communication between itself and associated *Remote Constant*

operators. It is continuously broadcasting the value that all associated operators should hold. When the parameter is adjusted at a remote station, the *Remote Constant* operator sends a message to the *Constant* operator. If the *Constant* operator accepts the remote station command it modifies its output value and broadcasts it over the network. The *Constant* operator broadcast of the new output value serves to confirm that the local station has accepted the remote station command and to alert all other *Remote Constant* operators at the other remote stations that they should update their output value.

In the context of a *FAIRNET* communication network, the *Constant* operator acts primarily as a broadcaster, continuously transmitting its output value. It therefore includes the pertinent networking properties of broadcast operators, namely the communication medium (RS-232 or RS-485) and the broadcast priority (critical or non-critical). The *Remote Constant* operator acts primarily as a receiver "tuned" to the continuous broadcasts of the *Constant* operator. It includes the pertinent networking properties of receiver operators, namely the communication medium (RS-232 or RS-485) and the maximum broadcast "dead-time".

6.1.4 Network-Enabled A/M Button *and* Remote A/M Button *Operators*

The *A/M Button* operator and the *Remote A/M Button* operator are special networking operators that work together to allow synchronized auto/manual operation of multiple control stations attached to a *FAIRNET* network. Suppose a design effort required that users be able to take manual control over the same process loop at any one of four control stations. To satisfy the design requirements, you could include an *A/M Button* operator in the schema for one control station (the "local" station) and associate a *Remote A/M Button* operator in the schemas of each remaining control station (the "remote" stations). *FAIRNET* takes care of synchronizing them. If a user switches any one of the four stations from automatic mode to manual mode, the other three stations will also switch to manual operation. If the user then adjusts the manual output from any one of the four stations, the other three will match that manual output. If another user at a different control station switched the process back into automatic mode, the other three stations will also switch into automatic mode.

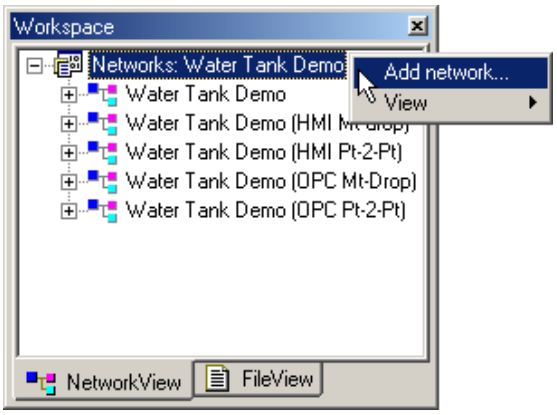
In order to associate a *Remote A/M Button* operator in one schema to an *A/M Button* operator in another, you must match the signal name property of the operators—just as you would to associate a receiver operator with a broadcast operator. Both *A/M Button* and *Remote A/M Button* operators have built-in network broadcasting and receiving capabilities. The *A/M Button* operator arbitrates communication between itself and associated *Remote A/M Button* operators. It is continuously broadcasting the output state that the *Remote A/M Button* operators should assume. When process adjustments are made at a remote station, the *Remote A/M Button* operator sends a message to the *A/M Button*. If the *A/M Button* accepts the remote station command it modifies its output state and broadcasts it over the network. The *A/M Button* operator broadcast of the new output state serves to confirm that the local station has accepted the remote station command and to alert all other *Remote A/M Button* operators at the other remote stations that they should update their output states.

In the context of a *FAIRNET* communication network, the *A/M Button* operator acts primarily as a broadcaster, continuously transmitting its output state. It therefore includes

the pertinent networking properties of broadcast operators, namely the communication medium (RS-232 or RS-485) and the broadcast priority (critical or non-critical). The *Remote A/M Button* operator acts primarily as a receiver "tuned" to the continuous broadcasts of the *A/M Button* operator. It includes the pertinent networking properties of receiver operators, namely the communication medium (RS-232 or RS-485) and the maximum broadcast "dead-time".

6.2 Defining *FAIRNET* Communication Networks

The previous section describes how to configure *Broadcast* and *Receiver* operators to communicate schema signals over a *FAIRNET* network. A *Broadcast* operator in one schema transmits a signal over the network and a *Receiver* operator in another schema captures it. How do you establish the signal link? Half of the effort is associating the broadcast/receiver operator pair—simply a matter of labeling them with the same signal name and assigning them to a common communication channel (RS-232 or RS-485). The other half is creating and configuring a network and then associating schema documents to that network.



To create a communication network in *Design Pad*, first select the *Network View* tab in the workspace window. Then, right click on the workspace root node and select the *Add Network* item from the drop down menu. *Design Pad* will respond with the Add Network to Workspace dialog shown in Figure 18. This dialog will prompt you to enter a name for the network, to identify the network type (multi-drop or point-to-point), and to set the critical broadcast latency time.

The network name is used to distinguish networks in a workspace. The workspace *Network View* tab contains a tree node for each network you have created—the node is labeled with the network name next to a graphic. There are no limitations on the number of networks that may be defined in a *Design Pad* workspace. When you create a new network, the Add Network to Workspace dialog displays a list of all the networks that exist in the workspace.

FAIRNET supports two types of networks: *point-to-point* and *multi-drop*. In a point-to-point network, two devices are linked directly to one another over a two-way channel (communication with additional devices is not possible on this channel). In a multi-drop network, several devices (two or more) share a common communications channel. Devices can be added to (or removed from) the common channel as required. The *FAC-2000* controller includes one RS-232 port for point-to-point networking and one RS-485 port for multi-drop networking.

FAIRNET supports deterministic and non-deterministic messaging. Deterministic messages are reserved for the most critical signals in a networked control system. A *FAIRNET*

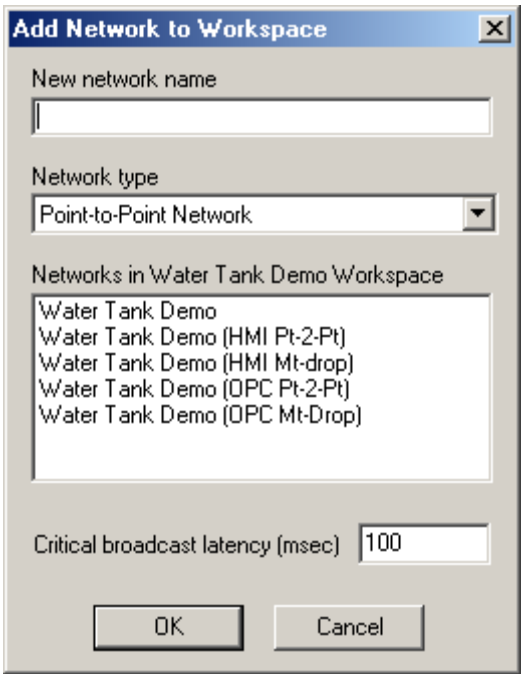


Figure 18. Add Network to Workspace dialog.

network guarantees that these signals are transmitted at regular intervals. The length of the interval between successive critical message transmissions is called the *Critical Broadcast Latency Time*. By default, Design Pad sets this value to 100 milliseconds (*i.e.*, by default, critical broadcast messages are transmitted every 100 milliseconds.)

Note: *FAIRNET* also transmits non-deterministic messages at regular intervals. However, it does not guarantee that it will always do so (occasionally, the transmission of a non-critical message could be delayed).

6.3 Linking Documents to Communication Networks

Once you have defined a communications network in *Design Pad*, you must specify which schema documents (*i.e.*, which control stations) will be part of that network. And if a PC will also be part of the network, you must associate the network with the appropriate HMI document or OPC server configuration document.

To assign a document (schema or HMI) to a controller network in *Design Pad*, first select the *File View* tab in the workspace window. Next, right click on document you would like to assign to a controller network, and select the Assign Network item from the drop down menu. *Design Pad* will respond with the Assign Schema to Network(s) dialog shown in Figure 19. Then select the network(s) you would like to associate with the document.

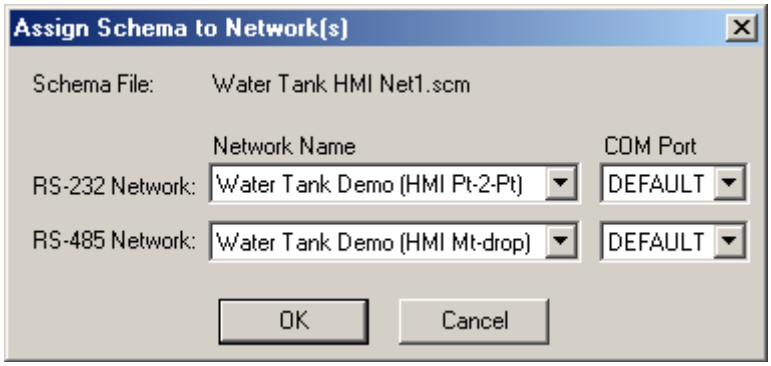


Figure 19. A schema document can be assigned to one point-to-point network (RS-232) and one multi-drop network (RS-485).

Schema documents are designed for execution in a *FAC-2000* controller. Since the *FAC-2000* controller includes one RS-232 port for point-to-point networking and one RS-485 port for multi-drop networking, you can associate a schema with up to two networks (one point-to-point network and one multi-drop network). The Assign Schema to Network(s) dialog (Figure 19) contains drop-down boxes that list the networks available in the workspace. (The drop-down box labeled RS-232 lists all the point-to-point networks that are available in the workspace. And the drop-down box labeled RS-485 lists all the multi-drop networks that are available in the workspace.) To assign a schema to a network, simply select the name of the network in the appropriate drop-down box.

When you assign a workspace HMI node to a network you must also identify the PC COM port that *Design Pad* should use when communicating with that network. (HMI documents are introduced in section 9.1.) For example, suppose you have a PC with an RS-485 network card configured to operate on the computer's COM5 port. If you needed to interface a controller multi-drop network to a *Design Pad* HMI document you would: (i) create a multi-drop network; (ii) create schemas for each control station and assign them to the network; and (iii) create an HMI document, assign it to the network, and set the COM port setting to COM5.

Note: When you assign a workspace schema node to a network, always set the PC COM port settings to "DEFAULT".

6.4 Accessing Network Configuration Information

The workspace maintains configuration information on each of its networks (*e.g.*, the network type, the latency time for critical broadcasts, the schemas assigned to each network, the broadcast signals that traffic each network, *etc.*). Design Pad encodes this information in network configuration files and packages them with a schema to program a *FAC-2000* control station. (See section 7.2 to learn how to export a schema and its associated network

configuration files to a controller). The network configuration files provide the control station with all the information it needs to participate in a controller network.

Design Pad provides two useful dialogs to obtain this "nuts and bolts" information about a controller network and its broadcast signals: the Network Properties dialog and the Network Broadcast Signal Properties dialog. To access either dialog, first select the *Network View* tab in the workspace window. Next, right click on the subject network, and select the desired item (Network Properties or Network Broadcast Signal Properties) from the drop down menu. Figure 20 shows a sample Network Properties dialog. It displays the following information:

Network Name. The name of the network (displayed in the *Network View* tab of the workspace window).

Network ID. A unique identifier generated by *Design Pad* when the network is created. *FAIRNET* uses this identifier to determine if a station belongs on the network. (When a controller is physically connected to a network, its Network ID must match that of other stations on the network.)

Network Version. An integer value that tracks changes in a network's configuration. *FAIRNET* uses this parameter to maintain a consistent set of configuration files for all stations on a network.

Network Type. *FAIRNET* supports point-to-point networking and multi-drop networking. In a point-to-point network, two devices are linked directly to one another over a two-way channel (communication with additional devices is not

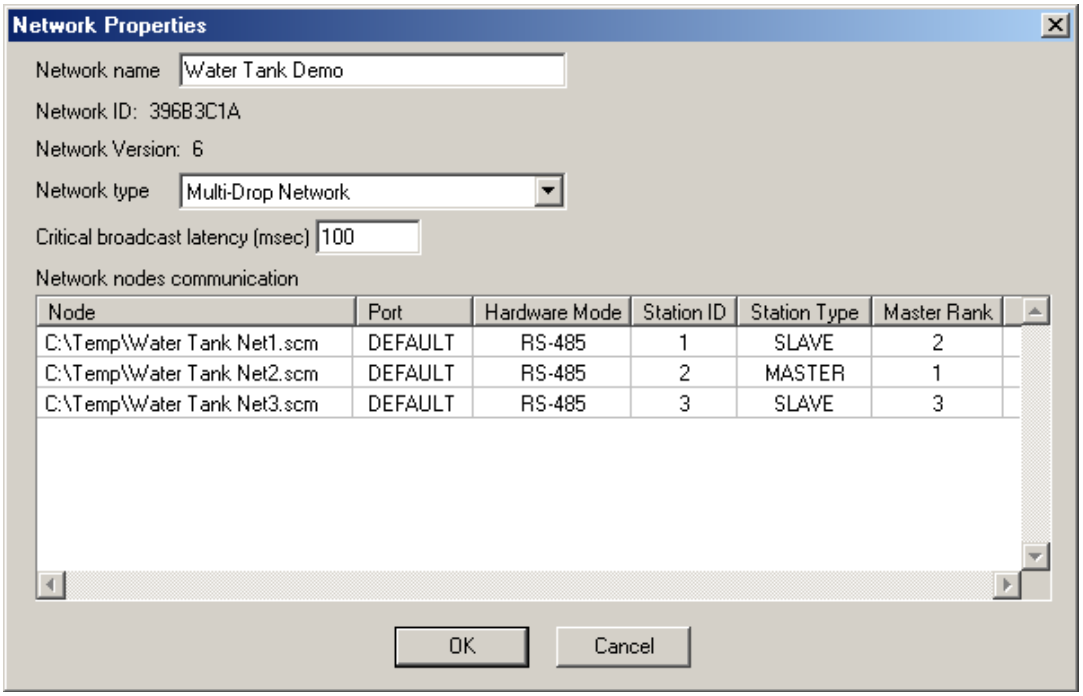


Figure 20. The network properties dialog.

possible on this channel). In a multi-drop network, several devices (two or more) share a common communications channel. Devices can be added to (or removed from) the common channel as required. The *FAC-2000* controller includes one RS-232 port for point-to-point networking and one RS-485 port for multi-drop networking.

Critical Broadcast Latency Time. *FAIRNET* supports deterministic and non-deterministic messaging. Deterministic messages are reserved for the most critical signals in a networked control system. A *FAIRNET* network guarantees that these signals are transmitted at regular intervals. This parameter specifies the length of the interval between successive critical message transmissions (expressed in milliseconds). By default, *Design Pad* sets this value to 100 milliseconds (*i.e.*, by default, critical broadcast messages are transmitted every 100 milliseconds.)

Network Nodes. Information about each node assigned to the network, including:

Node. The name of the node (schema document, HMI document, or OPC server configuration document)

Port. The PC COM port that *Design Pad* uses to communicate with the network node. For schema documents, the port should always be set to "DEFAULT". For HMI documents or OPC server configuration documents, this parameter indicates which PC port the HMI software uses to communicate with the network.

Hardware Mode. The hardware mode indicates the communications protocol (at the physical layer) that the node uses to participate in the network. For a point-to-point network, both nodes must use the RS-232 protocol. For a multi-drop network, all nodes must use either RS-485 or a *memory pipe* (all nodes must be one or the other). If a multi-drop network contains a schema document (*e.g.*, a *FAC-2000* control station), then all nodes must use RS-485. (The memory pipe option is reserved for virtual controller networks. A virtual network is a PC simulation of a controller network involving multiple applications that interact over a memory pipe. Each application represents a distinct station on the network.)

Station ID. A *Design Pad* generated integer value that uniquely identifies each node in network. *FAIRNET* uses this parameter to determine the source and destination of a given communication message.

Station Type. *FAIRNET* uses a master/slave architecture to arbitrate communication messages. One device on the network—the master device—coordinates all the network traffic. All other devices are slaves to the network master—they transmit information on the network only when prompted to do so by the master device. *Design Pad* automatically selects which station on the network will serve as the master.

Master Rank. Every station in a *FAIRNET* network has the capacity to function as the network master. If the master goes offline, one of the remaining slave

devices on the network will automatically take on the master role. This parameter determines which device on the network assumes the master role (the active station with the lowest master rank).

Figure 21 shows a sample Network Broadcast Signal Properties dialog. It displays the following information:

Signal Name. The name of the broadcast signal (corresponds to a broadcast operator in a schema).

Message ID. A unique identifier that Design Pad assigns to each broadcast signal. (Digital broadcast signals from the same station may have duplicate message identifiers.)

Bit Position. In order to optimize network usage, FAIRNET encodes digital broadcast signals from each station into 16-bit integer values (each integer value encodes the states of 16 digital signals). The bit position parameter indicates which bit in the encoded integer value represents the given digital broadcast signal.

Source Station ID. The station identifier for the node that transmits the given broadcast signal. (*Design Pad* assigns each node in a controller network a unique *Station ID*—see the Network Properties dialog.)

Operator ID. *Design Pad* assigns each broadcast operator a unique identifier, which it uses to associate broadcast and receiver operators.

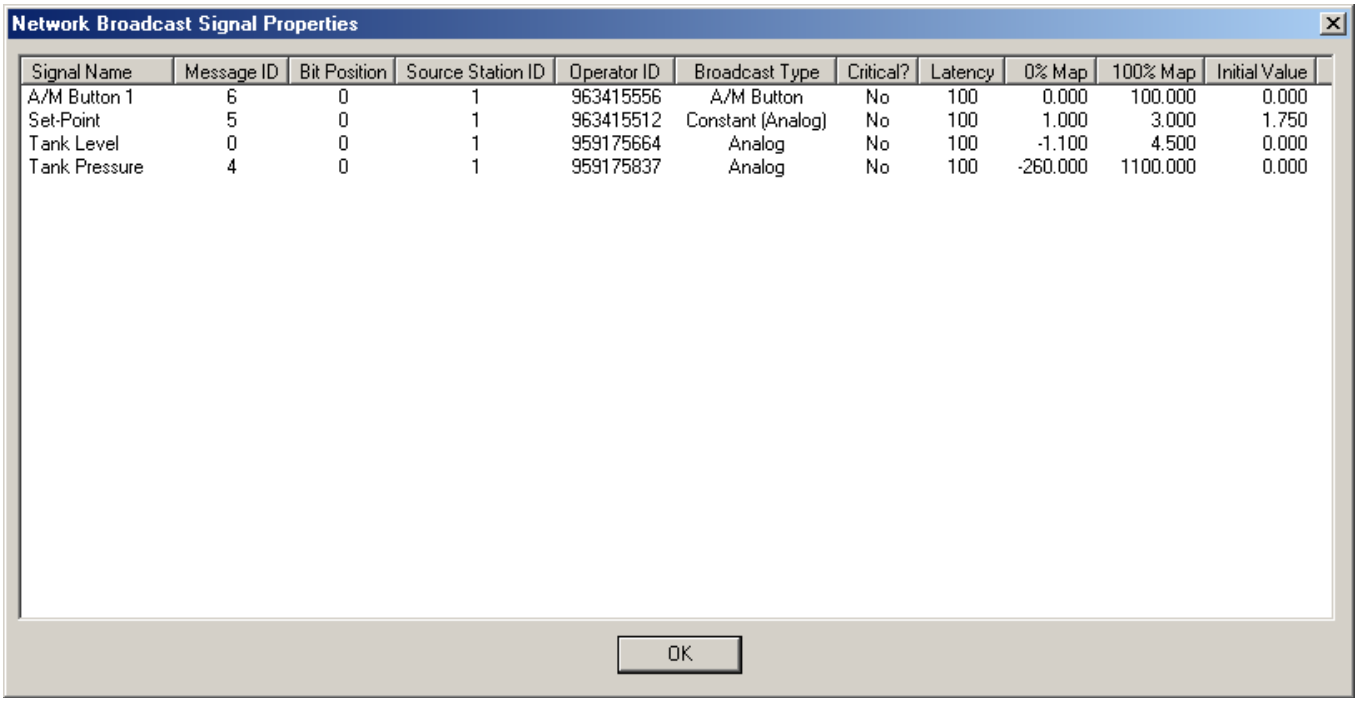


Figure 21. Broadcast signal properties dialog.

Broadcast Type. Design Pad provides several networking operators: analog broadcast, digital broadcast, mixed broadcast, network-enabled A/M Button, and network-enabled Constant. See section 6.1.1 for a detailed description of each.

Critical? *FAIRNET* supports deterministic and non-deterministic messaging. Deterministic messages are reserved for the most critical signals in a networked control system. A *FAIRNET* network guarantees that these signals are transmitted at regular intervals. FAIRNET also transmits non-critical messages at regular intervals but does not guarantee that it will always do so—occasionally, during periods of heavy network traffic, some non-critical message transmissions may be delayed. This parameter indicates if the given broadcast signal is or is not critical.

Latency. The time interval (in milliseconds) between successive broadcast transmissions. All critical broadcasts in a network have the same latency time (it is a property of the network). Non-critical broadcast message latency times can vary from one station to another (the non-critical broadcast message latency time is a property of the nnetwork node).

0% Map. The lower limit of the broadcast signal range.

100% Map. The upper limit of the broadcast signal range.

Initial Value. The initial broadcast signal value.

7. Communicating with a *FAC-2000* Controller

Design Pad has built-in communications software that enables two-way communication between a personal computer and a *FAC-2000* controller network. To take advantage of these communication features, connect a serial cable (Fairmount Automation Part Number 0100-4000) between an unused PC communications port and the COM1 input (for RS-232 communications) or the COM2 input (for RS-485 communications) of the *FAC-2000* unit.

The new *FAC-2000* embedded code incorporates many changes in order to enable controller networking. One significant modification is the protocol that the controllers use for communication. In the new protocol, *FAC-2000* controllers can process multiple communication messages simultaneously. In first-generation *FAC-2000* executable code versions (1.21 and below), controllers could only process a single communications message at a time. *Design Pad 2000* can communicate with controllers that use either first-generation or second-generation versions of the embedded code. (Older versions of *Design Pad* are only capable of communicating with controllers running a first-generation version of the embedded code.) When using *Design Pad 2000,* you must specify which communication mode to use: single-threaded mode for controllers with first-generation embedded code and multi-threaded mode for controllers with second-generation embedded code. You set the communications mode in the communications tab of the *Design Pad 2000* Preferences dialog (see section 10.4).

In this section, we describe all the PC-controller communication features available in *Design Pad 2000*.

Note. Prior versions of *Design Pad* prompted you for the PC communications port every time you initiated communication with a controller. In *Design Pad 2000*, you only need to specify the communications port once. You do so in the Communications tab of the *Design Pad 2000* Preferences dialog (see section 10.4).

7.1 Network Monitor Window

When *Design Pad 2000* communicates with a controller in multi-threaded mode, it displays communication results in the network monitor window. (When using single-threaded mode, *Design Pad* uses simple dialog boxes to convey information to you.) The network monitor window is normally docked at the bottom right corner of the *Design Pad* application window. It displays text messages that describe the results of PC-controller communications. For instance, when you execute the Analyze Network command (see section 7.7), the network monitor window will display the results of the analysis.

Network Monitor Window

You toggle the visible state of the network monitor window (shown or hidden) from the view menu. When you make the network monitor window visible, *Design Pad* displays a check mark next to the Network Monitor Window item in the View menu; it removes the check mark when the window is not visible. You can also toggle the visible state of the message window by pressing the corresponding button on the toolbar menu (shown to the right).

The network monitor window is always displayed on top of other windows in the *Design Pad* application environment. By default, it is docked on the bottom-right side of the screen. But it can be positioned (and resized) anywhere on the screen—it can be docked at the top, bottom, left side or right side of the client area or it can be floating anywhere on your computer screen (even outside of the application's client message). If the network monitor window is docked and you wish to position it elsewhere, click on the window handle (two raised lines adjacent to the close button) and drag it to the desired location. To resize the network monitor window while it is docked, click the right mouse button on the inner edge of the window and drag it to the desired location. When the network monitor window is floating, you can resize it as you would any other window.

7.2 Exporting a Schema to the *FAC-2000* Controller

Schema Export

Previous chapters in this instruction bulletin described how to create control schemas and how to configure controller networks. This section describes the final step in the *FAC-2000* programming process: how to download a schema and associated network configuration files into the device.

Design Pad 2000 includes communication features to export a schema from a PC to a *FAC-2000* controller and also to retrieve (or import) a schema from a controller back to the PC. The import feature is useful if you wish to view or edit the schema that a *FAC-2000* controller is executing (and you don't have it available on the PC). However, it also allows others to review the work you have done (the schema you created). To prevent unauthorized access to a schema you have exported to a controller, you can establish a communication password that a user must provide when attempting to import the schema into a PC. You set the communications password in the Password tab of the Controller Options dialog for a schema (see section 4.9). You can further protect your intellectual work by disabling the import functionality entirely when you export a schema to a controller. If you disable the schema import functionality, nobody will be able to access the schema executing in the controller—including you.

To program the *FAC-2000* controller, first open the workspace document that references the desired schema and contains the associated network configuration information. (If the station will not be part of a controller network, you need not open the workspace document—you can simply open the desired schema document.) Next, select the Export Schema with Import Enabled item or the Export Schema with Import Disabled item from the Controller menu. *Design Pad* will display a progress dialog (see Figure 22) with information about the programming process. When the process is complete, *Design Pad* will automatically remove the dialog.

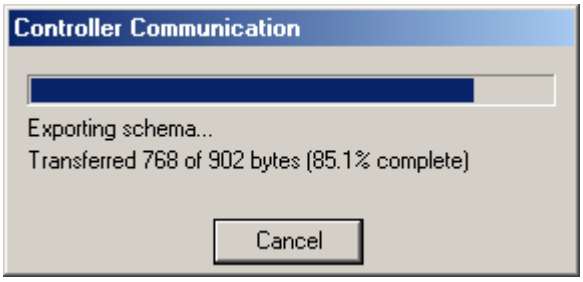
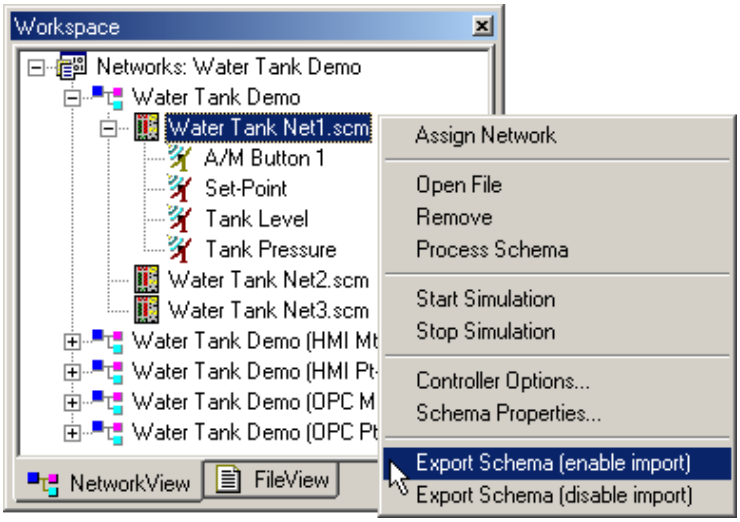


Figure 22. Progress dialog displayed while PC communicates with a *FAC-2000* control station..

When the *FAC-2000* receives a new schema, it verifies that it can properly execute it. Namely, it ensures that it can execute the schema faster than the user-defined loop sampling time, Δt. For instance, a very complex schema with scores of operators may require more than the default $\Delta t = 0.1$seconds to execute. In these cases, the *FAC-2000* controller will notify *Design Pad* that the loop time is too short. If you receive this message, you must increase the loop time and re-download the schema into the device. Otherwise, the *FAC-2000* may produce significant computational errors: any schema operator with an output that is a function of time will be adversely affected.



As an alternative to using the Controller menu, you can export a schema to a controller from the workspace window. If you use this method, you don't need to open the schema file. Just right-click on the desired schema document item in the workspace window and select one of the Export Schema items from the drop-down menu. (You can access the drop-down menu from both the *Files* and *Networks* tabs of the workspace window.)

7.3 Importing a Schema from the *FAC-2000* Controller

Occasionally, it may be useful to retrieve a schema executing in the controller and display it in *Design Pad*. To import a schema into *Design Pad*, connect the controller to your computer via serial cable and choose the Import Schema item from the Controller menu. If the schema executing in the device contains a communications password, you will be prompted to provide that password before *Design Pad* retrieves the schema. And if the schema was exported with import disabled, the controller will not allow you to import its schema at all.

When *Design Pad* has completed importing a controller's schema, it will display it in a new window, as an unnamed document.

7.4 Controller Calibration

The *FAC-2000* controller is calibrated at the factory prior to delivery. From time to time however, it may be necessary to re-calibrate the device. *Design Pad* provides two calibration methods for the controller's analog input and output channels. One method is fully automated; the other requires significant user interaction. Each method is described in detail in Fairmount Automation Technical Bulletin 9110-0003—the *FAC-2000* calibration procedure.

7.5 Updating the Controller Execution Code

Occasionally, Fairmount Automation will add new function blocks to *Design Pad's* operator suite. In order for schemas utilizing the new operators to work in the *FAC-2000*, the controller's execution code must be updated.

When Fairmount Automation makes changes to the controller software, it will distribute the execution code as a binary file. The file will be entitled `ctrlv###.$cc`, where ### is the execution code version number times 100. To determine the version of the code currently executing in your controller, select the Query Execution Code Version item from the Controller menu. (Of course, the query will fail if your computer is not properly connected to the controller.)

The process of updating the control code in the *FAC-2000* controller requires several minutes, will erase the schema file the controller is running, and will reset the controller. To do it, connect the controller to your computer via serial cable and select the Download Execution Code item from the Controller menu. After displaying a warning message, *Design Pad* will prompt you for the .$cc binary file. Locate the file, press the OK button, and wait *for Design Pad* to complete the code update.

7.6 FAC-2000 Network Licenses

The *FAC-2000* controller requires a network license to enable its networking features. Without a network license, a *FAC-2000* controller will not be able to communicate with other devices.

You can determine if a control station has a valid network license using the Query Network License Status command in the Controller menu. If the controller does not have a network license, you can obtain one electronically (via e-mail) by contacting Fairmount Automation. First you will need to extract a network key from the control station you wish to obtain a network license for. To do so, connect the controller unit to your PC via serial cable and select the Query Network License Key item from the Controller menu. *Design Pad* will query the controller and will prompt you with the dialog box shown in Figure 23. To obtain a license, complete the requested information, including your name, affiliated organization, and contact information (phone, fax, and address), as well as the controller particulars (model number, serial number, and a description of the application where it will be used). (Design Pad remembers your contact information from one session to another, so you will not have to repeatedly enter it.) Once you have completed the form, you can save the information to a text file (to include in a subsequent e-mail) or Design Pad can automatically prepare an e-mail message and launch your e-mail software for immediate transmission. Send all license requests to licensing@FairmountAutomation.com. A Fairmount Automation representative will reply to your request with a network license file for the controller. When you receive the license file, you export it to the device using the Download Network License File command from the Controller menu.

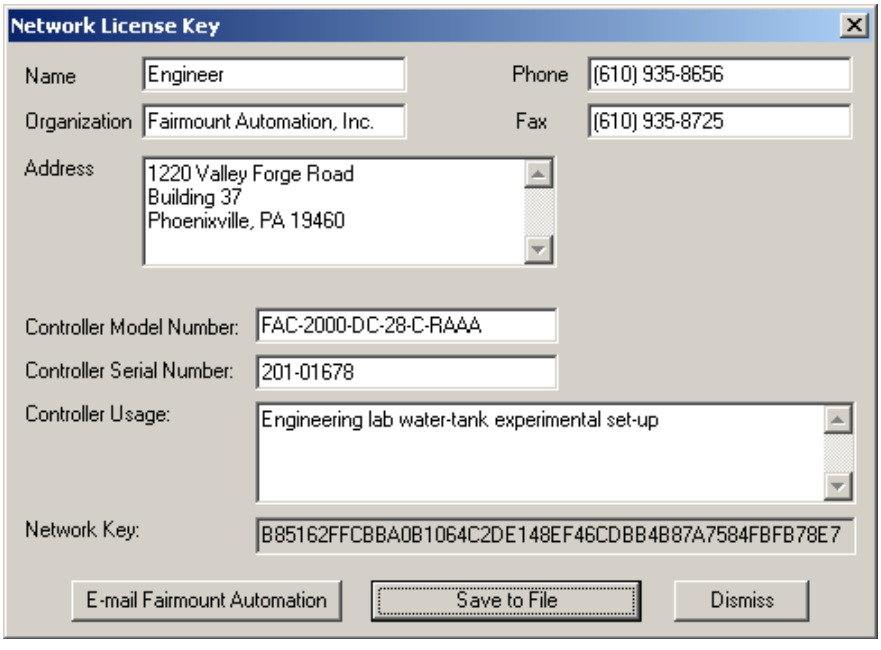


Figure 23. The network License Key dialog used to request a FAIRNET network license.

7.7 Analyzing Network Performance

Design Pad provides a convenient mechanism to analyze the performance of a FAIRNET network. The analysis includes statistics on transmission error rates, message transmission duration, and message transmission frequency. To initiate the analysis, first connect the network's master station to your PC using the RS-232 link (COM1 on the FAC-2000). Then, select the *Networks* tab in the workspace window, right-click on the desired network, and select the Analyze Network Messaging item from the drop-down menu. The master station will collect communication statistics over a period of 40 seconds and will then report the results to Design Pad; Design Pad displays these results in its network monitor window.

