



UM SWD EN Think & Do

User manual

User manual

UM SWD EN Think & Do

2013-06-17

Designation: UM SWD EN Think & Do

Revision: B

Order No.: —

This user manual is valid for:

Designation	Version	Order No.
Think & Do	8.1	

Please observe the following notes

User group of this manual

The use of products described in this manual is oriented exclusively to qualified application programmers and software engineers, who are familiar with the safety concepts of automation technology and applicable standards.

Explanation of symbols used and signal words



This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety measures that follow this symbol to avoid possible injury or death.

There are three different categories of personal injury that are indicated with a signal word.

DANGER This indicates a hazardous situation which, if not avoided, will result in death or serious injury.

WARNING This indicates a hazardous situation which, if not avoided, could result in death or serious injury.

CAUTION This indicates a hazardous situation which, if not avoided, could result in minor or moderate injury.



This symbol together with the signal word **NOTE** and the accompanying text alert the reader to a situation which may cause damage or malfunction to the device, hardware/software, or surrounding property.



This symbol and the accompanying text provide the reader with additional information or refer to detailed sources of information.

How to contact us

Internet

Up-to-date information on Phoenix Contact products and our Terms and Conditions can be found on the Internet at:

phoenixcontact.com

Make sure you always use the latest documentation.

It can be downloaded at:

phoenixcontact.net/products

Subsidiaries

If there are any problems that cannot be solved using the documentation, please contact your Phoenix Contact subsidiary.

Subsidiary contact information is available at phoenixcontact.com.

Published by

PHOENIX CONTACT GmbH & Co. KG
Flachsmarktstraße 8
32825 Blomberg
GERMANY

Should you have any suggestions or recommendations for improvement of the contents and layout of our manuals, please send your comments to:

tecdoc@phoenixcontact.com

Please observe the following notes

General terms and conditions of use for technical documentation

Phoenix Contact reserves the right to alter, correct, and/or improve the technical documentation and the products described in the technical documentation at its own discretion and without giving prior notice, insofar as this is reasonable for the user. The same applies to any technical changes that serve the purpose of technical progress.

The receipt of technical documentation (in particular user documentation) does not constitute any further duty on the part of Phoenix Contact to furnish information on modifications to products and/or technical documentation. You are responsible to verify the suitability and intended use of the products in your specific application, in particular with regard to observing the applicable standards and regulations. All information made available in the technical data is supplied without any accompanying guarantee, whether expressly mentioned, implied or tacitly assumed.

In general, the provisions of the current standard Terms and Conditions of Phoenix Contact apply exclusively, in particular as concerns any warranty liability.

This manual, including all illustrations contained herein, is copyright protected. Any changes to the contents or the publication of extracts of this document is prohibited.

Phoenix Contact reserves the right to register its own intellectual property rights for the product identifications of Phoenix Contact products that are used here. Registration of such intellectual property rights by third parties is prohibited.

Other product identifications may be afforded legal protection, even where they may not be indicated as such.

Table of Contents

1	Before You Begin.....	1-3
1.1	About This Manual.....	1-3
1.2	Manual Conventions.....	1-4
1.3	Starting Think & Do.....	1-5
1.4	Using Help.....	1-5
1.5	System Concepts	1-6
1.6	Phoenix Contact Technical Service	1-11
2	Creating a Project.....	2-3
2.1	Starting a New Project.....	2-3
2.2	Selecting a Runtime Target Type	2-8
2.3	Specifying Inputs, Outputs, and Data Items.....	2-10
2.4	Creating Control Programs	2-10
2.5	Creating an Operator Screen.....	2-12
3	Drivers, Devices, and Tags.....	3-3
3.1	Specifying Inputs, Outputs, and Data Items.....	3-3
3.2	Configuring I/O	3-5
3.3	Adding an I/O Driver	3-7
3.4	Connecting to I/O Devices.....	3-8
3.5	Mapping I/O Points to Data Items	3-11
3.6	Scanning and Monitoring I/O	3-13
3.7	Saving the I/O Configuration.....	3-14
3.8	Tips on Toolbar Functions	3-14
3.9	Watchdog Timeout Default I/O States	3-16
4	General Programming Techniques	4-3
4.1	Flow Charts and Subcharts	4-4
4.2	Exploring Flow Charts and Subcharts.....	4-5
4.3	Creating a Flow Chart.....	4-6
4.4	Changing Flow Chart Settings	4-7
4.5	FlowView Orientation.....	4-9
4.6	Displaying Multiple Flow Charts.....	4-12
4.7	Editing Flow Charts	4-13
4.8	Flow Chart Block Introduction.....	4-13
4.9	Connecting Flow Chart Blocks	4-16

Think & Do

4.10	Editing Flow Chart Block Expressions	4-17
4.11	Entering Block Description	4-18
4.12	Operator Screens	4-19
4.13	Scripting Languages in ScreenView	4-28
5	Flow Charts.....	5-3
5.1	Flow Chart Types and Categories	5-3
5.2	Basic Flow Chart Blocks.....	5-3
5.3	Using Subcharts	5-27
6	Operator Screen Techniques.....	6-3
6.1	Target Hardware Considerations.....	6-3
6.2	Exploring Screens	6-3
6.3	ScreenView Development Environment	6-5
6.4	Setting Screen Attributes	6-112
6.5	Screen Groups	6-114
6.6	Additional Worksheets.....	6-115
6.7	Using Scripts	6-131
7	Math and Data Operations.....	7-3
7.1	Exploring Data Items	7-3
7.2	Tagname Data Types and Formats	7-6
7.3	Understanding Arrays.....	7-8
7.4	Think & Do System Data Items	7-11
7.5	Counting with Number Data Items	7-15
7.6	Using the Move Block.....	7-16
7.7	Using the Calculation Block.....	7-25
7.8	Handling Analog Values	7-30
7.9	String Handling.....	7-34
7.10	SQL Database Operations	7-37
7.11	PID Block for Process Control	7-47
8	Communications.....	8-3
8.1	Serial Port Communications	8-3
8.2	Using OPC Server	8-10
8.3	Using DDE Server	8-16

	8.4 Setting Up Remote Communications.....	8-23
9	Building and Running Projects.....	9-3
	9.1 Building a Project.....	9-3
	9.2 Running a Project	9-4
	9.3 Adjusting the Scan Interval	9-5
	9.4 Stopping a Running Project	9-6
	9.5 Running Remote Projects.....	9-6
	9.6 Think & Do CE Watch for Windows CE Targets	9-9
	9.7 Instant Recall for Windows CE Target	9-12
10	Debugging Projects	10-3
	10.1 AppTracker.....	10-3
	10.2 Using FlowView in Monitor Mode	10-9
	10.3 Edit Mode	10-13
	10.4 Simulation Flow Charts.....	10-13
A	Appendix.....	A-1
	A 1 System Data Items	A-1
	A 2 Notes on Reading Error Data	A-5
	A 3 List of Figures	A-7
	A 4 List of Tables	A-17
B	Index.....	B-1

Section 1

This section informs you about:

- Features and Benefits of the Think & Do software
- System Concepts

Before You Begin	1-3
1.1 About This Manual.....	1-3
1.2 Manual Conventions.....	1-4
1.2.1 Text Conventions.....	1-4
1.2.2 Keyboard Commands.....	1-4
1.3 Starting Think & Do.....	1-5
1.4 Using Help.....	1-5
1.5 System Concepts	1-6
1.5.1 I/O Systems	1-6
1.5.2 Think & Do Methods and Terminology.....	1-7
1.5.3 Target Systems.....	1-8
1.5.4 Where's the Runtime Software?	1-9
1.6 Phoenix Contact Technical Service	1-11

1 Before You Begin

Welcome to Think & Do[®] Version 8 – an integrated control environment that lets you do more with a given PC platform. It supports development, deployment and operation of high-value automated control systems for material handling and manufacturing. Like all Phoenix Contact products, Think & Do provides an intuitive, open-architecture environment that readily integrates with hardware and software components from virtually all major suppliers.

Projects created with Think & Do integrate seamlessly with enterprise information systems to provide valuable data about system operation. Major components of Think & Do are:

- ProjectCenter: provides ready access to all project elements and the fully integrated tagname database using the Data Item Explorer.
- FlowView: where you create control logic.
- ScreenView: where you create HMI screens.
- I/O View: used to configure project I/O.
- AppTracker: a fast, graphical debugger.
- Runtime Engine: provides a robust, deterministic project execution environment.

Think & Do makes it easy to target your project to the Microsoft[®] Windows[®] operating system that best suits your needs. Whether you create a project for a Windows-based PC or Windows CE-based system, scaling for a different platform requires only minor adjustment.

1.1 About This Manual

This manual focuses on developing and running control programs and Human-Machine Interface (HMI) operator interface screens using Think & Do.

What's Inside

In this manual, you'll find the following chapters:

Section 1, "Before You Begin", this chapter, provides an overview of Think & Do and this manual.

Section 2, "Creating a Project", describes the steps required to create a project and start using Think & Do. It includes a brief introduction to the concept of Think & Do Runtime targets, and how to create input, output and data items using IOView. It also introduces flow charts and operator screens and how to create them using FlowView and ScreenView.

Section 3, "Drivers, Devices, and Tags", provides details on how to configure I/O using IOView.

Section 4, "General Programming Techniques", describes concepts for both flow charts and operator screens.

Section 5, "Flow Charts", provides details of flow chart programming.

Section 6, "Operator Screen Techniques", describes how to create operator screens. These screens are called the human-machine interface (HMI).

Section 7, "Math and Data Operations", defines the available data types in Think & Do and how to move data and use mathematical and relational expressions. It also defines how to use SQL statements and the PID block.

Section 8, "Communications", provides details on how to use serial port communications, DDE Server, OPC Server, and how to set up remote communications.

Section 9, “Building and Running Projects”, describes how to build and run projects.

Section 10, “Debugging Projects”, provides information on how to monitor and debug running projects.

The “Index” provides a handy cross reference to common terms.

1.2 Manual Conventions

This manual uses a few special symbols and conventions.

1.2.1 Text Conventions

This section discusses text conventions used throughout the manual. Folder and Filenames File names and folder paths appear in quotes. For example, “C:\Sample” is a folder path; “Project.pdb” is a filename.

Think & Do is a registered trademark of Think & Do Software, Inc. Windows, Windows XP, Windows Vista, Visual Basic, VBScript, and Visual C++ are trademarks of the Microsoft Corporation. OPC is a registered trademark of OPC Foundation. Modbus is a registered trademark of Schneider Automation, Inc. INTERBUS is a registered trademark of Phoenix Contact GmbH and Co.

Titles, Buttons, Keywords, Mnemonics

Command buttons, keywords and phrases found in dialog boxes appear in quotes. For example, the text might say, ‘click the “Apply” button’ to indicate there is a button in the dialog box labeled “Apply.” Menu selections also appear in quotes (for example, “File” menu) and submenu selections follow an ellipsis after the main menu (for example, “File... Print” menu).

There are a few hypertext links to Web pages. They appear as underlined text, such as www.phoenixcontact.com. When viewing this manual in Adobe Acrobat, hypertext links also appear as blue text. Clicking one of these links launches the web browser and attempts to connect to the link.

1.2.2 Keyboard Commands

Key names are shown as <Alt>, <Ctrl>, <Enter>, or <Home>. You'll find keyboard commands, key combinations, and key sequences as shown in Table 1-1:

Table 1-1 Manual Conventions for Keyboard Command

Keystroke(s)	Description
KEY1+KEY2	A plus sign (+) between key names means to press and hold the first key (KEY1) and type the second key (KEY2). For example, <Alt>+<F4> means to hold down the <Alt> key, press the <F4> key, and then release both the <Alt> and <F4> keys.
KEY1, KEY2	A comma (,) between key names means to type the keys individually in the sequence shown. For example, if instructed to type <R>, <Enter>, type the letter <R>, release it, then press the <Enter> key.

1.3 Starting Think & Do

After installation, an entry appears under “Start” for Think & Do. To start, follow these steps:

1. Select the “Start... Programs... Phoenix Contact... Think & Do... ProjectCenter” menu. This initiates execution of Think & Do.
2. Open or create a project by:
 - Selecting the “File...New” menu to create a project with the “untitled.pdb” name in the title bar.
 - Selecting the “File...Open” menu or the “Open” button to display a standard “Open” dialog box.

1.4 Using Help

There are several techniques for obtaining help from Think & Do – select “Help” from the menu bar, click a “Help” button, or press <F1>. Figure 1-1 illustrates the Help menu as it appears when selected from the menu bar.

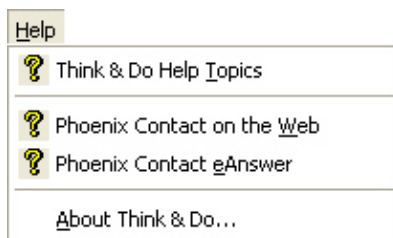


Figure 1-1 The Think & Do “Help” menu

The “Help... Think & Do Help Topics” menu item displays an “Think & Do Help” window showing the Contents tab with a list of all Help topics for Think & Do. You can also display Help Contents by pressing <F1>.