Figure 24, shows the results of a two-station FAIRNET network. The results include a summary section with a count of all network messages that were transmitted during the analysis period, the number of messages that were not correctly processed, and the number of data errors that occurred. It also includes a detailed breakdown of each type of message that was transmitted during the analysis period. For each message type, Design Pad displays the number of times it was transmitted; the number of times an error occurred during its transmission; the minimum, maximum, and average time (in milliseconds) time it took to

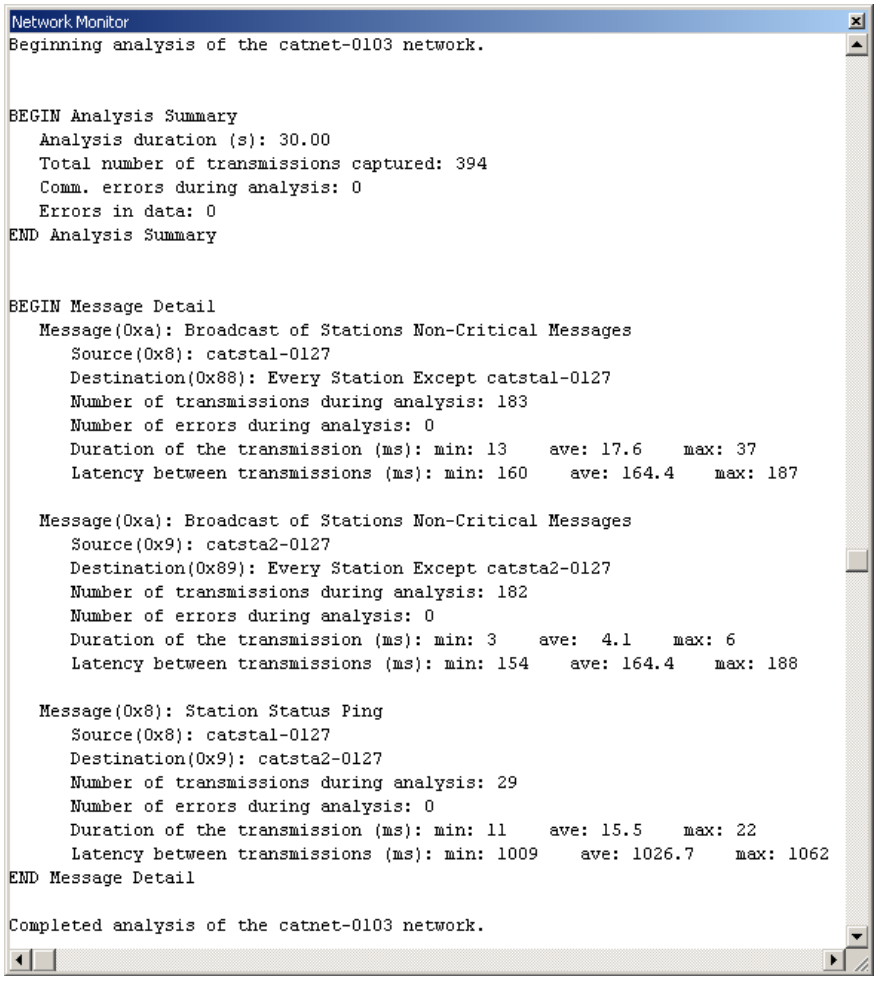


Figure 24. Results of FAIRNET analysis.

transmit the message; and the minimum, maximum, and average time (in milliseconds) between consecutive transmissions of the same message type.

7.8 Other Communication Features

Design Pad offers additional communication features to query and/or alter the state of the *FAC-2000* controller. These additional features are described below.

When you first received a *FAC-2000* controller from the factory, you may have noticed that the device does not contain a schema file to execute. When you powered a unit in this state, the alphanumeric displays contained the message "Awaiting Schema Download". You can return a programmed unit into this state by erasing the schema it is executing. To delete a controller's schema, connect the controller to your PC via serial cable and select the Erase Schema item from the Controller menu. (When communicating with a controller code first-generation code, you will need to restart the device in order for the command to take effect.)

You can restart the *FAC-2000* controller in one of two ways: Cycle the power to the device; or use *Design Pad*'s soft-reset feature. To reset the device, select the Re-start Device item from the Controller menu.

Fairmount Automation assigns a unique serial number to every *FAC-2000* controller it manufactures. This serial number is stamped on the bottom face of the controller. You can obtain the controller's serial number by visually inspecting the device or by querying it with *Design Pad*. To query the device with *Design Pad*, select the Query Serial Number item from the Controller menu.

8. RUNNING SIMULATIONS IN *DESIGN PAD*

A critical aspect of control schema design is testing. *Design Pad* offers built in simulation features that can be of great benefit when testing your schema. In this section, we develop a simulation for the example schema shown in Figure 1 (page 2). This schema was designed to regulate the water level in a tank. The controller processes measurements of the water level (the process variable) and regulates the position of an outlet valve to maintain the water level at a given set point.

We can express the water level in the tank, h, with the first-order differential equation

$$h(t) = \frac{1}{A}\int (f_{in} - f_{out})dt, \qquad h(0) = h_0$$

where h_0 is the initial water level, f_{in} is the inlet flow rate, f_{out} is the outlet flow rate, and A is the cross-sectional area of the tank. Differentiating both sides (with respect to t) we obtain

$$\dot{h} = \frac{f_{in} - f_{out}}{A}.$$

We can convert this continuous-time model into a discrete-time one by using the coarse approximation $dh/dt = \Delta h/\Delta t$:

$$\frac{\Delta h}{\Delta t} = \frac{h_i - h_{i-1}}{\Delta t} = \frac{f_{in} - f_{out}}{A}.$$

We solve for h_i to obtain

$$h_i = h_{i-1} + \left(\frac{f_{in} - f_{out}}{A}\right)\Delta t.$$

This discrete-time equation can be easily reproduced in *Design Pad* using its mathematical operators. Figure 25 contains the equivalent *Design Pad* formulation of the water tank model coupled to the schema developed in section 4 (see Figure 1). (A copy of this schema, named `tanksim.scm` is located in the `Fairmount Automation/Examples` folder.) Notice that, in this simulation schema, we have removed the *Analog Input 1* and *Analog Output 1* operators. In their place, we use the output of the plant model (the water level) to feed the *PV* input of the *PID Controller* operator. And, we use the output *M* of the PID block to drive the plant model (we use the Unit Conversion operator to map the PID output into the tank's outlet flow rate.)

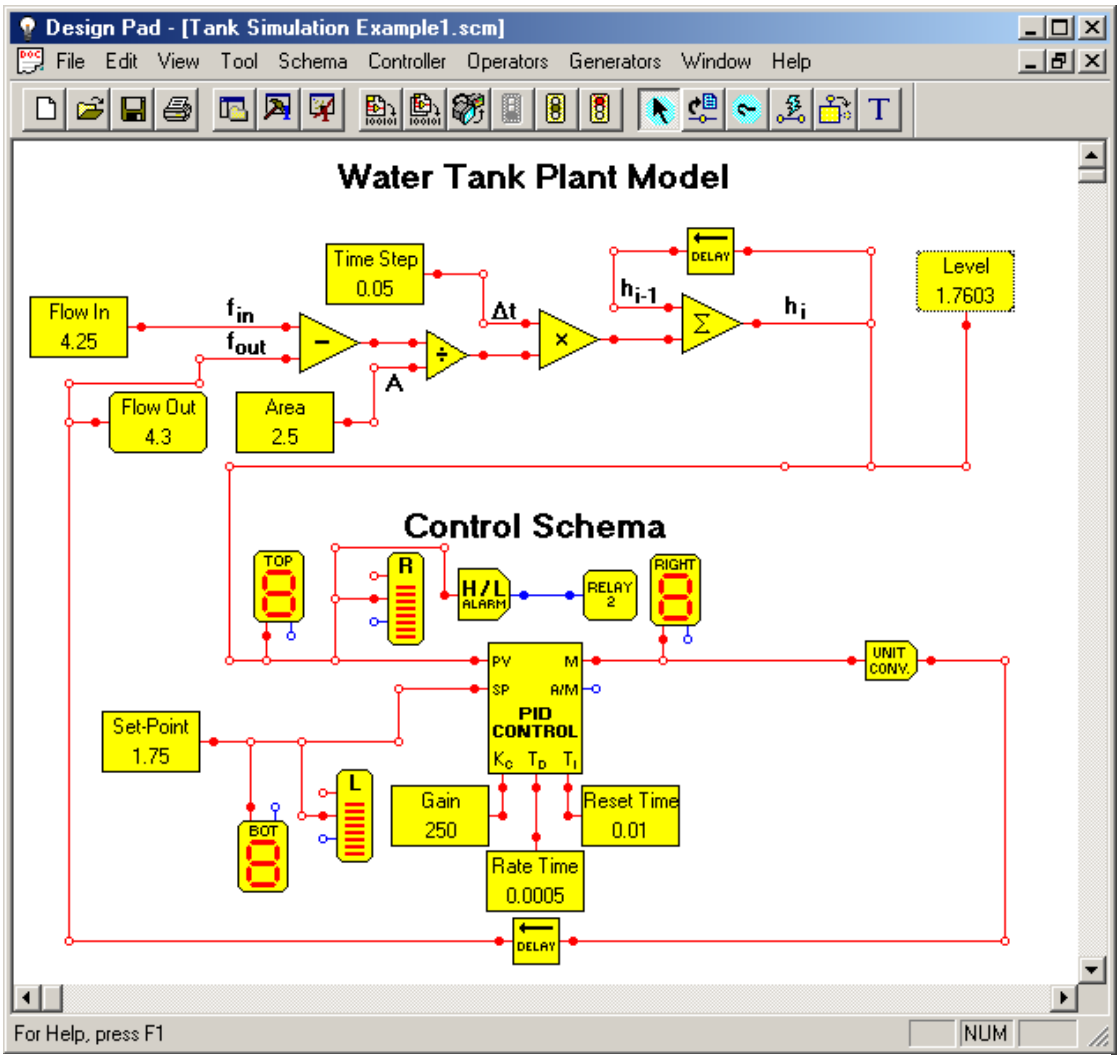


Figure 25. Simulation schema used to test the PID control strategy described in section 4 (see Figure 1, page2).

Start Sim

Stop Sim

Once you have synthesized a simulation schema, executing a simulation in *Design Pad* is easy—simply select the Start Simulation item from the Schema menu. (Alternative, you can click the "green traffic light" button on the toolbar menu.) You terminate a simulation with the Stop Simulation item from the Schema menu (or the "red traffic light" button on the toolbar menu).

8.1 The Probe Operator

When testing a schema, it is often useful to monitor signals as a simulation progresses. *Design Pad* provides probe operators that display signal values during a simulation. You use Probe operators like any other operator. First, you select the Probe (Analog or Digital) item from the Operators menu and position it within your schema. Then, simply connect the signal you wish to monitor to the one of object's input pins. (You can specify the number of inputs—up to five—in the operator's property sheet.)

Two probe operators are included in the simulation schema shown in Figure 25. One monitors the outlet flow rate and the other the water tank level. While a simulation executes, the probes display the present signal values. (Probes with multiple inputs display the value of the first input). Probe operators also allow you to view a dynamic plot of the signal(s) connected to it. To view this dynamic plot, activate the Selection tool, position the cursor over the probe, and double-click the right mouse button. For instance, in Figure 26 we show the (water level) *Probe* operator's graph window, as the controller responds to a set point change.

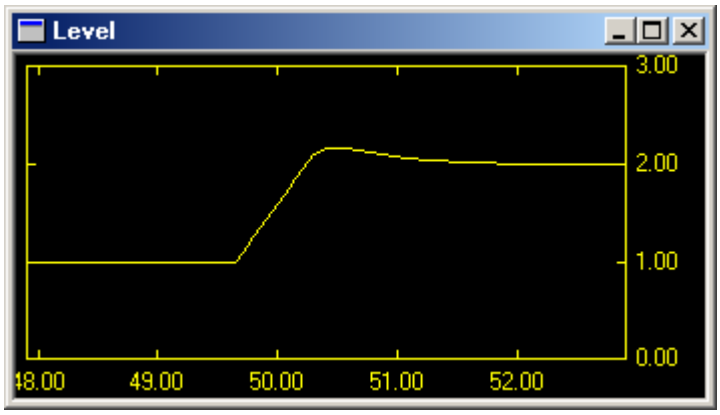


Figure 26. A *Probe* operator's graph window.

The dynamic graph will display all of the input signals connected to the probe operator. (See the figure in the front cover of this technical bulletin for an example.)

In Design Pad 2000 Professional, probe operators can record data directly to disk for subsequent analysis. Data is stored in a tab-delimited text file that can easily be imported to data analysis tools such as Microsoft Excel. This feature can be very useful when used as part of an HMI document that is linked to a FAIRNET network, as it provides a means to store sensor data acquired by an FAC2000 controller.

8.2 The Front Panel Window

An important aspect of schema development is the design of the user-interface—the *FAC-2000* front panel. You must decide which signals to present on the bargraph and numeric displays. You must determine how to order adjustable constant operators for presentation on

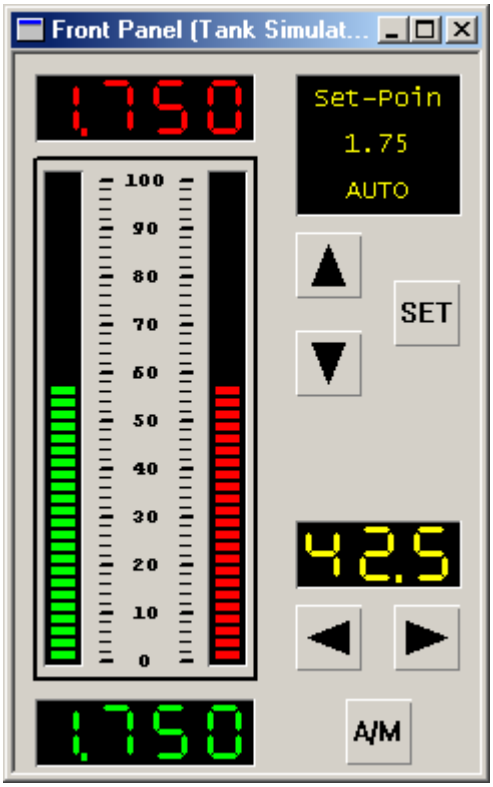


Figure 27. *Design Pad*'s' virtual front panel.

the *Set Menu*. And, you must choose what information to include on the alphanumeric displays. Rather than testing the user-interface directly on the controller, you can tweak your schema in *Design Pad*. When you run a simulation, you can test the user-interface with *Design Pad*'s front panel window. To activate it, select the Front Panel Window item from the View menu. As you can see in Figure 27, the virtual front panel looks very much like the real front panel. *Design Pad*'s front panel window not only looks like the real thing; it behaves like the real thing. During a simulation, all of the keypad buttons are active and behave just like they do on the device. However, there is one exception. Recall that to access 'Engineer Level' constants in the *Set Menu* (see section 4.7) you must simultaneously push the UP and DOWN arrow keys. But, clicking two buttons simultaneously in *Microsoft Windows* is not possible. How do you emulate pressing the UP and DOWN arrow keys simultaneously in *Design Pad*? You hold the `SHIFT` key down while you click either arrow button (UP or DOWN).

8.3 The Rear Panel Window

In the simulation example of section 8, we added a simple plant model to the sample water-tank control schema of Figure 1. The plant model replaced the *Analog Input 1* and *Analog Output 1* operators in the control schema to simulate a closed-loop system. As an alternative to creating a plant model, *Design Pad 2000 Professional* offers the capability to simulate input signals using a virtual rear-panel.

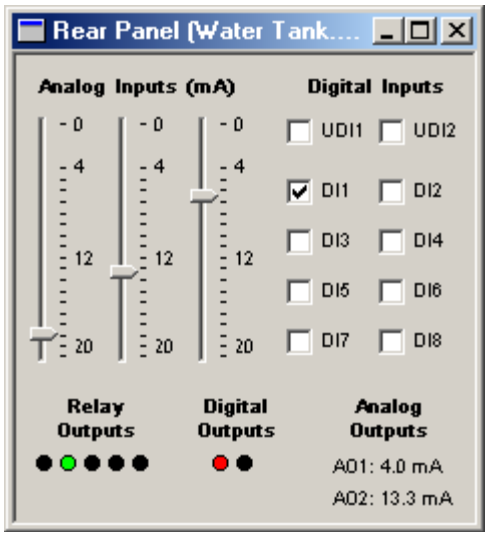


Figure 28. Design Pad's virtual rear-panel.

To display the Rear Panel Window shown in Figure 28, select the Rear Panel Window item from the View menu. The Rear Panel Window contains three slider controls (one for each analog input on the *FAC-2000*), ten check-box controls (for the universal digital inputs and standard digital inputs), seven indicator lights (five for the relay outputs and two for the digital outputs), and two text displays (for the analog outputs). When you are running a simulation, you can vary analog input values using the slider controls, set the digital input values with the check-boxes, and monitor the outputs with the indicator lights and text displays. (A green light indicates when a relay output is turned on and a red light indicates when a digital output is turned on.)

8.4 Signal Generators

The example simulation schema presented in section 8 assumes ideal operating conditions. It cannot predict how the controller might react to external disturbances or noisy measurements. You can analyze the robustness of your schema by adding signal generator blocks. The signal generators available in *Design Pad* include

- Impulse, step, and ramp functions
- Sine and cosine wave generators, and
- Random number generator

NOTE: Signal generator blocks are made available for simulation purposes only. They are not part of the *FAC-2000* operator suite. The controller will not accept schemas that contain signal generators.

To insert a signal generator into your schema, select the appropriate item from the Generators menu and position it at the desired location. Generator objects behave very much like Operator objects. You can select them with the Selection tool and adjust their position. You can change the position of their output pin(s) with the Pin Rotate Tool. And, you can connect them to other objects with the signal editing tools. Just like operator blocks, signal generators have properties that determine their behavior. For example, you can set the amplitude and frequency of the sine generator or the seed value for the random number generator. You set the functional characteristics of generators in their property sheet. To view the property sheet, activate the Selection tool, position the cursor over the generator, and double click the right mouse button.

In Figure 29, we demonstrate the use of signal generators on the simulation schema developed in section 8. (A copy of this schema, named `tanksim2.scm`, is located in the `Fairmount/Examples` directory.) In `tanksim2.scm`, we have added a random number generator to simulate the effects of measurement noise. And, we have added a pair of step functions to simulate the effects of an external disturbance (*e.g.*, a bucket of water is externally added to the tank). In Figure 30, we show the graph window of the *Tank Level* probe as the controller responds to these disturbances.

All the *Design Pad* generators and their associated properties are described in the *Signal Generator Reference* section of this bulletin.

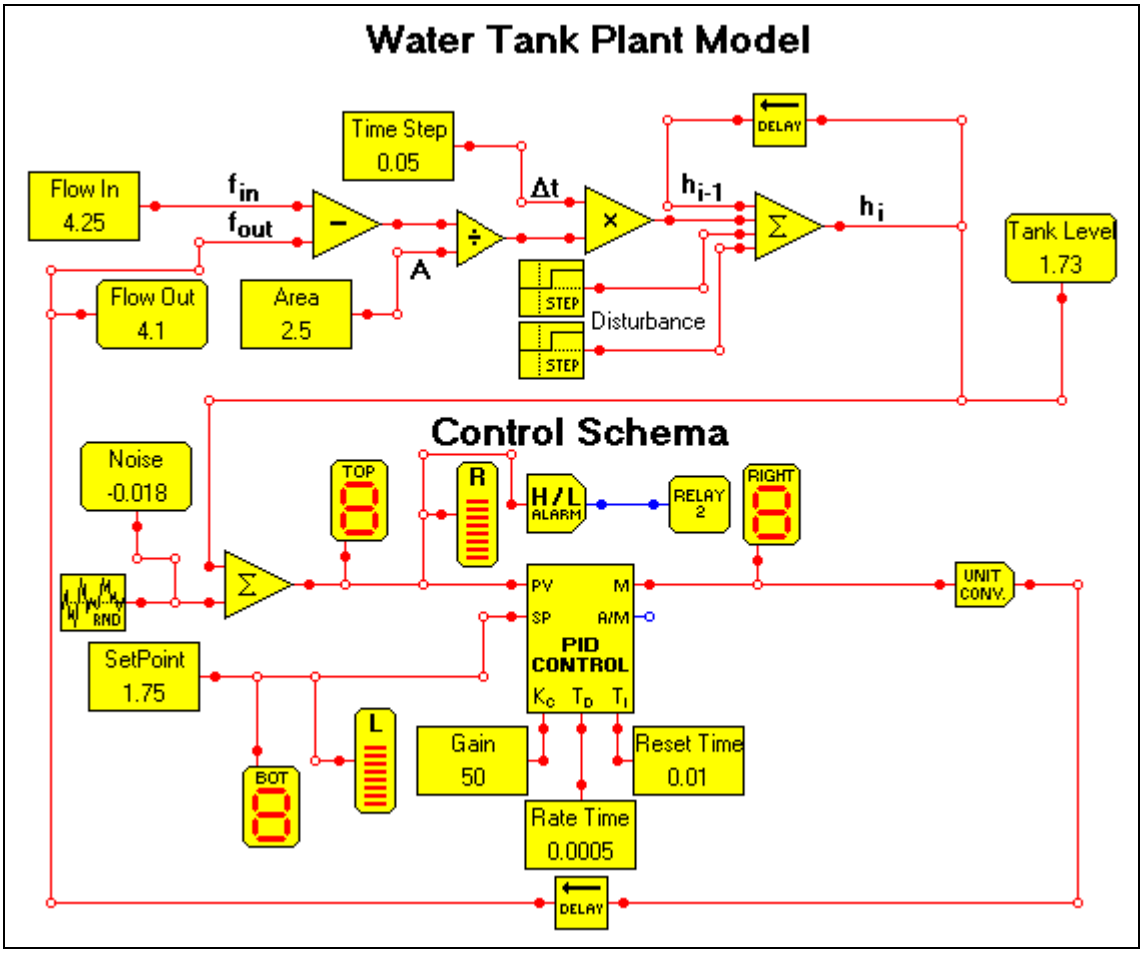


Figure 29. Revised simulation schema with signal generators to emulate effects of noise and external disturbances.

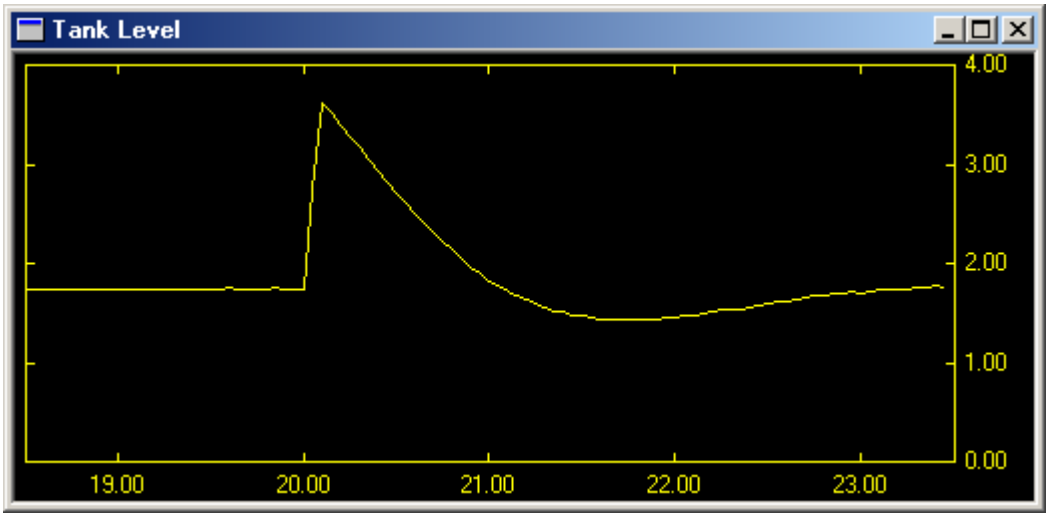


Figure 30. System response to external disturbances.

8.5 Simulating a Controller Network

In *Design Pad 2000 Professional*, you can simulate multiple schema files at the same time. If the schema files are assigned to a communication network, *Design Pad 2000* will also simulate the network traffic between the schemas. That is, a receiver operator in one schema will receive signals from broadcast operator in another.

To run a simulation of multiple control stations, open the schema document corresponding to each station and select the Start Simulation item from the Schema menu. You can also interact with the virtual front panel and/or virtual rear panel of each control station. To do so, select the Front Panel Window and/or Rear Panel Window commands from the View menu.

To stop the multi-station simulation, activate each simulating schema file and select the Stop Simulation item from the Schema menu.

9. Building Human-Machine Interfaces (HMI) with *Design Pad*

A significant benefit of a networked control system is the ease with which one can tap into the information that flows over the network bus. For instance, Human-Machine Interface (HMI) software can seamlessly link one or more computers to a *FAIRNET* controller network, thereby enabling a person to interact with the control stations from a remote location, to visualize the control processes in intuitive and insightful ways, and to monitor the results of real-time data analysis. You can link a *FAIRNET* network with any HMI package using Fairmount Automation's OPC Server software (the HMI software package must be capable of running as an OPC client). *Design Pad* also provides limited HMI functionality for direct access to a FAIRNET network.

9.1 Developing Design Pad HMI Documents

Design Pad 2000 Professional provides built-in functionality to create simple interfaces with *FAIRNET* networks. This release of Design Pad is not intended to be a full-featured HMI package. It includes only a limited number of user-interface elements to build an HMI screen. Its principal advantage over a full-featured third-party HMI software package is that it allows full replication of the FAC-2000 front panel.

You create a *Design Pad* HMI document in the same way that you create a schema document—by connecting operator blocks together (see section 4). And you associate an HMI document with a *FAIRNET* network in the same way that you associate a schema document with a *FAIRNET* network. (see section 6). All operators—including networking operators—available to you when you create a schema document are also available to you when you create an HMI document. Moreover, HMI documents can also use the signal generators and probe operators normally restricted to simulations.

In this section, we develop a simple HMI document to interact with the water tank level control schema described in section 2. Figure 31 shows the same schema of Figure 1 with broadcast operators added to enable information sharing over a *FAIRNET* network. (The "Set-Point" constant has also been enabled for networking.) Figure 31 shows a sample HMI document that receives broadcasted signals from the networked water tank schema. The intent of this sample HMI document is to fully replicate the control station front panel at a remote location. As such, it contains a receiver operator for each broadcast operator in the schema. The received signals are fed to display elements exactly as they are in the actual schema. For instance, in the schema the *Analog Input 1* signal representing the water level is fed to the *Top Numeric* and *Right Bargraph* displays; in the HMI document the receiver operator representing the water level is fed to the same two display elements.

In order for the sample *Design Pad* HMI (running on a personal computer) to receive broadcasted signals from the sample schema (running on a FAC-2000 control station) we must define a *FAIRNET* network in *Design Pad* and assign the two documents to that

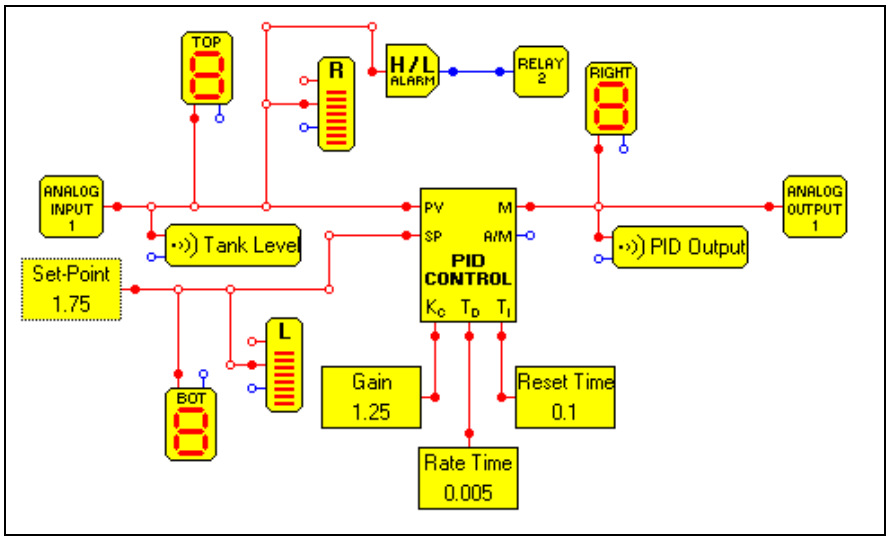


Figure 31. The water tank level control example of Figure 1 with broadcast operators added to enable information sharing over a FAIRNET network.

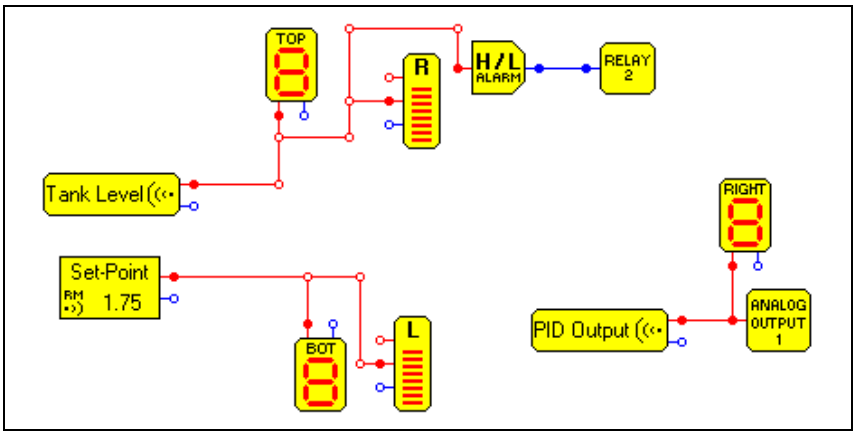


Figure 32. A sample HMI document used to remotely replicate the front-panel display of the control station executing the schema of Figure 31**.**

network. Figure 33 shows the properties for such a network: a multi-drop network named "Water Tank HMI".

To run the sample HMI, follow these steps:

1. Use a serial cable to connect the *FAC-2000* COM2 input to the RS-485 card (COM4) on your computer.
2. Download the schema of Figure 31 to the *FAC-2000* control station
3. In *Design Pad*, display the virtual front panel for the HMI document (activate the HMI document and select the Front Panel Window item from the View menu).

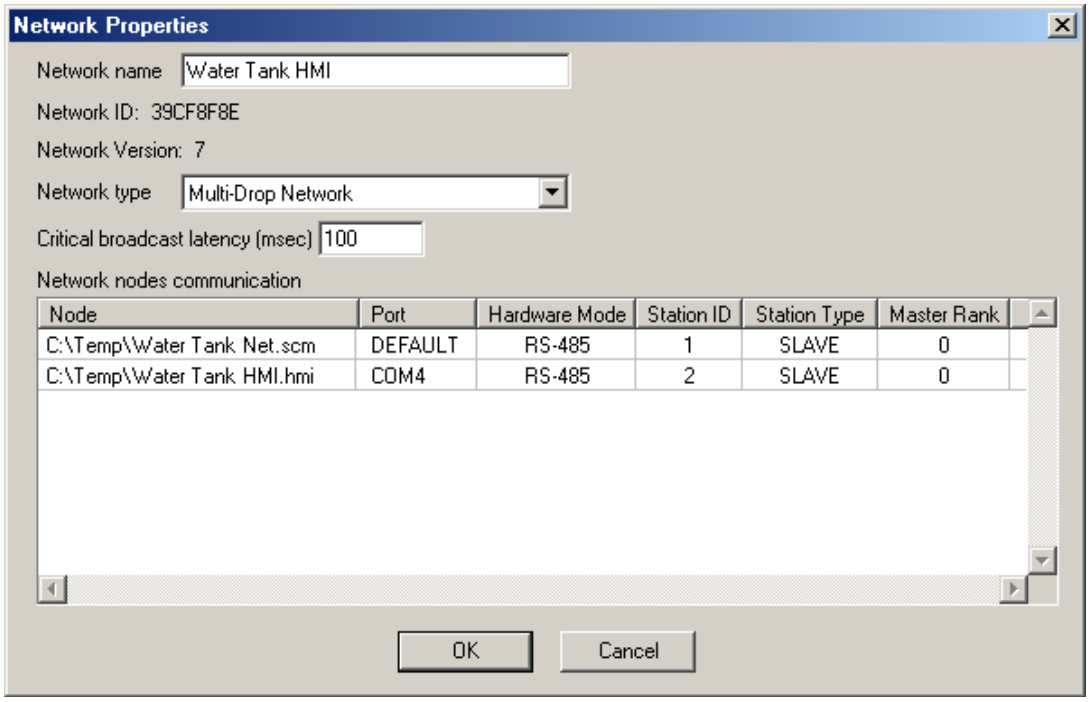


Figure 33. Properties of the *FAIRNET* network definition linking a schema document (running in a *FAC-2000* control station) and a Design Pad HMI document (running on a personal computer).

4. Press the Start Simulation button on the toolbar to execute the HMI

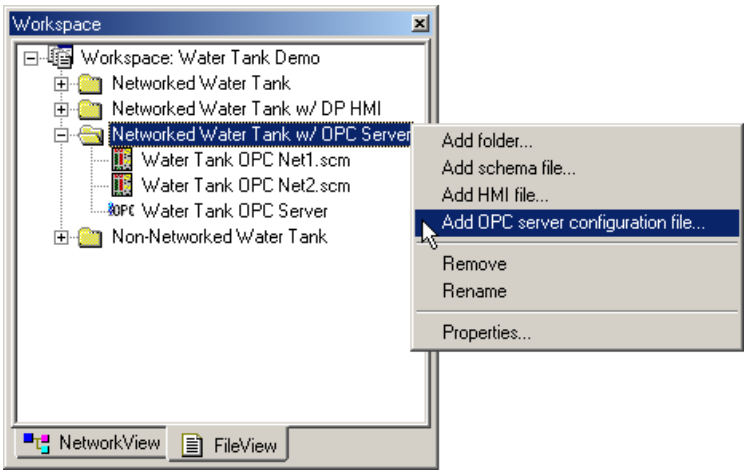
After following these four steps, the virtual front panel display in *Design Pad* should behave exactly as the real front panel does on the *FAC-2000* control station. It will show the same values on the text, numeric, and bargraph displays. And it will allow you to specify the process set-point just as you would from the control station. The only difference between the two is that the automatic/manual functionality built-into the PID operator of the control station will not be replicated in the computer HMI. (We could do so by replacing the single PID block with a combination of a *PID with External Feedback* operator and an *A/M Button* operator in the sample schema, and then adding a *Remote A/M Button* operator to the HMI document.)

Note: When a multi-schema simulation is executed, *Design Pad* emulates all the traffic on a *FAIRNET* network. When the HMI is executed, *Design Pad* is actually a participant (a node) in a *FAIRNET* network. There are situations where you may want to have the HMI interact with a real *FAIRNET* network and also interact with simulated schemas. To do so, you simply run a simulation for each schema or HMI that you want to execute. However, you must specify how each network operator (*Broadcast*, *Receiver*, network-enabled *Constant*, *Remote Constant*, network-enabled *A/M Button*, or *Remote A/M Button*) should behave: as part of a simulated network or as part of a real *FAIRNET* network. (An HMI document can have both.) You specify how a network operator should behave in its property dialog. (See the Operator Reference section for additional details.)

9.2 Configuring a *FAIRNET* OPC Server

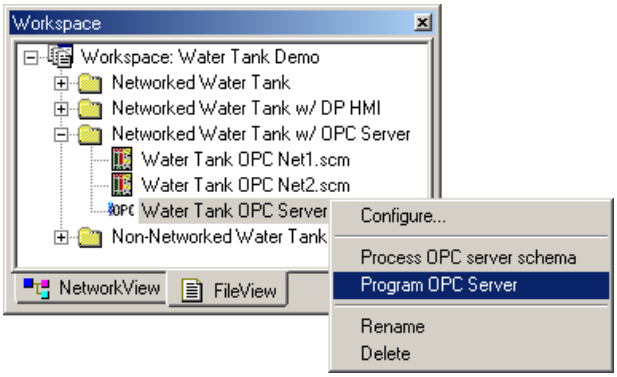
If you are already familiar with a third-party HMI software package, you may be more comfortable working with it to develop *FAIRNET* interfaces. Fairmount Automation provides an OPC Server package that acts as a gateway between the HMI package and *FAIRNET* networks. It allows any third-party HMI software package that can function as an OPC client to interact with a *FAC-2000* controller network.

Fairmount Automation's OPC Server software is a standalone package independent of *Design Pad*. However, you must use *Design Pad* to configure the OPC Server software. The configuration process involves creating an OPC Server node in the workspace, associating FAIRNET networks with the OPC Server, and declaring the signals that the OPC Server will broadcast on the network and that signals it will receive from the network.



To create a *FAIRNET* OPC Server, first select the *File View* tab of the workspace. Then, right-click on the root workspace node or on a workspace folder and select the Add OPC Server Configuration File item from the drop-down menu. *Design Pad* will create an OPC Server node with a generic name (e.g., "Server 1"). You can assign the node a name more relevant to your design by right-clicking on the node and selecting the Rename command from the drop-down menu.

To configure the OPC Server node, right-click on it and select the Configure command from the drop-down menu (or simply double-click on the node). *Design Pad* will present you with the OPC Server configuration dialog shown in Figure 34 to Figure 37. The dialog is divided into four sections: general, networks, broadcasts, and receivers. The specific entries for each section are described in the sections that follow.



After you complete the configuration process, you are ready to program the OPC Server software (*i.e.*, export to it the network configuration data). To do so, first launch the OPC Server. Then, right-click on the OPC Server node in workspace window and select the Program OPC Server item from the drop-down menu.

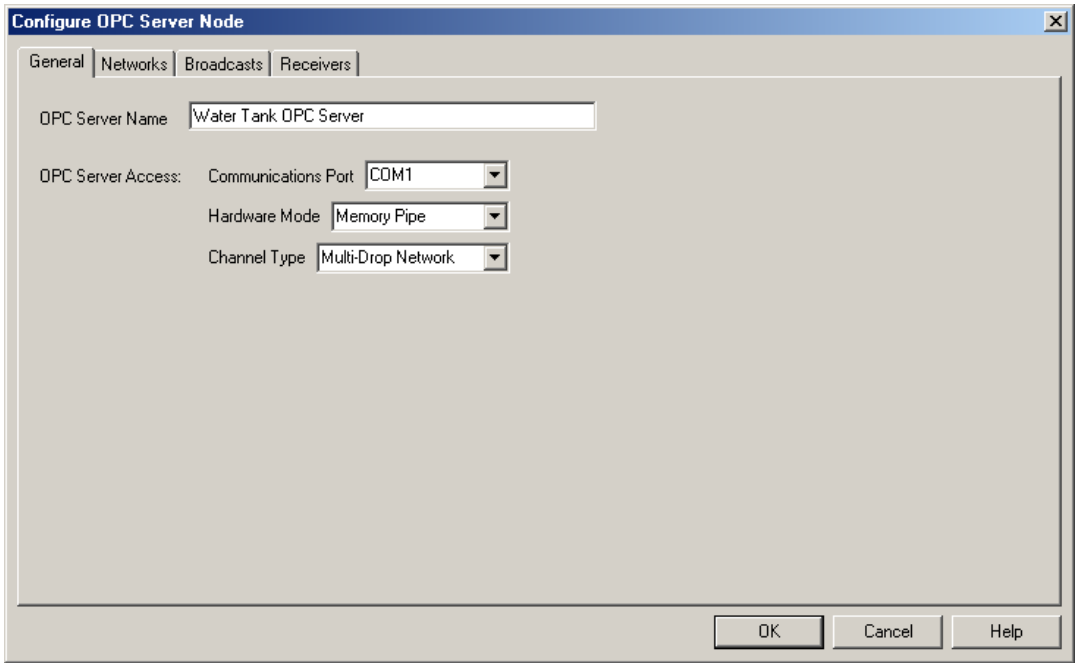


Figure 34. OPC Server Configuration dialog (general section).

9.2.1 OPC Server Configuration Dialog: General Section

The general section of the OPC Server Configuration dialog is shown in Figure 34. It includes a name that *Design Pad* uses to identify the OPC Server and additional parameters that determine how *Design Pad* interacts with the OPC Server software. This interaction takes place when *Design Pad* programs the OPC Server with the network configuration information. The programming process requires data sharing between the two applications. By default, this occurs over a memory pipe that *Design Pad* establishes between the two applications. Contact Fairmount Automation if you need to program the OPC Server using alternative methods not documented here.

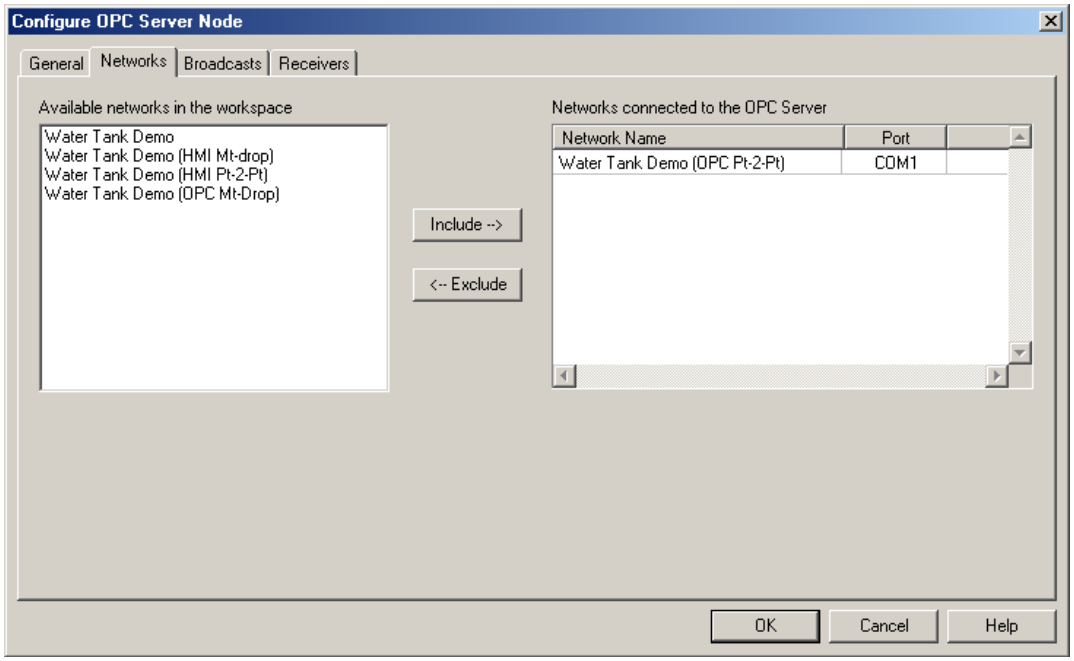


Figure 35. OPC Server Configuration dialog (networks section).

9.2.2 OPC Server Configuration Dialog: Networks Section

The networks section of the OPC Server Configuration dialog is shown in Figure 35. This section is used to specify the *FAIRNET* network(s) that the OPC Server will access (and which PC communication port(s) it will use to access them). The OPC Server package can serve as a gateway to multiple *FAIRNET* networks (the number of networks it can access is limited by the number of COM ports available on the PC.)

The first time you configure an OPC Server node, *Design Pad* lists all the available networks in the left list box. To provide the OPC Server access to the network signals on a particular network, select it and then press the *Include* button. *Design Pad* will move the network from the "available" list to the "connection" list. If you need to remove a network from the list of OPC Server connections, select it and press the *Exclude* button.

For each network in the "connection" list, you must specify a unique COM port that the OPC Server will use to access that network. (The OPC Server requires physically distinct serial connections between each *FAIRNET* network and PC COM port pair.)

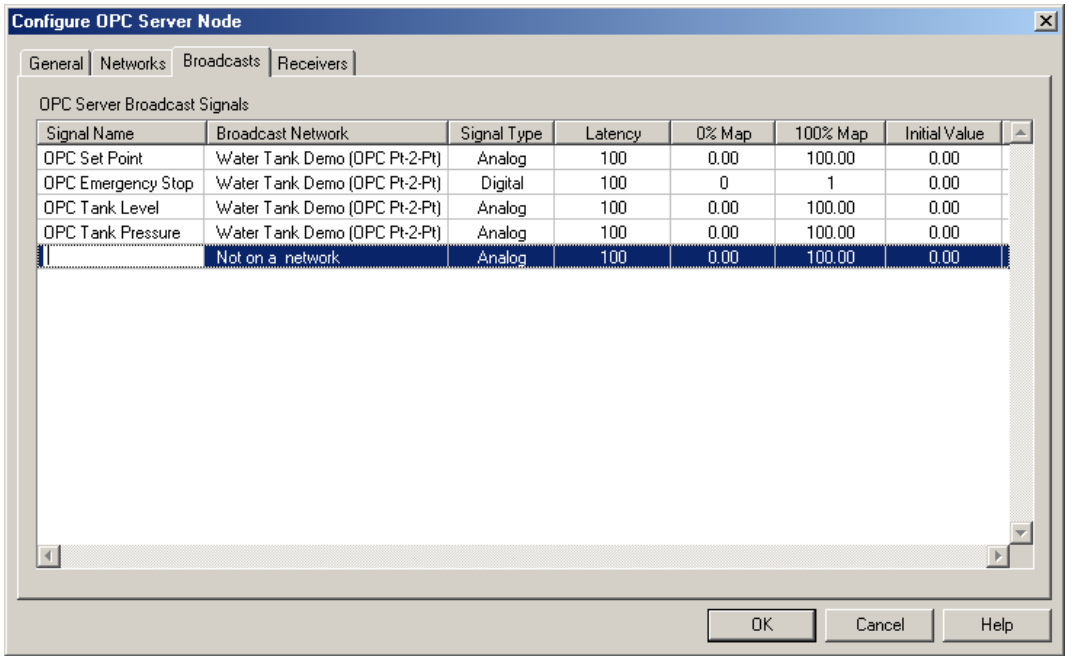


Figure 36. OPC Server Configuration dialog (broadcasts section).

9.2.3 OPC Server Configuration Dialog: Broadcasts Section

The broadcasts section of the OPC Server Configuration dialog is shown in Figure 36. In this section of the dialog, you can define signals that the OPC Server will broadcast over the network. The OPC Server broadcast signals are generated in the HMI (e.g., sliders or dials that a user can adjust.)

To define an OPC Server broadcast signal, you must completely fill-in a row in the signal list box—the same parameters that you specify for a broadcast operator:

Signal Name. A unique character-string that Design Pad uses to identify the broadcast signal (used this name to associate receiver operators in other schema files with this broadcast).

Broadcast Network. Indicates the *FAIRNET* network to broadcast this signal over. Clearly, you can only broadcast signals over a network that the OPC Server is associated with. (You define which networks the OPC Server has access to in the networks section of the dialog—see the previous section.)

Signal Type. Indicates the type of broadcast signal: Analog, Digital, or Mixed. (Refer to section 6.1.1 for a description of each.)

Latency. The latency time (in milliseconds) determines how often the broadcast signal will be transmitted. (Note that the OPC Server may not broadcast critical messages.)

0% Map. Specifies the lower limit of the broadcast signal range.

100% Map. Specifies the upper limit of the broadcast signal range.

Initial Value. Specifies the value that the broadcast signal should assume on startup. (If associated receiver operators do not receive the broadcast, their output will remain at this initial value.)

If you would like to delete a signal from the broadcast list, right-click on it and select the Remove Broadcast Signal item from the drop down menu.

9.2.4 OPC Server Configuration Dialog: Receivers Section

The receivers section of the OPC Server Configuration dialog is shown in Figure 37. In this section of the dialog, you define signals that the OPC Server will receive over the network. The OPC Server will provide these signals to the HMI package for display (*e.g.*, on gauges and indicators.)

To define an OPC Server receiver signal, you must completely fill-in a row in the signal list box—the same parameters that you specify for a receiver operator:

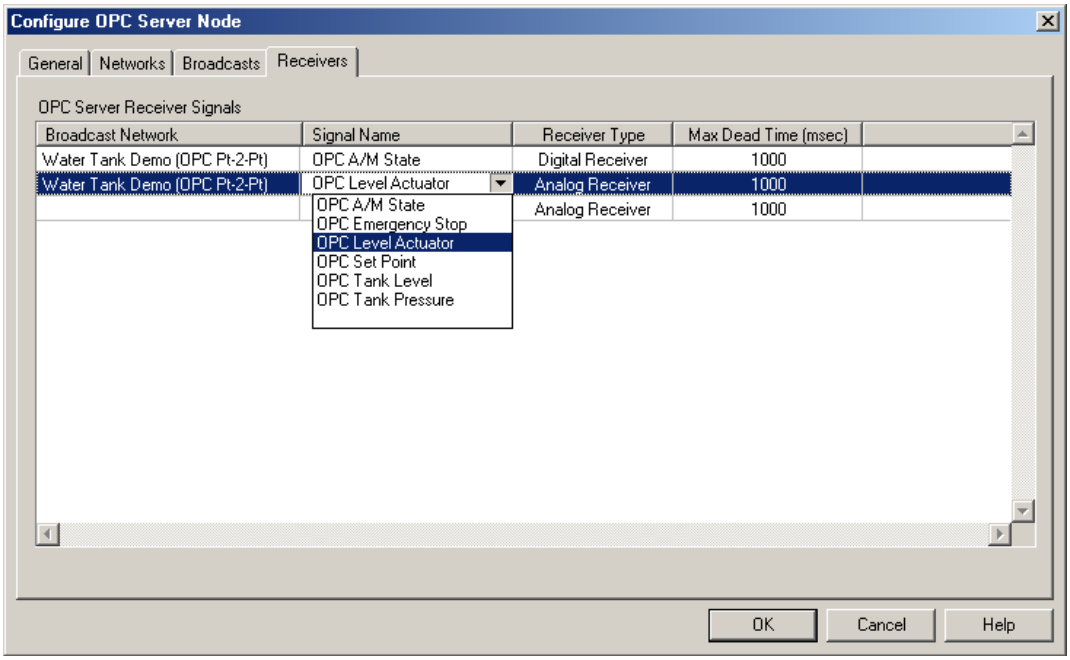


Figure 37. OPC Server Configuration dialog (receivers section).

Broadcast Network. Indicates the *FAIRNET* network to receive this signal over. Clearly, you can only receive signals over a network that the OPC Server is associated with. (You define which networks the OPC Server has access to in the networks section of the dialog—see the previous section.)

Signal Name. After you select the broadcast network, Design Pad will populate the signal name drop-down combo will the names of all the broadcast signals available on that network.

Receiver Type. Design Pad supports several types of network receivers: Analog Receivers, Digital Receivers, Mixed Receivers, Remote Constants, and Remote A/M Buttons. (Refer to section 6.1.2 for a description of each.)

Maximum Broadcast Dead Time. The period of time (in milliseconds) that may elapse before the receiver operator indicates that it has not received a broadcast signal (switches output *b* to *high*). Output *b* remains *low* while the operator actively receives the broadcasted signal.

10. CUSTOMIZING *DESIGN PAD*

You can customize some of functionality in Design Pad in its Preferences dialog (see Figure 30). To access this dialog, select the Preferences item from the File menu. The Design Pad features that you can customize are described in the following sections.

10.1 *Design Pad* Preferences: General Section

The general section of the *Design Pad* Preferences dialog includes the following parameters:

Reload last workspace at startup. If this item is selected, *Design Pad* will automatically open the workspace that was active during a previous Design Pad session. If it is not selected, *Design Pad* will not open any documents at startup.

Recent File List Size. Design Pad maintains a list of recently opened files (schema documents or HMI documents) in its file menu. This parameter determines how many recent files *Design Pad* should track. The maximum list size is 16 files.

Recent Workspace List Size. Design Pad maintains a list of recently opened workspace documents in its file menu. This parameter determines how many recent workspaces Design Pad should track. The maximum list size is 16 workspaces.

Schema Export Button on Toolbar. When you export a schema to a FAC-2000 control station, you can enable or disable schema importing from that station (see sections 7.2 and 7.3). To export a schema you can use the Controller menu or the toolbar button  . This preference item defines what the schema export button does: either download with import enabled, or download with import enabled.

Exporting Unopened Schemas from Workspace. When *Design Pad* exports a schema to a *FAC-2000* control station it records the download date in the schema file. (You can view the last download date in the Schema Properties dialog—see section 4.10.) If you use the workspace window to export an unopened schema document, *Design Pad* will prompt you to record to download date. (If you answer affirmatively, *Design Pad* may save the file using a more recent file format!) This property allows you to change Design Pad's default behavior. You can set it to never record the download date, to always record the download date, or to continue to prompt you when necessary.

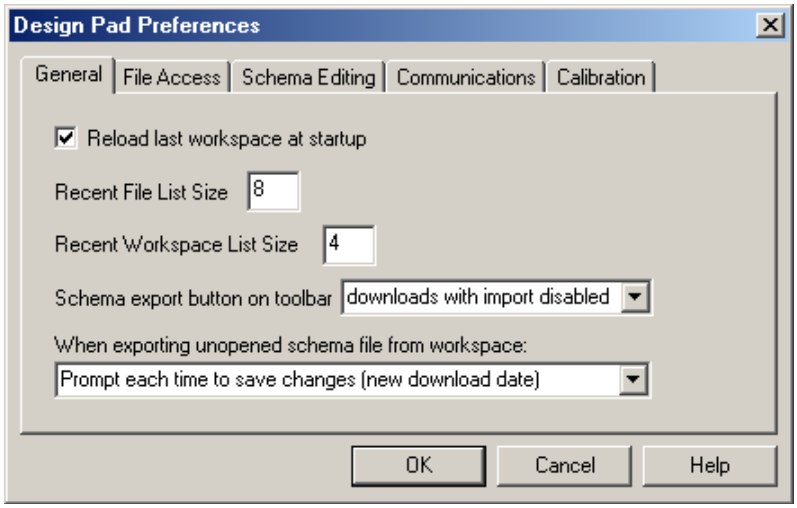


Figure 38. Design Pad Preferences (General Tab).

10.2 *Design Pad* Preferences: File Access Section

The file access section of the *Design Pad* Preferences dialog includes the following parameters:

Create backup files. If this item is selected, *Design Pad* will automatically create a backup (`.bak`) file whenever it saves a document (workspace, schema, or HMI). (The backup file name is the same as the original, but the file extension is `.bak`.

Auto-Recovery Enabled. On startup, *Design Pad* checks to see if it exited abnormally during a previous session, possibly resulting in data loss. If this parameter is selected, *Design Pad* will prompt you to recover lost work if an abnormal exit is detected.

Auto-Recovery File Save Frequency. If *Design Pad* exits abnormally, any unsaved changes you may have made to open documents may be lost. To prevent this, *Design Pad* regularly saves copies of open documents (while auto-recovery is enabled). This parameter specifies how often (in minutes) *Design Pad* updates its recovery files.

Default Document Directory. This item specifies the directory that Design Pad should reference whenever it displays the File Open or File Save dialogs.

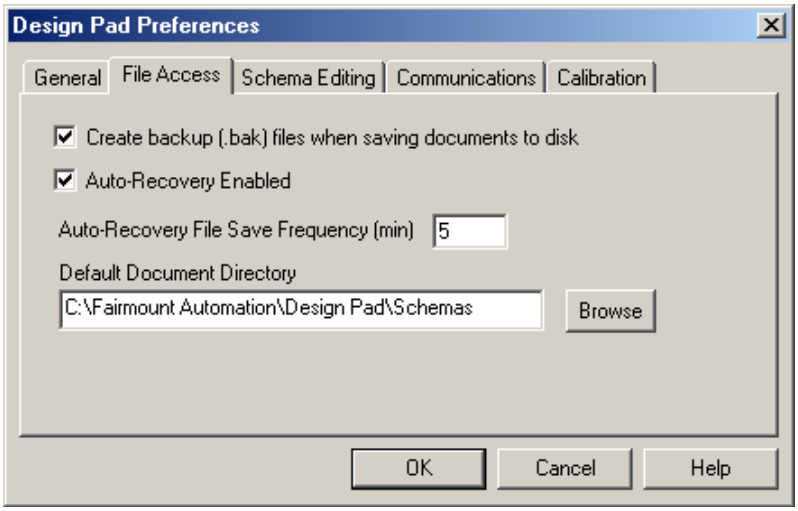


Figure 39. Design Pad Preferences dialog (File Access tab).

10.3 *Design Pad* Preferences: Schema Editing Section

The schema editing section of the *Design Pad* Preferences dialog includes the following parameters:

View Grid. If this item is selected, *Design Pad* will display a grid when it first opens a schema or HMI document. You can turn the grid on or off from the Edit menu (see section 4.4).

View Signal "Hot-Spots". If this item is selected, *Design Pad* will display the signal "hot-spots" when it first opens a schema or HMI document. You can toggle the display status of the signal "hot-spots" from the Edit menu (see section 4.4).

View Operator "Hot-Spots". If this item is selected, *Design Pad* will display the operator "hot-spots" when it first opens a schema or HMI document. You can toggle the display status of operator "hot-spots" from the Edit menu (see section 4.4).

Perpendicular Signal Editing Mode. If this item is selected, Design Pad will turn on perpendicular signal editing (PSE) mode when it first opens a schema or HMI document. You can turn the PSE mode on or off from the Edit menu (see section 4.4).

Orthogonal Editing Mode. If this item is selected, Design Pad will turn on orthogonal editing mode when it first opens a schema or HMI document. You can turn orthogonal mode on or off from the Edit menu (see section 4.4).

Mouse Displacement Required to Drag Objects. This parameter specifies the number of pixels you must displace an object (signal, operator, text box, *etc.*) in a schema or HMI document, before it begins to follow the mouse cursor.

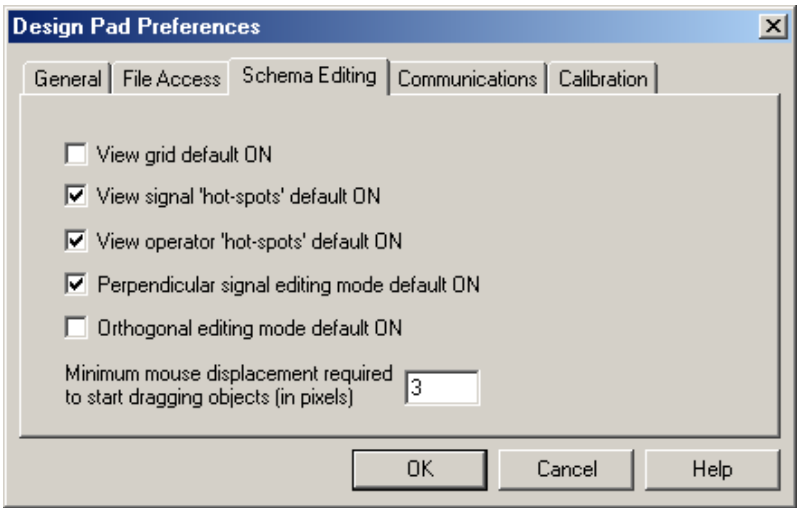


Figure 40. Design Pad Preferences dialog (Schema Editing tab).

10.4 *Design Pad* Preferences: Communications Section

The communications section of the *Design Pad* Preferences dialog includes the following parameters:

Default Communications Port. Indicates the default PC COM port that Design Pad should use when communicating with a *FAC-2000* control station.

Default Communications Hardware Mode. Indicates the communications protocol (RS-232 or RS-485) that Design Pad should use when communicating with a FAC-2000 control station..

Controller Communications Mode. *Design Pad* supports two methods of communicating with FAC-2000 control stations: single-threaded mode and multi-threaded mode. *FAC-2000* controllers executing first-generation executable code versions (version 1.21 and below) communicate in single-threaded mode. *FAC-2000* controllers executing second-generation executable code versions (version 2.00 and above) communicate in multi-threaded mode. Prior versions of Design Pad cannot communicate in multi-threaded mode.

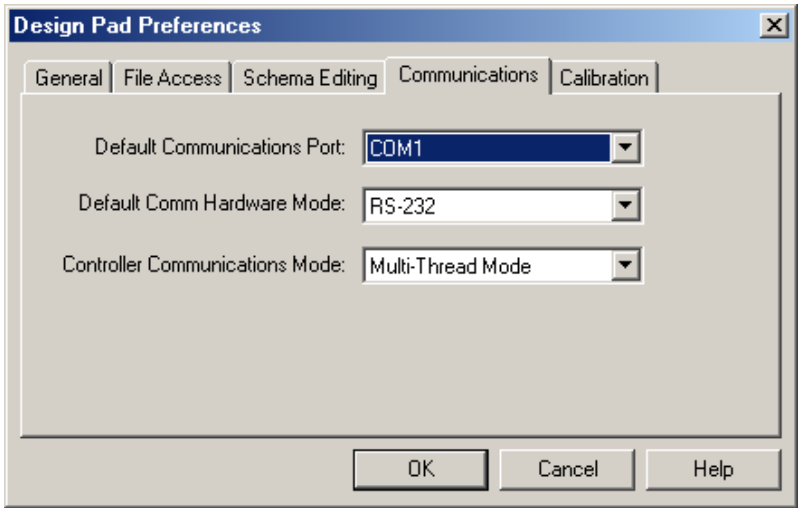


Figure 41. Design Pad Preferences dialog (Communications tab).

10.5 Calibration Tab

Default Power Supply COM Port. Indicates the PC COM port that Design Pad should use during the calibration process to manipulate the power supply settings.

Default Meter/Switch Unit COM Port. Indicates the PC COM port that Design Pad should use during the calibration process to manipulate the electronic meter or switch unit.

Maximum Calibration Error. Indicates the maximum allowable calibration error (in mA) for the analog inputs and analog outputs on the *FAC-2000* controller .

Default Path for Calibration File Results. After Design Pad completes the calibration process, it generates a file summarizing the calibration results for each analog channel. This parameter specifies the directory where the calibration results should be saved.

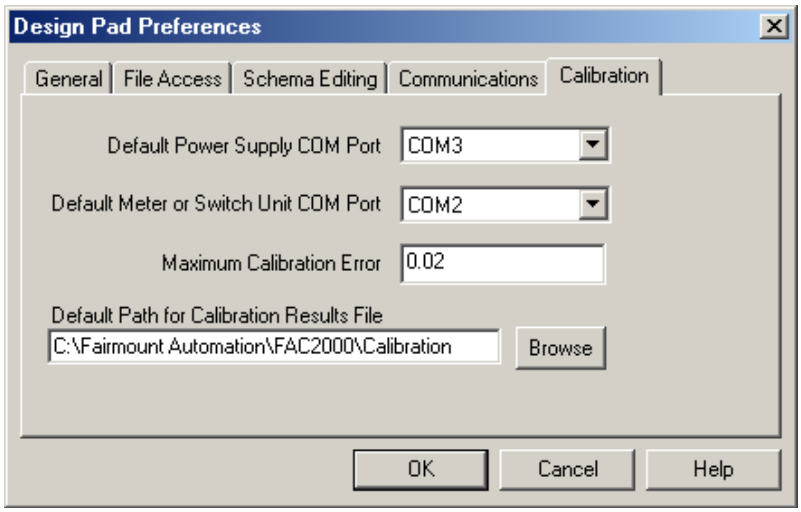


Figure 42. Design Pad Preferences dialog (Calibration).

11. OPERATOR REFERENCE

This section provides a detailed description of each operator available in the *FAC-2000*'s operator set.

The documentation for each operator includes a functional description, a property list, comments (if applicable), a 'see also' section, and the *Design Pad* menu location. The functional description explains what the operator does and how it works. The property list describes in detail the user-defined parameters associated with the operator. The comments section indicates any exceptional information particular to the operator. The 'see also' section refers the reader to related operators that may be of interest.

The following operators are new in Design Pad 2000:

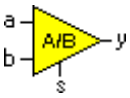
Absolute Value (section 11.2, page 83)
Broadcast (section 11.14, page 103)
Receiver (section 11.53, page 170)
Remote Auto/Manual Button (section 11.55, page 173)
Remote Analog Constant (section 11.56, page179)
Remote Digital Constant (section 11.57, page 182)

The following operators have additional computational functionality in Design Pad 2000:

Auto/Manual Button (section 11.9, page 92)
Analog Constant (section11.22, page 119)
Digital Constant (section 11.23, page 121)
Limiter (section 11.35, page 135)
Probe (section 11.50, page 162)
Ramp Profile (section 11.51, page 164)
Rate Limiter (section 11.52, page 168)

The following operators have additional properties in Design Pad 2000

Alpha-numeric Display (section 11.4, page 85)
Bargraph Display (section 11.11, page 98)
Numeric Display (section 11.43, page 143)



11.1 A/B SWITCH

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The output of the *A/B Switch* block is defined by

$$y = \begin{cases} a & \text{if} \quad s = 0 \\ b & \text{if} \quad s = 1 \end{cases}$$

where y is the output, s is a digital selector input, and where a and b are analog inputs.

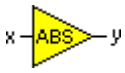
User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also:

Multiplexer, Demultiplexer



11.2 ABSOLUTE VALUE

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *absolute value* operator is

$$y = |x|$$

where y is the output, and x is the input.

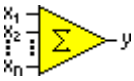
User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also:

Addition, Subtraction, Multiplication, Division, Power, Unit Conversion



11.3 ADDITION

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *addition* operator is the sum of its inputs,

$$y = \sum_{i=1}^{n} x_i$$

where y is the output, x_i (i = 1,…,n) are the inputs, and n is the number of inputs. *Design Pad* assumes that $x_i \equiv 0$, if input pin i is not connected.

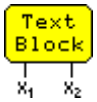
User-Defined Properties:

Object Name	A string label that identifies the operator.
Number of Inputs	The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See Also:

Subtraction, Absolute Value, Multiplication, Division, Power, Unit Conversion



11.4 ALPHANUMERIC DISPLAY

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *alphanumeric display* operator can be used to show text messages on the alphanumeric displays located on the front panel of the *FAC-2000* controller. The operator has two inputs: (x_1) a boolean input that activates it, and (x_2) an analog input used to embed numeric values in the text message.

When the operator is activated (input x_1 is *high*), a user-defined text message will be scrolled from left to right onto the display. The last eight characters of the message will remain on the screen. The duration of the scrolling period (the *entry time*), and the time the message remains on the screen (the *hold time*), are properties of the operator.

The *FAC-2000* controller is equipped with three alphanumeric display lines. The target line where the message is to be displayed is a property of the *alphanumeric display* operator. Other properties of this operator include the message priority level (1-3) and the number of times the message is to be repeated (0-255). Each target line maintains a queue of messages to display. Each queue is sorted based on the priority levels of the messages it contains. The message at the top of the queue is displayed first. After the *hold time* elapses, the message is removed from the queue and the next message is displayed. If a message is to be repeated, it is re-inserted into the queue, immediately following all messages of equal priority. A message with the *number repeat* property set to 255 will not be removed from the queue while input x_1 is *high* (it will be displayed indefinitely).

The text message to be displayed is formed by combining two character strings (operator properties *text before number* and *text after number*) with the numeric value from analog input x_2. In other words, the text message is a concatenation of

text before number + number from input x_2 + *text after number*

If analog input x_2 is not used (*i.e.*, if no connection is made to it) the text message that will be displayed is the character string defined in the *text before number* property.

User-Defined Properties:

Text Before Number — Text message (or part of text message) to be displayed on the front panel when input signal x_1 is *high* (1). If input signal x_2 is connected, the text message to be displayed is a combination of this property, the numeric value x_2, and the text after number property. Otherwise, if input x_2 is not connected, the property defines the full text to be displayed.

Text After Number The part of text message to be displayed that follows the numeric value x_2.

Number of Significant Digits to Display The number of significant digits to use when displaying the numeric value x_2. This property is ignored if input x_2 is not connected.

Target Line The *FAC-2000* controller has three alphanumeric display lines. This property specifies which display line the message should be written to (top, middle, or bottom).

Justification The justification of the characters that remain on the screen after the scrolling period: left, right, or centered. (This property applies only to text messages with less than eight alphanumeric characters.)

Presentation The style used to present the message: *Steady*, *Flash*, or *Fade*.

Number Repeat The number of times to repeat the message when the message is activated (input x_1 is *high*). If this property is set to 0, the message will be displayed once; if it is set to 1, the message will be displayed twice; if it is set to 2, the message will be displayed three times; and so on. If this property is set to 255, the message will be remain on the display as long as the x_1 is *high*.

Entry Time The number of milliseconds required to scroll the entire message across the display.

Hold Time The number of milliseconds to hold the message on the display after the scrolling period is completed.

Priority The message priority level: an integer in the range [1,3]. Messages of priority 1 have the lowest priority; messages of priority 3 have the highest priority.

Brightness A brightness setting for the LED digits in the range [0-15], with 15 indicating maximum brightness.

Display Enable Input If input x_1 is not connected *Design Pad* will issue a warning upon processing. This property determines if the input pin is visible in the schema diagram. If you do not check this property, the pin will not be displayed and *Design Pad* will not show a warning message upon processing the schema. In either case, if the pin is not connected, the operator behaves as if it is connected and x_1 = *high (i.e.,* it displays the message on the front panel).

Display Analog Input If input x_2 is not used *Design Pad* will issue a warning upon processing the schema. You can suppress the warning message by making the input pin invisible. When this property is checked the pin is visible; otherwise it is invisible.

Comments:

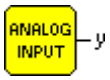
- Multiple alphanumeric display operators for the same target line may be active simultaneously. Each active message will be displayed only if all other active messages of higher priority (on the same target line) have been removed from the message queue.

- If a low-priority message is placed in the message queue, and the message queue contains a higher priority message with the '*Number Repeat*' set to 255, the low-priority message may never be displayed. (The higher priority message will be displayed until it is deactivated.)