The other Help menu items are:

Table 1-2 Help Sources

Source	Result
Phoenix Contact on the Web	Connects to the Phoenix Contact website so that you can get technical support directly from Phoenix Contact.
Phoenix Contact eAnswer	A web-based support system that includes previously answered questions and a place where you can ask your questions and receive answers. You can also submit product suggestions.
About Think & Do...	Displays important information about the Think & Do and ProjectCenter, such as the copyright and version.

To locate specific information about Think & Do, click “Help... Think & Do Help Topics.” You’ll also find that some Help topics refer to other topics. You can view these related items by clicking on underlined words or phrases.

For additional information on using Help, see the operating system documentation.

1.5 System Concepts

It is important to learn key system concepts and terms before creating flow charts for your application. Please read this section to understand Think & Do structure.

1.5.1 I/O Systems

The traditional PC (personal computer) has I/O (input/output devices) oriented to office computing: printer ports, keyboard and mouse connections, etc. When using a PC for industrial control, the PC must be equipped to control industrial I/O devices. Factory I/O can include one or more racks (bases) of I/O modules (Figure 1-2), or a full factory network with many kinds of devices. We give these the collective name I/O subsystem. In order to connect with and control a machine or process, most PC-based control solutions, such as Think & Do, use either external rack(s) of Input/Output (I/O) modules or a factory network of nodes and devices.

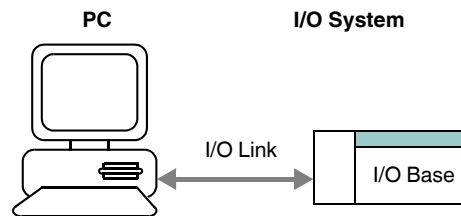


Figure 1-2 PCs connect to I/O bases in racks

Some factory networks such as DeviceNet and INTERBUS use a trunk line with drops as shown in Figure 1-3. The device type can vary from a limit switch to a motor controller, but each one is identified by a node address. By installing the appropriate I/O scanner (adapter) card in the PC, Think & Do easily controls a factory network.

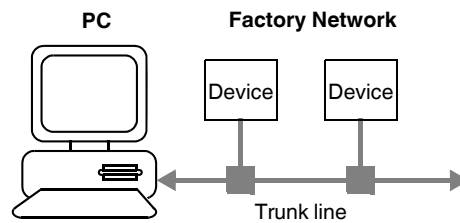


Figure 1-3 A PC can connect to a network with drops to devices

The Ethernet topology (Figure 1-4) uses an Ethernet adapter card in the PC and a central hub. The hub distributes individual connections to I/O controllers such as the popular Ethernet Base Controllers H2-EBC and H4-EBC from Automationdirect.com. The base controllers are similar to I/O concentrators in other networks, and occupy one network address on the Ethernet network.

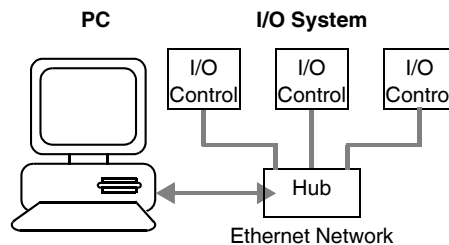


Figure 1-4 PC connected to multiple I/O controllers

The arrival of the Windows CE-based controllers makes it possible for an embedded system running Think & Do software to fit in the base with the I/O modules (Figure 1-5). The Think & Do WinPLC includes the I/O system in the base.

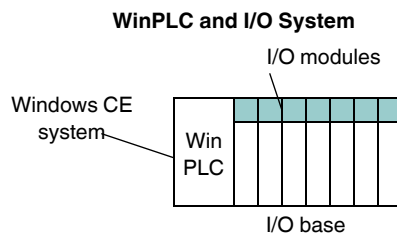


Figure 1-5 WinPLC and its one base of I/O modules

1.5.2 Think & Do Methods and Terminology

The software you create to control applications is called a project. A project can have one or more types of components (flow charts, HMI screens, I/O configuration, etc.).

Think & Do consists of interrelated development tools. ProjectCenter is the top-level project management tool, and includes a spreadsheet-like tool for creating a tagname database. FlowView is a development environment for designing flow charts; ScreenView provides the environment to create HMI screens; and IOView is a tool for creating the I/O configuration for the project. You can switch seamlessly from one tool to another during project development. During development, you open a project and have access to all its components through each tool.

Programmers may be used to thinking of software development in these steps:

1. Write program code.
2. Compile the code into an executable object.
3. Run the executable.
4. Test the program and debug until complete.
5. Commission the machine, wait for a machine problem or obsolescence, start at Step 1.

Think & Do equivalent activities have these steps:

1. Use FlowView to create or edit project flow charts.
2. Use ScreenView to create or edit HMI screens.
3. Build the project into appropriate Runtime components.
4. Run the project.
5. Test the project, using AppTracker and EasyTrac to debug until complete.
6. Commission the machine.

The word *Runtime* refers to the system's execution of the project created; it runs the project at Runtime. When project development is completed, use Runtime software to run the compiled project.

A *Runtime target* is the intended hardware platform that will control the final application. The target may be an industrial PC or an embedded controller. When the target is a different platform from the PC used for development, the terms *remote* and *local* are used (with respect to the programmer) for differentiation. The next section describes typical configurations in these terms.

1.5.3 Target Systems

Many applications require the PC used for project development to also run the project. However, you can create a project on a PC to run on a different platform at a later time. In either case, the target system is the PC, which has the Think & Do Runtime software and runs the project. If you are developing and running the project on the same PC the target is a "Certified PC".

The concept of having a target machine run a project that is different from the development platform provides flexibility for various scenarios. In all cases, the development system must run on a 32-bit Windows-based PC. The Runtime may run on a 32-bit Windows-based PC

or Windows CE system. In fact, there can be several Runtime platforms attached to a single development system, as shown below. This allows the targeting of a particular machine from the group to run the project just developed.

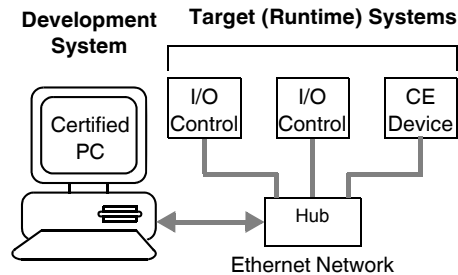


Figure 1-6 Group development system



In Think & Do, a PC running the Windows XP, Vista or 7 operating system is called a “Certified PC”. This manual and Help use the term Certified PC or Windows-based PC when referring to a PC that is using one of these operating systems.

1.5.4 Where’s the Runtime Software?

Each Think & Do package includes a CD-ROM that equips one Windows-based PC to develop and run Think & Do projects. But what about Windows CE hardware platforms? Windows CE-based systems, such as the WinPLC, come with a Think & Do Runtime license and have the Runtime software (Windows CE version) already loaded in its firmware. When you purchase a Think & Do version of a Windows CE platform, it already knows how to run Think & Do projects. Just select the station name of the Windows CE device and run. The PC used to develop the project downloads that project to the Windows CE engine to run. In this way, you can use one Think & Do software package on a Windows-based PC to develop projects for many Windows CE machines.

When you have completed the project for the target system, you can permanently disconnect the development system PC. The Windows CE system contains the Think & Do Runtime and the project – everything it needs to be a stand-alone controller.



When developing projects targeted to run on an off-the-shelf Windows-based PC, a Think & Do package must be purchased for each PC, just as each one must have a copy of the Windows XP/Vista/7 operating system.

1.5.4.1 Local and Remote Systems

When developing an application and running it on the same PC, the target system is said to be the local PC. This configuration is the simplest to do and is shown in Figure 1-7. An I/O scanner card connects the PC to the I/O subsystem.

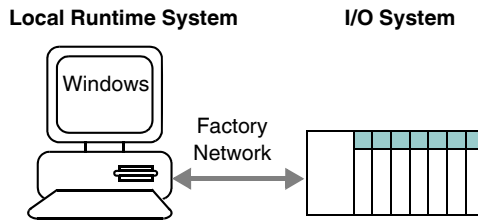


Figure 1-7 Local PC used for development and Runtime

In some factory situations, it may be necessary to access the PC-based control system from another location. The project engineer may create the application on a development system in an office environment, while the Runtime system exists some distance away in a manufacturing environment. The company network can connect the development system to the Runtime system as shown in Figure 1-8. In another common scenario, a project engineer will use a laptop as a local system and attach it to the factory-floor controller as a remote PC.

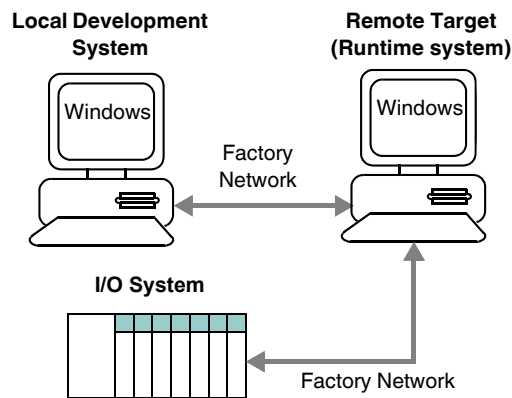


Figure 1-8 Development and Runtime control on different systems

When using remote and local systems, each PC platform must have a Windows XP/Vista/7 operating system and be equally capable of running Think & Do. Each PC must also have its own installation (and license) of Think & Do.

Local and remote systems communicate using Windows built-in messaging called DCOM (Distributed Component Object Model). This is included as part of Microsoft's Windows XP/Vista/7 operating systems. To implement a local/remote connection, both PCs must be configured for this connection. Just follow the instructions in "Setting Up Remote Communications" on page 8-23.

1.6 Phoenix Contact Technical Service

If you have a question about the Think & Do and can't find the answer in this manual or the online help system, contact Phoenix Contact Technical Service by phone, fax, e-mail, or the Web. Our staff will give you the advice you need to get the most from the Think & Do.

We suggest that you try to duplicate the problem before calling Technical Service. During this process, write down each step that you perform and any error messages that appear. To obtain the best possible support, please be at your computer when you call, and have the following available:

- This manual.
- The version number on your Think & Do CD.
- The serial number on your license. You can view this by selecting "Start... Programs... Phoenix Contact... Think & Do... System... Hardware Key Information" to display the "Read Hardware Key" window.
- Windows version number and Service Pack level. To find this information, select the Control Panel, and double-click the System icon.
- Information about your computer hardware: type and model of computer, monitor, video card, I/O boards installed in the system, and the amount and type of installed memory.

PHOENIX CONTACT

Internet: www.phoenixcontact.com



Users outside the United States should contact their local sales office.

This section informs you about:

- Starting a new project
- Selecting a Runtime target
- Specifying inputs, outputs, and data items
- Basics on creating control programs and operator screens

Creating a Project.....	2-3
2.1 Starting a New Project.....	2-3
2.1.1 Creating a New Project.....	2-4
2.1.2 Naming a New Project.....	2-5
2.1.3 Opening a Project.....	2-6
2.1.4 Project Explorer Bar.....	2-6
2.1.5 Flow Chart and Screens Explorer Bars.....	2-7
2.1.6 Data Item Explorer Bar.....	2-8
2.1.7 Other Explorer Bars.....	2-8
2.2 Selecting a Runtime Target Type.....	2-8
2.2.1 Local/Remote.....	2-9
2.2.2 Selecting a Windows CE Target.....	2-9
2.3 Specifying Inputs, Outputs, and Data Items.....	2-10
2.4 Creating Control Programs.....	2-10
2.4.1 Action Block.....	2-11
2.4.2 Branching Block.....	2-11
2.4.3 Building Flow Charts.....	2-11
2.5 Creating an Operator Screen.....	2-12
2.5.1 HMI Overview.....	2-12
2.5.2 Screens as Part of a Project.....	2-12

2 Creating a Project

In everyday life, a project usually refers to a series of tasks completed in order to accomplish a specific goal. Similarly, the goal of using Think & Do is to create a Runtime program that does whatever you design it to do. This is done by working on small, manageable pieces of the project with special tools within Think & Do. So, when using the development tools, you will be creating a project that consists of the following parts:

- A flow chart, or collection of flow charts
- An I/O configuration for the devices being controlled
- HMI screen(s) which provide an operator interface (optional for some applications or not available on some Windows CE-based devices)

The flow charts and screens must communicate with each other and the I/O to control and monitor the application. To make this possible, create a common tagname database of meaningful names. The database contains lists of system inputs, outputs, and internal variables.