- Text messages may not be displayed while the device is in *Set Menu* or *A/M Menu* modes (*e.g.,* while a plant operator is adjusting the process set point or manual output value). While in these modes, the *FAC-2000* device may use the alphanumeric displays to display parameter information.

See Also:

Bargraph Display, Numeric Display



11.5 ANALOG INPUT

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *analog input* operator provides access to one of the three *FAC-2000* analog inputs. The operator maps a 4-20mA input signal into a user-defined 0-100% representation, where 4mA is equivalent to the 0% value and 20mA is equivalent to the 100% value. The mapping is

$$y = x_{0\%} + \frac{\left(20.83333x - 4\right)\left(x_{100\%} - x_{0\%}\right)}{16}$$

where y is the operator's representation of the input signal, x is the 4-20mA signal, $x_{0\%}$ is the user-defined 0% value equivalent to 4mA, and $x_{100\%}$ is the user-defined 100% value equivalent to 20mA. (Note: the maximum input current recognized by the device is 20.83333mA.)

User-Defined Properties:

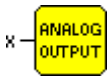
Object Name	A string label that identifies the operator.
0% Value	The operator representation of a 4mA signal.
100% Value	The operator representation of a 20mA signal.

Comments:

- The 0% value may be greater than the 100% value.
- If the input signal is outside of the 4-20mA range, the operator representation may be outside of the 0-100% range.

See also:

Digital Input, Universal Digital Input, Analog Output, Digital Output, Relay



11.6 ANALOG OUTPUT

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *analog output* operator provides access to one of the two *FAC-2000* analog outputs. The operator maps a user-defined 0-100% value into a 4-20mA output signal, where 4mA is equivalent to the 0% value and 20mA is equivalent to the 100% value. The mapping is:

$$OUT = 4 + 16\frac{x - x_{0\%}}{x_{100\%} - x_{0\%}}$$

where *OUT* is the 4-20mA output signal, x is a schema signal nominally in the 0-100% range, $x_{0\%}$ is the user-defined 0% value equivalent to 4mA, and $x_{100\%}$ is the user-defined 100% value equivalent to 20mA.

User-Defined Properties:

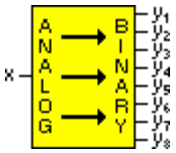
Object Name	A string label that identifies the operator.
Initial Value	The output value on startup (at time t=0).
0% Value	The operator representation of a 4mA signal.
100% Value	The operator representation of a 20mA signal.

Comments:

- The 0% value may be greater than the 100% value.
- If the schema signal x is outside of the 0-100% range, the output signal may be outside of the 4-20mA range. The *FAC-2000* output range is 0-20.5mA.

See also:

Digital Output, Analog Input, Digital Input, Universal Digital Input, Relay



11.7 ANALOG to BINARY CONVERTER

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The *Analog to Binary Converter* block accepts an analog input x, and computes its binary representation y_i (i=1,2,…,8). Two of the operator properties, x_{min} and x_{max}, define the range of the input signal. They are used to map the input signal into an integer value between 0 and 255. The mapping is

$$Y_{10} = \begin{cases} 0 & \text{if} \quad x < x_{min} \\ \text{round}\left(\dfrac{255(x - x_{min})}{x_{max} - x_{min}} \right) & \text{if} \quad x_{min} \le x \le x_{max} \\ 255 & \text{if} \quad x > x_{max} \end{cases}$$

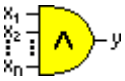
where Y_{10} is the base-10 output value to be mapped into the eight digital signals, y_i (i=1,…,8). The round() function above rounds its argument to the nearest integer.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Minimum (0) Value	The input value corresponding to a base-10 output value of 0 (x_{min}).
Maximum (255) Value	The input value corresponding to a base-10 output value of 255 (x_{max}).

Comments: None.

See Also: Binary to Analog Converter, Multiplexer, Demultiplexer, A/D Conversion, D/A Conversion



11.8 AND-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the *AND-Gate* operator is

$$y = x_1 \wedge x_2 \wedge \cdots \wedge x_n ,$$

where y is the output, x_i (i = 1,…,n) are the inputs, and n is the number of inputs. In other words, the output state is *high* (1) if all inputs are *high*. If any input is *low* (0), the output is *low*. Below is the truth table for a 3-input AND-gate:

x_1	x_2	x_3	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Design Pad assumes that $x_i \equiv 1$, if input pin i is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Number of Inputs The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See Also:

NAND-Gate, OR-Gate, NOR-Gate, NOT-Gate, XOR-Gate, XNOR-Gate, RS Flip-Flop

11.9 AUTO/MANUAL (A/M) BUTTON

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *Auto/Manual Button* operator provides access to the A/M button on the *FAC-2000* faceplate. It allows the user to set the device (or a subset of the active schema) into automatic or manual mode. When in manual mode, the operator output y is set by the left-right arrow keys on the *FAC-2000* front panel. When in automatic mode, the operator output is equal to its primary input, *i.e.*, $y = x$. The boolean output b indicates the mode: it is *high* (1) when the operator is in manual mode; it is *low* (0) when the operator is in auto mode.

When the operator transitions from automatic mode to manual mode, the output remains at the transition value until modified from the front panel.

When the operator switches from manual mode to automatic mode, the transition may be abrupt or smooth (bumpless). If the bumpless transfer property is not selected, the output will switch instantly from the manual output to the automatic output. When bumpless transfer property is checked, the transition from manual to automatic is guaranteed to be smooth. The nature of the transition (*i.e.*, the transfer profile) can be either exponential or linear.

Let S_{am} be the difference between the manual output and the automatic output at the instant the operator switches from manual to automatic, $t=t_{am}$. Under an exponential transfer profile, the operator output will be (approximately)

$$y(t) = x(t) + S_{am} e^{-60(t-t_{am})/r} \quad \forall t \geq t_{am}$$

where r is a time constant expressed in minutes. Under a linear transfer profile, the operator output will be (approximately)

$$y(t) = \begin{cases} x(t) + \max\left[0, S_{am} - r\left(t - t_{am}\right)\right] & \text{if} \quad S_{am} > 0 \\ x(t) + \min\left[0, S_{am} + r\left(t - t_{am}\right)\right] & \text{if} \quad S_{am} < 0 \end{cases}$$

where r is a linear decay rate expressed in (input) units/minute.

Generally, a person will use the keypad buttons to switch a control station from operating in automatic mode to operating in manual mode (and vice versa). You would also typically rely on a person to make manual output adjustments. However, under certain conditions you may wish to automatically switch the A/M operator from one mode to another—and automatically adjust the "manual" output—without user involvement. The A/M operator has two secondary inputs that allow you to do just that. You can use digital input x_{so} to automatically set the auto/manual operating mode. And you can use analog input x_{mo} to automatically set the "manual" output. These two inputs are edge-triggered, meaning that

they override the keypad buttons and affect the operator output only when their value changes. While these inputs remain fixed, they do not affect the outputs.

When the operator is in manual mode and input x_{so} switches from 1 (manual) to 0 (auto), the operator will switch into automatic mode and its output $y(t)$ will (over time) match its input $x(t)$. (If the bumpless transfer feature is enabled, the output will vary according to the equations above.) If the operator is in automatic mode and input x_{so} switches from 1 (manual) to 0 (auto), then it will remain in automatic mode.

When the operator is in automatic mode and input x_{so} switches from 0 (auto) to 1 (manual) at time t_o, the operator will switch into manual mode and its output $y(t)$ will move toward the value of input x_{mo} at the time of the switch. The transfer of the output—from $y = y(t_o)$ to $y = x_{mo}(t_o)$—occurs in increments and is rate-limited by the *Schema Key Repeat Time*, Δ_{KRAM} (see section 4.9). The output will vary as if a user presses the A/M key to switch to manual and then presses—and holds down—one of the arrow keys until the output reaches the desired value, $x_{mo}(t_o)$. Mathematically, we can express this as:

$$y(t) = \begin{cases} x_{mo}(t_o) & \text{if} \quad t > t_o + \Delta_{KRAM}\dfrac{x_{mo}(t_o) - y(t_o)}{M} \\ y(t_o) + M\dfrac{t - t_o}{\Delta_{KRAM}} & \text{if} \quad t_o \geq t \leq t_o + \Delta_{KRAM}\dfrac{x_{mo}(t_o) - y(t_o)}{M} \end{cases}$$

Whenever the manual-override input x_{mo} changes value, the output will move toward that value according to the equation above. (Of course, this occurs only while the operator is in manual mode; otherwise, the x_{mo} input is ignored). In order for the manual input x_{mo} to affect the output, its value must change. Moreover, the magnitude of the change must exceed the *Manual Input Activation Threshold* A_T over a period of time called the *Manual Input Activation Time* T. In other words,

$$|x_{mo}(t) - x_{mo}(t_o)| \geq A_T \quad \forall t \in (t_o, T]$$

must be satisfied in for the manual output to begin moving toward $x_{mo}(t_o)$.

Do the A/M, LEFT ARROW, and RIGHT ARROW keys on the *FAC-2000* faceplate have precedence over the inputs x_{so} and x_{mo}? The answer is that the operator will follow its most recent command. If it is commanded by its inputs to automatically adjust its manual output value, it will do so. If in the process of automatically adjusting the manual output value a person presses one of the arrow keys, the A/M button operator will follow the key commands. It will not resume its automatic adjustments unless input x_{mo} changes value—remember that this input is edge-triggered!

In *Design Pad 2000* the A/M operator doubles as a special networking operator designed to synchronize the operation of multiple A/M stations. You can enable the A/M operator to broadcast its outputs over a controller network and to receive information from one or more Remote A/M operators (see section 6.1.4) from other stations on the network. Together, the network-enabled A/M operator and Remote A/M operators offer synchronized auto/manual functionality between multiple stations. For example, consider the case where one station contains a network-enabled A/M operator and two other stations on the same controller network each contain a corresponding remote A/M operator. If you switch any one of the three stations from automatic mode to manual mode, the other two stations will also switch to manual operation. If you then adjust the manual output from any one of the three stations, the other two will match that manual output. If you switch any one of the stations back into automatic mode, the other two stations will also switch into automatic mode. When the stations operate in automatic mode, the output signal of the remote A/M operators will follow the output of the A/M operator.

In order to associate an A/M operator in one schema to a remote A/M operator in another, you must match their signal name property. If you want to associate multiple remote A/M operators with an A/M operator, then you set the signal name property of all operators to the same case-sensitive character string.

As with other networking operators, in addition to setting the signal name property, you must also specify if the broadcast-signal is critical and the type of network that this operator is intended for (RS-232 or RS-485).

User-Defined Properties:

Signal Name	A string label that identifies the operator. When networking is enabled, this character string is used to associate (synchronize) an A/M operator in one schema with remote A/M operator in other schemas.
Initial State	The operator mode (*auto* or *manual*) on startup (at time t=0).
Initial Value	The analog output value on startup (at time t=0). Applicable only if the initial state is set to manual.
Minimum	The minimum analog output value. Applies to both auto and manual modes.
Maximum	The maximum analog output value. Applies to both auto and manual modes.
Increment Magnitude	The magnitude of manual output increments when in manual mode. The manual output will be incremented or decremented by this parameter when (i) a person presses the LEFT ARROW or RIGHT ARROW buttons on the *FAC-2000* front panel (ii) input x_{mo} changes value (see manual input activation threshold property and manual input activation hysteresis property).

Bumpless Transfer When the operator switches from manual mode to automatic mode, the output signal may differ from input *x*. If this property is not checked, there may be a discontinuity in the output. When bumpless transfer is checked, the transition from manual to automatic is guaranteed to be smooth.

Transfer Profile When the bumpless transfer property is set, the output will transition smoothly from manual mode to automatic mode. This property characterizes the nature of the transition: exponential or linear. When the exponential profile is selected, the difference between the manual output and automatic output will decay exponentially. When the linear profile is selected, the difference will decay linearly.

Transfer Rate The rate at which the difference between manual output and automatic output will decay when the bumpless transfer property is set. When the exponential profile is selected, this parameter defines the exponential decay rate expressed in minutes. When the linear profile is selected, this parameter defines the linear decay rate in input units/minute.

Use Alphanumerics When this property is selected, the output value will be presented on the alphanumeric displays while the operator is in manual mode.

Number of Significant Digits to Display The number of significant digits to use when displaying the manual output on the alphanumeric displays.

Lock in Manual State When this property is selected, the operator will only function in manual mode. A person will not be able to switch it into automatic operation.

Enable Network Access When this property is selected, the operator will broadcast its outputs over a *FAIRNET* communications network and will receive information (over the network) from associated remote A/M operators. Network operators are associated by the signal name property.

Communication Medium *FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator communicates over.

Broadcast Priority This parameter indicates if the network signal is critical or non-critical. Critical signals are deterministic; a *FAIRNET* network guarantees that consecutive critical broadcasts occur within a specified time interval (the critical broadcast interval is a property of the network that the operator is part of). Non-critical signals

are non-deterministic; consecutive broadcasts generally occur within a specified time-interval but may occasionally take more time (the non-critical broadcast interval is a property of the schema that the operator is part of).

Display A/M State Input

The state input x_{so} is used to automatically switch the operator state—automatic or manual—without front-panel user interaction. This input is not used in most applications and can remain disconnected (but *Design Pad* will issue a warning upon processing). When this property is not checked, *Design Pad* will not display the input pin and will not issue a warning upon processing.

Display Manual Input

The manual input x_{mo} is used to automatically adjust the "manual" output without front-panel user interaction. This input is not used in most applications and can remain disconnected (but *Design Pad* will issue a warning upon processing). When this property is not checked, *Design Pad* will not display the input pin and will not issue a warning upon processing.

Manual Input Activation Threshold

When the A/M operator is in manual mode, the output can be adjusted automatically using input x_{mo}. In order for automatic adjustments to begin, the manual input must change value. The magnitude of the change must exceed this parameter. (And it must exceed this value for a period of time defined by the *Manual Input Activation Time* property below.)

Manual Input Activation Time (msec)

When the A/M operator is in manual mode, the output can be adjusted automatically using input x_{mo}. In order for automatic adjustments to begin, the manual input must change value. The magnitude of the change must exceed a threshold value over a period of time (in milliseconds) defined by this parameter.

Comments:

- The PI, PD, and PID operators have A/M button functionality built-in to them. This A/M button operator is made available for alternative control strategies.

See Also:

PID Controller, PI Controller, PD Controller, PID with External Feedback, Remote Auto/Manual Button

11.10 A/D CONVERSION

Design Pad Menu Location: Operators—General Purpose

Functional Description:

Design Pad supports two signal types—analog signals (floating point values) drawn in red, and digital signals (boolean value, *i.e.,* true/false or high/low) drawn in blue. These signal types cannot be directly mixed—analog signals may not be directly connected to digital ones. But, analog signals may be converted into digital ones (and vice-versa). The *A/D Conversion* operator converts a floating-point analog value into a high/low (1/0) boolean value. The boolean output of the *A/D Conversion* operator is

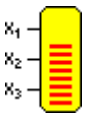
$$y = \begin{cases} 0 & \text{if} \quad |x| < Z \\ 1 & \text{otherwise.} \end{cases}$$

User-Defined Properties:

Object Name	A string label that identifies the operator.
Z	The maximum magnitude of the analog input equivalent to the boolean *low* state.

Comments: None.

See Also: D/A Conversion, Analog to Binary Conversion, Binary to Analog Conversion



11.11 BARGRAPH DISPLAY

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *bargraph display* operator can be used to show any schema signal on an *FAC-2000* front panel bargraph display. Each *FAC-2000* bargraph display contains 40 LED elements. Typically, the number of elements illuminated is proportional to the signal value. Alternatively, each bargraph can be programmed to illuminate a variable-height band of LEDs above, below, or centered about the signal value of interest.

To show a signal on the bargraph display, simply connect it to analog input x_2. The number of LEDs activated on the bargraph display, N_{led}, will be proportional to the input signal x_2,

$$N_{led} = \begin{cases} 0 & \text{if} \quad x_2 < x_{\min} \\ 40\dfrac{x_2 - x_{\min}}{x_{\max} - x_{\min}} & \text{if} \quad x_{\min} \le x_2 \le x_{\max} \\ 40 & \text{if} \quad x_2 > x_{\max} \end{cases}$$

where x_{min} is the minimum input signal value and x_{max} is the maximum input signal value. (Parameters x_{min} and x_{max} are properties of this operator.) For example, if x_{min} = 0%, x_{max} = 100%, and x_2(t)=50%, the 20 lowest LEDs—from 0% to 50%—will be activated.

The example above demonstrates the typical configuration of the *bargraph display* operator. In certain applications however, the user may wish to activate a band of LEDs not necessarily anchored at the 0% reference point. The *bargraph display* operator allows the signal value (x_2) to be displayed relative to a reference point input (x_1). In fact, signal x_2 may be displayed above, below, or centered about the reference point x_1. (The alignment—above, center, or below—is a property of this operator.)

If the alignment property is set to *above*, the number of LEDs that are activated is

$$N_{led} = \begin{cases} 0 & \text{if} \quad x_1 > x_{\max} \ \text{ or } \ x_2 < x_{\min} \\ 40 - 40\dfrac{x_1 - x_{\min}}{x_{\max} - x_{\min}} & \text{if} \quad \begin{array}{l} x_2 > x_{\max} + x_{\min} - x_1 \\ \text{and } x_{\min} \le x_1 \le x_{\max} \end{array} \\ 40\dfrac{x_2 - x_{\min}}{x_{\max} - x_{\min}} & \text{otherwise} \end{cases}.$$

The equation above indicates the following:

- No LEDs are activated if the reference point input (x_1) is above the maximum signal value.
- No LEDs are activated if the signal value (x_2) is less than minimum signal value.
- The maximum number of LEDs that can be activated is

$$N_{\max} = 40 - 40\frac{x_1 - x_{\min}}{x_{\max} - x_{\min}}$$

  For example, if x_{min} = 0%, x_{max} = 100%, and x_1(t)=50%, the maximum number of LEDs that can be activated is 20.

- In general, the number of LEDs activated is

$$N_{led} = 40\frac{x_2 - x_{\min}}{x_{\max} - x_{\min}}$$

  When the result of this formula is $N_{led} > N_{max}$, N_{max} LEDs are activated.

Similarly, when the alignment property is set to *below*, the number of LEDs that are activated is

$$N_{led} = \begin{cases} 0 & \text{if} \quad x_1 < x_{\min} \text{ or } x_2 < x_{\min} \\ 40 - 40\dfrac{x_1 - x_{\min}}{x_{\max} - x_{\min}} & \text{if} \quad \begin{array}{l} x_2 > x_{\max} + x_{\min} - x_1 \\ \text{and } x_{\min} \le x_1 \le x_{\max} \end{array} \\ 40\dfrac{x_2 - x_{\min}}{x_{\max} - x_{\min}} & \text{otherwise} \end{cases}$$

When the alignment property is set to *center*, the signal value is centered with respect to the reference point. The maximum number of LEDs that can be activated is

$$N_{\max} = 40 - 40\frac{\left| x_1 - \dfrac{x_{\max} - x_{\min}}{2} \right| - x_{\min}}{x_{\max} - x_{\min}}$$

The number of LEDs activated is

$$N_{led} = \begin{cases} 0 & \text{if} \quad x_2 < x_{\min} \\ 40\dfrac{x_2 - x_{\min}}{x_{\max} - x_{\min}} & \text{if} \quad 40\dfrac{x_2 - x_{\min}}{x_{\max} - x_{\min}} < N_{\max} \\ N_{\max} & \text{otherwise} \end{cases}$$

The digital input x_3 controls the presentation of the signal on the display. If $x_3 = 1$, the bargraph will flash; otherwise, if $x_3 = 0$, the bargraph display will be steady. Flashing a bargraph display may be useful to indicate particular operating states. For instance, it could be used to indicate that the controller is in MANUAL mode, or that some problem condition has arisen.

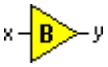
User-Defined Properties:

Alignment	This property determines how input signal x_2 is displayed relative to reference point x_1: above, center, or below.
Minimum	The minimum input signal value, x_{min}. It corresponds to having no LEDs active on the bargraph display.
Maximum	The maximum input signal value, x_{max}. It corresponds to having all LEDs active on the bargraph display.
Brightness	A brightness setting for the LED digits in the range [0-15], with 15 indicating maximum brightness.
Display Reference Input	If input x_1 is not used *Design Pad* will issue a warning upon processing the schema. You can suppress the warning message by making the input pin invisible. When this property is checked the pin is visible; otherwise it is invisible.
Display Blinking Input	If input x_3 is not used *Design Pad* will issue a warning upon processing the schema. You can suppress the warning message by making the input pin invisible. When this property is checked the pin is visible; otherwise it is invisible.

Comments:

- The minimum value x_{min} must be greater than the maximum value x_{max}.
- If input x_1 is not connected, *Design Pad* assumes that $x_1 = 0$.
- *Design Pad* will issue an error on processing if an input pin x_2 is not connected.

See Also: Numeric Display



11.12 BIAS

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

The output of the *bias* operator is

$$y = x + B$$

where y is the output, x is the input, and B is the biasing term.

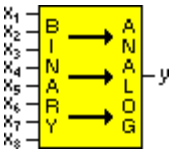
Design Pad will issue an error on processing if the input pin is not connected.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Bias	The biasing term, B.

Comments: None.

See Also: Gain, Unit Conversion, Addition, Subtraction



11.13 BINARY to ANALOG CONVERTER

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The *Binary to Analog Converter* block accepts eight digital inputs x_i (i=1,2,…,8), and computes its base-10 representation y. Two of the operator properties, y_{min} and y_{max}, define the range of the output signal. They are used to map the base-10 representation of the eight input signals into an analog value between y_{min} and y_{max}. The mapping is

$$y = \frac{255 - X_{10}}{x_{max} - x_{min}} + x_{min}$$

where X_{10} is the base-10 representation of the eight digital input signals, x_i (i=1,…,8).

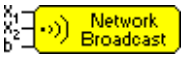
User-Defined Properties:

Object Name	A string label that identifies the operator.
Minimum (0) Value	The output value corresponding to a base-10 input value of 0 (y_{min}).
Maximum (255) Value	The output value corresponding to a base-10 input value of 255 (y_{max}).

Comments: None.

See Also:

Analog to Binary Converter, Multiplexer, Demultiplexer, A/D Conversion, D/A Conversion



11.14 BROADCAST

Design Pad Menu Location: Operators—Network Operators

Functional Description:

Broadcast operators encode their input signals x_1 and x_2 into a communication message that they broadcast over a controller network. All controllers that are part of the network will receive the broadcasted message. But, only those controllers with a receiver operator "tuned" to that broadcast will actually decode the message.

There are three types of broadcast operators: analog, digital, and mixed broadcasts. Analog and digital broadcast operators do not have an input x_2.

Analog broadcast operators transmit analog signals. They have two inputs—one analog and one digital. Analog input x_1 is the signal that the operator is to broadcast. Digital input *b* enables (*high*) or disables broadcasting (*low*). If the digital input is not connected, broadcasting is always enabled.

Digital broadcast operators transmit digital signals. They have two digital inputs. Digital input x_1 is the broadcast signal and digital input *b* is the enable pin.

Mixed broadcast operators transmit communication messages consisting of both analog and digital signals. They have three inputs—one analog and two digital. Digital input *b* enables/disables broadcasting for the operator. Analog input x_1 and digital input x_2 are combined into a single communication message that the operator broadcasts over the network.

Analog broadcast operators use a 16-bit integer to encode the analog input signal. A 16-bit integer can assume $2^{16} = 65536$ distinct values (0-65535). The communication message *m*—the 16-bit integer—that the analog broadcast operator transmits is determined from:

$$m = \begin{cases} 0 & if \quad x_1 < x_{0\%} \\ \dfrac{65535(x_1 - x_{0\%})}{x_{100\%} - x_{0\%}} & if \quad x_{0\%} \le x_1 \le x_{100\%} \\ 65535 & if \quad x_1 > x_{100\%} \end{cases}$$

where $x_{0\%}$ is the minimum value that x_1 is expected to assume and $x_{100\%}$ is the maximum value that x_1 is expected to assume. The signal mappings $x_{0\%}$ and $x_{100\%}$ are properties of the broadcast operator. Note that if input x_1 represents a 4-20mA signal, then the resolution of an analog broadcast transmission is $2^{-12} = 0.000244$mA. Mixed broadcast operators use a 16-bit integer to encode the analog input signal x_1 and the digital input signal x_2 (it uses 15 bits

for the analog signal and 1 bit for the digital signal). If input x_1 represents a 4-20mA signal, then the resolution of a mixed broadcast transmission is 2^{-11}=0.000488mA.

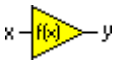
User-Defined Properties:

Signal Name	A string label that identifies the broadcast signal. It is used to associate this broadcast operator in one schema with receiver operator(s) in other schemas.
Initial Broadcast Value	The signal value transmitted initially.
Communication Medium	*FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator communicates over.
Broadcast Priority	This parameter indicates if the network signal is critical or non-critical. Critical signals are deterministic; a *FAIRNET* network guarantees that consecutive critical broadcasts occur within a specified time interval (the critical broadcast interval is a property of the network that the operator is part of). Non-critical signals are non-deterministic; consecutive broadcasts generally occur within a specified time-interval but may occasionally take more time (the non-critical broadcast interval is a property of the schema that the operator is part of).
0% Mapping	The lower limit of the x_1 input signal range, $x_{100\%}$.
100% Mapping	The upper limit of the x_1 input signal range, $x_{0\%}$.
Display Enable Input	Digital input *b* enables broadcast transmissions. In most applications, continuous transmissions are desired so this input is not used and can remain disconnected (but *Design Pad* will issue a warning upon processing). When this property is not checked, *Design Pad* will not display the input pin and will not issue a warning upon processing.

Comments: None.

See Also:

Receiver



11.15 CHARACTERIZER

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

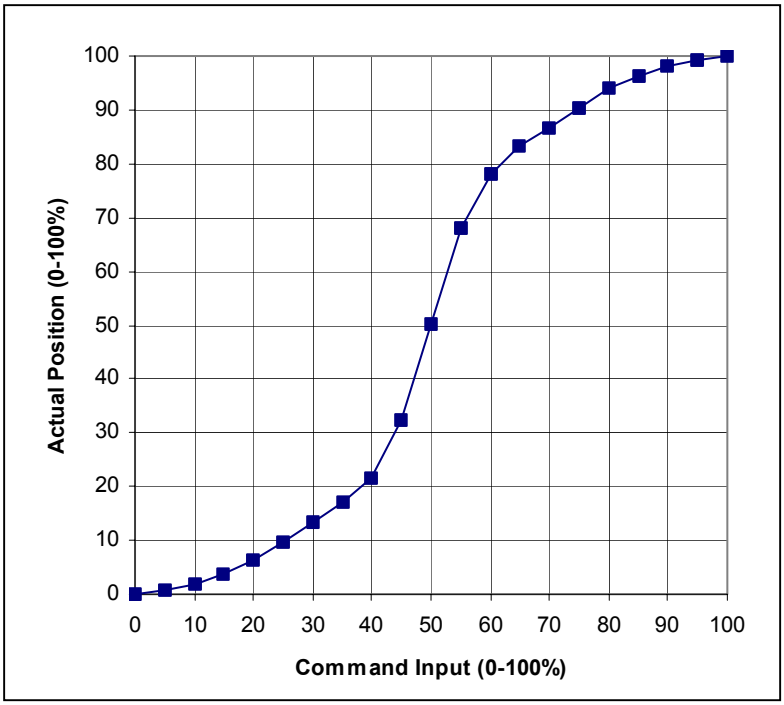
The *characterizer* operator is a piecewise-linear interpolation mapping. Its output y is defined by

$$y = y_i + \left(y_{i+1} - y_i\right)\frac{x - x_i}{x_{i+1} - x_i}, \qquad \text{if } x_i \le x \le x_{i+1}, \qquad i=0,1,...,19,$$

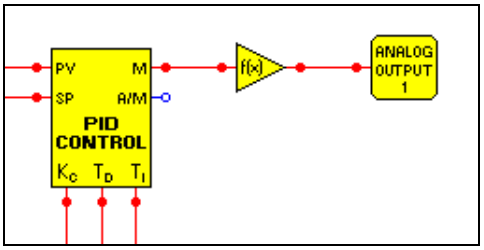
where x is the characterizer input, and $\left\{\left(x_i, y_i\right) \mid x_i \le x_{i+1}\right\}$, is a 21-point user-defined data set to be interpolated.

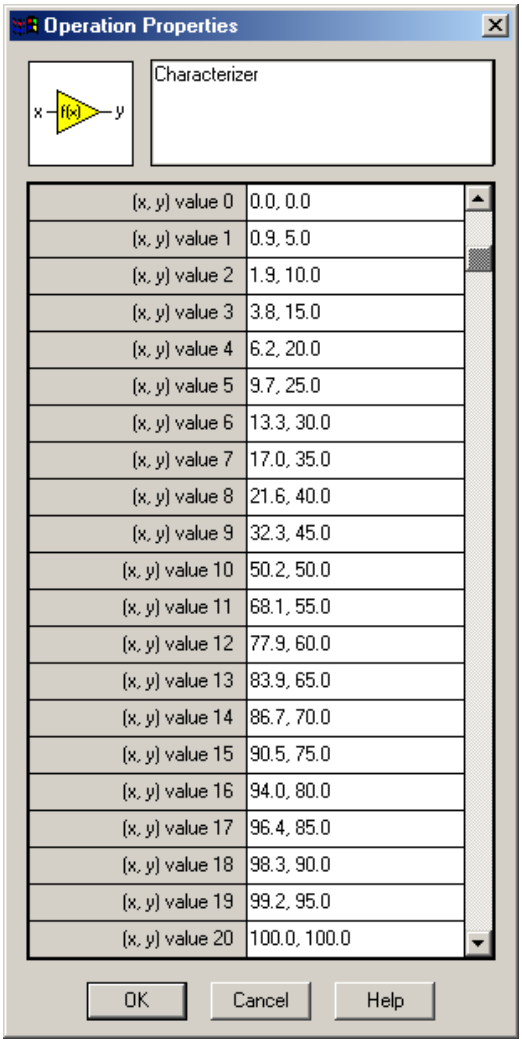
The characterizer operator is ideally suited for situations where an accurate mathematical model of equipment operation is not available, but measurement data is. For instance, suppose that in the water tank schema shown in Figure 1 (see page 2), analog output 1 drives a proportional valve. Suppose further that the valve response is not linear throughout its range of operation. Assume that experiments characterizing its performance yield the following test data:

Command Input	Actual Position
0.0	0.0
5.0	0.9
10.0	1.9
15.0	3.8
20.0	6.2
25.0	9.7
30.0	13.3
35.0	17.0
40.0	21.6
45.0	32.3
50.0	50.2
55.0	68.1
60.0	77.9
65.0	83.1
70.0	86.7
75.0	90.5
80.0	94.0
85.0	96.4
90.0	98.3
95.0	99.2
100.0	100.0



In the table, the values in the left column represent the position the valve was commanded to reach (0-100% its range of operation). The data in the right column is the measured steady-state valve position in response to the corresponding command input.





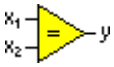
A pronounced non-linear valve response (such as the one shown in the preceding graph) may adversely affect overall system performance. PID tuning constants that yield good performance when the valve is operating in its linear range may yield inadequate performance when the valve is operating in its saturation region. In the water tank example referenced above, you could counteract this problem by adding a characterizer operator between the PID block and the analog output block. You could set up the characterizer operator to invert the valve operating characteristics, thereby linearizing the plant. The required characterizer data set (the inverse of the valve performance data) is reproduced here.

With this arrangement, the characterizer and valve positioner combination would respond to PID command signals linearly. For example, a PID output of 77.9% will be translated (by the characterizer) into a command input (to the valve) of 60.0%. When this 60.0% command input is applied to the valve, the resulting valve position will be 77.9%.

User-Defined Properties:

Object Name	A string label that identifies the operator.
(x_0, y_0)	First data point to interpolate.
(x_1, y_1)	Second data point to interpolate.
⋮	
(x_{20}, y_{20})	Last data point to interpolate.

Comments: The data abscissas must be monotonically increasing: $x_i \leq x_{i+1}$, $i = 0,1,\ldots 19$.



11.16 COMPARATOR: =

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *equality comparator* operator compares two analog input signals. Its digital output is set *high* (1) if the signals are equal (*i.e.*, within a user-defined dead-band); and it is set *low* (0) if they are not. The output state will change only if the equality/inequality condition is maintained for a user-specified length of time—the switching dead-time. That is, the output is defined by

$$y(t)=\begin{cases} 1 & \text{if } \operatorname{abs}\left[x_1(\bar{t})-x_2(\bar{t})\right]\le B \ \ \forall \bar{t}\in[t,t-T] \\ 0 & \text{if } \operatorname{abs}\left[x_1(\bar{t})-x_2(\bar{t})\right]> B \ \ \forall \bar{t}\in[t,t-T] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where y is the digital output, x_i (i=1,2) are the analog inputs, B is the equality dead-band, T is the switching dead-time, Δt is the loop sampling time, and where abs[] denotes the absolute value operator.

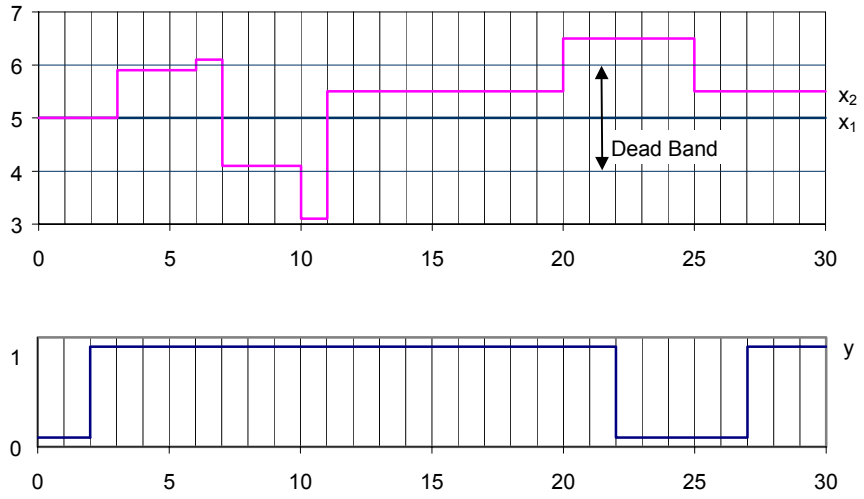
For example, suppose that the initial output is $y(0) = 0$, input signal x_1 is fixed $(x_1 \equiv 5.0)$, the dead-band is set to B=1.0, the switching dead-time is set to T=2 seconds, and signal x_2 has the following profile:

$$x_2(t)=\begin{cases} 5.0 & 0\le t<3 \\ 5.9 & 3\le t<6 \\ 6.1 & 6\le t<7 \\ 4.1 & 7\le t<10 \\ 3.1 & 10\le t<11 \\ 5.5 & 11\le t<20 \\ 6.5 & 20\le t<25 \\ 5.5 & t\ge 25 \end{cases}$$

Under these conditions, the output of the equality comparator will be

$$y(t)=\begin{cases} 0 & 0\le t<2 \\ 1 & 2\le t<22 \\ 0 & 22\le t<27 \\ 1 & t\ge 27 \end{cases}$$

A plot of the input signals and a plot of the operator output are shown next.



Initially, the output is $y(0)=0$ (as specified by the initial condition property). The output remains *low* (0), until two seconds elapse (the switching dead-time). The output goes *high* after two seconds, $y(2)=1$, and remains *high* while $x_2(t)=5.9$ because the input signals are within the dead-band $B=1.0$. It also remains *high* while $x_2(t)=6.1$ because the inequality condition is not maintained for at least two seconds. Similarly, it remains *high* while $x_2(t)=4.1$ (within dead-band) and while $x_2(t)=3.1$(inequality condition not maintained for two seconds). At $t=20$ seconds, the input signals satisfy the inequality condition, but the output does not switch to *low* until $t=22$ seconds. (when the input signals have remained unequal for at least two seconds.) Likewise, the output will go *high* (and remain *high* thereafter) at $t=27$ seconds.

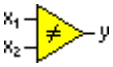
User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time $t=0$).
Dead-Band	The equality dead-band, B.
Switching Dead-Time	The length of time a signal must satisfy the equality (inequality) condition, for the output to switch from 0 to 1 (from 1 to 0). The switching dead-time is denoted by T in the equation above, and is expressed in seconds.

Comments:

- The equality dead-band must be greater than or equal to zero, $B \geq 0$.
- The switching dead-time must be greater than or equal to zero, $T \geq 0$.

See Also: ≠, >, <, ≥, and ≤ Comparators; High/Low Alarm; Thresholding



11.17 COMPARATOR: ≠

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *inequality comparator* operator compares two analog input signals. Its digital output is set *low* (0) if the signals are equal (*i.e.*, within a user-defined dead-band); and it is set *high* (1) if they are not. The output state will change only if the equality/inequality condition is maintained for a user-specified length of time—the switching dead-time. That is, the output is defined by

$$y(t) = \begin{cases} 0 & \text{if abs}\left[x_1(\bar{t}) - x_2(\bar{t})\right] \le B \;\; \forall \bar{t} \in [t, t-T] \\ 1 & \text{if abs}\left[x_1(\bar{t}) - x_2(\bar{t})\right] > B \;\; \forall \bar{t} \in [t, t-T] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where y is the digital output, x_i (i=1,2) are the analog inputs, B is the equality dead-band, T is the switching dead-time, Δt is the loop sampling time, and where abs[] denotes the absolute value operator.

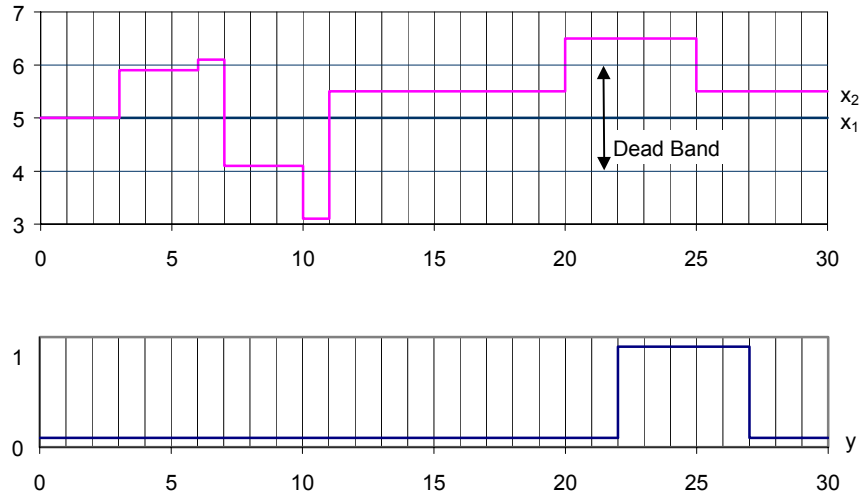
For example, suppose that the initial output is $y(0)$=0, input signal x_1 is fixed $(x_1 \equiv 5.0)$, the dead-band is set to B=1.0, the switching dead-time is set to T=2 seconds, and signal x_2 has the following profile:

$$x_2(t) = \begin{cases} 5.0 & 0 \le t < 3 \\ 5.9 & 3 \le t < 6 \\ 6.1 & 6 \le t < 7 \\ 4.1 & 7 \le t < 10 \\ 3.1 & 10 \le t < 11 \\ 5.5 & 11 \le t < 20 \\ 6.5 & 20 \le t < 25 \\ 5.5 & t \ge 25 \end{cases}$$

Under these conditions, the output of the inequality comparator will be

$$y(t) = \begin{cases} 0 & 0 \le t < 22 \\ 1 & 22 \le t < 27 \\ 0 & t \ge 27 \end{cases}$$

A plot of the input signals and a plot of the operator output are shown below.



Initially, the output is $y(0)=0$ (as specified by the initial condition property). It remains *low* (0) while $x_2(t)=5.0$ since the signals are equal to each other. It also remains *low* while $x_2(t)=5.9$ because the input signals are within the dead-band $B=1.0$. And, it remains *low* while $x_2(t)=6.1$ because the inequality condition is not maintained for at least two seconds. Similarly, the output stays *low* while $x_2(t)=4.1$ (signals within dead-band), while $x_2(t)=3.1$(inequality condition not maintained for two seconds), and while $x_2(t)=5.5$ (signals within dead-band). The output does not switch to *high* (1) until $t=22$, when the two signals are not equal for more than two seconds. At time $t=25$ seconds, the input signals satisfy the equality condition, but the output does not switch to *low* until $t=27$ seconds. The output remains *low* thereafter.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time $t=0$).
Dead-Band	The equality dead-band, B.
Switching Dead-Time	The length of time a signal must satisfy the inequality (equality) condition, for the output to switch from 0 to 1 (from 1 to 0). The switching dead-time is denoted by T in the equation above, and is expressed in seconds.

Comments:

- The equality dead-band must be greater than or equal to zero, $B \geq 0$.
- The switching dead-time must be greater than or equal to zero, $T \geq 0$.

See Also: =, >, <, ≥, and ≤ Comparators; High/Low Alarm; Thresholding

11.18 COMPARATOR: >

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *greater-than comparator* operator compares two analog input signals. Its digital output is set to *high* (1) if input signal x_1 is greater than input signal x_2; and it is set to *low* (0) otherwise. The output state will change only if the greater than condition (or less than or equal to condition) is maintained for a user-specified length of time—the switching dead time. That is, the output is defined by

$$y(t)=\begin{cases} 1 & \text{if} \quad x_1(\bar{t}) > x_2(\bar{t}) \; \forall \bar{t} \in [t, t-T] \\ 0 & \text{if} \quad x_1(\bar{t}) + B \le x_2(\bar{t}) \; \forall \bar{t} \in [t, t-T] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where y is the digital output, x_i (i=1,2) are the analog inputs, B is the comparison dead-band, T is the switching dead-time, and Δt is the loop sampling time.

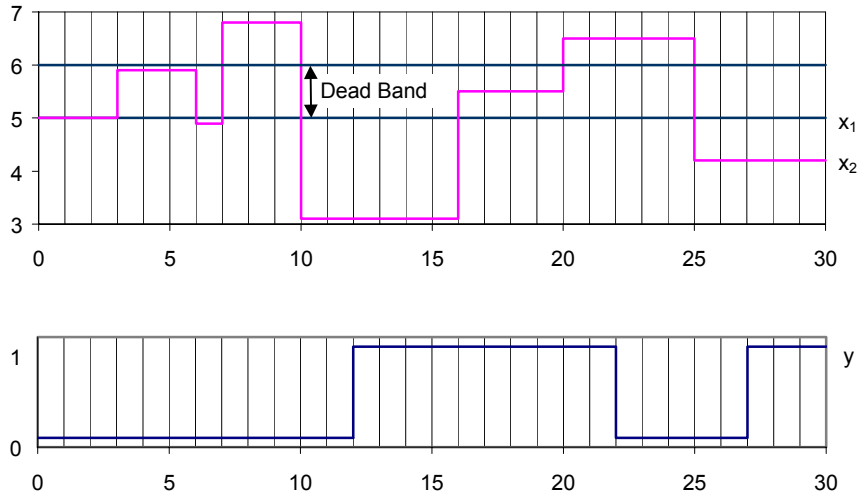
For example, suppose that the initial output is $y(0)$=0, input signal x_1 is fixed $(x_1 \equiv 5.0)$, the dead-band is set to B=1.0, the switching dead-time is set to T=2 seconds, and signal x_2 has the following profile:

$$x_2(t)=\begin{cases} 5.0 & 0 \le t < 3 \\ 5.9 & 3 \le t < 6 \\ 4.9 & 6 \le t < 7 \\ 6.8 & 7 \le t < 10 \\ 3.1 & 10 \le t < 16 \\ 5.5 & 16 \le t < 20 \\ 6.5 & 20 \le t < 25 \\ 4.2 & t \ge 25 \end{cases}$$

Under these conditions, the output of the *greater-than* comparator will be

$$y(t)=\begin{cases} 0 & 0 \le t < 12 \\ 1 & 12 \le t < 22 \\ 0 & 22 \le t < 27 \\ 1 & t \ge 27 \end{cases}$$

A plot of the input signals and a plot of the operator output are shown below.



Initially, the output is $y(0)=0$ (as specified by the initial condition property). It remains *low* (0) while $x_2(t)=5.0$ and while $x_2(t)=5.9$ since the greater-than condition is not met. It also remains *low* while $x_2(t)=4.9$ because the greater-than condition is not maintained for at least two seconds. The output does not switch to *high* (1) until $t=12$, when $x_1(t)>x_2(t)$ for more than two seconds. The output remains *high* while $x_2(t)=5.5$ because the greater-than condition is met (within the dead-band, *i.e.,* $5.0 + 1.0 > 5.5$). At time $t=20$ seconds, input x_1 is less than x_2, but the output does not switch to *low* (0) until $t=22$ seconds (when the less-than condition has been maintained for at least two seconds). The output will switch to *high* (and remain *high* thereafter) at time $t=27$ seconds.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time $t=0$).
Dead-Band	The comparison dead-band, B.
Switching Dead-Time	The length of time a signal must satisfy the greater-than condition, for the output to switch from 0 to 1 (likewise for the output to transition from 1 to 0). The switching dead-time is denoted by T in the equation above, and is expressed in seconds.

Comments:

- The comparison dead-band must be greater than or equal to zero, $B \geq 0$.
- The switching dead-time must be greater than or equal to zero, $T \geq 0$.

See Also: =, ≠, <, ≥, and ≤ Comparators; High/Low Alarm; Thresholding

11.19 COMPARATOR: ≥

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *greater-or-equal comparator* operator compares two analog input signals. Its digital output is set to *high* (1) if input signal x_1 is greater than or equal to input signal x_2; and it is set to *low* (0) otherwise. The output state will change only if the greater than or equal to condition (or less than condition) is maintained for a user-specified length of time—the switching dead time. That is, the output is defined by

$$y(t) = \begin{cases} 1 & \text{if} \quad x_1(\bar{t}) \geq x_2(\bar{t}) \; \forall \bar{t} \in [t, t-T] \\ 0 & \text{if} \quad x_1(\bar{t}) + B < x_2(\bar{t}) \; \forall \bar{t} \in [t, t-T] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where y is the digital output, x_i (i =1,2) are the analog inputs, B is the comparison dead-band, T is the switching dead-time, and Δt is the loop sampling time.

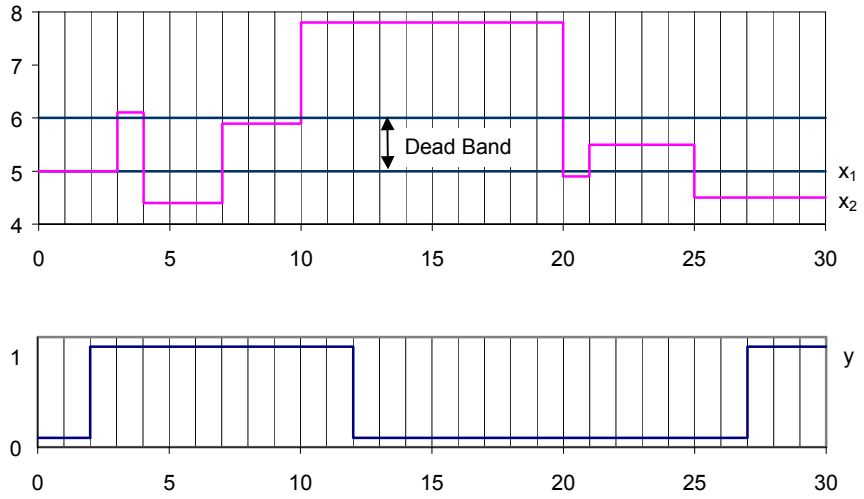
For example, suppose that the initial output is $y(0)$=0, input signal x_1 is fixed $(x_1 \equiv 5.0)$, the dead-band is set to B=1.0, the switching dead-time is set to T=2 seconds, and signal x_2 has the following profile:

$$x_2(t) = \begin{cases} 5.0 & 0 \leq t < 3 \\ 6.1 & 3 \leq t < 4 \\ 4.4 & 4 \leq t < 7 \\ 5.9 & 7 \leq t < 10 \\ 7.8 & 10 \leq t < 20 \\ 4.9 & 20 \leq t < 21 \\ 5.5 & 21 \leq t < 25 \\ 4.5 & t \geq 25 \end{cases}$$

Under these conditions, the output of the *greater-or-equal* comparator will be

$$y(t) = \begin{cases} 0 & 0 \leq t < 2 \\ 1 & 2 \leq t < 12 \\ 0 & 12 \leq t < 27 \\ 1 & t \geq 27 \end{cases}$$

A plot of the input signals and a plot of the operator output are shown below.



Initially, the output is $y(0)=0$ (as specified by the initial condition property). The output remains *low* (0) for the first two seconds (the switching dead-time). It goes *high* (1) after two seconds because the greater-or-equal condition is satisfied. It remains *high* while $x_2(t)=6.1$ because the less-than condition is not maintained for at least two seconds. It also remains *high* while $x_2(t)=5.9$ because the greater-or-equal condition is satisfied (within the dead-band, *i.e.,* 5.0 + 1.0 > 5.9). The output does not switch to *low* (0) until $t=12$, when $x_1(t)+B<x_2(t)$ for more than two seconds. The output remains *low* while $x_2(t)=4.9$ because the greater-than condition is not maintained for at least two seconds. At time $t=25$ seconds, input x_1 is greater than x_2, but the output does not switch to *high* until $t=27$ seconds (when the greater-or-equal condition has been maintained for at least two seconds).

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time $t=0$).
Dead-Band	The comparison dead-band, B.
Switching Dead-Time	The length of time a signal must satisfy the greater-than-or-equal-to condition for the output to switch from 0 to 1 (likewise for the output to transition from 1 to 0). In the equation above, T denotes the switching dead-time. It is expressed in seconds.

Comments:

- The comparison dead-band must be greater than or equal to zero, $B \geq 0$.
- The switching dead-time must be greater than or equal to zero, $T \geq 0$.

See Also: =, ≠, >, <, and ≤ Comparators; High/Low Alarm; Thresholding

11.20 COMPARATOR: <

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *less-than comparator* operator compares two analog input signals. Its digital output is set to *high* (1) if input signal x_1 is less than input signal x_2; and it is set to *low* (0) otherwise. The output state will change only if the less than condition (or greater than or equal to condition) is maintained for a user-specified length of time—the switching dead time. That is, the output is defined by

$$y(t)=\begin{cases} 1 & \text{if} \quad x_1(\bar{t}) < x_2(\bar{t}) \,\forall \bar{t} \in [t, t-T] \\ 0 & \text{if} \quad x_1(\bar{t}) - B \geq x_2(\bar{t}) \,\forall \bar{t} \in [t, t-T] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where y is the digital output, x_i (i=1,2) are the analog inputs, B is the comparison dead-band, T is the switching dead-time, and Δt is the loop sampling time.

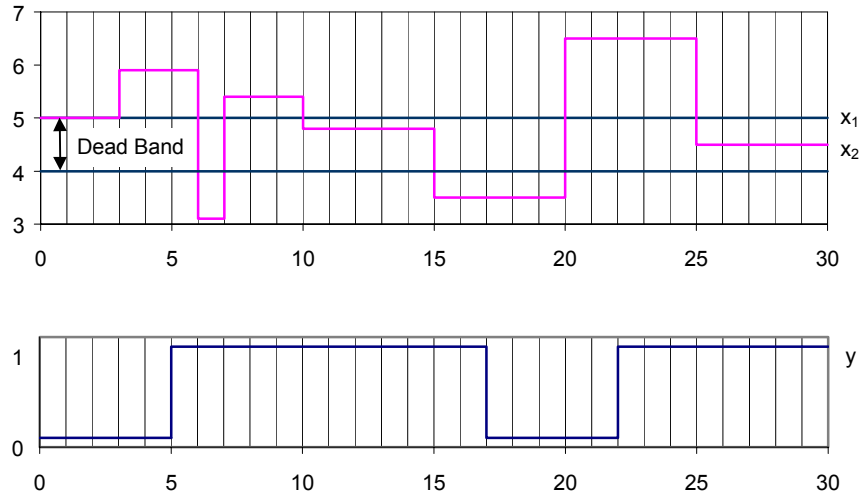
For example, suppose that the initial output is $y(0)$=0, input signal x_1 is fixed $(x_1 \equiv 5.0)$, the dead-band is set to B=1.0, the switching dead-time is set to T=2 seconds, and signal x_2 has the following profile:

$$x_2(t)=\begin{cases} 5.0 & 0 \leq t < 3 \\ 5.9 & 3 \leq t < 6 \\ 3.1 & 6 \leq t < 7 \\ 5.4 & 7 \leq t < 10 \\ 4.8 & 10 \leq t < 15 \\ 3.5 & 15 \leq t < 20 \\ 6.5 & 20 \leq t < 25 \\ 4.5 & t \geq 25 \end{cases}$$

Under these conditions, the output of the *less-than* comparator will be

$$y(t)=\begin{cases} 0 & 0 \leq t < 5 \\ 1 & 5 \leq t < 17 \\ 0 & 17 \leq t < 22 \\ 1 & t \geq 22 \end{cases}$$

A plot of the input signals and a plot of the operator output are shown below.



Initially, the output is $y(0)$=0 (as specified by the initial condition property). It remains *low* (0) while $x_2(t)$=5.0 because the less-than condition is not met. At t=3 seconds, the less-than condition is satisfied, but the output does not switch to *high* (1) until t=5 seconds (after the condition is maintained for at least two seconds). The output remains *high* while $x_2(t)$=3.1 because the greater-than condition is not maintained for at least two seconds. And, the output remains *high* while $x_2(t)$=4.8 because the input signals satisfy the less than condition (within the dead-band, *i.e.*, 5.0 + 1.0 < 4.8). The output goes *low* at t=17 seconds since $x_1(t) + B \geq x_2(t)$ for two consecutive seconds. It switches back to *high* (and remains *high* thereafter) at t=22, when x1(t)<x2(t) for two consecutive seconds.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time t=0).
Dead-Band	The comparison dead-band, B.
Switching Dead-Time	The length of time a signal must satisfy the less-than condition for the output to switch from 0 to 1 (likewise for the transition from 1 to 0). The switching dead-time is denoted by T in the equation above, and is expressed in seconds.

Comments:

- The comparison dead-band must be greater than or equal to zero, $B \geq 0$.
- The switching dead-time must be greater than or equal to zero, $T \geq 0$.

See Also: =, ≠, >, ≥, and ≤ Comparators; High/Low Alarm; Thresholding

11.21 COMPARATOR: ≤

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *less-or-equal comparator* operator compares two analog input signals. Its digital output is set to *high* (1) if input signal x_1 is less than or equal to input signal x_2; and it is set to *low* (0) otherwise. The output state will change only if the less or equal condition (or greater than condition) is maintained for a user-specified length of time—the switching dead time. That is, the output is defined by

$$y(t)=\begin{cases} 1 & \text{if} \quad x_1(\bar{t}) \le x_2(\bar{t}) \; \forall \bar{t} \in [t, t-T] \\ 0 & \text{if} \quad x_1(\bar{t}) - B > x_2(\bar{t}) \; \forall \bar{t} \in [t, t-T] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where y is the digital output, x_i (i =1,2) are the analog inputs, B is the comparison dead-band, T is the switching dead-time, and Δt is the loop sampling time.

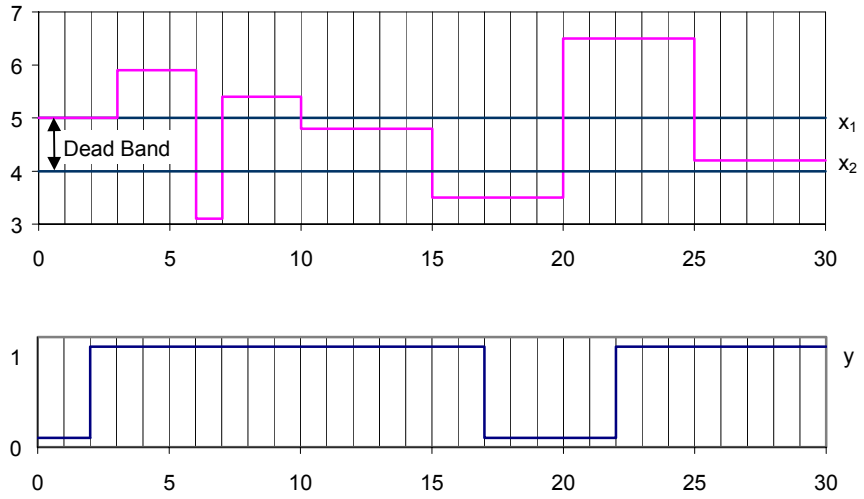
For example, suppose that the initial output is $y(0)$=0, input signal x_1 is fixed $(x_1 \equiv 5.0)$, the dead-band is set to B=1.0, the switching dead-time is set to T=2 seconds, and signal x_2 has the following profile:

$$x_2(t)=\begin{cases} 5.0 & 0 \le t < 3 \\ 5.9 & 3 \le t < 6 \\ 3.1 & 6 \le t < 7 \\ 5.4 & 7 \le t < 10 \\ 4.8 & 10 \le t < 15 \\ 3.5 & 15 \le t < 20 \\ 6.5 & 20 \le t < 25 \\ 4.2 & t \ge 25 \end{cases}$$

Under these conditions, the output of the *less-than* comparator will be

$$y(t)=\begin{cases} 0 & 0 \le t < 2 \\ 1 & 2 \le t < 17 \\ 0 & 17 \le t < 22 \\ 1 & t \ge 22 \end{cases}$$

A plot of the input signals and a plot of the operator output are shown below.



Initially, the output is $y(0)=0$ (as specified by the initial condition property). The output remains *low* (0) for the first two seconds (the switching dead-time). It goes *high* (1) after two seconds because the less-or-equal condition is satisfied. It remains *high* while $x_2(t)=5.9$ and even while $x_2(t)=3.1$ because the greater-than condition is not maintained for at least two seconds. And, the output remains *high* while $x_2(t)=4.8$ because the input signals satisfy the less-or-equal condition (within the dead-band, *i.e.*, $5.0 + 1.0 \leq 4.8$). The output goes *low* at $t=17$ seconds since $x_1(t) + B > x_2(t)$ for two consecutive seconds. It switches back to *high* (and remains *high* thereafter) at $t=22$, when $x_1(t) \leq x_2(t)$ for two consecutive seconds.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time $t=0$).
Dead-Band	The comparison dead-band, B.
Switching Dead-Time	The length of time a signal must satisfy the less-than-or-equal-to for the output to switch from 0 to 1 (likewise for the transition from 1 to 0). In the equation above, T denotes the switching dead-time. It is expressed in seconds.

Comments:

- The comparison dead-band must be greater than or equal to zero, $B \geq 0$.
- The switching dead-time must be greater than or equal to zero, $T \geq 0$.

See Also: , ≠, >, ≥, and < Comparators; High/Low Alarm; Thresholding

11.22 CONSTANT (ANALOG)

Design Pad Menu Location: Operators

Functional Description:

The output of the *constant* operator is the fixed value

$$y = K$$

where y is the output, and K is the user-defined constant value.

In *Design Pad 2000* the constant operator doubles as a special networking operator designed to synchronize a parameter value (*e.g.*, process set-point) accessible on multiple controller stations. You can enable the constant operator to broadcast its output value over a controller network and to receive information from one or more Remote Constant operators (see section 6.1.3) from other stations on the network. The network-enabled Constant operator and the Remote Constant operator work together to maintain a common output value that can be adjusted from the front-panel of multiple control stations. For example, consider the case where one station contains a network-enabled Constant operator and two other stations on the same controller network each contain a corresponding Remote Constant operator. If you adjust the parameter value at any one of the three stations, the other two stations will automatically match your changes. You can update the parameter value at any one of the three stations—the other two will automatically follow.

In order to associate a Constant operator in one schema to a Remote Constant operator in another, you must match their signal name property. If you want to associate multiple Remote Constant operators with a particular Constant operator, then you must set the signal name property of all operators to the same case-sensitive character string.

As with other networking operators, in addition to setting the signal name property, you must also specify if the broadcast-signal is critical and the type of network that this operator is intended for (RS-232 or RS-485).

User-Defined Properties:

Signal Name	A string label that identifies the operator. When networking is enabled, this character string is used to associate (synchronize) an A/M operator in one schema with remote A/M operator in other schemas.
Constant Value	The constant value, *K*.
Front Panel Access	This property can assume three states: (i) None (no front panel

	access); (ii) Operator level access (constant can be adjusted from the *FAC-2000* front panel); and (iii) Engineer level access (constant can be adjusted only after password is provided.)
Minimum	The minimum value the constant can assume when modified from the front panel.
Maximum	The maximum value the constant can assume when modified from the front panel.
Increment	The magnitude of constant value increments when modified from the *FAC-2000* front panel.
Number of Significant Digits to Display	The number of significant digits to use when displaying the constant value on the alphanumeric displays.
Enable Network Access	When this property is selected, the operator will broadcast its output value over a *FAIRNET* communications network and will receive information (over the network) from associated Remote Constant operators. Network operators are associated by the signal name property.
Communication Medium	*FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator communicates over.
Broadcast Priority	This parameter indicates if the network signal is critical or non-critical. Critical signals are deterministic; a *FAIRNET* network guarantees that consecutive critical broadcasts occur within a specified time interval (the critical broadcast interval is a property of the network that the operator is part of). Non-critical signals are non-deterministic; consecutive broadcasts generally occur within a specified time-interval but may occasionally take more time (the non-critical broadcast interval is a property of the schema that the operator is part of).

Comments: None.

See Also: Constant (Digital), Remote Constant, Unit Conversion

11.23 CONSTANT (DIGITAL)

Design Pad Menu Location: Operators

Functional Description:

The output of the digital *constant* operator is the fixed value

$$y = K$$

where y is the output, and K is the user-defined boolean value.

In *Design Pad 2000* the constant operator doubles as a special networking operator designed to synchronize a parameter value (*e.g.*, process set-point) accessible on multiple controller stations. You can enable the constant operator to broadcast its output value over a controller network and to receive information from one or more Remote Constant operators (see section 6.1.3) from other stations on the network. The network-enabled Constant operator and the Remote Constant operator work together to maintain a common output value that can be adjusted from the front-panel of multiple control stations. For example, consider the case where one station contains a network-enabled Constant operator and two other stations on the same controller network each contain a corresponding Remote Constant operator. If you adjust the parameter value at any one of the three stations, the other two stations will automatically match your changes. You can update the parameter value at any one of the three stations—the other two will automatically follow.

In order to associate a Constant operator in one schema to a Remote Constant operator in another, you must match their signal name property. If you want to associate multiple Remote Constant operators with a particular Constant operator, then you must set the signal name property of all operators to the same case-sensitive character string.

As with other networking operators, in addition to setting the signal name property, you must also specify if the broadcast-signal is critical and the type of network that this operator is intended for (RS-232 or RS-485).

User-Defined Properties:

Signal Name	A string label that identifies the operator. When networking is enabled, this character string is used to associate (synchronize) an A/M operator in one schema with remote A/M operator in other schemas.
Constant Value	The constant value, *K* (HIGH/LOW)
Front Panel Access	This property can assume three states: (i) None (no front panel access); (ii) Operator level access (constant can be adjusted from the *FAC-2000* front panel); and (iii) Engineer level access

(constant can be adjusted only after password is provided.)

Low State Label — By default, the state K=0 is indicated as "LOW". However, in certain cases, other terms may be more appropriate (*e.g.*, NO, FALSE, OFF, *etc.*)

High State Label — By default, the state K=1 is indicated as "HIGH". However, in certain cases, other terms may be more appropriate (*e.g.*, YES, TRUE, ON, *etc.*)

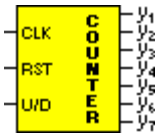
Enable Network Access — When this property is selected, the operator will broadcast its output value over a *FAIRNET* communications network and will receive information (over the network) from associated Remote Constant operators. Network operators are associated by the signal name property.

Communication Medium — *FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator communicates over.

Broadcast Priority — This parameter indicates if the network signal is critical or non-critical. Critical signals are deterministic; a *FAIRNET* network guarantees that consecutive critical broadcasts occur within a specified time interval (the critical broadcast interval is a property of the network that the operator is part of). Non-critical signals are non-deterministic; consecutive broadcasts generally occur within a specified time-interval but may occasionally take more time (the non-critical broadcast interval is a property of the schema that the operator is part of).

Comments: None.

See Also: Constant (Analog), Remote Constant, Unit Conversion



11.24 COUNTER

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The Counter operator is a digital counter that adjusts its output value with each clock pulse. It has three digital inputs, *CLK*, *RST*, and *U/D*; and eight digital outputs y_i (i=1,2,…8). The digital outputs are the binary representation of the counter state, S, an integer in the range [S_{min}, S_{max}]. Parameters S_{min} and S_{max} are user-defined properties of the operator that must satisfy $0 \le S_{\min} \le S_{\max} \le 255$.

The counter state, S, is updated with the rising edge of the *CLK* input as follows:

$$S(t_{clk}) = \begin{cases} S(t_{clk} - \Delta t) + 1 & \text{if} \quad U/D = 1 \text{ and } S \neq S_{\max} \\ S_{\min} & \text{if} \quad U/D = 1 \text{ and } S = S_{\max} \\ S(t_{clk} - \Delta t) - 1 & \text{if} \quad U/D = 0 \text{ and } S \neq S_{\min} \\ S_{\max} & \text{if} \quad U/D = 0 \text{ and } S = S_{\min} \end{cases}$$

where t_{clk} is the time instant that *CLK* changes from 0 to 1.

In addition, the counter state is reset to its minimum value, *i.e.*, $S = S_{\min}$ whenever the *RST* input is *high* (1).

User-Defined Properties:

Object Name	A string label that identifies the operator.
Min. Value (0-255)	The minimum counter state, S_{min}.
Max. Value (0-255)	The maximum counter state, S_{max}.

Comments: None.

See Also: Binary to Analog Converter

11.25 DELAY ELEMENT (ANALOG or DIGITAL)

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The output of the *delay element* operator is the time-shifted value of its input

$$y(t + M\Delta t) = x(t)$$

where y is the output, x is the input, Δt is the loop sampling time, and M is the number of samples to delay the signal by. Multiple M must be an integer in the range [1-100].

User-Defined Properties:

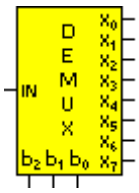
Object Name	A string label that identifies the operator.
Time Delay Multiple	An integer multiple of the loop sampling time Δt. It indicates the number of time steps to delay the input signal. It must be an integer in the range [1-100].
Initial Condition	The output value on startup (at time $t = 0$).

Comments:

The default time delay multiple is M=1.

See Also:

Moving Average, Timer, PID Controller, PI Controller, PD Controller



11.26 DEMULTIPLEXER

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The de*multiplexer* operator passes its analog input signal to one of its eight analog outputs. The analog output that passes the analog input is determined by state of the three digital inputs, as shown in the table below:

b_2	b_1	b_0	Selected Output
0	0	0	$x_0 = IN$
0	0	1	$x_1 = IN$
0	1	0	$x_2 = IN$
0	1	1	$x_3 = IN$
1	0	0	$x_4 = IN$
1	0	1	$x_5 = IN$
1	1	0	$x_6 = IN$
1	1	1	$x_7 = IN$

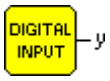
In the table, x_i (i=0,1,…,7) represent the analog output signals, *IN* represents the analog input signal, and b_i (i=0,1,2) are the digital selector inputs. The outputs that are not selected are set to zero, *i.e.*, x_i=0 for $i = 0,1,\ldots 7, i \neq \hat{i}$, where $\hat{i}$ is the index of the selected signal.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also: Multiplexer, A/B Switch, Analog to Binary Converter



11.27 DIGITAL INPUT

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *digital input* operator provides access to one of the eight *FAC-2000* digital inputs. When the digital input signal is below the activation threshold (see the *FAC-2000* Hardware Reference Guide—Fairmount Automation Technical Bulletin 9110-0001), the operator state is *low* (0). When the input signal is above the threshold, the operator state is *high* (1). The operator state will change only if the input signal is maintained for at least the hysteresis time.

User-Defined Properties:

Object name.	A string label that identifies the operator.
Initial State	The input state (*high* or *low*) at startup (at time *t*=0). Ignored when hysteresis time is zero.
Hysteresis	The minimum length of time (in seconds) the input voltage level must remain above the activation threshold (or below the deactivation threshold) for the operator state to switch from *high* to *low* (or from *low* to *high*).