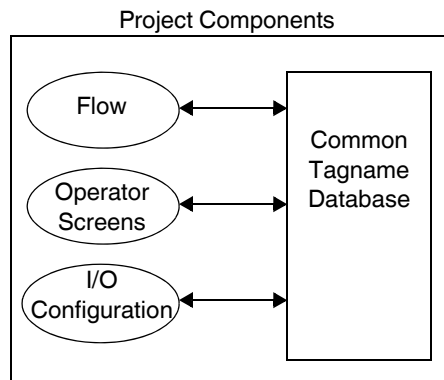


Figure 2-1 Project components

2.1 Starting a New Project

This section provides basic procedures for creating a project and a very simple flow chart program using Think & Do FlowView. It then shows how to create an operator screen, with ScreenView. Later chapters go into more detail on flow charts and screens.

2.1.1 Creating a New Project



ProjectCenter Icon

To create a new project, follow these steps:

1. From the Windows desktop, click “Start... All Programs... Phoenix Contact... Think & Do... ProjectCenter” to launch ProjectCenter or double-click the “ProjectCenter” icon on the desktop to start.
2. Select the “File... New” menu, and then take a moment to orient yourself to ProjectCenter (see Figure 2-2).

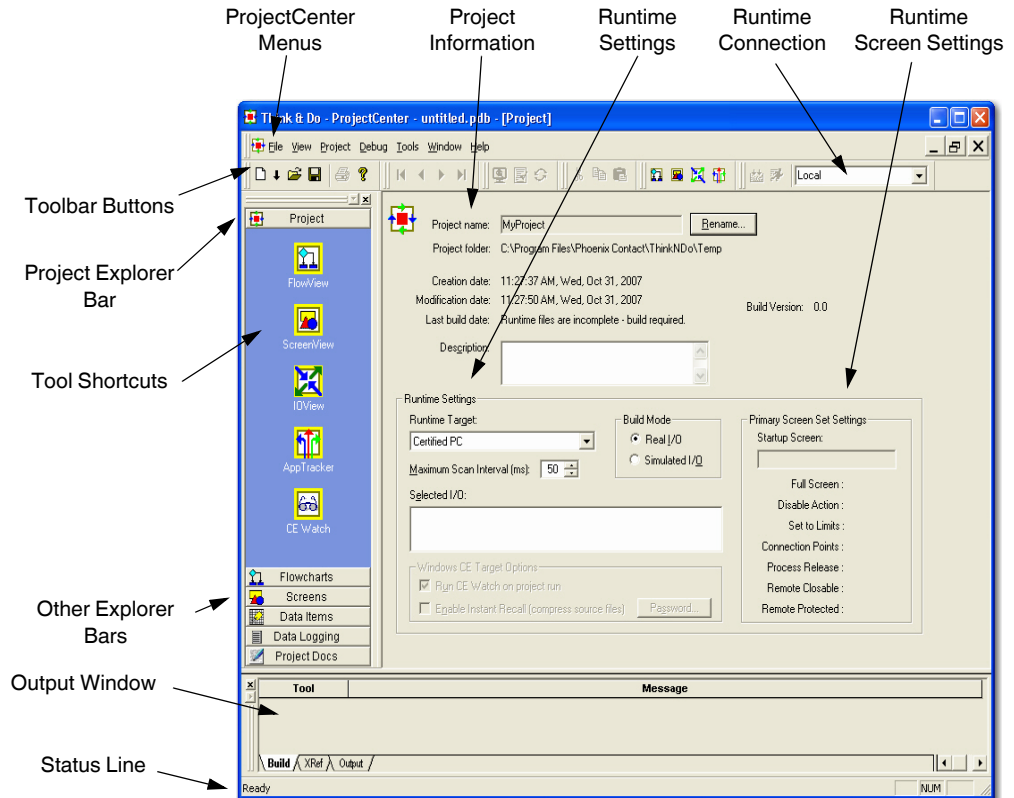


Figure 2-2 ProjectCenter provides access to Think & Do development tools

2.1.2 Naming a New Project

To name the project, follow these steps:

1. Click the “Rename” button next to the “Project Name” field. This displays the “Rename Project” dialog box shown in Figure 2-3.

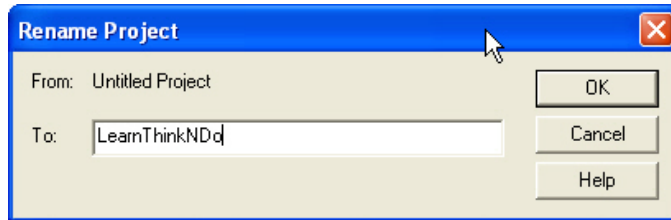


Figure 2-3 The “Rename Project” dialog box with a name filled in

2. Enter the name of the project in the “To” field.
3. Click the “OK” button.

To save the project, follow these steps:

1. In ProjectCenter, select the “File... Save As...” or “File... Save” menu, or click the “Save” button to display the “Save As” dialog box shown in Figure 2-4.

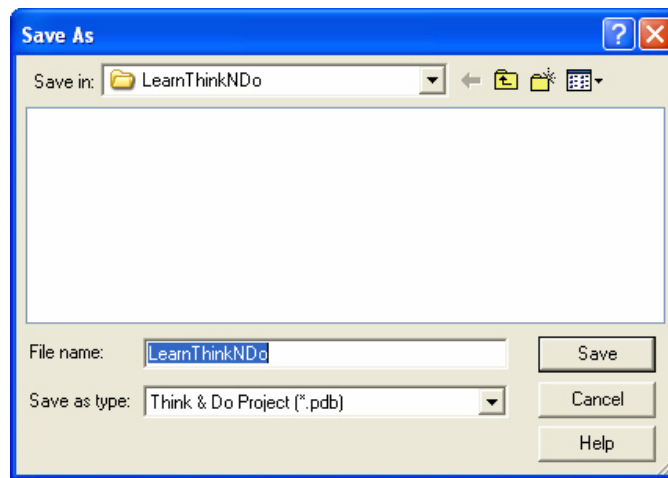


Figure 2-4 Use the “Save As” dialog box to define the project name

2. To use the default path (Program Files... Phoenix Contact... Think & Do... Projects) and project name, simply click the “Save” button.



Renaming a new project before saving it automatically creates a new directory with the name of the project.



Only one project can be stored in a directory.



Create New Folder Button

3. To use a different directory, use the “Save In” drop-down to navigate to the desired drive and directory. Create a new directory by clicking the “Create a New Folder” button. When the new folder appears in the “Save As” list, type the desired name and press the <Enter> key. Then double-click the folder to open it.
4. To change the name of the project, enter a new name in the “File name” field. Note that only alphanumeric characters are valid.
5. Click the “Save” button to save the project.



Open Button

2.1.3 Opening a Project

To open a project, follow these steps:

1. In ProjectCenter, select “File... Open...” or click the “Open” toolbar button to display a file selection browser.
2. Navigate to the folder that has the “.pdb” file.
3. Select the file and click “Open” or double-click the file in the browser.

2.1.4 Project Explorer Bar

Click the “Project” Explorer bar (at top of pane) to view the Think & Do tool shortcuts. Click any of these shortcut icons to directly access the Think & Do tools (see Figure 2-5).

- FlowView
- ScreenView
- IOView
- AppTracker
- CE Watch

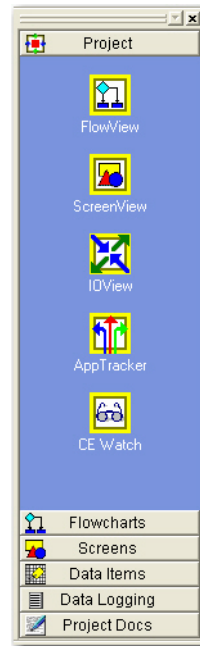


Figure 2-5 The “Project” Explorer bar provides access to Think & Do tools

2.1.5 Flow Chart and Screens Explorer Bars

Click the second and third Explorer bars to view the project's flow charts and screens. The initial folders name the categories that can be expanded after flow charts and screens exist, listing them in a tree structure for project management.

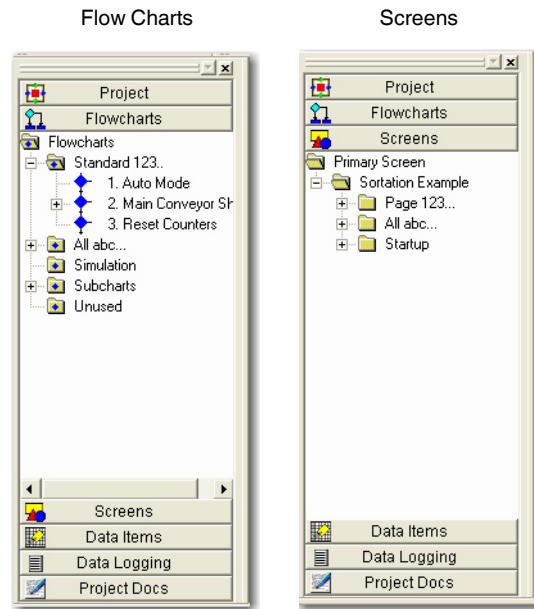


Figure 2-6 The “Flow Charts” and “Screens” Explorer bars

2.1.6 Data Item Explorer Bar

The “Data Item” Explorer bar, fourth on the list, shows folders categorizing project data items (or variables). For a complete discussion of data item types, see “Tagname Data Types and Formats” on page 7-6.

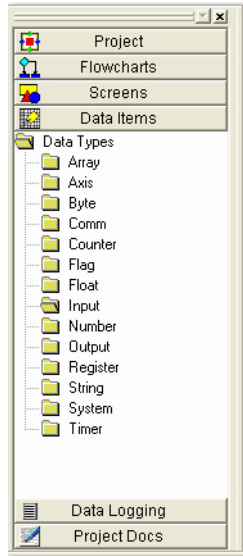


Figure 2-7 The “Data Items” Explorer bar

2.1.7 Other Explorer Bars

The “Data Logging” and “Project Docs” Explorer bars provide additional capability discussed in later chapters.

2.2 Selecting a Runtime Target Type

When creating a project, it is essential to select the type of target for the project at Runtime. For many users it will be the same PC on which they are developing the project. Others may have remote or embedded controllers.

To select the Runtime target:

1. Click the “Project” Explorer bar (see Figure 2-2 on page 2-4). The project information displays in the main ProjectCenter window.
2. In the “Runtime Settings” group (see Figure 2-2 on page 2-4), choose the proper Runtime target from the drop-down list.
3. If you select one of the “Windows CE” options, there are additional available options. You can choose to have CE Watch run automatically whenever the project runs (see “Think & Do CE Watch for Windows CE Targets” on page 9-9). There’s also an option to “Enable Instant Recall” (see “Instant Recall for Windows CE Target” on page 9-12).

2.2.1 Local/Remote

If you select “Certified PC” as the Runtime target, ProjectCenter lists “Local” (the default name for the current PC) in the “Runtime Connection” drop-down menu (see Figure 2-2 on page 2-4). To use a remote PC for the Runtime, type the name of that PC in the Runtime target field. (For more information on connecting to a remote PC, see “Setting Up Remote Communications” on page 8-23).

2.2.2 Selecting a Windows CE Target

One or several Windows CE systems may be on the network, connected to the local PC. When several systems are attached, ProjectCenter can only target one at a time. Think & Do interacts with the proper device for project uploads/download, project run/halt actions, etc.



The PC must have a network configuration with TCP/IP protocol, a network adapter card, and special network settings. Otherwise, the procedure that follows may not work. Please refer to the Getting Started booklet that accompanied the Windows CE hardware for instructions. It may also be necessary install the NWLink IPX/SPX protocol if the target device cannot be located and must be reset to establish a compatible IP address.

To select a Windows CE target:

1. Choose one of the “Windows CE” options in the “Runtime Target” field. The “Runtime Connection” field in the toolbar will display “Find CE Runtime...”
2. Click the drop-down arrow and then click “Find CE Runtime...” from the drop-down list.



If the network is disconnected or power is removed from target devices, the “Find CE Runtime” may display the “Target Picker error” dialog box. If that occurs, check the network connections and power supply to target device(s).

3. The “Find/Configure CE Runtime” dialog box, shown in Figure 2-8, appears. Review the list of all Windows CE devices found on the network.

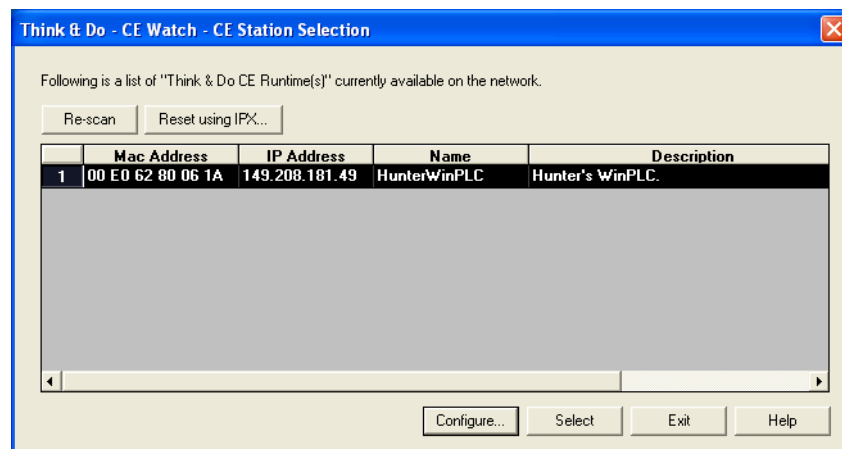


Figure 2-8 The “Find...Configure CE Runtime” dialog box

For example, Figure 2-8 lists a WinPLC at a particular network IP address, which Think & Do uses to identify the device.