Comments:

The hysteresis time must be greater or equal to zero.

See also:

Analog Input, Universal Digital Input, Analog Output, Digital Output, Relay

11.28 DIGITAL OUTPUT

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *digital output* operator provides access to one of the two *FAC-2000* digital outputs. When the operator state is *low* (0), the digital output is deactivated; when the operator state is *high* (1), the output is activated (see *FAC-2000* Hardware Reference Guide—Fairmount Automation Technical Bulletin 9110-0001 for more information). The output signal will change only if the operator state is maintained for at least the hysteresis time.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time $t = 0$).
Hysteresis	The minimum length of time (in seconds) the operator state must remain *high* (or *low*) for the output to activate (or deactivate).

Comments:

The hysteresis time must be greater or equal to zero.

See also:

Relay, Analog Output, Digital Input, Universal Digital Input, Analog Input

11.29 DIVISION

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *division* operator is the quotient of its inputs

$$y = \frac{x_1}{x_2}$$

where y is the output, x_i (i=1,2) are the inputs. *Design Pad* will issue an error on processing if an input pin is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments:

If the denominator is equal to zero, x_2=0, then the output is set to the maximum 4-byte floating point value: $y = 3.37 \times 10^{38}$.

See also:

Multiplication, Power, Addition, Subtraction, Unit Conversion

11.30 D/A CONVERSION

Design Pad Menu Location: Operators—General Purpose

Functional Description:

Design Pad supports two signal types—analog signals (floating point values) drawn in red, and digital signals (boolean value, *i.e.,* true/false or high/low) drawn in blue. These signal types cannot be directly mixed—analog signals may not be directly connected to digital ones. But, analog signals may be converted into digital ones (and vice-versa). The *D/A Conversion* operator converts a high/low (1/0) boolean value into an analog floating point value. The analog output of the *D/A Conversion* operator is

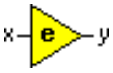
$$y = \begin{cases} 0.0 & \text{if} \quad x=0 \\ 1.0 & \text{if} \quad x=1. \end{cases}$$

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also: A/D Conversion, Binary to Analog Conversion, Analog to Binary Conversion



11.31 EXPONENTIAL

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *exponential* operator is

$$y = e^x$$

where y is the output and x is the input.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments:

None.

See also:

Natural Logarithm, Power

11.32 GAIN

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

The output of the *gain* operator is

$$y = Gx$$

where y is the output, x is the input, and G is the gain factor.

Design Pad will issue an error on processing if the input pin is not connected.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Gain	The gain factor, *G*.

Comments: None.

See also:

Bias, Unit Conversion, Multiplication, Division

11.33 HIGH/LOW ALARM

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *High/Low Alarm* operator compares an analog input signal to high and low alarm levels. The operator's digital output is set *low* (0) if the input signal is between the alarm levels, and is set *high* (1) otherwise. The output will change only if the alarm condition is maintained for a user-specified length of time—the hysteresis time. That is, the output y is defined by

$$y(t) = \begin{cases} 0 & if \quad x_{\min} < x(\bar{t}) < x_{\max} \ \forall \bar{t} \in [t, t-H] \\ 1 & if \ x(\bar{t}) \le x_{\min} \ or \ x(\bar{t}) \ge x_{\max} \ \forall \bar{t} \in [t, t-H] \\ y(t-\Delta t) & \text{otherwise} \end{cases}$$

where x is the input, x_{max} is the high alarm level, and x_{min} is the low alarm level, H is the hysteresis time, and Δt is the loop sampling time.

For example, suppose that the initial output is $y(0)$=0, the high alarm level is x_{max}=80, the low alarm level is x_{min}=20, the hysteresis time is H=2 seconds, and the input signal has the following profile:

$$x(t) = \begin{cases} 50 & 0 \le t < 3 \\ 90 & 3 \le t < 4 \\ 70 & 4 \le t < 7 \\ 85 & 7 \le t < 10 \\ 75 & 10 \le t < 11 \\ 85 & 15 \le t < 20 \\ 25 & 20 \le t < 25 \\ 10 & t \ge 25 \end{cases}$$

Under these conditions, the output of the *High/Low Alarm* operator will be

$$y(t) = \begin{cases} 0 & 0 \le t < 9 \\ 1 & 9 \le t < 22 \\ 0 & 22 \le t < 27 \\ 1 & t \ge 27 \end{cases}$$

Initially, the output is $y(0)$=0 (as specified by the initial condition property). The output remains *low* (0) while $x(t)$=50 since an alarm condition is not met. Although an alarm condition exists while $x(t)$=90, the output remains *low* because the condition is not sustained for two seconds (the hysteresis time). The output switches to *high* (1) at t=9 seconds because

an alarm condition is maintained for at least two seconds. It remains *high* even while *x(t)*=75 (within alarm levels) because the condition $x_{min}<x<x_{max}$ is not sustained for two seconds. At t=20 seconds, the input signal satisfies the no-alarm condition, but the output does not switch to *low* until t=22 seconds. (when the input signal has remained within the alarm levels for at least two seconds.) Likewise, the output will go *high* (and remain *high* thereafter) at t=27 seconds (because the low alarm condition is sustained).

User-Defined Properties:

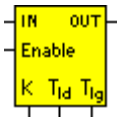
Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time t=0).
Low Alarm	The low alarm level x_{min}.
High Alarm	The high alarm level, x_{max}.
Hysteresis	The hysteresis time, H, expressed in seconds.

Comments:

The high alarm level must exceed the low alarm level, $x_{max} > x_{min}$

See Also:

=, ≠, >, <, ≥, and ≤ Comparators; Thresholding



11.34 LEAD-LAG CONTROLLER

Design Pad Menu Location: Operators—Controller Blocks

Functional Description:

The *Lead-Lag Controller block* implements a lead-lag compensator with transfer function

$$K\frac{T_{ld}s+1}{T_{\lg}s+1}$$

where T_{ld} and $T_{\lg}$ are compensator time constants, and K is the compensator gain. The corresponding discrete-time implementation is

$$y(t)=y(t-\Delta t)+\left[1-e^{-\Delta t/T_{\lg}}\right]\left[Kx(t)-y(t-\Delta t)\right]+K\frac{T_{ld}}{T_{\lg}}\left[x(t)-x(t-\Delta t)\right],$$

where y is the compensator output (*OUT*), x is the compensator input (*IN*), and Δt is the controller's loop sampling time.

In addition to the analog signal input (*IN*) and compensator coefficient inputs (K, T_{ld}, and T_{lg}), the lead-lag controller block has an *Enable* input. When the *Enable* input is *high*, the output is computed as indicated above; when the *Enable* input is *low*, the compensator is disabled and the output simply passes the input, *i.e.*, $y(t) = x(t)$.

User-Defined Properties:

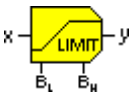
Object Name	A string label that identifies the operator.
Initial Condition	The output value on startup (at time t=0).
Minimum Output	The minimum output value.
Maximum Output	The maximum output value.

Comments:

- The output signal (*OUT*) is limited to user-defined minimum and maximum limits.
- The coefficients T_{ld}, and T_{lg} are expressed in minutes.
- The lag coefficient must be at least 60 milliseconds, *i.e.*, $T_{\lg} \geq 0.001$ min.

See Also:

PD Controller, PI Controller, PID Controller, PID Controller with External Feedback



11.35 LIMITER

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

The *limiter* operator confines its input signal $x(t)$ within an operating band. The output $y(t)$ is defined by

$$y(t) = \begin{cases} B_L(t) & \text{if } x(t) \le B_L(t) \\ B_H(t) & \text{if } x(t) \ge B_H(t) \\ x(t) & \text{otherwise} \end{cases}$$

where limiting inputs $B_L(t)$ and $B_H(t)$ define the lower and upper boundaries of the operating band.

If the signal boundaries do not change over time, inputs $B_L(t)$ and $B_H(t)$ need not be connected. Instead, the signal limits can be defined by the operator properties $x_{\min}$ and $x_{\max}$.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Low Limit	The lower limit of the operating band, $x_{\min}$. This parameter is ignored if input B_L is connected.
High Limit	The upper limit of the operating band, $x_{\max}$. This parameter is ignored if input $B_H(t)$ is connected.
Display Low Limit Pin	This property determines if the input pin B_L is visible in the schema diagram. *Design Pad* issues a warning upon processing the schema when the pin is visible but not connected. The warning message is not issued when the pin is not visible (when this property is not checked).
Display High Limit Pin	This property determines if the input pin B_H is visible in the schema diagram. *Design Pad* issues a warning upon processing the schema when the pin is visible but not connected. The warning message is not issued when the pin is not visible (when this property is not checked).

See also: Rate Limiter, High/Low Alarm

11.36 MOVING AVERAGE

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

The *moving average* operator computes the average of the past N_s input values

$$y(t) = \frac{1}{N_s} \sum_{i=0}^{N_s-1} x(t - i\Delta t)$$

where $y_1(t)$ is the output at time t, Δt is the loop sampling time, N_s are the number of samples, and $x(t - i\Delta t)$ (i= 0,…,N_s-1) are the past N_s input values.

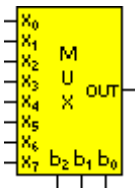
User-Defined Properties:

Object Name	A string label that identifies the operator.
Number of Samples	The number of samples, N_s.

Comments: None.

See also:

Delay Element, Addition, Division, Multiplication



11.37 MULTIPLEXER

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The output of the *multiplexer* operator is equal to one of its eight analog inputs. The analog input that is passed through to the output is determined by state of the three digital inputs, as shown in the table below:

b_2	b_1	b_0	*OUT*
0	0	0	x_0
0	0	1	x_1
0	1	0	x_2
0	1	1	x_3
1	0	0	x_4
1	0	1	x_5
1	1	0	x_6
1	1	1	x_7

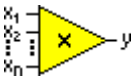
In the table, *OUT* represents the output signal, x_i (i=0,1,…,7) are the analog input signals, b_i (i=0,1,2) are the digital selector inputs.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See also: Demultiplexer, A/B Switch, Analog to Binary Converter



11.38 MULTIPLICATION

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *multiplication* operator is the product of its inputs

$$y = \prod_{i=1}^{n} x_i$$

where y is the output, x_i (i=1,…,n) are the inputs, and n is the number of inputs. *Design Pad* assumes that $x_i \equiv 1$, if input pin i is not connected.

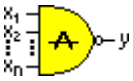
User-Defined Properties:

Object Name	A string label that identifies the operator.
Number of Inputs	The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See also:

Division, Power, Addition, Subtraction, Unit Conversion



11.39 NAND-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the *NAND-Gate* operator is

$$y = \overline{x_1 \wedge x_2 \wedge \cdots \wedge x_n}$$

where y is the output, x_i (i=1,…,n) are the inputs, and n is the number of inputs. In other words, the output state is *low* (0) if all inputs are *high* (1). If any input is *low*, the output is *high*. Below is the truth table for a 3-input NAND-gate:

x_1	x_2	x_3	y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Design Pad assumes that $x_i \equiv 1$, if input pin i is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Number of Inputs The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See Also:

AND-Gate, OR-Gate, NOR-Gate, NOT-Gate, XOR-Gate, XNOR-Gate, RS Flip-Flop

11.40 NATURAL LOGARITHM

Design Pad Menu Location: Operators—Math Functions

Functional Description

The output of the *natural log* operator is

$$y = ln(x)$$

where y is the output and x is the input.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments:

If the input is less than zero, $x < 0$, then the output is set to the minimum 4-byte floating point value: $y = -3.37 \times 10^{38}$.

See Also:

Exponential, Power

11.41 NOR-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the *NOR-Gate* operator is

$$y = \overline{x_1 \vee x_2 \vee \cdots \vee x_n}$$

where y is the output, x_i (i=1,…,n) are the inputs, and n is the number of inputs. In other words, the output state is *low* (0) if any input is *high* (1). If all inputs are *low*, the output is *high*. Below is the truth table for a 3-input OR-gate:

x_1	x_2	x_3	y
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Design Pad assumes that $x_i \equiv 0$, if input pin i is not connected.

User-Defined Properties:

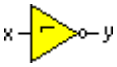
Object Name A string label that identifies the operator.

Number of Inputs The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See Also:

OR-Gate, NOT-Gate, XOR-Gate, XNOR-Gate, NAND-Gate, RS Flip-Flop



11.42 NOT-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the N*OT-Gate* operator is the negation of its input

$$y = \overline{x}$$

where *y* is the output, and *x* are the inputs. In other words, the output state is *low* (0) if the input state *high* (1) and the output state is *high* if the input state is *low*.

Design Pad will issue an error on processing if the input pin is not connected.

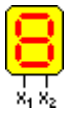
User-Defined Properties:

Object Name	A string label that identifies the operator.

Comments: None.

See Also:

NAND-Gate, NOR-GATE, XNOR-Gate, AND-Gate, OR-Gate, XOR-Gate, RS Flip-Flop



11.43 NUMERIC DISPLAY

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *numeric display* operator can be used to show any schema signal on the *FAC-2000* front panel numeric displays. To show a signal on a numeric display, simply connect it to analog input x_1. The value displayed at time t, $d(t)$, is the moving average of the last N_s samples of input x_1

$$d(t) = \frac{1}{N_s} \sum_{i=0}^{N_s-1} x_1(t - i\Delta t)$$

where Δt is the loop sampling time.

Digital input x_2 controls the presentation of signal x_1 on the display. If $x_2 = 1$, the x_1 value will flash on the numeric display; otherwise, if $x_2 = 0$, the value will be shown continuously. Flashing the contents of the numeric display may be useful to indicate particular operating states. For instance, it could be used to indicate that the controller is in MANUAL mode, or that some problem condition has arisen.

User-Defined Properties:

Decimal Point Position — The position of the decimal point on the numeric display. The decimal point position can be floating, or it can be fixed after the first, second, third, or fourth (if applicable). When in floating mode, the decimal point position will depend on the value of input signal x_1. It will display the signal at the maximum resolution possible (using all the available digits).

Number of Samples — The number of samples N_s to use when computing the signal to be displayed (*i.e.*, the number of samples in the moving average computation).

Brightness — A brightness setting for the LED digits in the range [0-15], with 15 indicating maximum brightness.

Display Blinking Input — If input x_2 is not used *Design Pad* will issue a warning upon processing the schema. You can suppress the warning message by making the input pin invisible. When this property is checked the pin is visible; otherwise it is invisible.

Comments:

If the value of input signal x_1 exceeds the capacity of the display (*e.g.*, 9999 for the four-digit numeric displays when the decimal point is floating, or 9.99 for the three-digit display when the decimal point position is set to 'After First Digit'), the displays will show HHHH or HHH. Likewise, if x_1 is less than the minimum value for the chosen setting (*e.g.*, -99.9 for a four-digit display with a 'After Second Digit' decimal point position, or -99 for a three-digit display with a floating decimal point), the displays will show LLLL or LLL.

See Also:

Bargraph Display, Alphanumeric Display

11.44 OR-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the *OR-Gate* operator is

$$y = x_1 \vee x_2 \vee \cdots \vee x_n$$

where y is the output, x_i (i=1,…n) are the inputs, and n is the number of inputs. In other words, the output state is *high* (1) if any input is *high*. If all inputs are *low* (0), the output is *low*. Below is the truth table for a 3-input OR-gate:

x_1	x_2	x_3	y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Design Pad assumes that x_i=0, if input pin i is not connected.

User-Defined Properties:

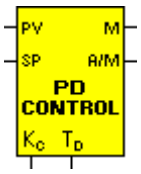
Object Name A string label that identifies the operator.

Number of Inputs The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See Also:

NOR-Gate, XOR-Gate, XNOR-Gate, NOT-Gate, AND-Gate, NAND-Gate, RS Flip-Flop



11.45 PD CONTROLLER

Design Pad Menu Location: Operators—Controller Blocks

Functional Description:

The *PD block* implements the position form of a Proportional-Derivative control algorithm[3]. It computes a control signal M, as the sum of proportional and derivative (rate) control modes:

$$M(t) = B_P(t) + B_D(t).$$

Each mode is based on measurements (samples) of an error signal E. For direct-acting control, this error signal is defined as

$$E = PV - SP,$$

where PV is the process variable and SP is the set-point. For reverse-acting control, the error is defined as

$$E = SP - PV.$$

The direct/reverse acting attribute of a PD controller refers to the relative direction of movement of the process variable and controller output. Under a direct acting configuration, an increase in the process variable results in an increase in the controller output. Under a reverse acting configuration, an increase in the process variable results in a decrease in controller output. The proper setting depends upon the process itself. In most processes, the controller should be set to reverse acting. (Hence, the *FAC-2000* controller defaults to reverse acting.)

The proportional mode term, B_P, is calculated as the product of the proportional gain, K_c, and the error term $E(t)$,

$$B_P(t) = K_c\, E(t).$$

The derivative mode term, B_D, is obtained from

[3] See Corripio, A.B., *Tuning of Industrial Control Systems* (Research Triangle Park, NC: ISA, 1990).

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t} \right] \left[E(t) - E(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

where T_D is the derivative time expressed in minutes and Δt is the loop sampling time expressed in seconds. With this definition, a change in the process set-point can cause undesirable pulses in the derivative mode term. These undesirable pulses, known as derivative kick, can also result from discretization effects and from the noise-amplification effect of differentiation. Derivative kick can be avoided by having the derivative mode act on the process variable, rather than on the error. To avoid derivative kick in direct acting processes, B_D is calculated from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t + \alpha\ 60\ T_D} \right] \left[\alpha\, B_D(t - \Delta t) + PV(t) - PV(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

To avoid derivative kick in reverse acting processes, B_D is calculated from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t + \alpha\ 60\ T_D} \right] \left[\alpha\, B_D(t - \Delta t) - PV(t) + PV(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

The parameter α in the equations above acts as a noise filter. Its reciprocal, $1/\alpha$, is called the "dynamic gain limit" and corresponds to the maximum amplification of the error signal. By default, the *FAC-2000* controller is configured to avoid derivative kick. And, the default value of the filter parameter is set to $\alpha = 0.1$.

In addition to the control signal output *M*, the PD block provides a boolean output labeled *A/M* that indicates if the block is operating in automatic or manual mode. If output *A/M*=0, the operator is in automatic mode; if *A/M*=1, the operator is in manual mode.

When the operator transitions from automatic mode to manual mode, the output remains at the transition value until modified from the front panel.

When the operator switches from manual mode to automatic mode, the transition may be abrupt or smooth (bumpless). If the bumpless transfer property is not selected, the output will switch instantly from the manual output to the automatic output. When bumpless transfer property is checked, the transition from manual to automatic is guaranteed to be

smooth. To achieve bumpless transfer, an artificial reset term is used with time constant T_{mr}^{-1} (the manual reset time expressed in minutes). The PD output after the manual to auto transition will be (approximately)

$$M(t) = B_P(t) + B_D(t) + S_{am} e^{-60(t-t_{am})/T_{mr}} \quad \forall t > t_{am}$$

where S_{am} is the difference between the manual output and automatic output at the time of the switch:

$$S_{am} = M(t_{am}) - B_P(t_{am}) - B_D(t_{am}) .$$

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial Condition	The output value on startup, *i.e.*, $M(0)$.
Initial State	The operator mode (*auto* or *manual*) on startup (at time t=0).
A/M Panel Access	A boolean property indicating if the PD block can be switched between automatic and manual modes from the *FAC-2000* front panel (using the A/M button).
Minimum Output	The minimum controller output value. Applicable to both automatic and manual modes.
Maximum Output	The maximum controller output value. Applicable to both automatic and manual modes.
Manual Increment	The magnitude of output value increments when the operator block is in manual mode.
Bumpless Transfer	When the operator switches from manual mode to automatic mode, the output signal may differ from input x. If this property is not checked, there may be a discontinuity in the output. When bumpless transfer is checked, the transition from manual to automatic is guaranteed to be smooth.
Manual Reset Time (min)	When the integral term is not used (*i.e.*, if $T_I \le 0$), bumpless transfer is achieved with an artificial reset term with time constant T_{mr}^{-1} (the manual reset time expressed in minutes).
Use Alpha-numeric Displays	When this property is selected, the output value will be presented on the alphanumeric displays while the operator is in manual mode.

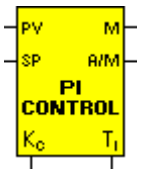
Number of Significant Digits to Display	The number of significant digits to use when displaying the manual output on the alphanumeric displays.
Direct Acting	A boolean property that governs the relative direction of movement of the process variable and controller output. When the property is checked, an increase in the process variable results in an increase in the controller output. When it is not checked, an increase in the process variable results in a decrease in controller output. By default, this property is left unchecked (in most applications, the PD controller should be reverse acting.)
Derivative on Measure	A boolean property that governs the computation of the derivative-mode term. When this property is checked, the derivative mode term is based on the process variable; when it is not checked, it is based on the error. The latter case may result in *derivative kick* when the set-point is modified, or when the operator switches between automatic and manual modes. By default, this property is checked to eliminate derivative kick and to limit the noise amplification effect of differentiation.
Filter parameter, α	A scalar parameter used when the '*Derivative on Measure*' property is checked. It serves to limit the effects of noise and discretization on the derivative-mode term.

Comments:

- The rate time (T_D) inputs of the *PD Controller* operator is measured in minutes.
- If the proportional gain (K_c) input is less than 0.001, 0.001 will be used as the gain in the computation of the output signal.

See Also:

PI Controller, PID Controller, PID with External Feedback, Auto/Manual Button.



11.46 PI CONTROLLER

Design Pad Menu Location: Operators—Controller Blocks

Functional Description:

The *PI block* implements the position form of a Proportional-Integral control algorithm[4]. It computes a control signal *M*, as the sum of proportional and integral (reset) control modes:

$$M(t) = B_P(t) + B_I(t).$$

Each mode is based on measurements (samples) of an error signal *E*. For direct-acting control, this error signal is defined as

$$E = PV - SP,$$

where *PV* is the process variable and *SP* is the set-point. For reverse-acting control, the error is defined as

$$E = SP - PV.$$

The direct/reverse acting attribute of a PI controller refers to the relative direction of movement of the process variable and controller output. Under a direct acting configuration, an increase in the process variable results in an increase in the controller output. Under a reverse acting configuration, an increase in the process variable results in a decrease in controller output. The proper setting depends upon the process itself. In most processes, the controller should be set to reverse acting. (Hence, the *FAC-2000* controller defaults to reverse acting.)

The proportional mode term, B_P, is calculated as the product of the proportional gain, K_c, and the error term $E(t)$,

$$B_P(t) = K_c\, E(t).$$

The integral mode term B_I, is obtained from

[4] See Corripio, A.B., *Tuning of Industrial Control Systems* (Research Triangle Park, NC: ISA, 1990).

$$B_I(t) = \begin{cases} K_c \left[\dfrac{\Delta t}{60\ T_I} \right] S(t) & \text{if} \quad T_I > 0 \\ \\ 0 & \text{if} \quad T_I \le 0 \end{cases}$$

where T_I is the integral or reset time expressed in minutes, Δt is the loop sampling time expressed in seconds, and $S(t)$ is the sum of all past errors, computed from

$$S(t) = \ S(t - \Delta t) + E(t), \quad S(0) = 0$$

In addition to the control signal output M, the PI block provides a boolean output labeled *A/M* that indicates if the block is operating in automatic or manual mode. If output *A/M*=0, the operator is in automatic mode; if *A/M*=1, the operator is in manual mode.

When the operator transitions from automatic mode to manual mode, the output remains at the transition value until modified from the front panel.

When the operator switches from manual mode to automatic mode, the transition may be abrupt or smooth (bumpless). If the bumpless transfer property is not selected, the output will switch instantly from the manual output to the automatic output. When bumpless transfer property is checked, the transition from manual to automatic is guaranteed to be smooth. To achieve bumpless transfer, the integral term is matched to the difference between the manual output and automatic output at the time of the switch, t_{am},

$$B_I\left(t_{am}\right) = M(t_{am}) - B_P(t_{am})$$

When the integral term is not used (*i.e.*, if $T_I \le 0$), bumpless transfer is achieved with an artificial reset term with time constant T_{mr}^{-1} (the manual reset time expressed in minutes). The PI output after the manual to auto transition will be (approximately)

$$M(t) = B_P(t) + S_{am} e^{-60\left(t - t_{am}\right)/T_{mr}} \quad \forall t > t_{am}$$

where S_{am} is the difference between the manual output and automatic output at the time of the switch:

$$S_{am} = M(t_{am}) - B_P(t_{am}).$$

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial Condition	The output value on startup, *i.e.*, $M(0)$.
Initial State	The operator mode (*auto* or *manual*) on startup (at time t=0).

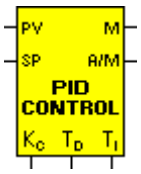
A/M Panel Access	A boolean property indicating if the PIblock can be switched between automatic and manual modes from the *FAC-2000* front panel (using the A/M button).
Minimum Output	The minimum controller output value. Applicable to both automatic and manual modes.
Maximum Output	The maximum controller output value. Applicable to both automatic and manual modes.
Manual Increment	The magnitude of output value increments when the operator block is in manual mode.
Bumpless Transfer	When the operator switches from manual mode to automatic mode, the output signal may differ from input *x*. If this property is not checked, there may be a discontinuity in the output. When bumpless transfer is checked, the transition from manual to automatic is guaranteed to be smooth.
Manual Reset Time (minutes)	When the integral term is not used (*i.e.*, if $T_I \leq 0$), bumpless transfer is achieved with an artificial reset term with time constant T_{mr}^{-1} (the manual reset time expressed in minutes).
Use Alpha-numeric Displays	When this property is selected, the output value will be presented on the alphanumeric displays while the operator is in manual mode.
Number of Significant Digits to Display	The number of significant digits to use when displaying the manual output on the alphanumeric displays.
Direct Acting	A boolean property that governs the relative direction of movement of the process variable and controller output. When the property is checked, an increase in the process variable results in an increase in the controller output. When it is not checked, an increase in the process variable results in a decrease in controller output. By default, this property is left unchecked (in most applications, the PD controller should be reverse acting.)

Comments:

- The reset time (T_I) input of the *PI Controller* operator is measured in minutes.
- If the proportional gain (K_c) input is less than 0.001, 0.001 will be used as the gain in the computation of the output signal.

See Also:

PD Controller, PID Controller, PID with External Feedback, Auto/Manual Button.



11.47 PID CONTROLLER

Design Pad Menu Location: Operators—Controller Blocks

Functional Description:

The *PID block* implements the position form of a Proportional-Integral-Derivative control algorithm[5]. It computes a control signal *M*, as the sum of proportional, integral (reset), and derivative (rate) control modes:

$$M(t) = B_P(t) + B_I(t) + B_D(t).$$

Each mode is based on measurements (samples) of an error signal *E*. For direct-acting control, this error signal is defined as

$$E = PV - SP,$$

where *PV* is the process variable and *SP* is the set-point. For reverse-acting control, the error is defined as

$$E = SP - PV.$$

The direct/reverse acting attribute of a PID controller refers to the relative direction of movement of the process variable and controller output. Under a direct acting configuration, an increase in the process variable results in an increase in the controller output. Under a reverse acting configuration, an increase in the process variable results in a decrease in controller output. The proper setting depends upon the process itself. In most processes, the controller should be set to reverse acting. (Hence, the *FAC-2000* controller defaults to reverse acting.)

The proportional mode term, B_P, is calculated as the product of the proportional gain, K_c, and the error term $E(t)$,

$$B_P(t) = K_c\, E(t).$$

The integral mode term B_I, is obtained from

[5] See Corripio, A.B., *Tuning of Industrial Control Systems* (Research Triangle Park, NC: ISA, 1990).

$$B_I(t) = \begin{cases} K_c \left[\dfrac{\Delta t}{60\ T_I} \right] S(t) & \text{if} \quad T_I > 0 \\ 0 & \text{if} \quad T_I \le 0 \end{cases},$$

where T_I is the integral or reset time expressed in minutes, Δt is the loop sampling time expressed in seconds, and $S(t)$ is the sum of all past errors, computed from

$$S(t) = \ S(t - \Delta t) + E(t), \quad S(0) = 0$$

Finally, the derivative mode term, B_D, is obtained from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t} \right] \left[E(t) - E(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

where T_D is the derivative time, expressed in minutes. With this definition, a change in the process set-point can cause undesirable pulses in the derivative mode term. These undesirable pulses, known as derivative kick, can also result from discretization effects and from the noise-amplification effect of differentiation. Derivative kick can be avoided by having the derivative mode act on the process variable, rather than on the error. To avoid derivative kick in direct acting processes, B_D is calculated from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t + \alpha\ 60\ T_D} \right] \left[\alpha\, B_D(t - \Delta t) + PV(t) - PV(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

To avoid derivative kick in reverse acting processes, B_D is calculated from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t + \alpha\ 60\ T_D} \right] \left[\alpha\, B_D(t - \Delta t) - PV(t) + PV(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

The parameter α in the equations above acts as a noise filter. Its reciprocal, $1/\alpha$, is called the "dynamic gain limit" and corresponds to the maximum amplification of the error signal.

By default, the *FAC-2000* controller is configured to avoid derivative kick. And, the default value of the filter parameter is set to $\alpha = 0.1$.

In addition to the control signal output *M*, the PID block provides a boolean output labeled *A/M* that indicates if the block is operating in automatic or manual mode. If output *A/M*=0, the operator is in automatic mode; if *A/M*=1, the operator is in manual mode.

When the operator transitions from automatic mode to manual mode, the output remains at the transition value until modified from the front panel.

When the operator switches from manual mode to automatic mode, the transition may be abrupt or smooth (bumpless). If the bumpless transfer property is not selected, the output will switch instantly from the manual output to the automatic output. When bumpless transfer property is checked, the transition from manual to automatic is guaranteed to be smooth. To achieve bumpless transfer, the integral term is matched to the difference between the manual output and automatic output at the time of the switch, t_{am},

$$B_I(t_{am}) = M(t_{am}) - B_P(t_{am}) - B_D(t_{am})$$

When the integral term is not used (*i.e.*, if $T_I \leq 0$), bumpless transfer is achieved with an artificial reset term with time constant T_{mr}^{-1} (the manual reset time expressed in minutes). The PID output after the manual to auto transition will be (approximately)

$$M(t) = B_P(t) + B_D(t) + S_{am} e^{-60(t-t_{am})/T_{mr}} \quad \forall t > t_{am}$$

where S_{am} is the difference between the manual output and automatic output at the time of the switch:

$$S_{am} = M(t_{am}) - B_P(t_{am}) - B_D(t_{am}).$$

User-Defined Properties:

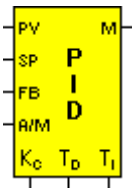
Object Name	A string label that identifies the operator.
Initial Condition	The output value on startup, *i.e.*, *M*(0).
Initial State	The operator mode (*auto* or *manual*) on startup (at time *t*=0).
A/M Panel Access	A boolean property indicating if the PID block can be switched between automatic and manual modes from the *FAC-2000* front panel (using the A/M button).
Minimum Output	The minimum controller output value. Applicable to both automatic and manual modes.
Maximum Output	The maximum controller output value. Applicable to both automatic and manual modes.

Manual Increment	The magnitude of output value increments when the operator block is in manual mode.
Bumpless Transfer	When the operator switches from manual mode to automatic mode, the output signal may differ from input *x*. If this property is not checked, there may be a discontinuity in the output. When bumpless transfer is checked, the transition from manual to automatic is guaranteed to be smooth.
Manual Reset Time (minutes)	When the integral term is not used (*i.e.*, if $T_I \leq 0$), bumpless transfer is achieved with an artificial reset term with time constant T_{mr}^{-1} (the manual reset time expressed in minutes).
Use Alpha-numeric Displays	When this property is selected, the output value will be presented on the alphanumeric displays while the operator is in manual mode.
Number of Significant Digits to Display	The number of significant digits to use when displaying the manual output on the alphanumeric displays.
Direct Acting	A boolean property that governs the relative direction of movement of the process variable and controller output. When the property is checked, an increase in the process variable results in an increase in the controller output. When it is not checked, an increase in the process variable results in a decrease in controller output. By default, this property is left unchecked (in most applications, the PD controller should be reverse acting.)
Derivative on Measure	A boolean property that governs the computation of the derivative-mode term. When this property is checked, the derivative mode term is based on the process variable; when it is not checked, it is based on the error. The latter case may result in *derivative kick* when the set-point is modified, or when the operator switches between automatic and manual modes. By default, this property is checked to eliminate derivative kick and to limit the noise amplification effect of differentiation.
Filter Parameter, α	A scalar parameter used when the '*Derivative on Measure*' property is checked. It serves to limit the effects of noise and discretization on the derivative-mode term.

Comments:

- The reset time (T_I) and rate time (T_D) inputs of the *PID Controller* operator are measured in minutes.
- If the proportional gain (K_c) input is less than 0.001, 0.001 will be used as the gain in the computation of the output signal.

See Also: PD Controller, PI Controller, PID with External Feedback, Auto/Manual Button.



11.48 PID with EXTERNAL FEEDBACK

Design Pad Menu Location: Operators—Controller Blocks

Functional Description:

The *PID with External Feedback block* implements the position form of a Proportional-Integral-Derivative control algorithm[6]. Like the *PID Controller* block, it computes a control signal *M*, as the sum of proportional, integral (reset), and derivative (rate) control modes:

$$M(t) = B_P(t) + B_I(t) + B_D(t).$$

Unlike the *PID Controller* block, it computes the reset mode B_I from an external feedback signal *FB*

$$B_I(t) = B_I(t - \Delta t) + \left[FB(t) - B_I(t - \Delta t)\right]\frac{60\Delta T}{T_I},$$

where T_I is the integral or reset time, expressed in minutes, and where Δt is the loop sampling time. If the signal that is fed back is simply the output signal *M, i.e.,*

$$FB(t) = M(t - \Delta t),$$

then, the *PID Controller* operator and the *PID with External Feedback* operator will function identically. The external feedback version is useful if you intend to do additional processing between the output signal *M* and the end actuator (see the schema named '`PID2.SCM`' in the `Fairmount Automation/Examples` directory for a demonstration). The *PID Controller* block should be used when output *M* drives an actuator directly.