4. Highlight the desired Windows CE target device in the list.
5. Click the "Select" button to choose the selected target device and exit the dialog box.



The "Find/Configure CE Runtime" dialog box in Figure 2-8 provides utilities for resetting and configuring Windows CE Runtime targets. The IP address is user-configurable, but the MAC address (media access control) is a unique hardware address for each device that cannot be changed. Please refer to the Windows CE device documentation for details on how to reset and configure the IP address.

The name of the selected target will appear in the list box for the "Runtime Target" field (see Figure 2-2 on page 2-4). Later, when the project runs, ProjectCenter automatically sends compiled programs to run on the selected target.

2.3 Specifying Inputs, Outputs, and Data Items



IOView Icon

Think & Do uses a separate application, IOView, to define a project's inputs, outputs, and data items. To launch IOView, select "Tools... IOView" or click the "IOView" icon in the "Project" Explorer bar.

The IOView defines the drivers in the project and shows them graphically. For many drivers, IOView can automatically detect and display connected devices. For more information on IOView, see Section 3, "Drivers, Devices, and Tags".

2.4 Creating Control Programs

Think & Do uses flow chart control programs developed in FlowView. A flow chart is a type of diagram that uses symbols to illustrate a control process. The symbols in the flow chart depict two classes of items, action blocks and branching blocks (see Figure 2-9). These are defined in the following sections.

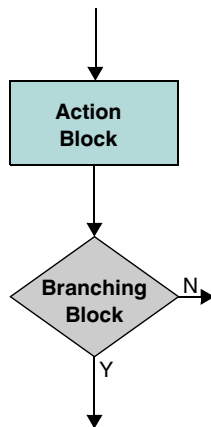


Figure 2-9 Flow chart showing an action and branching block

2.4.1 Action Block

A box representing an operation on the data in the system, which may include input or output. In the flow chart, an action block has one entry point (top) and one exit point (bottom).

2.4.2 Branching Block

A diamond-shaped box represents a branch in the control path based on available data. The decision may compare one variable to another, a variable to a constant, or a true or false condition. Branching blocks have one entry point (top) and two exit points (either side and bottom, or both sides).

With just branching and action blocks, a program can have complete command of the I/O sub-system, or control a machine or process. A project may consist of one or several flow charts.

2.4.3 Building Flow Charts

Use FlowView to create flow charts. To create a new flow chart follow these steps:

- From the ProjectCenter, launch FlowView using one of the following techniques:
 - Select "Tools... FlowView."
 - Click the "FlowView" icon in the Project Explorer bar.
 - Select "File... New... Flow Chart" to launch FlowView, the flow chart drawing tool. If using this technique, continue with Step 3.
- In FlowView, select "File... New... Flow Chart".
- In FlowView, the "Name Flow Chart" dialog box appears (see Figure 2-10). Type a name for the flow chart.



FlowView Icon

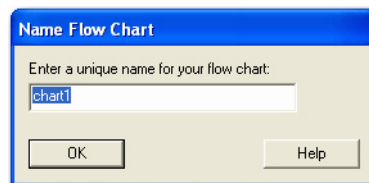


Figure 2-10 "Name Flow Chart" dialog box

- Click the "OK" button.

For more information on how to use FlowView, see Section 4, "General Programming Techniques". For a complete description of flow chart programming, see "Flow Charts" on page 5-3.

2.5 Creating an Operator Screen

Think & Do provides more than just PC-based control with flow charts – it includes a full-featured operator interface or HMI (Human-Machine Interface). Use ScreenView to create operator screens.



Windows CE target Runtime systems – some devices, such as the WinPLC, do not have direct screen output available to its operating system.

2.5.1 HMI Overview

The final output of the development work is a screen or series of screens. These appear in a window on the PC at Runtime. The purpose of a screen is to communicate important information to a machine operator or process engineer. It also provides an opportunity for operator-controlled input. Screens typically include:

- The current state of the machine or process (such as Idle, Run, or Stop)
- Alarm conditions, if they exist
- Production data (such as number of parts, percentage defects, downtime reasons, and trends)
- Diagram of process (on-screen diagram to help orientate the operator)

ScreenView is a full-featured HMI or operator interface design package, which helps create screens that really communicate. In addition to standard drawing constructs, the screen object shape toolbars provide an ability to create dynamic objects (change color, size, position, or rotate) and active objects (alarms, trend controls, ActiveX controls, and .NET controls). Operator input support includes function keys and data entry with text fields, check boxes, radio buttons, push buttons, and drop-down selection lists.

2.5.2 Screens as Part of a Project

The screens developed using ScreenView are part of a project. At Runtime, the screens appear in an independent window on the PC desktop. Only one screen is visible in the window at a time. You can resize or minimize HMI screens, just as any window. However, you cannot exit or quit screen windows (the Runtime must be stopped instead, which shuts down the operator screens in the project). Data entry or function key entry requires that the screen be in the active window.



When a project runs, its flow charts are the highest priority task – not screens. During the “Pause” phase of each scan period, the system executes screen tasks and other applications. Adding screens affects system loading, so be sure the scan time setting is large enough to allow the Runtime to service operator screen(s).

To create a new screen in the project:

1. In ProjectCenter, click “File... New... Screen” to open ScreenView and display the “Screen Attributes” dialog box (Figure 2-11).

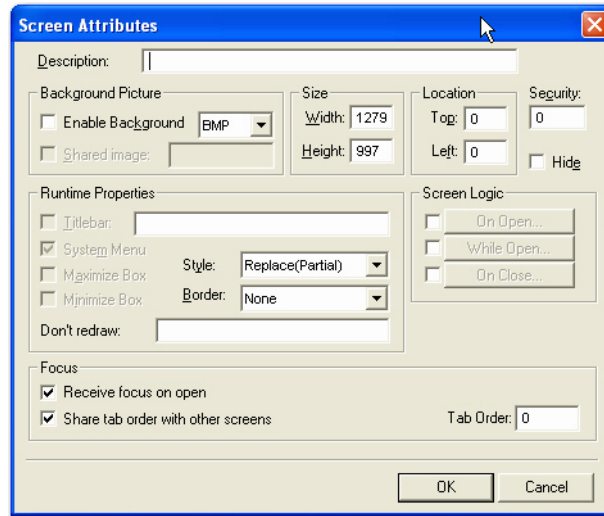


Figure 2-11 “Screen Attributes” dialog box

2. Enter a description for the screen, and make any other required changes.
3. Click “OK” to create the screen.

For more information on how to use ScreenView, see “General Programming Techniques” on page 4-3. For more information on HMI screen development, see “Operator Screen Techniques” on page 6-3.

This section informs you about

- Specifying inputs, outputs, and data items
- Configuring I/O
- Adding I/O drivers and devices
- Mapping I/O
- Scanning and monitoring I/O, saving the I/O configuration
- Using a watchdog timeout

Drivers, Devices, and Tags	3-3
3.1 Specifying Inputs, Outputs, and Data Items.....	3-3
3.1.1 Configuring External I/O.....	3-3
3.2 Configuring I/O	3-5
3.3 Adding an I/O Driver	3-7
3.4 Connecting to I/O Devices	3-8
3.4.1 Manually Adding an I/O Module	3-9
3.4.2 Editing an I/O Module	3-10
3.4.3 Deleting an I/O Module	3-10
3.5 Mapping I/O Points to Data Items	3-11
3.6 Scanning and Monitoring I/O	3-13
3.7 Saving the I/O Configuration	3-14
3.8 Tips on Toolbar Functions	3-14
3.9 Watchdog Timeout Default I/O States	3-16
3.9.1 Startup/Shutdown Output State	3-16

3 Drivers, Devices, and Tags

This section discusses the techniques required to define input, output and internal data items used in control programs and HMI screens. Think & Do uses a common tagname database for all I/O and variables throughout the project.

3.1 Specifying Inputs, Outputs, and Data Items

All variables in the control system may be classified into two simple categories:

- External: Inputs and Outputs or any tagnames mapped to external I/O. These may include I/O bases, motion axes, or serial port devices.
- Internal: internal variables for storage, calculations, and manipulation

All data items can have meaningful tagnames of up to 30 characters. All data types are for internal use in a Think & Do project. However, you can map Inputs, Outputs, Counters, Numbers, Floats, Strings, and Comm data items to external I/O devices.



The space character is not valid for tagnames. This ensures that tagnames comply with common standards to transfer data items across networks and between different applications.

3.1.1 Configuring External I/O

Think & Do has many available software drivers that provide the connection between the PC and I/O network or scanner cards. Each card has a corresponding driver. An I/O network or scanner card may have one or more nodes. For example, the Momentum Ethernet and DeviceNet cards support multiple nodes, each with a single *module*. A module is either an I/O base (rack) or a field device. Some network cards may have nodes that support multiple modules. For example, Seriplex and INTERBUS cards have a single node that supports multiple modules. Finally, there are scanner cards that support multiple nodes with multiple modules on each node, such as Automationdirect.com Ethernet and Opto 22 Optomux I/O.

Adding Data Items

To add new data items:

1. In ProjectCenter, click the “Data Items” Explorer bar. The “Data Types” folder tree appears in the “Data Items” Explorer bar (Figure 3-1).

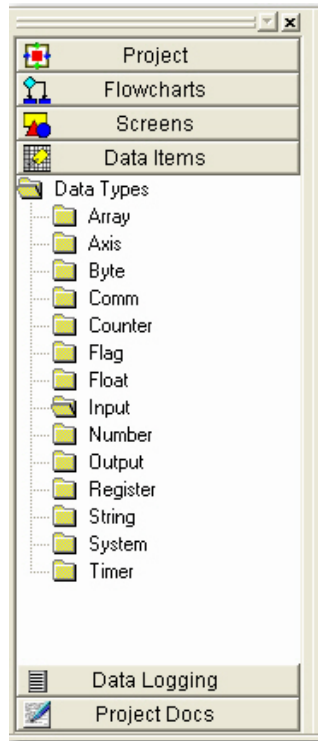


Figure 3-1 ProjectCenter Data Items

2. Click the data type folder for the data type desired to display the Data Item grid. The Data Item grid shown in Figure 3-2 is for Counter data types. The Data Item grid has a fixed “C” (for Counter) in the left-most column of each row.

Index	Tagname	Initial	Retentive	Description	Mapped to...
C 1	WhiteAccumulationCounter	0	<input type="checkbox"/>		
C 2	GreenAccumulationCounter	0	<input type="checkbox"/>		
C 3	RedAccumulationCounter	0	<input type="checkbox"/>		
C 4	YellowAccumulationCounter	0	<input type="checkbox"/>		
C 6	WhiteDiverterPosition	7	<input type="checkbox"/>		
C 7	GreenDiverterPosition	12	<input type="checkbox"/>		
C 8	RedDiverterPosition	18	<input type="checkbox"/>		
C 10	WhitePackageID	1	<input type="checkbox"/>		
C 11	GreenPackageID	2	<input type="checkbox"/>		
C 12	RedPackageID	3	<input type="checkbox"/>		
C 23	YellowPackageID	4	<input type="checkbox"/>		
C			<input type="checkbox"/>		

Figure 3-2 The Data Items grid for Counter data types

3. In the main Data Item grid, type the tagname desired. Press the <Enter> key or click outside the tagname field to enter.



By default, Think & Do increments the Index as tagnames are entered in the Data Item grid. You can, however, change these values to non-sequential numbers. The data type and index uniquely identify each tagname throughout Think & Do.



Save Button

4. Click the “Save” button in the toolbar or select “File...Save” to save the data items added.

Deleting Data Items

To delete a row in the Data Item grid:

1. Select a row – Move the mouse cursor to the left-most column in the row to delete. The cursor changes to a right-pointing arrow.
2. Click to select the entire row, which will then be highlighted.
3. Press the key to delete the row.



You cannot cut-and-paste an entire row because tagnames must be unique within a project.

Editing Text in the Data Item Grid

To copy and paste text in Data Item grid cells:

1. Select the text within a cell.
2. Click the “Copy” toolbar button.
3. Click a different cell and then click the “Paste” toolbar button.

3.2 Configuring I/O

Use the IOView to map data items to I/O points. To launch IOView, follow these steps:

1. In ProjectCenter, click the “Project” Explorer bar (left-most pane, top bar) to view the tool shortcuts.
2. Click the IOView icon to launch IOView.

IOView is a separate tool (not a window in ProjectCenter). Using it produces a single I/O configuration for the project that Think & Do stores with all other project components.

The IOView window (see Figure 3-3) draws a picture of the I/O network as you configure it. After configuration, the I/O network or scanner card(s) in the PC are in the upper-left pane.



IOView Icon

I/O network and scanner cards connect to *modules*. Modules can be either I/O bases or field devices. Modules appear in the upper-right pane. Configuration information appears in the bottom pane of the window as modules are select modules and different tabs at the bottom of the pane.

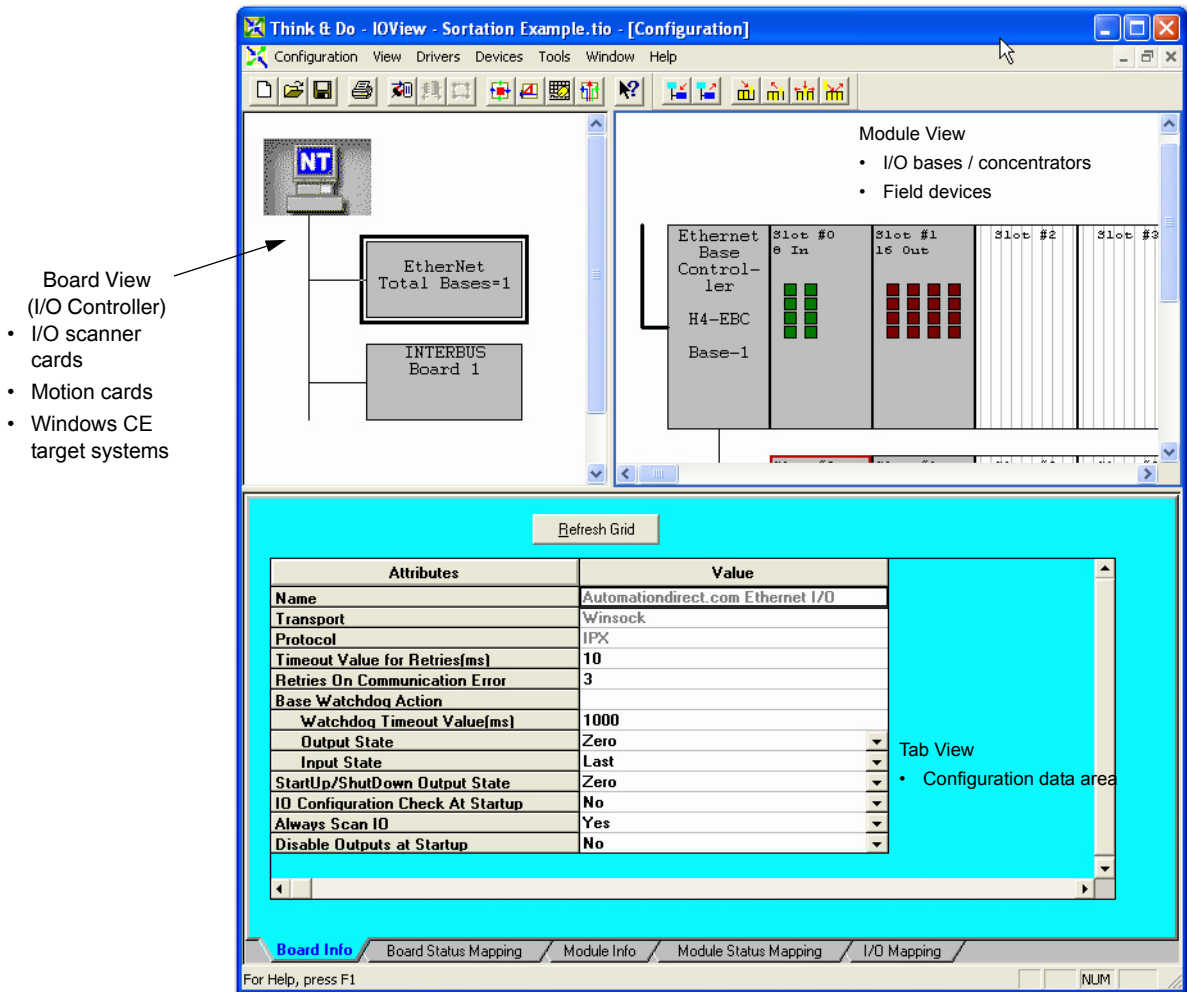


Figure 3-3 IOView window includes a board view, module view, and tab view

The general steps required to configure and map an I/O network are:

1. If a scanner card is available, install it in the PC along with the required driver software that came with the card.
2. Add the appropriate driver for the I/O network or scanner card associated with the Board View I/O controller (the I/O controller may be an I/O scanner or network card, or a Windows CE Runtime target).
3. Do one of the following:
 - If I/O devices are available:
 - a. Connect the scanner card to the I/O devices on the network and build a physical I/O network.

- b. Use Auto-Discovery to find all devices on the network.
- c. Monitor inputs with IOView in real time and turn on the outputs by clicking on the graphic.
- If I/O devices are not available or the Think & Do driver doesn't support Auto-Discovery, generate the configuration manually using these steps:
 - a. Add the device or module manually, as described in "Manually Adding an I/O Module" on page 3-9.
 - b. Map the input points to tagnames.
 - c. Map the output points to tagnames.
- 4. Save the I/O configuration.

3.3 Adding an I/O Driver

You must add a driver for the I/O network, scanner card, or backplane (also called an I/O adapter) connected to the Runtime target. The driver is a special piece of software that Think & Do uses to communicate over the I/O link to physical devices.

To add an I/O driver:

1. Either select "Drivers...Add" or click the "Add Driver" toolbar button.
2. From the list of I/O drivers (Figure 3-4), select the appropriate one for the I/O connected to the target hardware.



Add Driver
Button

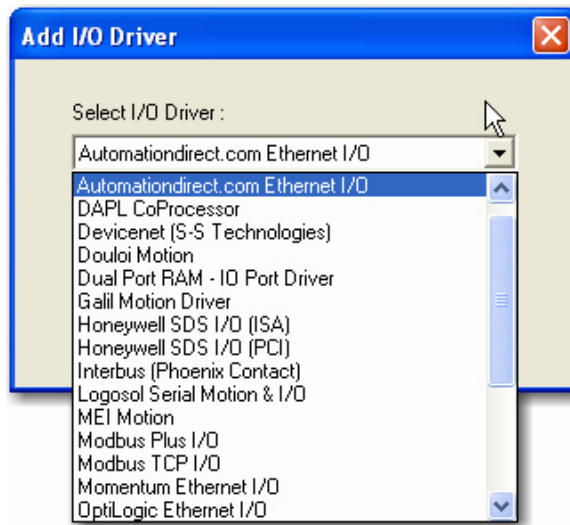


Figure 3-4 The "Add I/O Driver" dialog box and its driver listing



Phoenix Contact is constantly adding new drivers and device types to stay current with the industry's latest products. Visit www.phoenixcontact.com to check the latest news on supported I/O.

3. The next dialog box may prompt for board number, serial port number, etc., depending on the I/O type.

4. Click the “OK” button.

IOView displays an image of the I/O adapter as shown in Figure 3-5 (multiple cards appear in a tree structure). IOView also displays board information related to the I/O card in the lower pane of the window, where you may edit some of the parameters. Generally, it’s a good idea to leave all of the parameters at their default settings, unless you know for certain that they require changing (see online help for details).

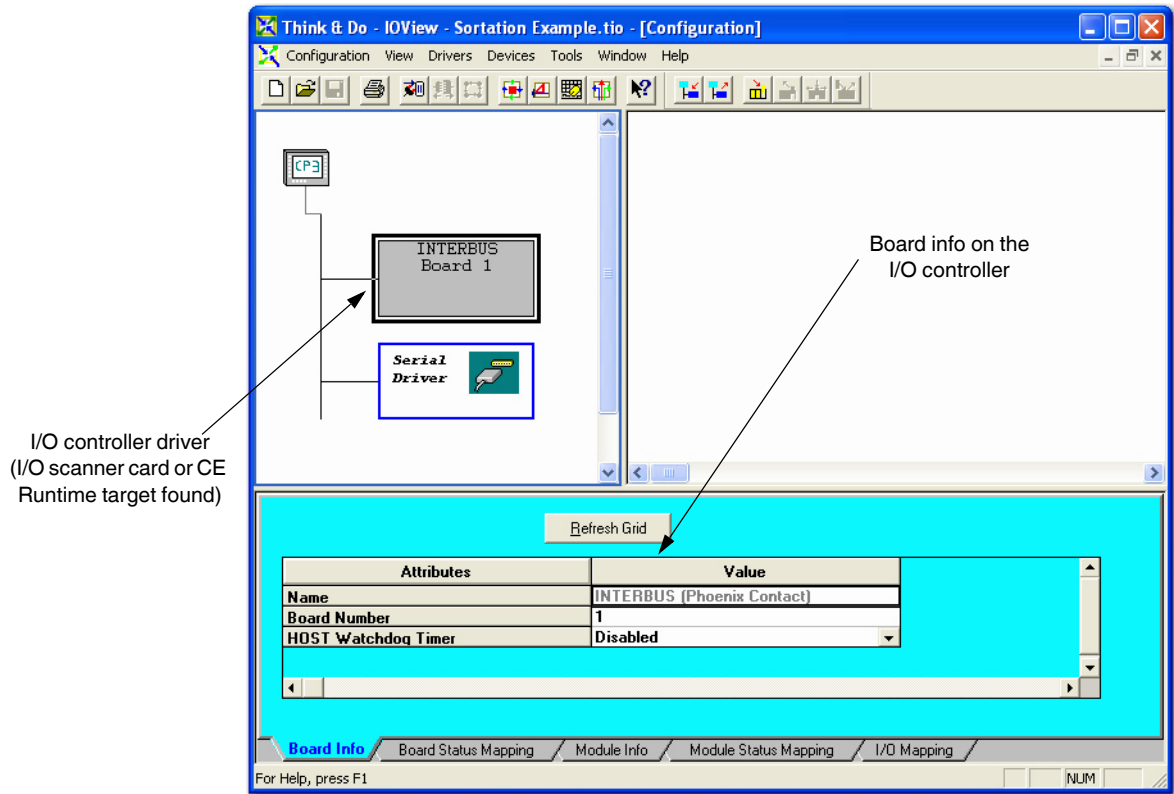


Figure 3-5 IOView displays an image of the I/O scanner card or Runtime target

3.4 Connecting to I/O Devices

After adding an I/O driver to the configuration, you can connect it to actual I/O, if it exists and is connected to the I/O network or scanner card in the PC.

To connect to an I/O system:

- From the “Configuration...Connect” menu (or use the “Connect” toolbar button).

IOView instructs the I/O adapter card to connect to the bases or devices on the network immediately. It reads the status (I/O count and type) from all devices in order to create an image of the physical I/O. Note that not all I/O networks support this capability.



Connect Button

When IOView connects to the I/O, it creates an image similar to the one shown in Figure 3-6. This example matches a DL205 base, which uses an Ethernet Base Controller in the left-most slot. The modules adjacent to it include an 8-point input simulator (F2-08SIM) and an 8-point relay output module (D2-08TR).

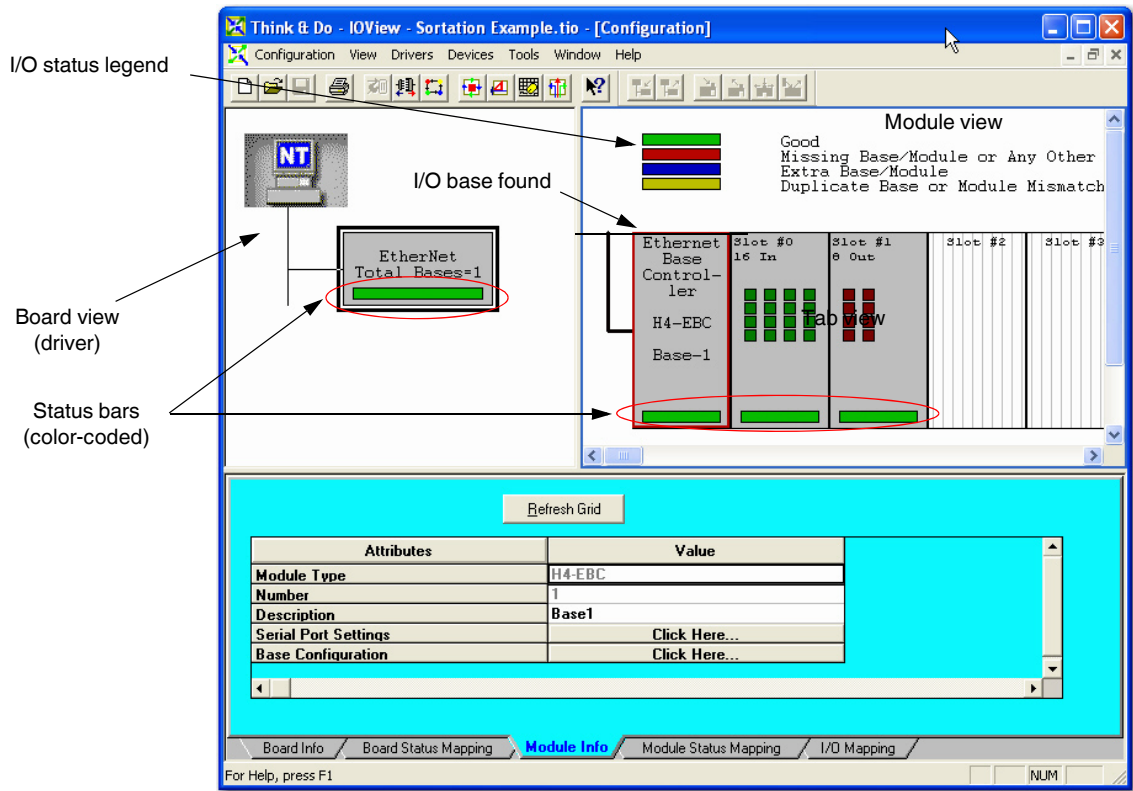


Figure 3-6 IOView showing the "Module Info" tab for selected module

3.4.1 Manually Adding an I/O Module



To manually add an I/O module (or device), follow these steps:

1. Select the driver in the Board view of the IOView.
2. Click the "Devices...Add" menu (or click the "Add Devices" toolbar button) to display the "Add/Insert Device Type" dialog box. This dialog box has one or more drop-down lists depending on the type of driver the device connects to.
3. Choose the device from the drop-down list and make any other selections required.
4. Click the "OK" button to display a picture of the device/module in IOView.