The proportional and rate modes are based on measurements (samples) of an error signal *E*. For direct-acting control, this error signal is defined as

$$E = PV - SP,$$

where *PV* is the process variable and *SP* is the set-point. For reverse-acting control, the error is defined as

$$E = SP - PV.$$

The direct/reverse acting attribute of a PID controller refers to the relative direction of movement of the process variable and controller output. Under a direct acting configuration,

[6] See Corripio, A.B., *Tuning of Industrial Control Systems* (Research Triangle Park, NC: ISA, 1990).

an increase in the process variable results in an increase in the controller output. Under a reverse acting configuration, an increase in the process variable results in a decrease in controller output. The proper setting depends upon the process itself. In most processes, the controller should be set to reverse acting. (Hence, the *FAC-2000* controller defaults to reverse acting.)

The proportional mode term, B_P, is calculated as the product of the proportional gain, K_c, and the error term $E(t)$,

$$B_P(t) = K_c\, E(t).$$

The derivative mode term, B_D, is obtained from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t} \right] \left[E(t) - E(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

where T_D is the derivative time, expressed in minutes. With this definition, a change in the process set-point can cause undesirable pulses in the derivative mode term. These undesirable pulses, known as derivative kick, can also result from discretization effects and from the noise-amplification effect of differentiation. Derivative kick can be avoided by having the derivative mode act on the process variable, rather than on the error. To avoid derivative kick in direct acting processes, B_D is calculated from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t + \alpha\ 60\ T_D} \right] \left[\alpha\, B_D(t - \Delta t) + PV(t) - PV(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

To avoid derivative kick in reverse acting processes, B_D is calculated from

$$B_D(t) = \begin{cases} K_c \left[\dfrac{60\ T_D}{\Delta t + \alpha\ 60\ T_D} \right] \left[\alpha\, B_D(t - \Delta t) - PV(t) + PV(t - \Delta t) \right] & \text{if} \quad T_D > 0 \\ 0 & \text{if} \quad T_D \le 0 \end{cases}$$

The parameter α in the equations above acts as a noise filter. Its reciprocal, $1/\alpha$, is called the "dynamic gain limit" and corresponds to the maximum amplification of the error signal. By default, the *FAC-2000* controller is configured to avoid derivative kick. And, the default value of the filter parameter is set to $\alpha = 0.1$.

The *PID with External Feedback* block provides a boolean input labeled *A/M* that indicates the operating mode: automatic or manual. If input *A/M*=0, the operator is in automatic mode and the output is computed as explained above. If input *A/M*=1, the operator is in manual mode; the output is tied directly to the feedback signal, *i.e.,*

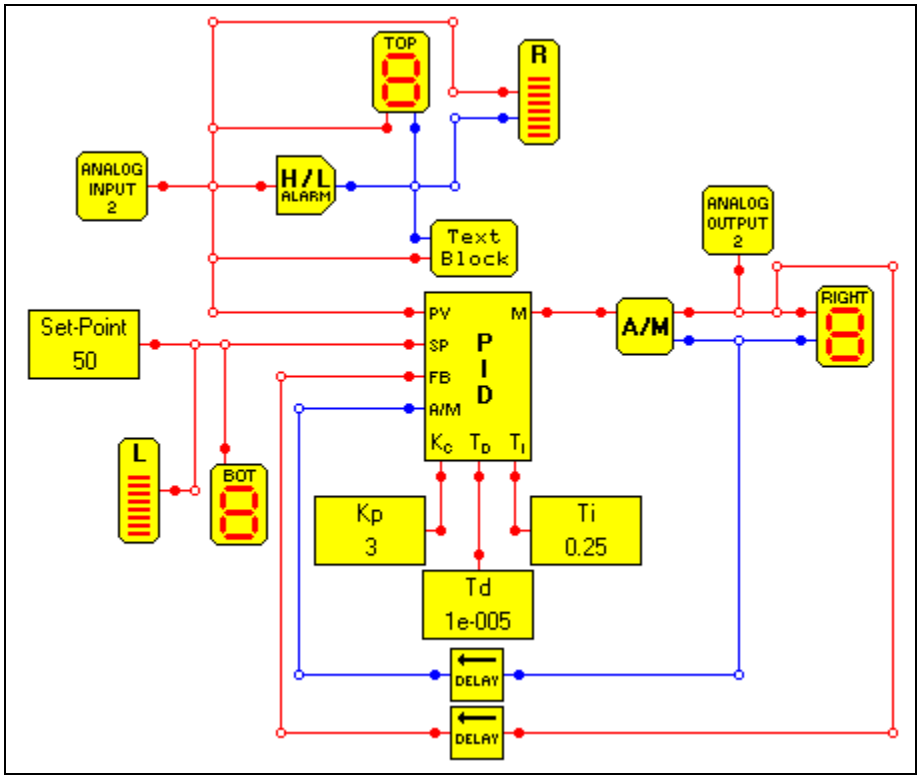
$$M(t) = FB(t) \text{ if } A/M=1.$$

And, the integral term is matched to the difference between the feedback signal and the automatic output:

$$B_I(t) = FB(t) - B_P(t) - B_D(t).$$

With this arrangement, the transition between automatic and manual modes will be bumpless.

A sample schema demonstrating the use of this *PID with External Feedback* block is shown in the figure below.



In the figure, the *PID with External Feedback* block is used in conjunction with an external *A/M Button* block. Note that the *Bumpless Transfer* property of the A/M block should not be selected, since bumpless transfer between automatic and manual modes is built-into the controller block.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial Condition	The output value on startup, *i.e.*, $M(0)$.
Minimum Output	The minimum controller output value.
Maximum Output	The maximum controller output value.
Direct Acting	A boolean property that governs the relative direction of movement of the process variable and controller output. When the property is checked, an increase in the process variable results in an increase in the controller output. When it is not checked, an increase in the process variable results in a decrease in controller output. By default, this property is left unchecked (in most applications, the PD controller should be reverse acting.)
Derivative on Measure	A boolean property that governs the computation of the derivative-mode term. When this property is checked, the derivative mode term is based on the process variable; when it is not checked, it is based on the error. The latter case may result in *derivative kick* when the set-point is modified, or when the operator switches between automatic and manual modes. By default, this property is checked to eliminate derivative kick and to limit the noise amplification effect of differentiation.
Filter Parameter, α	A scalar parameter used when the '*Derivative on Measure*' property is checked. It serves to limit the effects of noise and discretization on the derivative-mode term.

Comments:

- The reset time (T_I) and rate time (T_D) inputs of the *PID Controller* operator are measured in minutes.
- If the reset time (T_I) input is less than 0.001 minutes, 0.001 will be used as the reset time in the computation of the integral term.
- If the proportional gain (K_c) input is less than 0.001, 0.001 will be used as the gain in the computation of the output signal.

See Also:

PD Controller, PI Controller, PID Controller, Auto/Manual Button.

11.49 POWER

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *power* operator is:

$$y = (x_1)^{x_2}$$

where y is the output, x_i (i=1,2) are the inputs.

Design Pad will issue an error on processing if an input pin is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also:

Exponential, Multiplication

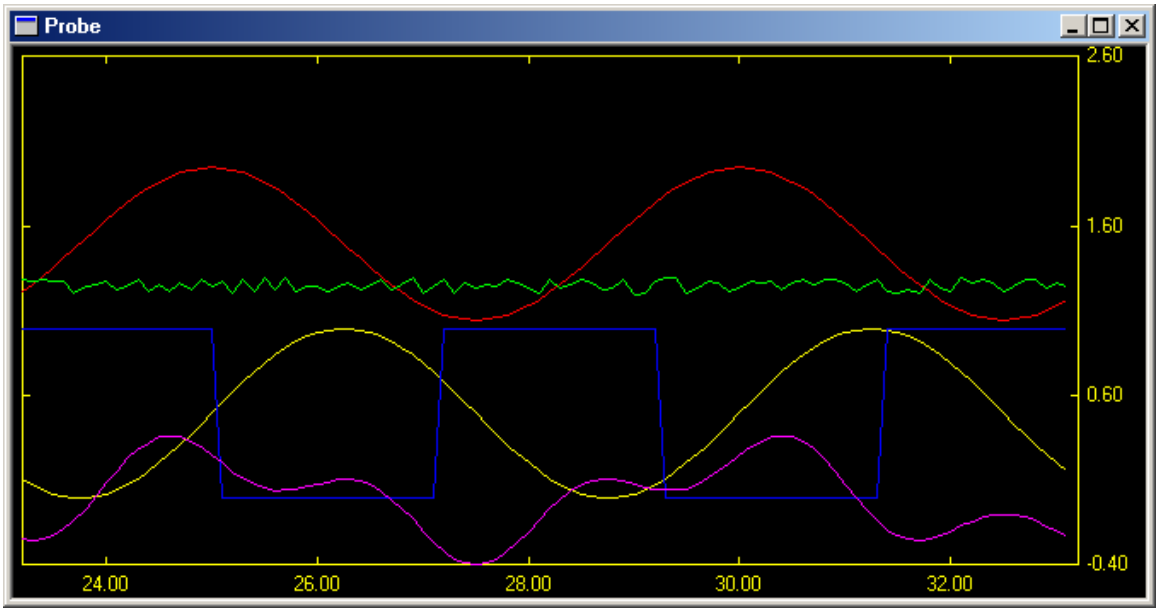
11.50 PROBE (ANALOG or DIGITAL)

Design Pad Menu Location: Operators

Functional Description:

The probe operator displays the value of its input signal x_1 during a *Design Pad* simulation. The time-evolution of the signal is also displayed in graphic form when the operator is double-clicked with the right mouse button.

The probe operator can graph up to five input signals. A different color is used to represent each input signal: yellow for x_1, red for x_2, green for x_3, blue for x_4, and purple for x_5.



In addition to graphing signals, probe operators can also serve as data recorders, streaming their input signals to disk. When you execute a simulation, the probe operator record its input signal values in a tab-delimited text file. This file format can be interpreted by most data analysis tools (*e.g.*, Microsoft Excel). By default, the probe data is not recorded to disk. You must explicitly enable disk streaming (see operator properties below) and you must provide the name of a file where the data is to be recorded.

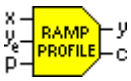
User-Defined Properties:

Object Name	A string label that identifies the operator. The object name is used as the window title for the operator's graph window.

Number of Inputs	The number of signals to be plotted on the operator graph window.
Number of Significant Digits to Display	During a simulation the probe operator icon displays the value of input x_1. This parameter determines the number of significant digits used to display the input signal.
Stream to Disk	If this property is selected, the input signal values will be recorded to a file. The format of the file is tab-delimited text.
Output Filename	The name of the file (including the directory path) used to record the simulation data. If the file already exists it will be overwritten when a simulation executes.

Comments:

Design Pad issues a warning message upon processing the schema if one of the probe input pins is not connected.



11.51 RAMP PROFILE

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The *ramp profile* operator produces a sequence of linear ramps defined by an eleven-point (time, output value) data set:

$$\left\{(t_i, y_i) \,\middle|\, 0 \le t_i \le t_{i+1}\right\}.$$

The operator has two analog inputs *x* and y_e, one digital input *p*, one analog output *y*, and one digital output *c*. Digital input *p* activates the ramp sequence. While *p* is *low* (0), the operator tracks the input, *i.e., y(t)=x(t)*. While *p* is *high* (1), the operator executes the ramp sequence, beginning with the ramp segment where the output value matches the input y_e. That is, the output *y* of the ramp profile operator is

$$y(t) = \begin{cases} x(t) & \text{if} \quad p = 0 \\ y_0 & \text{if} \quad p = 1 \ \text{ and } \ t - t_p < t_0 \\ t_i + (y_{i+1} - y_i)\dfrac{t - t_p - t_i}{t_{i+1} - t_i} & \text{if} \quad p = 1 \ \text{ and } \ t_i \le t - t_p < t_{i+1} \ \ (i = 0,1,\ldots,9) \\ y_{10} & \text{if} \quad p = 1 \ \text{ and } \ t - t_p \ge t_{10} \end{cases}$$

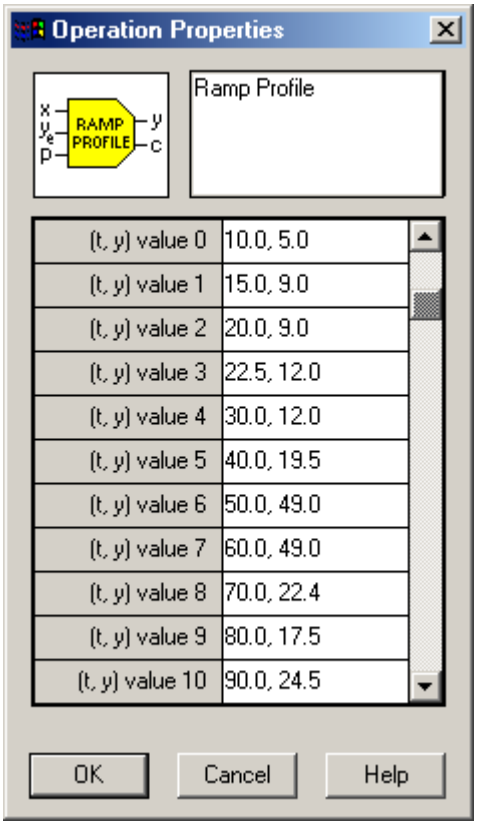
where t_p is a time reference indicating when the ramp sequence begins. If input y_e is not connected, then t_p is equal to the instant that digital input *p* switches to *high* (*i.e.*, $t_p = t_{sp}$). Otherwise, if input y_e is connected, then the time reference t_p is determined from:

$$t_p = t_{sp} + \frac{(t_{i+1} - t_i)(y_e - y_i)}{y_{i+1} - y_i}$$

for the smallest index $i \in [0,9]$ that satisfies either $y_i \le y_e \le y_{i+1}$ or $y_i \ge y_e \ge y_{i+1}$. If no such index exists (*i.e.*, if $y_e > \max y_i$ and $y_e < \min y_i$) then the entire ramp sequence is executed (*i.e.*, $t_p = t_{sp}$).

Digital output *c* indicates when the ramp sequence is complete. It is defined by

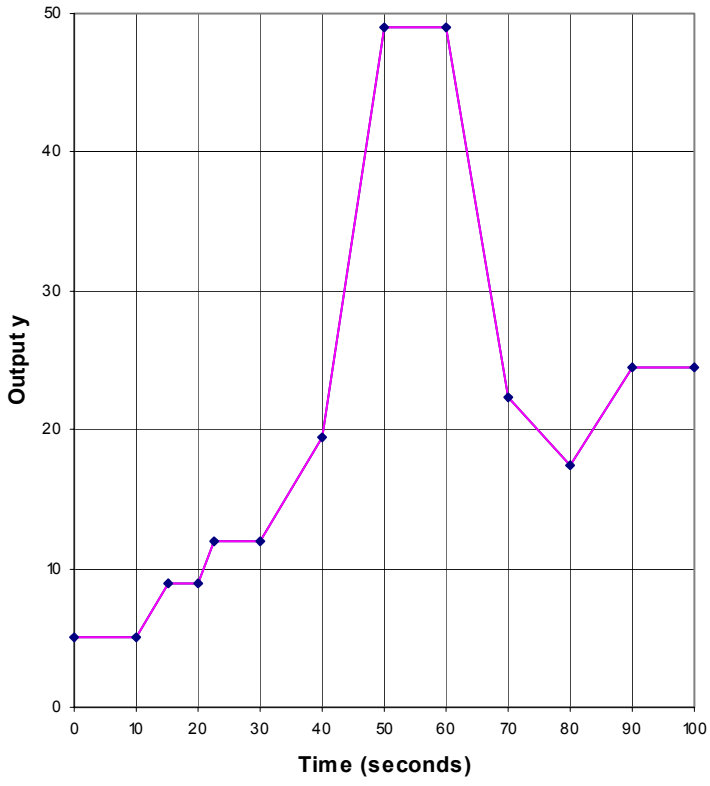
$$c = \begin{cases} 1 & \text{if} \quad p = 1 \ \text{ and } \ t - t_p \ge t_{10} \\ 0 & \text{if} \quad \text{otherwise} \end{cases}$$

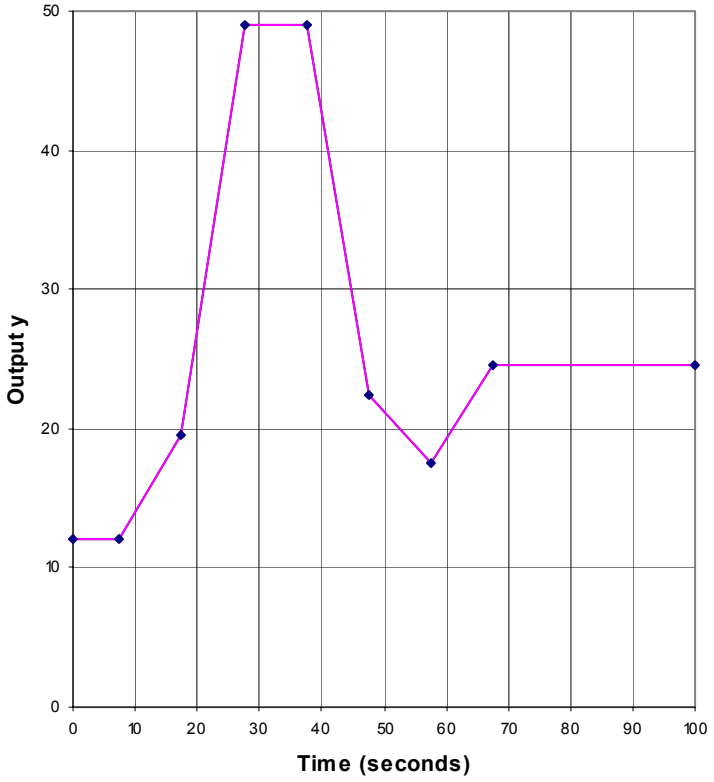


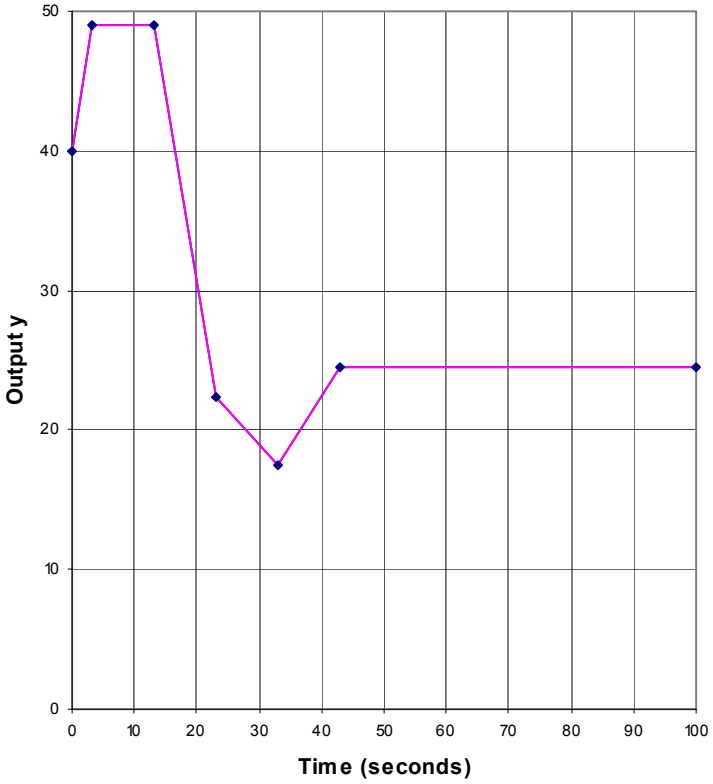
Consider the property sheet shown to the left for a sample ramp profile operator. When input *p* is activates the operator, the portion of the ramp sequence that will be executed will depend on the value of input y_e at the activation time.

If input y_e is not connected then the entire ramp sequence will be executed, as shown in the figure below. The entire sequence will also be executed if input y_e is outside of the ramp range (*i.e.*, if $y_e < 5.0$ or $y_e > 49.0$).

When input y_e is connected, the operator will begin the ramp sequence with the first output value matching y_e. For instance, when $y_e = 12.0$, the ramp output will begin with the fourth segment of the ramp sequence (skipping the first 22.5 seconds). And when $y_e = 40.0$, the output will begin with the seventh segment of the ramp sequence (skipping the first 46.77 seconds).




Output y
Time (seconds)
0
10
20
30
40
50
60
70
80
90
100


Output y
Time (seconds)
0
10
20
30
40
50
60
70
80
90
100

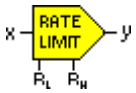
User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial Condition	The output value on startup (at time t=0).
(t_0, y_0)	First (time, output value) data point in ramp sequence.
(t_1, y_1)	Second (time, output value) data point in ramp sequence.
⋮	
(t_{10}, t_{10})	Last (time, output value) data point in ramp sequence.

Comments:

- The time coordinates must be non-negative and may not exceed 28,800 seconds (8 hours), *i.e.,* $0 \le t_i \le 28800, i = 0,1,...,10.$
- The time coordinates of the ramp sequence must be monotonically increasing, *i.e.,* $t_i \le t_{i+1}, i = 0,1,...,9.$
- Output y will be discontinuous if $y_0 \ne x(t_p)$.

See Also: Characterizer



11.52 RATE LIMITER

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

The *rate limiter* operator restricts a signal's rate of change to a specified operating band: It prevents a signal from growing faster than the rising rate limit, input $R^+(t)$. And, it prevents a signal from decreasing in value faster than the fall rate limit, $R^-(t)$.

In mathematical terms, the rate limiter output is defined by

$$y(t) = \begin{cases} y(t-\Delta t) + R^+(t)\Delta t & \text{if} \quad x(t) \geq x(t-\Delta t) \quad \text{and} \quad \Delta x \geq R^+(t)\Delta t \\ y(t-\Delta t) + \Delta x \Delta t & \text{if} \quad x(t) \geq x(t-\Delta t) \quad \text{and} \quad \Delta x < R^+(t)\Delta t \\ y(t-\Delta t) - R^-(t)\Delta t & \text{if} \quad x(t) < x(t-\Delta t) \quad \text{and} \quad \Delta x \geq R^-(t)\Delta t \\ y(t-\Delta t) - \Delta x \Delta t & \text{if} \quad x(t) < x(t-\Delta t) \quad \text{and} \quad \Delta x < R^-(t)\Delta t \end{cases}$$

where y is the output, $\Delta x = |x(t) - x(t-\Delta t)|$ is the change in the input value, Δt is the loop sampling time, input $R^+(t) \geq 0$ is the maximum positive rate of change, and input $R^-(t) \geq 0$ is the maximum negative rate of change.

If the rate limits will not change over time then inputs $R^+(t)$ and $R^-(t)$ need not be connected. Instead, the rate limits can be defined by the operator properties $\Delta x^+_{\max}$ and $\Delta x^-_{\max}$.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Rise Rate Limit	The maximum positive rate of change in the input signal, $\Delta x^+_{\max} \geq 0$.
Fall Rate Limit	The maximum negative rate of change in the input signal, $\Delta x^-_{\max} \geq 0$.
Display Rise Rate Limit	This property determines if the input pin R^+ is visible in the schema diagram. *Design Pad* issues a warning upon processing the schema when the pin is visible but not connected. The warning

message is not issued when the pin is not visible (when this property is not checked).

Display Fall Rate Limit

This property determines if the input pin R^- is visible in the schema diagram. *Design Pad* issues a warning upon processing the schema when the pin is visible but not connected. The warning message is not issued when the pin is not visible (when this property is not checked).

Comments: None.

See Also:

Limiter, Delay Element

11.53 RECEIVER

Design Pad Menu Location: Operators—Network Operators

Functional Description:

Receiver operators decode messages that were broadcasted over a communications network. The receiver outputs y_1 and y_2 are the broadcast signals obtained from the network. And output *b* indicates if the broadcast transmissions are being received regularly (*i.e.*, indicates if the network link is good).

There are three types of analog receivers: analog, digital, and mixed. Analog and digital receivers do not have an output y_2.

Analog receiver operators decode analog signals broadcasted over a communications network. They have two outputs—one analog and one digital. Analog output y_1 is the broadcast signal that the operator receives. Digital output *b* indicates if the operator is actively receiving the broadcasted signal: the output is *low* (0) while transmissions are regularly received; the output is *high* (1) if the operator has not received a broadcast in a user-specified time interval (the broadcast dead-time property of the receiver).

Analog broadcast signals are encoded into a 16-bit integer that can assume $2^{16} = 65536$ distinct values (0-65535). The receiver operator decodes the 16-bit integer into an analog value in the range $y_1 \in [y_{0\%}, y_{100\%}]$ as follows:

$$y_1 = y_{0\%} + \frac{m(y_{100\%} - y_{0\%})}{65535}$$

where *m* the 16-bit integer. The signal mappings $y_{0\%}$ and $y_{100\%}$ are properties of the receiver operator. The broadcast operator uses its own signal mapping properties ($x_{0\%}$ and $x_{100\%}$) to encode the broadcast signal. If the receiver mapping properties are not the same as the broadcast mapping properties, the output of the receiver operator will not be the same as the input to the broadcast operator.

Digital receiver operators decode digital signals broadcasted over a communications network. They have two digital outputs. Digital output y_1 is the decoded broadcast signal and digital output *b* is the broadcast status.

Mixed receiver operators decode communication messages consisting of both analog and digital signals. They have three outputs—one analog and two digital. Digital output *b* indicates if the operator is actively receiving broadcasts. Analog output y_1 and digital output y_2 are the analog and digital components of decoded communication message that the operator receives. (The communications message uses 15 bits to represent the analog signal value and 1 bit to represent the digital signal value.)

In order to associate a receiver operator to a particular network broadcast, you must set the signal name property of the receiver to the broadcast signal name. The signal name in the receiver must match the broadcast signal name exactly. (Signal names are case sensitive—*Design Pad* considers the names *OilPressure*, *oilPressure*, and *OILPRESSURE* to refer to different signals.*)*

User-Defined Properties:

Signal Name	A string label that identifies the broadcast signal that the receiver is "tuned" to. *Design Pad* associates this receiver operator in one schema with a broadcast operator in another schema that has the same signal name..
Communication Medium	*FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the receiver operator should be "listening" to.
Maximum Broadcast Dead-Time	The period of time (in milliseconds) that may elapse before the receiver operator indicates that it has not received a broadcast signal (switches output *b* to *high*). Output *b* remains *low* while the operator actively receives the broadcasted signal.
0% Mapping	The lower limit of the y_1 output signal range, $y_{0\%}$.
100% Mapping	The upper limit of the y_1 output signal range, $y_{100\%}$.
Simulation Mode	This parameter applies only to operators that are part of HMI (`.hmi`) documents; it does not apply to operators that are part of schema (`.scm`) documents. When executing an HMI document simulation, *Design Pad* can emulate a controller communications network (select *Use Virtual PC Network*) or it can be part of an actual *FAIRNET* network (select *Use FAIRNET Network*).

Comments: None.

See Also:

Broadcast

11.54 RELAY OUTPUT

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *relay output* operator provides access to one of the four *FAC-2000* relay outputs. When the operator state is *low* (0), the relay is deactivated; when the operator state is *high* (1), the relay is activated. The relay activation state will change only if the operator state is maintained for at least the hysteresis time.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time t=0).
Hysteresis	The minimum length of time (in seconds) the operator state must remain *high* (or *low*), for the relay to activate (or de-activate).

Comments:

The hysteresis time must greater than or equal to zero.

See Also:

Digital Output, Digital Input, Universal Digital Input, Analog Output, Analog Input

11.55 REMOTE AUTO/MANUAL BUTTON

Design Pad Menu Location: Operators—Network Operators

Functional Description:

The *Remote A/M Button* operator is a special type of network operator that complements the *Auto/Manual Button* operator. Together, they allow synchronized auto/manual operation of multiple control stations attached to a *FAIRNET* network. For example, consider the case where one station contains a network-enabled *A/M Button* operator and two other stations on the same controller network each contain a corresponding *Remote A/M Button* operator. If you switch any one of the three stations from automatic mode to manual mode, the other two stations will also switch to manual operation. If you then adjust the manual output from any one of the three stations, the other two will match that manual output. If you switch any one of the stations back into automatic mode, the other two stations will also switch into automatic mode. When the stations operate in automatic mode, the output signal of the *Remote A/M Button* operator follows the output of the *A/M Button* operator.

In order to associate a *Remote A/M Button* operator in one schema to an *A/M Button* operator in another, you must match the signal name property of the operators. To synchronize multiple control stations, the schema of one station must have an *A/M Button* operator and the schemas of each additional station must each have a *Remote A/M Button* operator (and the signal name property of all operators must have be same case-sensitive character string).

The *Remote A/M Button* operator has one analog output and two digital outputs. Digital output b_1 indicates the operating mode: it is *high* (1) when the operator is in manual mode; it is *low* (0) when the operator is in automatic mode. Digital output b_2 serves as a network status signal indicating if the *Remote A/M Button* operator is actively receiving broadcasts from its associated *A/M Button* operator. The output is *low* (0) while transmissions are regularly received; the output is *high* (1) if the operator has not received a broadcast in a user-specified time interval (the broadcast dead-time property of this operator).

Analog output y depends on the operating mode (automatic or manual) and on the network status. When in manual mode, output y is set by the LEFT and RIGHT ARROW keys of the device front panel (or, while the network is active, the arrow keys of associated devices). When in automatic mode, output y tracks the output of the associated *A/M Button* operator while the network is active. If the operator has ceased receiving network transmissions and it is in automatic mode, output y tracks input x, (*i.e.*, $y = x$).

Remote A/M Button and *A/M Button* operators are synchronized over a *FAIRNET* network. When the output(s) of one operator changes, the output(s) of the associated operators will follow. But what happens when conflicting commands are simultaneously issued from different control stations? Say a user in one station increases the manual output while a user in another simultaneously reduces the manual output. The station executing the schema with the *A/M Button* operator arbitrates conflicting commands. The *A/M Button* operator

continuously broadcasts the output state that all associated *Remote A/M Button* operators should assume. When a *Remote A/M Button* operator deviates from this broadcasted state (*e.g.*, one of its front-panel keys is pressed), it sends a message to the *A/M Button*. If the *A/M Button* accepts this command it modifies its output state and broadcasts it over the network. The broadcast of the new output state serves to confirm that the *A/M Button* has accepted the *Remote A/M Button* command and to alert all other *Remote A/M Button* operators that they should update their output states. If the *Remote A/M Button* does not receive confirmation that its message has been accepted after T_D seconds, it reverts to tracking the output state of the *A/M Button*. Parameter T_D is a property of the *Remote A/M Button* operator labeled *Deviation Time from Remote Signal*.

When the operator transitions from automatic mode to manual mode, output *y* remains at the transition value until modified from the front panel (of any associated control station).

When the operator switches from manual mode to automatic mode, the transition may be abrupt or smooth (bumpless). If the bumpless transfer property is not selected, the output will switch instantly from the manual output to the automatic output. When bumpless transfer property is checked, the transition from manual to automatic is guaranteed to be smooth. The nature of the transition (*i.e.*, the transfer profile) can be either exponential or linear.

Let S_{am} be the difference between the manual output and the automatic output at the instant the operator switches from manual to automatic, $t=t_{am}$. Under an exponential transfer profile, the operator output will be (approximately)

$$y(t) = x(t) + S_{am} e^{-60(t-t_{am})/r} \quad \forall t \geq t_{am}$$

where *r* is a time constant expressed in minutes. Under a linear transfer profile, the operator output will be (approximately)

$$y(t) = \begin{cases} x(t) + \max\left[0, S_{am} - r\left(t - t_{am}\right)\right] & \text{if} \quad S_{am} > 0 \\ x(t) + \min\left[0, S_{am} + r\left(t - t_{am}\right)\right] & \text{if} \quad S_{am} < 0 \end{cases}$$

where *r* is a linear decay rate expressed in (input) units/minute.

Note: The following description of the use of input x_{so} and input x_{mo} applies only while the network connection is not active. While the network connection is active, these two inputs do not affect the output of the *Remote A/M Button* operator.

Generally, a person will use the keypad buttons to switch a control station from operating in automatic mode to operating in manual mode (and vice versa). You would also typically rely on a person to make manual output adjustments. However, under certain conditions you may wish to automatically switch the *Remote A/M Button* operator from one mode to another—and automatically adjust the "manual" output—without user involvement. The *Remote A/M Button* operator has two secondary inputs that allow you to do just that. You can use digital

input x_{so} to automatically set the auto/manual mode. And you can use analog input x_{mo} to automatically set the "manual" output. These two inputs are edge-triggered, meaning that they override the keypad buttons and affect the operator output only when their value changes. While these inputs remain fixed, they do not affect the outputs. (Of course, when these inputs affect the output of the *Remote A/M Button* operator, they also affect the output of associated *Remote A/M Button* operators and *A/M Button* operators.)

When the operator is in manual mode and input x_{so} switches from 1 (manual) to 0 (auto), the operator will switch into automatic mode and its output $y(t)$ will (over time) match its input $x(t)$. (If the bumpless transfer feature is enabled, the output will vary according to the equations above.) If the operator is in automatic mode and input x_{so} switches from 1 (manual) to 0 (auto), then it will remain in automatic mode.

When the operator is in automatic mode and input x_{so} switches from 0 (auto) to 1 (manual) at time t_o, the operator will switch into manual mode and its output $y(t)$ will move toward the value of input x_{mo} at the time of the switch. The transfer of the output—from $y = y(t_o)$ to $y = x_{mo}(t_o)$—occurs in increments and is rate-limited by the *Schema Key Repeat Time*, Δ_{KRAM} (see section 4.9). The output will vary as if a user presses the A/M key to switch to manual and then presses—and holds down—one of the arrow keys until the output reaches the desired value, $x_{mo}(t_o)$. Mathematically, we can express this as:

$$y(t) = \begin{cases} x_{mo}(t_o) & \text{if} \quad t > t_o + \Delta_{KRAM} \dfrac{x_{mo}(t_o) - y(t_o)}{M} \\ y(t_o) + M \dfrac{t - t_o}{\Delta_{KRAM}} & \text{if} \quad t_o \geq t \leq t_o + \Delta_{KRAM} \dfrac{x_{mo}(t_o) - y(t_o)}{M} \end{cases}$$

Whenever the manual-override input x_{mo} changes value, the output will move toward that value according to the equation above. (Of course, this occurs only while the operator is in manual mode; otherwise, the x_{mo} input is ignored). In order for the manual input x_{mo} to affect the output, its value must change. Moreover, the magnitude of the change must exceed the *Manual Input Activation Threshold* A_T over a period of time called the *Manual Input Activation Time* T. In other words,

$$\left| x_{mo}(t) - x_{mo}(t_o) \right| \geq A_T \quad \forall t \in (t_o, T]$$

must be satisfied in for the manual output to begin moving toward $x_{mo}(t_o)$.