3.4.2 Editing an I/O Module

To select a module and view its I/O configuration, follow these steps:

1. Click the module image in the module view pane of IOView. When selected, the module outline has a red border.
2. Click the tabs in the tab view area to see information about the selected module.

The four colored bars above the I/O base form a legend to help interpret the indicator status bars located on each module. Each module's status bar is green if the selected module is okay.



The image of the base in the figure has more slots than the base which is used in the I/O system. IOView uses the maximum slot count when the base status information does not include slot count.

3.4.3 Deleting an I/O Module

To delete an I/O module (or device), follow these steps:

1. Select the device or module in IOView.
2. Select the "Devices...Remove" menu (or click the "Remove Devices" toolbar button) to delete the device or module without any confirmation.



Remove
Device Button

3.5 Mapping I/O Points to Data Items

After selecting the input module, click the "I/O Mapping" (right-most) tab at the bottom of the View window. The tab view (Figure 3-7) displays the I/O mapping information for the selected module above it.

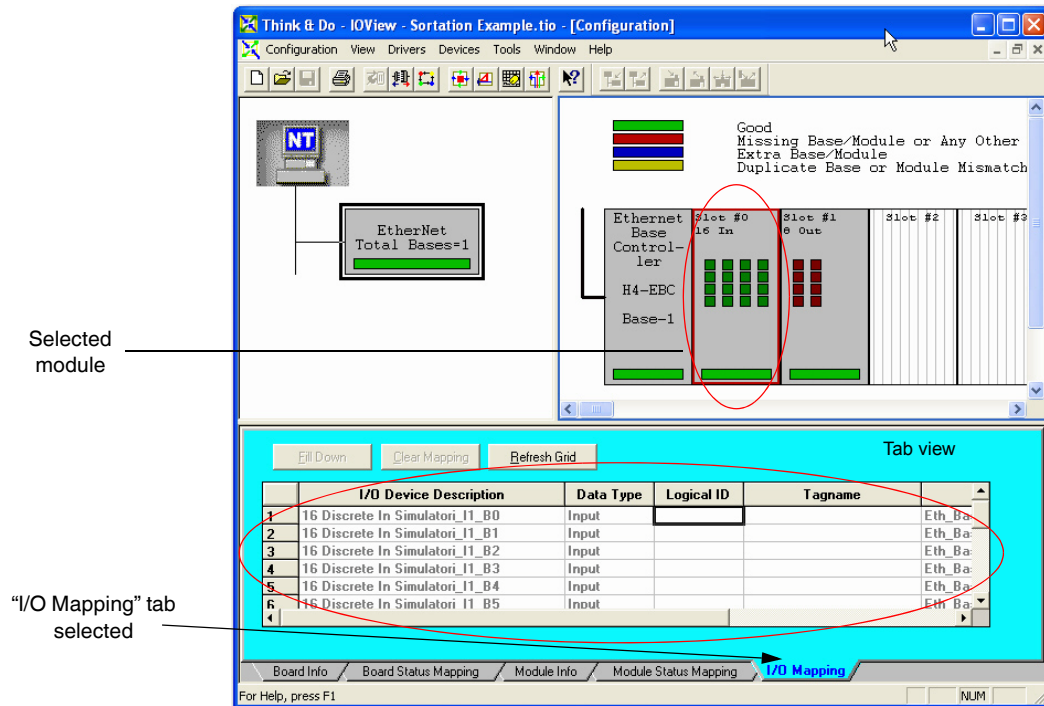


Figure 3-7 IOView showing the "I/O Mapping" tab for selected module

The "I/O Mapping" tab of the IOView window presents information about the selected module in a spreadsheet format.

The column titles and their meanings are:

- "I/O Device Description": Each bit in an I/O module has a table entry and a descriptive prefix that can be user-defined in the "I/O Device Description" field in the "Module Info" tab.
- "Data Type": The data type corresponds to the classification of the tagname in the Data Item grid in ProjectCenter.
- "Logical ID": This is the number the common database uses to access each I/O point. The next section describes how to map Logical IDs.
- "Tagname": The name of the input or output point. This is assigned in ProjectCenter's Data Item grid, but IOView displays it here when I/O points are mapped to data items.
- "Physical I/O": Lists the type of I/O network, base number, slot number, and bit number.

The I/O configuration is determined by mapping physical I/O points to logical ID numbers or tagnames (Figure 3-8). In this example, the first four connections go straight across, but the last four connections cross each other. You can also choose a straight-through mapping or

re-map I/O points by changing the relative order of logical ID (such as I-0, I-1, I-2, I-7, I-4, etc.) to the physical I/O. You can assign unique tagnames to each logical ID or group of IDs (depending on data type).

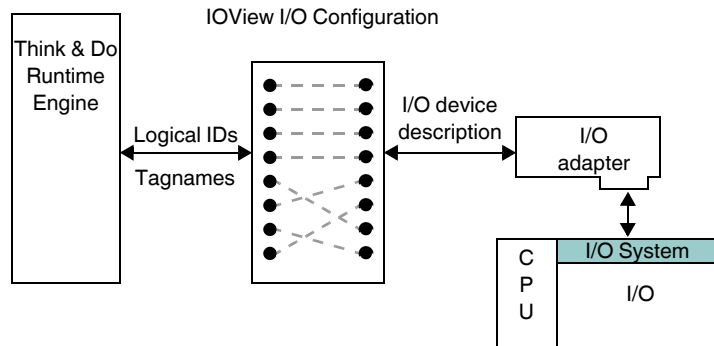


Figure 3-8 Physical I/O points map to logical I/O points in any sequence

To map the input or output module points:

1. Select a module in IOView.
2. In the “Tab” panel, select the “I/O Mapping” tab.
3. Click the cursor in the “Logical ID” column of the first row.
4. Type the logical ID number desired to define (for example, if an input module is selected, you might enter 0 as the first logical ID), and then press the <Enter> key. The Logical ID (following the example, I-0) appears, along with the corresponding tagname defined in ProjectCenter (see “Adding Data Items” on page 3-4).



A convenient way to map I/O points is to double-click any cell in the “Logical ID” or “Tagname” column, and IOView displays the Data Item grid in ProjectCenter to selection of the desired tagname from the tagname database. To automatically copy a logical ID and tagname to the I/O mapping table, double-click the first column of the “Data Item” table, or select the row and click the “OK” button in the dialog box.

Since many projects use a simple, straight-through mapping, IOView provides an auto-complete function, called Fill Down. The Fill Down feature automatically enters sequential, logical IDs and tagnames.

To use Fill Down to complete I/O mappings:

1. Enter the first Logical ID number in the top cell of the Logical ID column.
2. Select the entry in the top cell in the Logical ID column.
3. Click and drag the cursor downward, releasing it over the bottom row. This selects the remaining logical ID cells and highlights them in black. This enables the “Fill Down” button that appears above the grid.
4. Click the “Fill Down” button. IOView automatically fills in the selected cells, incrementing the logical ID number for each cell. The associated tagnames appear automatically.

Think & Do ensures tagname changes are only made once. For example, changing a tagname in ProjectCenter’s Data Item grid is reflected automatically in IOView or in any other part of Think & Do.



Edit the default “I/O Device Description” field to create a meaningful input/output point name for the application (located in the “Module Info” tab).



Fill Down Button

3.6 Scanning and Monitoring I/O



Connect
Button



Scan Button

IOView has the ability to monitor actual inputs and turn on outputs to test wiring independent of control logic.

To monitor I/O from IOView:

1. Click the “Configuration... Connect” menu, or use the “Connect” toolbar button.
2. Click “Configuration... Scan” menu, or use the “Scan” toolbar button.

To check an input point:

1. Turn an input ON or OFF.
2. Watch the on-screen image of the input module as the physical input changes to monitor its status.

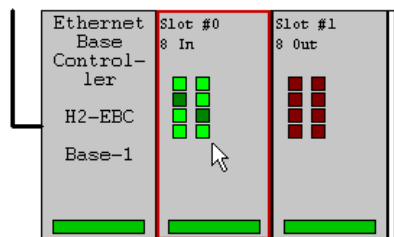


Figure 3-9 The on-screen images in IOView reflect input and output states

Turn on outputs by clicking the LED on the graphic (available in several I/O drivers). A warning message appears before the actual I/O update.

Set analog values by clicking on the graphic of the module and channel. IOView displays a dialog box to input the desired value for the output channel.

3.7 Saving the I/O Configuration



Disconnect Button

After configuring and checking the status of I/O, it's a good idea to save the I/O configuration for the project.

To save a newly scanned configuration:

1. Disconnect the I/O by selecting the "Configuration...Disconnect" menu or clicking the "Disconnect" button. (This automatically makes I/O scan inactive).
2. The "Save" dialog box appears. Click the "Yes" button to save the configuration with the project.

Each project can have only one I/O configuration associated with it. Edit an I/O configuration by using the "Devices" menu selections. Click the "Save" menu button after any I/O configuration edits.

3.8 Tips on Toolbar Functions

The toolbar in IOView includes special-purpose buttons for convenience.

- The buttons "Connect," "Disconnect" or "Scan" access the I/O directly from the toolbar.

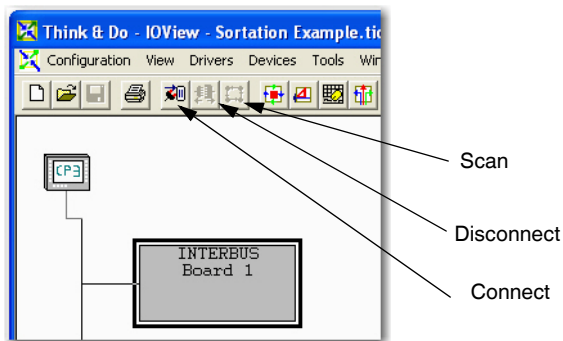


Figure 3-10 Connection-related buttons available in the toolbar

- Connect and Disconnect are opposite (and alternately available) functions.
- Scan works only when the Connect function is active.
- A Disconnect while Scan is active automatically makes Scan inactive.

The second set of toolbar buttons make off-line editing easier, when either the devices are not available to scan or the driver does not support scanning.

- The “Add Driver” and “Remove Driver” buttons configure I/O adapters, motion control boards, or serial ports in the target Runtime.

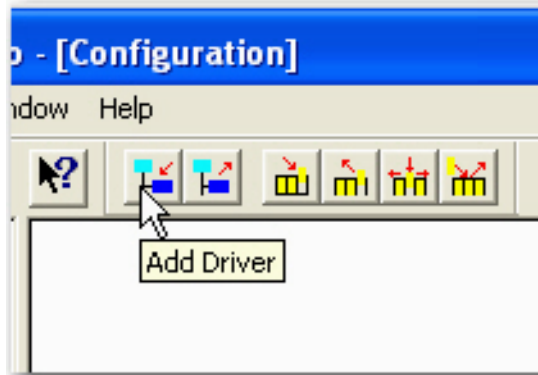


Figure 3-11 “Add” and “Remove” driver buttons

- Each driver’s I/O controller is shown in the Board view pane of IOView in a tree diagram.
- The remaining buttons on the toolbar perform the Add Device, Remove Device, Insert Device, and Replace Device functions. For most I/O these buttons can configure a desired I/O network for the project before the actual devices are connected to the PC.

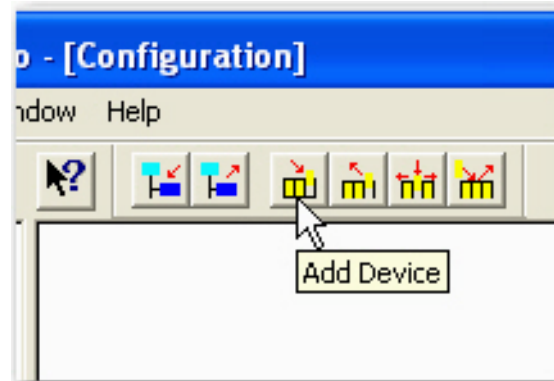


Figure 3-12 “Add,” “Remove,” “Insert” and “Replace” device buttons

- Undocking a toolbar creates a floating window of buttons by clicking and dragging on an empty part of the toolbar. You can dock a toolbar window on any window border by dragging the window to the border and releasing the mouse button after the window snaps into place.

3.9 Watchdog Timeout Default I/O States

For most projects, the actual state of I/O points during a communications failure or during startup and shutdown is a major concern. Many I/O devices support special features to specify the state of the I/O at these critical times with a specified communications watchdog timer action. IOView provides complete access for configuring preferences for these situations.

To configure watchdog timeout states:

1. Click the module graphic for the desired module to configure.
2. Click the "Board Info" tab and note the "Output State" or "Input State" attributes in the "Watchdog Action" group. The default settings are "Zero" (off) for outputs and "Last State" (like a latch) for inputs. The accompanying list boxes select "Zero", "Last State", or "Pattern" for inputs and outputs.

3.9.1 Startup/Shutdown Output State

This attribute defines what the outputs will do during startup or shutdown periods. For example, the Ethernet base controller supports two modes:

FailSafe

If the network communication times out between the Runtime and the base controller, you can configure output and input states.

Startup/Shutdown

This feature defines the on-off pattern used for outputs when the Runtime starts and stops. Choosing a pattern as an I/O state permits a combination of on/off states for input or output modules. This requires further definition of the pattern for each module in the base. Select the "Module Info" tab, and note the Fail-safe I/O Pattern attribute, with the "Click Here..." prompt.



The pattern specified applies to either or both "Watchdog time-out" and "Startup/Shutdown" categories, if pattern is the specified I/O state.

To configure an I/O Pattern, follow these steps:

1. Use the "Click Here..." button to access the Pattern dialog box. The entry field is in hex format and Think & Do decodes and displays the value in binary (MSB to LSB, left to right) to confirm individual I/O point states.
2. Click the "OK" button when finished.

Section 4

This section informs you about

- General techniques for creating and editing flow charts and subcharts
- General techniques for creating and editing operator screens

General Programming Techniques	4-3
4.1 Flow Charts and Subcharts	4-3
4.2 Exploring Flow Charts and Subcharts.....	4-4
4.3 Creating a Flow Chart.....	4-5
4.4 Changing Flow Chart Settings	4-6
4.4.1 Changing Execution Order.....	4-6
4.4.2 Changing a Flow Chart's Category	4-7
4.4.3 Renaming, Copying, Deleting, or Printing Flow Charts	4-7
4.5 FlowView Orientation	4-8
4.5.1 Adding Flow Chart Blocks to a Flow Chart.....	4-9
4.5.2 Turning the Grid On and Off.....	4-9
4.5.3 Increasing the Number of Columns.....	4-9
4.5.4 Changing Zoom Level.....	4-10
4.6 Displaying Multiple Flow Charts.....	4-11
4.7 Editing Flow Charts	4-12
4.7.1 Cut-Copy-Paste	4-12
4.8 Flow Chart Block Introduction.....	4-12
4.8.1 Enable Block.....	4-12
4.8.2 Freeze Block.....	4-13
4.8.3 Types of Action Blocks	4-13
4.8.4 Types of Branching Blocks	4-13
4.8.5 Notes	4-15
4.9 Connecting Flow Chart Blocks	4-15
4.10 Editing Flow Chart Block Expressions	4-16
4.11 Entering Block Description	4-17
4.12 Operator Screens	4-18
4.12.1 Screens as Part of a Project.....	4-18
4.12.2 Creating Operator Screens.....	4-19
4.12.3 ScreenView Development Environment	4-19
4.12.4 Object Properties Dialog Box.....	4-21
4.12.5 Creating and Using Tags in ScreenView.....	4-26
4.13 Scripting Languages in ScreenView	4-27

4 General Programming Techniques

Think & Do uses flow chart control programs. The symbols in the flow chart depict two classes of items: action blocks and branching blocks.

Action Block

An action block is a box (see Figure 4-1) representing an operation on the data in the system, which may include input or output data. In the flow chart, an action block has one entry point (top) (top, left, or right) and one exit point (left, right, or bottom). Think & Do has four types of action blocks that are described later (see “Types of Action Blocks” on page 4-13).

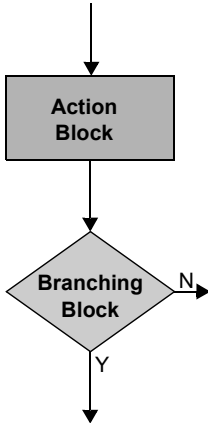


Figure 4-1 Flow chart segment showing an action and branching block

Branching Block

A branching block is a diamond-shaped box (see Figure 4-1) that represents a branch in the control path based on available data. A branching block may compare one variable to another, a variable to a constant, or a true or false condition. Branching blocks have one entry point (top) and two exit points (either side and bottom). Think & Do has two types of branching blocks that are described later (see “Compare Block” on page 5-10 and “Decision Block” on page 5-14).

Using just action and branching blocks, flow charts can have complete control of I/O systems, or control a machine or process. A project may consist of one or several flow charts.

4.1 Flow Charts and Subcharts

Flow chart programs in Think & Do provide two chart types for hierarchical structure: *Standard* flow charts and *Subcharts*. Their relationship is similar to main program and subroutine in BASIC language programs. A flow chart Call block calls a subchart, which transfers execution to the subchart. The subchart does a return to the original flow chart, transferring execution back again (Figure 4-2).

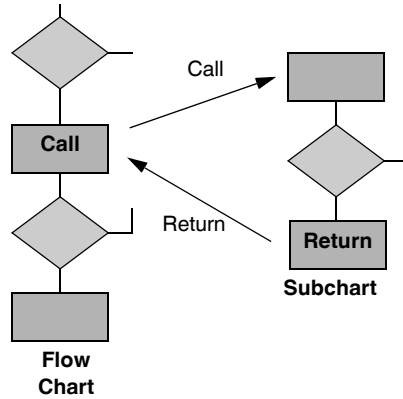


Figure 4-2 A Call block calls a subchart, which returns to the original flow chart

Using subcharts with flow charts provides great flexibility in project design. A flow chart can pass variables to a subchart through parameters you define. Then the subchart can pass results back to the calling flow chart. Well-designed subcharts are generic, making them easy to re-use in future projects. This saves time in future project development. This aspect of project design is covered in detail in “Flow Charts” on page 5-3.

4.2 Exploring Flow Charts and Subcharts

In the ProjectCenter “Project” Explorer bar, click the “Flow Charts” Explorer bar to see a list of the project’s flow charts (see Figure 4-3).

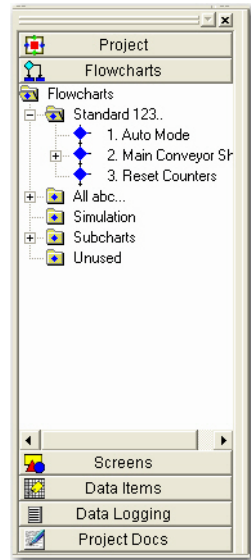


Figure 4-3 The “Project” Explorer bar showing the Flow Chart explorer

Folders group flow charts by categories. The categories are not necessarily exclusive, meaning that a flow chart can appear in multiple folders. The categories are:

- “Standard 123...”: lists flow charts in the order that they are executed during each scan.
- “All abc...”: lists both flow charts and subcharts together in alphabetical order.
- “Simulation”: lists simulation flow charts (can write to input points).
- “Subcharts”: lists the project’s subcharts (if any) in alphabetical order. Note that the actual order of execution is determined by when they are called.
- “Unused”: lists any unused flow charts (a category to temporarily disable execution).



You can also place unfinished flow charts in the “Unused” category.

When the project has both standard flow charts and subcharts, the hierarchical relationship is shown in ProjectCenter's "Flow Chart" Explorer bar (see Figure 4-4).

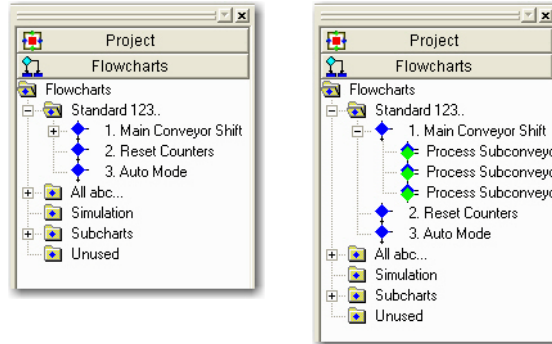


Figure 4-4 Heirarchical sublist list collapsed (left), expanded (right)

In the example in Figure 4-4, the "Main Conveyor Shift" flow chart has one or more associated subcharts as indicated by the + to its left.

To expand the Flow Chart explorer tree:

- Click the + beside the folder to expand ("Main Conveyor Shift" in the example).
- Press the keypad <+> key to expand the "Standard 123..." the selected folder. Use the keypad <-> key to collapse a selected folder

In Figure 4-4, three subcharts are called from the flow chart "Main Conveyor Shift". Note that more than one standard flow chart can call a particular subchart, so an instance of that subchart appears in the directory tree under each standard flow chart that calls it. In addition, an instance appears for each call from a flow chart. The subchart appears once in the "All 123..." folder and the "Subchart" folder.



FlowView Icon

4.3 Creating a Flow Chart

To create a new flow chart, follow these steps:

1. In ProjectCenter, launch the FlowView using one of the following techniques:
 - a. Click the "Tools...FlowView" menu.
 - b. Click the "FlowView" icon in the "Project" Explorer bar.
 - c. Click the "File... New... Standard Chart" menu. This will launch FlowView, the Think & Do flow charting tool. If using this technique, continue with Step 3.
2. In FlowView, select "File... New... Standard Chart" menu.

3. In FlowView, a “Name Flow Chart” dialog box appears (see Figure 4-5). Type a name for the flow chart.

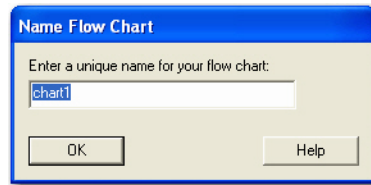


Figure 4-5 “Name Flow Chart” dialog box

4. Click the “OK” button.

4.4 Changing Flow Chart Settings

All standard flow charts in a project execute once each logic scan. The “Standard 123...” folder under the “Flow Chart” Explorer bar lists the execution order (top one first, bottom last).

4.4.1 Changing Execution Order

To change the order of flow chart execution:

1. Click the + beside the “Standard 123...” folder to expand it and show all standard flow charts.
2. Place the cursor over the diamond symbol by the flow chart name to re-order.
3. Click and drag the flow chart to a new location on the list. The tree listing then shows the new order of execution.



Only the “Standard 123...” listing of flow charts may be re-ordered. Other folders simply list the category of flow charts they contain in alphabetical order.

4.4.2 Changing a Flow Chart's Category

To change a flow chart to/from Simulation or Unused category:

1. Select the flow chart name in the "Flow Chart" explorer window.
2. Do one of the following:
 - In the main ProjectCenter window, click the check box for the desired setting as shown in Figure 4-6.

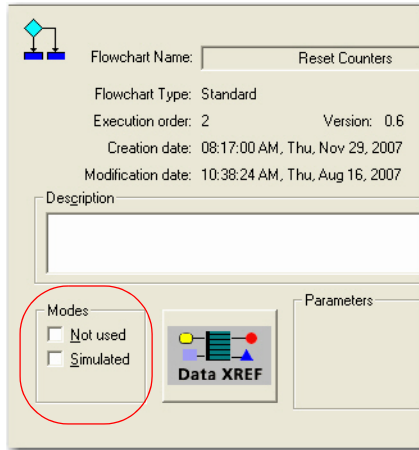


Figure 4-6 Select the "Unused" or "Simulation" check box to move a flow chart

- Drag the flow chart to the appropriate folder.

4.4.3 Renaming, Copying, Deleting, or Printing Flow Charts

To rename, duplicate, delete, or print an existing flow chart:

1. Move the cursor over the name of the flow chart in the Flow Chart Explorer pane.
2. Right-click to get the pop-up menu, and select the desired action.



Before creating a new flow chart in a new project, it's best to name the project first, by saving it to a new directory (each project must be in a unique directory).