Do the A/M, LEFT ARROW, and RIGHT ARROW keys on the *FAC-2000* faceplate have precedence over the inputs x_{so} and x_{mo}? The answer is that the operator will follow its most recent command. If it is commanded by its inputs to automatically adjust its manual output value, it will do so. If in the process of automatically adjusting the manual output

value a person presses one of the arrow keys, the *Remote A/M Button* operator will follow the key commands. It will not resume its automatic adjustments unless input x_{mo} changes value—remember that this input is edge-triggered!

User-Defined Properties:

Signal Name	A string label that identifies the operator. This character string is used to associate (synchronize) a *Remote A/M Button* operator in one schema with an *A/M Button* operator in another schema.
Initial State	The operator mode (*auto* or *manual*) on startup (at time *t*=0).
Initial Value	The analog output value on startup (at time *t*=0). Applicable only if the initial state is set to manual.
Minimum	The minimum analog output value. Applies to both auto and manual modes.
Maximum	The maximum analog output value. Applies to both auto and manual modes.
Increment Magnitude	The magnitude of manual output increments when in manual mode. The manual output will be incremented or decremented by this parameter when (i) a person presses the LEFT ARROW or RIGHT ARROW buttons on the *FAC-2000* front panel (ii) input x_{mo} changes value (see manual input activation threshold property and manual input activation hysteresis property).
Bumpless Transfer	When the operator switches from manual mode to automatic mode, the output signal may differ from input *x*. If this property is not checked, there may be a discontinuity in the output. When bumpless transfer is checked, the transition from manual to automatic is guaranteed to be smooth.
Transfer Profile	When the bumpless transfer property is set, the output will transition smoothly from manual mode to automatic mode. This property characterizes the nature of the transition: exponential or linear. When the exponential profile is selected, the difference between the manual output and automatic output will decay exponentially. When the linear profile is selected, the difference will decay linearly.
Transfer Rate	The rate at which the difference between manual output and automatic output will decay when the bumpless transfer property is set. When the exponential profile is selected, this parameter defines the exponential decay rate expressed in minutes. When the linear profile is selected, this parameter defines the linear decay rate in input units/minute.

Use Alpha-Numeric Displays When this property is selected, the output value will be presented on the alphanumeric displays while the operator is in manual mode.

Number of Significant Digits to Display The number of significant digits to use when displaying the manual output on the alphanumeric displays.

Lock in Manual State When this property is selected, the operator will only function in manual mode. A person (or an associated *A/M Button* or *Remote A/M Button* operator) will not be able to switch it into automatic operation.

Communication Medium *FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator should be communicating over.

Maximum Broadcast Dead-Time The period of time (in milliseconds) that may elapse before the operator indicates that it has not received a broadcast signal from its associated *A/M Button* operator (switches output *b* to *high*). Output *b* remains *low* while the operator actively receives the broadcasted signal.

Display A/M State Input The state input x_{so} is used to automatically switch the operator state—automatic or manual—without front-panel user interaction. This input is not used in most applications and can remain disconnected (but *Design Pad* will issue a warning upon processing). When this property is not checked, *Design Pad* will not display the input pin and will not issue a warning upon processing.

Display Manual Input The manual input x_{mo} is used to automatically adjust the "manual" output without front-panel user interaction. This input is not used in most applications and can remain disconnected (but *Design Pad* will issue a warning upon processing). When this property is not checked, *Design Pad* will not display the input pin and will not issue a warning upon processing.

Manual Input Activation Threshold When the *A/M Button* operator is in manual mode, the output can be adjusted automatically using input x_{mo}. In order for automatic adjustments to begin, the manual input must change value. The magnitude of the change must exceed this parameter. (And it must exceed this value for a period of time defined by the *Manual Input Activation Time* property below.)

Manual Input When the *A/M Button* operator is in manual mode, the output can

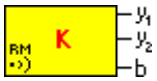
Activation Time (msec) be adjusted automatically using input x_{mo}. In order for automatic adjustments to begin, the manual input must change value. The magnitude of the change must exceed a threshold value over a period of time (in milliseconds) defined by this parameter.

Deviation Time from Remote Signal While the network is active, the *Remote A/M Button* operator is "tuned" to the broadcasted output state of an associated *A/M Button* operator. However, it may deviate from the *A/M Button* when its station's front-panel keys (A/M, LEFT ARROW, or RIGHT ARROW) are pressed. When this happens, the operator sends a message over the network to the *A/M Button* and waits for confirmation that the *A/M Button* has accepted its command. While it waits for confirmation, its output state may deviate from the *A/M Button* operator broadcasts. This parameter indicates how long (in milliseconds) the operator can deviate from the *A/M Button* broadcasts. (If it has not received confirmation after this time elapses, the operator will resume tracking the broadcasted output state from the associated *A/M Button*.)

Simulation Mode This parameter applies only to operators that are part of HMI (`.hmi`) documents; it does not apply to operators that are part of schema (`.scm`) documents. When executing an HMI document simulation, *Design Pad* can emulate a controller communications network (select *Use Virtual PC Network*) or it can be part of an actual *FAIRNET* network (select *Use FAIRNET Network*).

Comments: None.

See Also: PID Controller, PI Controller, PD Controller, PID with external feedback



11.56 REMOTE CONSTANT (ANALOG)

Design Pad Menu Location: Operators—Network Operators

Functional Description:

The *Remote Constant* operator is a special type of network operator that complements the *Constant* operator. Together, they allow synchronized parameter adjustment over multiple control stations attached to a *FAIRNET* network. For example, suppose your design requires that the same process set-point parameter be adjustable from three control stations. You could set up one station with a network-enabled *Constant* operator and two other stations with an associated *Remote Constant* operator. *FAIRNET* takes care of the synchronization: if a user adjusts the set-point value in any one of the three stations, the value will be automatically updated at the other two stations.

In order to associate a *Remote Constant* operator in one schema to a network-enabled *Constant* operator in another, you must match the signal name property of the operators. To synchronize multiple control stations, the schema of one station must have a network-enabled *Constant* operator and the schemas of each additional station must each have a *Remote Constant* operator (and the signal name property of all operators must have be same case-sensitive character string).

The *Remote Constant* operator has one analog output and one digital output. The analog output y is the constant value, *i.e.*, $y = K$. The digital output b serves as a network status signal indicating if the operator is actively receiving broadcasts from its associated network-enabled *Constant* operator. The output is *low* (0) while transmissions are regularly received; the output is *high* (1) if the operator has not received a broadcast in a user-specified time interval (the *Maximum Broadcast Dead-Time* property of this operator).

Remote Constant and *Constant* operators are synchronized over a *FAIRNET* network. When the output of one operator changes, the output of the associated operators will follow. But what happens when conflicting commands are simultaneously issued from different control stations? Say a user in one station increases the parameter value while a user in another simultaneously reduces it. The station executing the schema with the *Constant* operator arbitrates conflicting commands. The *Constant* operator continuously broadcasts the output value that all associated *Remote Constant* operators should assume. When a *Remote Constant* operator deviates from this broadcasted value (*e.g.*, one of its front-panel keys is pressed), it sends a message to the *Constant* operator. If the *Constant* operator accepts this command it modifies its output value and broadcasts it over the network. The broadcast of the new output value serves to confirm that the *Constant* operator has accepted the *Remote Constant* command and to alert all other *Remote Constant* operators that they should update their output value. If the *Remote Constant* does not receive confirmation that its message has been accepted after T_D seconds, it reverts to tracking broadcasted the output value of the *Constant* operator. Parameter T_D is a property of the *Remote Constant* operator labeled *Deviation Time from Remote Signal*.

User-Defined Properties:

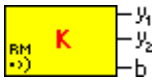
Signal Name	A string label that identifies the operator. When networking is enabled, this character string is used to associate (synchronize) a *Constant* operator in one schema with *Remote Constant* operators in other schemas.
Constant Value	The constant value, *K*.
Front Panel Access	This property can assume three states: (i) None (no front panel access); (ii) Operator level access (constant can be adjusted from the *FAC-2000* front panel); and (iii) Engineer level access (constant can be adjusted only after password is provided.)
Minimum	The minimum value the constant can assume when modified from the front panel (or from another station on the network).
Maximum	The maximum value the constant can assume when modified from the front panel (or from another station on the network).
Increment	The magnitude of constant value increments when modified from the *FAC-2000* front panel (or from another station on the network).
Number of Significant Digits to Display	The number of significant digits to use when displaying the constant value on the alphanumeric displays.
Communication Medium	*FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator communicates over.
Maximum Broadcast Dead-Time	The period of time (in milliseconds) that may elapse before the operator indicates that it has not received a broadcast signal from its associated *Constant* operator (switches output *b* to *high*). Output *b* remains *low* while the operator actively receives the broadcasted signal.
Deviation Time from Remote Signal	While the network is active, the *Remote Constant* operator is "tuned" to the broadcasted output state of an associated *Constant* operator. However, it may deviate from the *Constant* operator when its station's front-panel keys (SET, UP ARROW, or DOWN ARROW) are pressed. When this happens, the operator sends a message over the network to the *Constant* and waits for confirmation that the *Constant* has accepted its command. While it waits for confirmation, its output value may deviate from the *Constant* operator broadcasts. This parameter indicates how long (in milliseconds) the operator can deviate from the *Constant*

operator broadcasts. (If it has not received confirmation after this time elapses, the operator will resume tracking the broadcasted output value from the associated *Constant* operator.)

Simulation Mode This parameter applies only to operators that are part of HMI (`.hmi`) documents; it does not apply to operators that are part of schema (`.scm`) documents. When executing an HMI document simulation, *Design Pad* can emulate a controller communications network (select *Use Virtual PC Network*) or it can be part of an actual *FAIRNET* network (select *Use FAIRNET Network*).

Comments: None.

See Also: Constant (Analog)



11.57 REMOTE CONSTANT (Digital)

Design Pad Menu Location: Operators—Network Operators

Functional Description:

The *Remote Constant* operator is a special type of network operator that complements the *Constant* operator. Together, they allow synchronized parameter adjustment over multiple control stations attached to a *FAIRNET* network. For example, suppose your design requires that the same process set-point parameter be adjustable from three control stations. You could set up one station with a network-enabled *Constant* operator and two other stations with an associated *Remote Constant* operator. *FAIRNET* takes care of the synchronization: if a user adjusts the set-point value in any one of the three stations, the value will be automatically updated at the other two stations.

In order to associate a *Remote Digital Constant* operator in one schema to a network-enabled *Digital Constant* operator in another, you must match the signal name property of the operators. To synchronize multiple control stations, the schema of one station must have a network-enabled *Constant* operator and the schemas of each additional station must each have a *Remote Constant* operator (and the signal name property of all operators must have be same case-sensitive character string).

The *Remote Constant* operator has one analog output and one digital output. The analog output y is the constant value, *i.e.*, $y = K$. The digital output b serves as a network status signal indicating if the operator is actively receiving broadcasts from its associated network-enabled *Constant* operator. The output is *low* (0) while transmissions are regularly received; the output is *high* (1) if the operator has not received a broadcast in a user-specified time interval (the *Maximum Broadcast Dead-Time* property of this operator).

Remote Constant and *Constant* operators are synchronized over a *FAIRNET* network. When the output of one operator changes, the output of the associated operators will follow. But what happens when conflicting commands are simultaneously issued from different control stations? Say a user in one station increases the parameter value while a user in another simultaneously reduces it. The station executing the schema with the *Constant* operator arbitrates conflicting commands. The *Constant* operator continuously broadcasts the output value that all associated *Remote Constant* operators should assume. When a *Remote Constant* operator deviates from this broadcasted value (*e.g.*, one of its front-panel keys is pressed), it sends a message to the *Constant* operator. If the *Constant* operator accepts this command it modifies its output value and broadcasts it over the network. The broadcast of the new output value serves to confirm that the *Constant* operator has accepted the *Remote Constant* command and to alert all other *Remote Constant* operators that they should update their output value. If the *Remote Constant* does not receive confirmation that its message has been accepted after T_D seconds, it reverts to tracking broadcasted the output value of the *Constant* operator. Parameter T_D is a property of the *Remote Constant* operator labeled *Deviation Time from Remote Signal*.

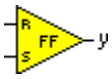
User-Defined Properties:

Signal Name	A string label that identifies the operator. When networking is enabled, this character string is used to associate (synchronize) a *Constant* operator in one schema with *Remote Constant* operators in other schemas.
Constant Value	The constant value, *K*.
Front Panel Access	This property can assume three states: (i) None (no front panel access); (ii) Operator level access (constant can be adjusted from the *FAC-2000* front panel); and (iii) Engineer level access (constant can be adjusted only after password is provided.)
Low State Label	By default, the state K=0 is indicated as "LOW". However, in certain cases, other terms may be more appropriate (*e.g.*, NO, FALSE, OFF, *etc.*)
High State Label	By default, the state K=1 is indicated as "HIGH". However, in certain cases, other terms may be more appropriate (*e.g.*, YES, TRUE, ON, *etc.*)
Communication Medium	*FAC-2000* controllers are equipped with two communication channels: one RS-232 channel and one RS-485 channel. They can be attached to two distinct networks—one on each channel. This parameter specifies which of the two channels the operator communicates over.
Maximum Broadcast Dead-Time	The period of time (in milliseconds) that may elapse before the operator indicates that it has not received a broadcast signal from its associated *Constant* operator (switches output *b* to *high*). Output *b* remains *low* while the operator actively receives the broadcasted signal.
Deviation Time from Remote Signal	While the network is active, the *Remote Constant* operator is "tuned" to the broadcasted output state of an associated *Constant* operator. However, it may deviate from the *Constant* operator when its station's front-panel keys (SET, UP ARROW, or DOWN ARROW) are pressed. When this happens, the operator sends a message over the network to the *Constant* and waits for confirmation that the *Constant* has accepted its command. While it waits for confirmation, its output value may deviate from the *Constant* operator broadcasts. This parameter indicates how long (in milliseconds) the operator can deviate from the *Constant* operator broadcasts. (If it has not received confirmation after this time elapses, the operator will resume tracking the broadcasted output value from the associated *Constant* operator.)

Simulation Mode This parameter applies only to operators that are part of HMI (`.hmi`) documents; it does not apply to operators that are part of schema (`.scm`) documents. When executing an HMI document simulation, *Design Pad* can emulate a controller communications network (select *Use Virtual PC Network*) or it can be part of an actual *FAIRNET* network (select *Use FAIRNET Network*).

Comments: None.

See Also: Constant (Analog)



11.58 RS FLIP-FLOP

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The *RS Flip-Flop* is a logic gate with the following truth-table:

R	S	$y(t)$
0	0	$y(t-\Delta t)$
0	1	1
1	0	0
1	1	$\overline{y(t-\Delta t)}$

In the table, y represents the output, R is the reset input, S is the set input, and Δt is the loop sampling time.

Design Pad will issue an error on processing if an input pin is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Initial State The output state (*high* or *low*) on startup (at time t=0).

Comments: None.

See Also:

AND-Gate, NAND-Gate, OR-Gate, NOR-Gate, NOT-Gate, XOR-Gate, XNOR-Gate

11.59 SET-POINT TRACKING

Design Pad Menu Location: Operators—Controller Blocks

Functional Description:

The *Set-Point Tracking* operator is designed for use with the *PD*, *PI*, and *PID Controller* blocks. It provides a means to match the controller set-point to the process variable while the controller is operating in manual mode. This strategy is sometimes used to achieve bumpless transfer when the controller is switched from manual to automatic.

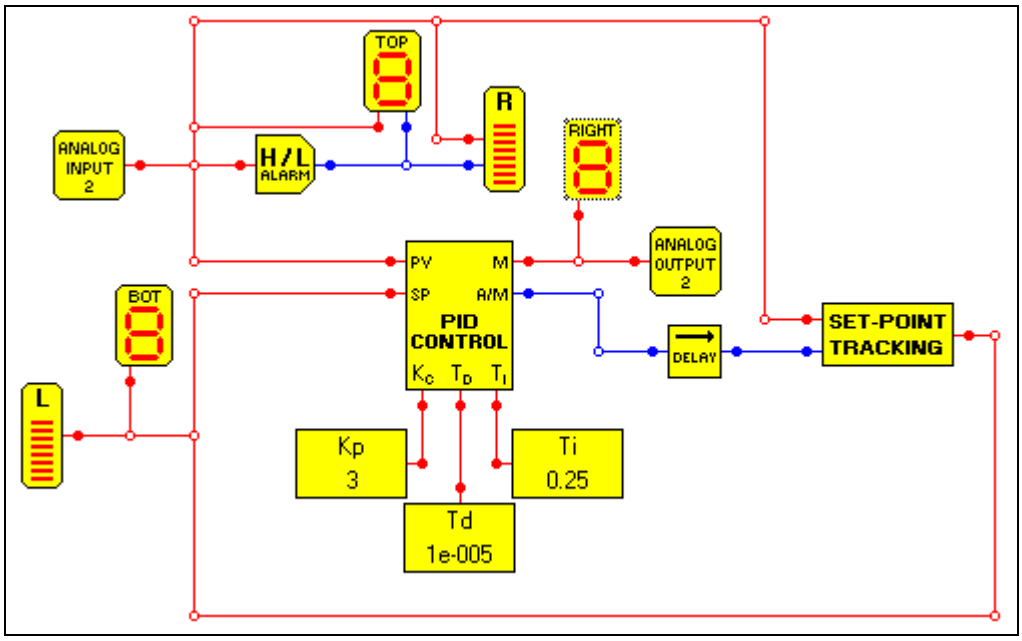
The *Set-Point Tracking* operator has one analog input, x, one digital input b, and one analog output y. When the digital input is *high* (b=1), the operator tracks the analog input, *i.e.,*

$$y = \begin{cases} y_{\min} & \text{if} \quad x > y_{\min},\ b = 1 \\ y_{\max} & \text{if} \quad x < y_{\max},\ b = 1 \\ x & \text{if} \quad y_{\min} \le x \le y_{\max},\ b = 1 \end{cases}$$

where limiting values y_{min} and y_{max}, are user-defined operator properties. When the digital input is *low* (b=0), the operator output is adjustable only from the front panel. In other words, the output maintains its value until modified via the *Set Menu* with the UP/DOWN arrow keys on the controller faceplate. The magnitude of output adjustment is a user-defined property of the operator.

A typical configuration of the *Set-Point Tracking* operator with the *PID Controller* block is shown in the following figure. In the figure, the *Set-Point Tracking* operator tracks analog input 2 (the process variable). The output of the *Set-Point Tracking* operator feeds the set-point input (*SP*) of the *PID Controller* block. With this arrangement, the process set-point will be adjustable from the controller front panel while the PID block operates in automatic mode. When the PID block operates in manual mode, the set-point will match the process variable.

Design Pad will issue an error on processing if either input pin is not connected.



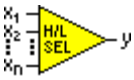
User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial Value	The output value on startup (at time t=0).
Minimum Value	The minimum output value, y_{min}.
Maximum Value	The maximum output value, y_{max}.
Increment Magnitude	The magnitude of output value increments when modified from the *FAC-2000* front panel (when input *b* is *low*).
Number of Significant Digits to Display	The number of significant digits to use when displaying the set-point value on the alphanumeric displays.

Comments: None.

See Also:

PD Controller, PI Controller, PID Controller



11.60 SIGNAL SELECTOR

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The *Signal Selector* accepts *n* input signals and produces one output signal. The output signal value is equal to the minimum or maximum input signal value. The selection of the minimum or maximum value is a property of the operator (see below).

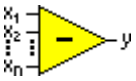
User-Defined Properties

Object Name	A string label that identifies the operator.
Number of Inputs	The number of input pins *n*, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.
Signal Select	Determines if the operator should select the minimum input signal value, or maximum input signal value.

Comments: None.

See Also:

=, ≠, >, <, ≥, and ≤ Comparators; Demultiplexer, Multiplexer



11.61 SUBTRACTION

Design Pad Menu Location: Operators—Math Functions

Functional Description:

The output of the *subtraction* operator is the difference:

$$y = x_1 - x_2 - x_3 - \cdots - x_n$$

where y is the output, x_i (i=1,…,n) are the inputs, and n is the number of inputs. *Design Pad* assumes that $x_i \equiv 0$, if input pin i is not connected.

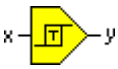
User-Defined Properties:

Object Name	A string label that identifies the operator.
Number of Inputs	The number of input pins n, where $2 \le n \le 6$. This property cannot be modified while any signals are attached to the operator.

Comments: None.

See Also:

Addition, Multiplication, Division, Unit Conversion



11.62 THRESHOLDING

Design Pad Menu Location: Operators—Signal Comparators

Functional Description:

The *Thresholding* operator compares an analog input signal to high and low threshold levels. The operator's digital output is set *high* (1) if the input signal is greater than the upper limit; it is set *low* (0) if the input signal is less than the lower limit; and it remains unchanged while the input signal is between the threshold levels. For the output to change from *low* to *high*, the input must exceed the threshold level for a user-specified length of time—the rising dead-time. Likewise, for the output to switch from *high* to *low*, the input signal must be less than the threshold level for a user-specified length of time—the falling dead-time. That is, the output y is defined by

$$y(t) = \begin{cases} 1 & \text{if} \quad x(t) \geq x_{\max} \ \forall t \in [t, t-R] \\ y(t-\Delta t) & \text{if} \quad x_{\min} < x < x_{\max} \\ 0 & \text{if} \quad x(t) \leq x_{\min} \ \forall t \in [t, t-F] \end{cases}$$

where x is the input, Δt is the loop sampling time, $x_{\max}$ is the high threshold level, $x_{\min}$ is the low threshold level, R is the rising dead-time, and F is the falling dead-time.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Initial State	The output state (*high* or *low*) on startup (at time t=0).
Low Threshold	The low threshold level, $x_{\min}$.
High Threshold	The high threshold level, $x_{\max}$.
Falling Dead-Time	The falling dead time, F, expressed in seconds.
Rising Dead-Time	The rising dead time, R, expressed in seconds.

Comments: The high threshold level must exceed the low threshold level, $x_{\max} > x_{\min}$.

See Also: =, ≠, >, <, ≥, and ≤ Comparators; High/Low Alarm

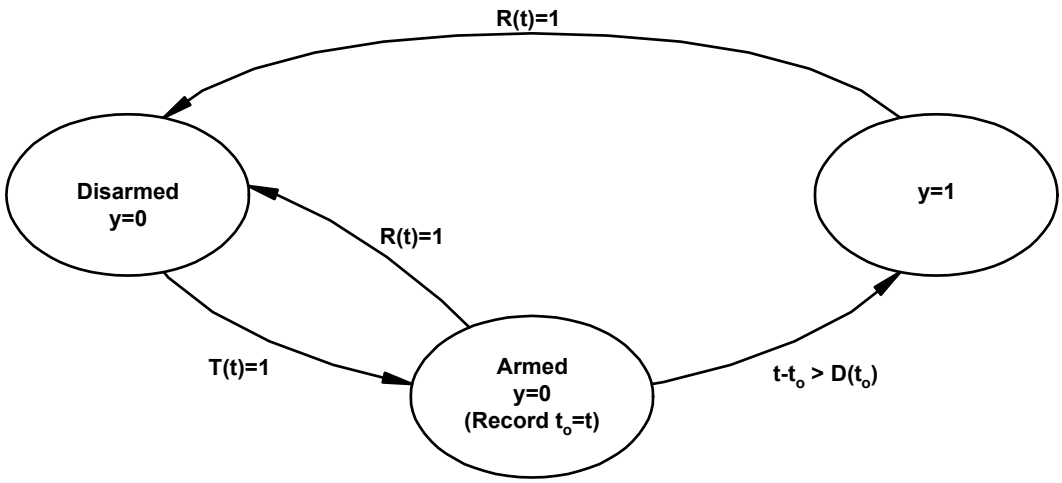
11.63 TIMER

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The *Timer* operator measures elapsed time. It is activated when the Trigger (*T*) input is *high* (1). After the *Timer* is activated, the output remains *low* (0) until the time specified by the Delay (*D*) input (in seconds) elapses. After the Delay time elapses, the output will go *high*, and remain *high* until the Reset (*R*) input goes *high*. After the reset pulse is issued, the timer can be triggered again.

The state transition diagram shown below illustrates the functional behavior of the Timer operator.

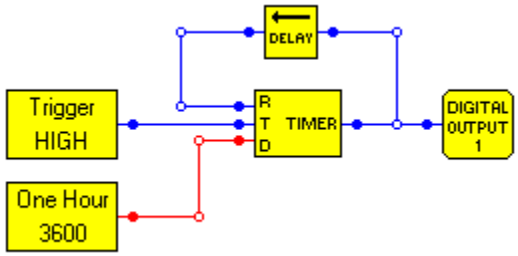


It consists of three states: (i) a disarmed state with the output *low*—the initial state; (ii) an armed state with the output *low*; and (iii) a *high* output state. The operator transitions from the disarmed state to the armed state when input $T(t) = 1$. The time instant when the *Timer* is armed or triggered is recorded as $t_0 = t$. Once armed, the *Timer* will begin to record the elapsed time. It will transition from the armed state to the *high* output state when

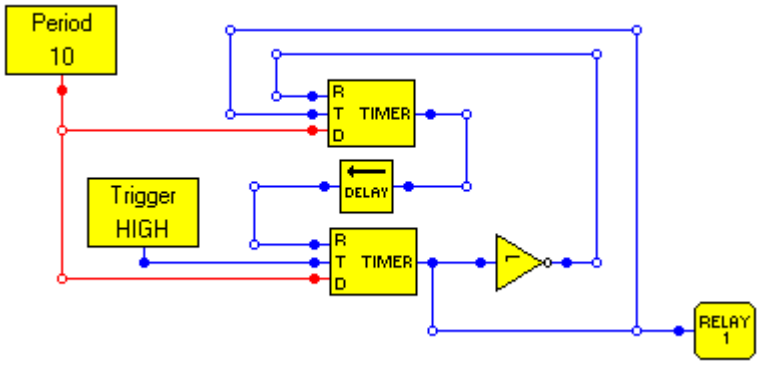
$$t - t_0 \geq D(t_0).$$

The operator transitions to the disarmed state whenever input $R(t) = 1$.

The *Timer* operator can be configured to issue a pulse at regular intervals. For example, simple schema shown in the figure below will produce a short pulse every hour.



The Timer operator can also be configured to function as a square-wave generator. For example, in the schema shown below, two Timer operators continually activate and deactivate Relay 1 of the *FAC-2000* in alternating 10 second periods.



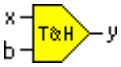
User-Defined Properties:

Object Name A string label that identifies the operator.

Comments:

The Timer operator cannot measure events greater than 8 hours (28,800 seconds). Therefore, input D may not exceed 28,800 when the timer is activated.

See Also: Delay Element



11.64 TRACK & HOLD

Design Pad Menu Location: Operators—Signal Conditioning

Functional Description:

The *Track & Hold* operator has an analog input *x*, a digital input *b*, and an analog output *y*. While input *b* is *low* (*b*=0), output *y* tracks input *x* (*i.e.*, *y*=*x*). When input *b* switches to *high* (*b*=1), *y* will thereafter "hold" the value of *x* at the time of input *b*'s transition. In other words, the operator output is defined by

$$y(t) = \begin{cases} x(t) & \text{if} \quad b(t) = 0 \\ x(\bar{t}) & \text{if} \quad b(t) = 1 \end{cases}$$

where $\bar{t}$ is the time instant that *b* switched to the hold state (*b*=1).

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments:

The operator output may be discontinuous when switching from the hold condition to the track condition.

11.65 UNIT CONVERSION

Design Pad Menu Location: Operators—General Purpose

Functional Description:

The *unit conversion* operator should be used to map between units of measurement. For example, a 4-20*mA* signal can be converted to a pressure reading of 1100-1400*Psi*. The output of the *unit conversion* operator is

$$y = \begin{cases} x_{\min} & \text{if} \quad Mx + S < x_{\min} \\ x_{\max} & \text{if} \quad Mx + S > x_{\max} \\ Mx + S & \text{otherwise} \end{cases}$$

where y is the output, x is the input, M is the conversion multiplier, and S is the conversion offset, $x_{\min}$ is the minimum operator output value, and $x_{\max}$ is the maximum operator output value.

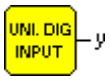
User-Defined Properties:

Object Name	A string label that identifies the operator.
Multiplier	The conversion multiplier, M.
Offset	The conversion offset, S.
Min. Output Value	The minimum value, $x_{\min}$.
Max. Output Value	The maximum value, $x_{\max}$.
Input Signal Unit	The unit of measurement of the input signal.
Output Signal Unit	The unit of measurement of the output signal.

Comments: None.

See Also:

Gain, Bias, Addition, Multiplication, Subtraction, Division, Power



11.66 UNIVERSAL DIGITAL INPUT

Design Pad Menu Location: Operators—Controller Hardware

Functional Description:

The *universal digital input* operator provides access to one of the two *FAC-2000* universal digital inputs. The operator state is *low* (0) while the digital input signal is below the deactivation threshold (see the *FAC-2000* Hardware Reference Guide—Fairmount Automation Technical Bulletin 9110-0001). The operator state is *high* (1) when the input signal is above the activation threshold. The operator state will change only if the input signal is above/below the threshold for at least the hysteresis time.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Hysteresis	The minimum length of time (in seconds) the input signal must remain below the deactivation threshold (or above the activation threshold) for the operator state to switch from *high* to *low* (or from *low* to *high*).

Comments:

The hysteresis time must be greater or equal to zero.

See Also:

Digital Input, Analog Input, Digital Output, Analog Output, Relay

11.67 XNOR-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the XNOR-*Gate* operator is

$$y = \overline{x_1 \veebar x_2}$$

where y is the output, and x_i (i=1,2) are the inputs. In other words, the output state is *high* (1) if the input states are equal. . If the input states differ, the output is *low* (0). Below is the truth table for an XNOR-gate:

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	1

Design Pad will issue an error on processing if an input pin is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also:

XOR-Gate, OR-Gate, NOR-GATE, NOT-Gate, AND-Gate, NAND-Gate, RS Flip-Flop

11.68 XOR-GATE

Design Pad Menu Location: Operators—Logic Functions

Functional Description:

The output of the *XOR-Gate* operator is

$$y = x_1 \veebar x_2$$

where y is the output, and x_i (i=1,2) are the inputs. In other words, the output state is *low* (0) if the input states are equal. . If the input states differ, the output is *high* (1). Below is the truth table for an XOR-gate:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Design Pad will issue an error on processing if an input pin is not connected.

User-Defined Properties:

Object Name A string label that identifies the operator.

Comments: None.

See Also

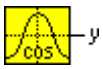
XNOR-Gate, OR-Gate, NOR-GATE, NOT-Gate, AND-Gate, NAND-Gate, RS Flip-Flop

12. SIGNAL GENERATOR REFERENCE

The following section provides a detailed description of each signal generator available in *Design Pad*.

The documentation for each generator includes a functional description, a property list, comments (if applicable), and a 'see also' section. The functional description explains what the generator does and how it works. The property list describes in detail the user-defined parameters associated with the generator. The comments section indicates any exceptional information particular to the generator. And, the 'see also' section refers the reader to related generators that may be of interest.

NOTE: Signal generator blocks are made available for simulation purposes only. They are not part of the *FAC-2000* operator suite. The controller will not accept schemas that contain signal generators.



12.1 COSINE WAVE

Functional Description:

The output of the *Cosine Wave* generator is

$$y(t) = A\cos 2\pi ft$$

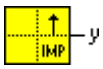
where y is the output, A is the amplitude, and f the frequency in hertz.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Amplitude	The cosine wave amplitude, A.
Frequency	The cosine wave frequency in hertz.

Comments: None.

See Also: Sine Wave



12.2 IMPULSE FUNCTION

Functional Description:

The output of the *Impulse* generator is

$$y(t) = \begin{cases} A & if \quad t = t_a \\ 0 & \text{if} \quad \text{otherwise} \end{cases}$$

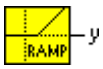
where y is the output, A is the pulse amplitude, and t_a is the activation time in seconds.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Activation Time (sec)	The time t_a (in seconds) when the generator produces an impulse.
Amplitude	The impulse amplitude, A.

Comments: None.

See Also: Ramp Function, Step Function



12.3 RAMP FUNCTION

Functional Description:

The output of the *Ramp* generator is

$$y(t) = \begin{cases} 0 & if \quad t < t_a \\ \\ R(t - t_a) & \text{if} \quad t \geq t_a \end{cases}$$

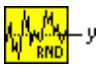
where y is the output, R is ramp slope or rate (in units/second), and t_a is the activation time (in seconds).

User-Defined Properties:

Object Name	A string label that identifies the operator.
Activation Time (sec)	The time t_a (in seconds) when the generator begins to ramp.
Rate (units/sec)	The slope of the ramp, R.

Comments: None.

See Also: Impulse Function, Step Function



12.4 RANDOM NUMBER (UNIFORM DISTRIBUTION)

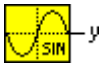
Functional Description:

The Random Number generator produces a uniformly distributed random number in the range [*A*, *B*].

User-Defined Properties:

Object Name	A string label that identifies the operator.
Minimum	The lower bound *A*.
Maximum	The upper bound *B*.

Comments: None.



12.5 SINE WAVE

Functional Description:

The output of the *Sine Wave* generator is

$$y(t) = A \sin 2\pi f t$$

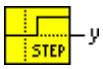
where y is the output, A is the amplitude, and f the frequency in hertz.

User-Defined Properties:

Object name	A string label that identifies the operator.
Amplitude	The sine wave amplitude, A.
Frequency	The sine wave frequency in hertz.

Comments: None.

See Also: Cosine Wave



12.6 STEP FUNCTION

Functional Description:

The output of the *Step* generator is

$$y(t) = \begin{cases} A & if \quad t \geq t_a \\ \\ 0 & \text{if} \quad t < t_a \end{cases}$$

where y is the output, A is the step amplitude, and t_a is the activation time in seconds.

User-Defined Properties:

Object Name	A string label that identifies the operator.
Activation Time (sec)	The time t_a (in seconds) when the generator produces a step change.
Amplitude	The amplitude of the step function, A.

Comments: None.

See Also: Impulse Function, Ramp Function

13. INDEX

P

S

T

V

W

1220 Valley Forge Road, Building 37
Phoenixville, PA 19460

Phone: (610) 935-8656
Fax: (610) 935-8725