4.5 FlowView Orientation

When you create a new flow chart, FlowView shows it maximized in the workspace (see Figure 4-7). Use the “Window” menu to show multiple open flow charts by selecting “Cascade” or one of the tile options. Switch to other open flow charts either using the list in the “Window” menu, or by pressing the <F6> or <Ctrl>+<Tab> (<Shift>+<F6> or <Shift>+<Tab> to reverse the order) key.

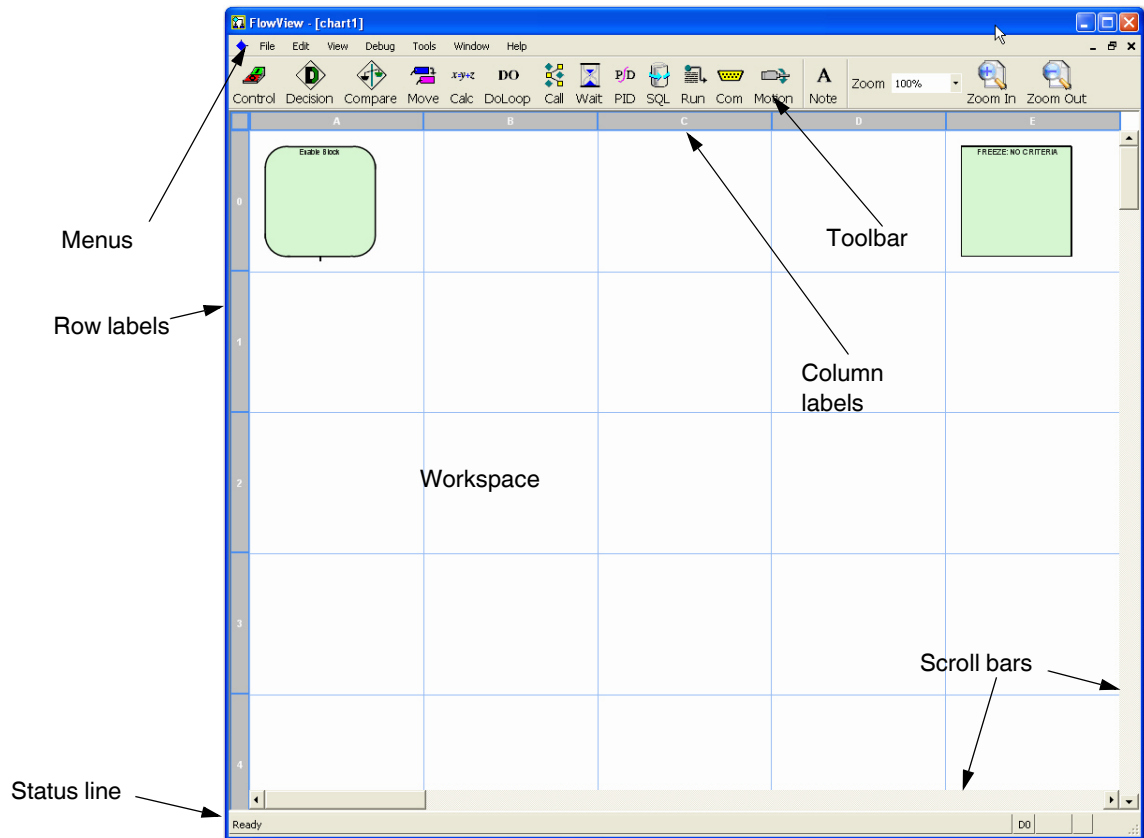


Figure 4-7 FlowView showing a new Flow Chart

The key elements of the FlowView window, shown in Figure 4-7, are:

- Menu: Provide access to all the commands in FlowView, as well as online help.
- Toolbar: Contains toolbar buttons that trigger actions. These actions duplicate many commands that are available in the menus.
- Column labels: Provide a guide for locating shapes in the drawing area. FlowView provides five columns for flow chart objects.
- Row labels: Provide a guide for locating shapes in the drawing area. FlowView provides an unlimited number of rows as they are needed... limited only by available memory.
- Drawing area: Build flow charts in the drawing area. Horizontal and vertical grid lines appear by default. Blocks must appear within a grid cell with only one block per cell.

- scroll bars: Use the scroll bars to bring portions of the drawing area that are off-screen into view. To scroll in small increments, click the arrows at the end of each scroll bar.
- Status line: Provides information about FlowView or selected shapes, etc.

Use multiple flow charts for all but the simplest projects with each flow chart controlling a single process, machine, or major function. A modular approach to developing flow chart programs makes it easier to create, debug, and maintain control projects.

All enabled flow charts execute in the order they appear in the “Standard 123...” list. The project can programmatically control which flow charts execute during any given scan cycle. For more information on enabling flow charts, see “Enable Block” on page 5-4.

To create a flow chart, select the “File... New... Standard Chart” menu. To open a flow chart, do one of the following:

- Select the “File... Open Flow Chart...” menu, and then select a flow chart from the cascaded menu list.
- To display a list of current flow charts in the project, use the Flow Chart Explorer in ProjectCenter. Open a flow chart from the list by double-clicking it.

4.5.1 Adding Flow Chart Blocks to a Flow Chart

To add a flow chart block to the current flow chart, follow these steps:

1. Click the icon in the toolbar for the flow chart block that you want to add.
2. Click the cell where you want to place the flow chart block.



Each cell of the flow chart can have only one flow chart block.

4.5.2 Turning the Grid On and Off

To turn the page grid on or off, follow these steps:

1. Select the “File... Preferences” menu to view the “Preferences” dialog box.
2. In the “Preferences... General” tab, clear the “Show Grid Lines” check box to turn off the grid lines. Select the check box to turn the grid lines back on.

The full design of a flow chart may be large, and any page may be larger than the viewable window. You can see more of the page by using the zoom factor setting.

4.5.3 Increasing the Number of Columns

To increase the number of columns, do one of the following sequence of steps:

For the Current Flow Chart

1. Select the “File... Chart Properties” menu to display the “Chart Properties” dialog box.
2. Type a number or use the up/down buttons in the “Columns” field to change the number of columns.
3. Click the “OK” button to close the dialog box.

For All Flow Charts

1. Select the “File... Preferences” menu to display the “Preferences” dialog box.
2. In the “Preferences... General” tab, type a number or use the up/down buttons in the “Columns” field to change the number of columns.
3. In the “Save Preferences” dialog box that appears, choose the option desired. You can choose to:
 - Apply the Changes to this project only
 - Apply the Changes to this project only, and all future projects created from this point forward
 - Leave the preferences set the way they are
 - Reset the project to the default settings for new projects
 - Abandon changes. Reset this project and all future projects to the factory default settings.
4. Click the “OK” button to close the dialog box.

4.5.4 Changing Zoom Level

To change the page magnification:

- Click the “Zoom” drop-down list on the toolbar (Figure 4-8) or click a magnification tool in the menu bar (+) or (–) to increment the zoom to the next respective setting.

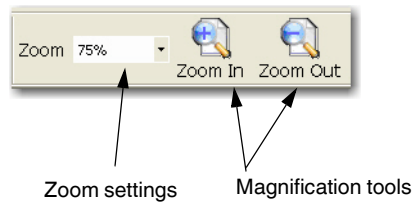


Figure 4-8 Zoom tools in FlowView

- Another method is to select the “View...Zoom Factor...” and select the desired setting.

4.6 Displaying Multiple Flow Charts

You can open multiple flow charts in FlowView. This is very useful for copying flow chart logic from one to another. As shown in Figure 4-9, each flow chart occupies its own window within FlowView.

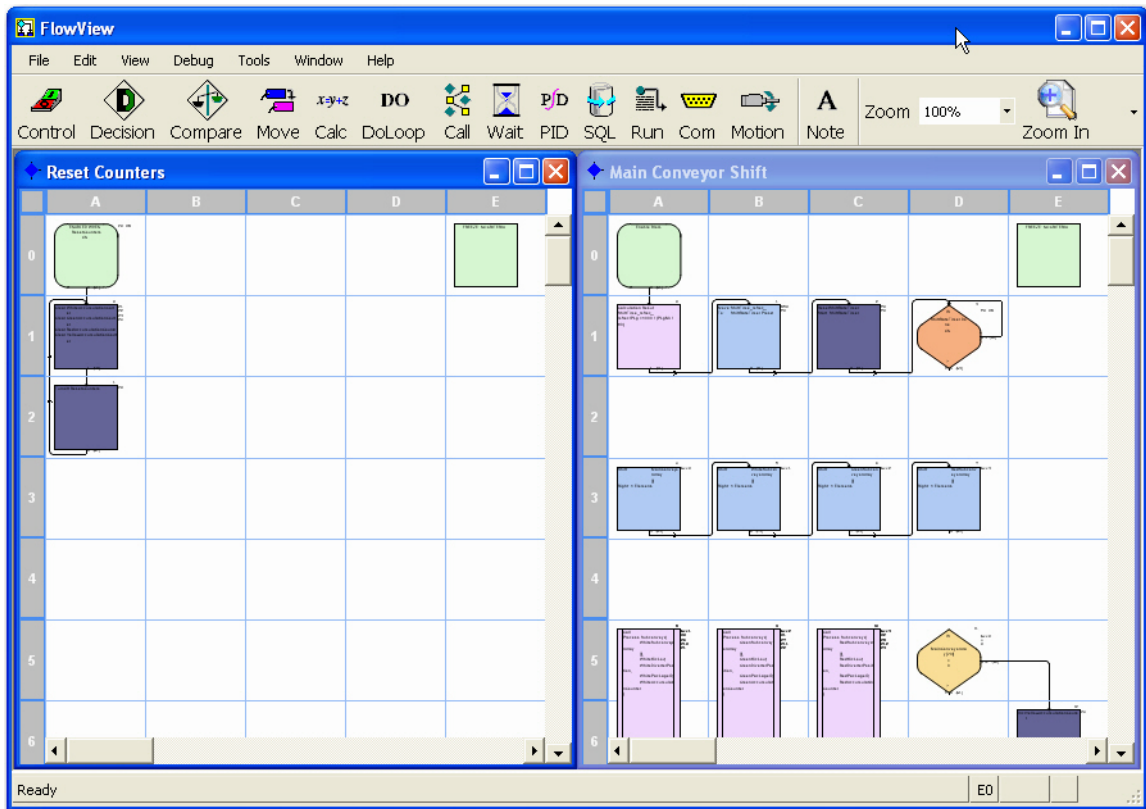


Figure 4-9 Open multiple flow charts in FlowView, each with its own window

The window arrangement in Figure 4-9 is called *tiled*. A *cascaded* arrangement places one behind the other, slightly offset. You can also maximize, minimize, or close any flow chart window by using the “Minimize”, “Maximize”, and “Close” buttons in the upper right corner.

To choose tiled or cascaded windows, click the “Window...Cascade” or “Window...Tile Vertically” or “Window... Tile Horizontally” menu from the FlowView main menu.

4.7 Editing Flow Charts

FlowView provides intuitive editing capability to move or cut-and-paste flow chart blocks from one grid location to another. Many functions are available with a mouse right-click, or use toolbar buttons as described:

To select flow chart elements for editing:

1. Click a block or connection to select it.



By default, FlowView is always in selection mode, except after clicking a block's exit connector anchor that doesn't have a flow line connecting to another block (see "Connecting Flow Chart Blocks" on page 4-15).

2. To select multiple blocks or connections do one of the following:
 - Select a rectangular area by pressing (hold down) the left mouse button, dragging the cursor diagonally to size a rectangular area, and then releasing the mouse button.
 - Hold the <Ctrl> key while clicking blocks or connections. Clicking a selected block or connection with the <Ctrl> key pressed deselects that object without deselecting any others.
3. Perform the action (such as cut, copy, move, or delete).

4.7.1 Cut-Copy-Paste

After selecting the object(s), use the cut (<Ctrl>+<x>), copy (<Ctrl>+<c>), and paste (<Ctrl>+<v>) shortcuts to speed up flow chart development, or use the menu selections from the "Edit" menu. With multiple flow charts open, it is possible to cut/copy and paste from one flow chart to another.

4.8 Flow Chart Block Introduction

This section provides a brief introduction to flow charts. For a more complete discussion, see Section 5, "Flow Charts".

To place a new flow chart block on the page, follow these steps:

1. In the toolbar, click the icon of the block type to place.
2. Click in the desired location on the drawing page.

4.8.1 Enable Block

The Enable block always appears at the top of a flow chart. It allows the flow chart to begin executing. It can contain an expression that conditionally activates the flow chart while the project is running. To develop the control program, add flow chart blocks below the Enable block.

4.8.2 Freeze Block

The Freeze block (upper right corner of flow chart page) does not connect to other flow chart blocks. When configured, its expression can conditionally freeze or resume a flow chart execution.

4.8.3 Types of Action Blocks

FlowView has the following types of action blocks (in the order they appear in the toolbar):

- Control: Control blocks initiate an action such as turning on an output or resetting a timer. One Control block can have up to ten control expressions. Control expressions set values for flags, turn outputs on and off, and control timers and counters.
- Move: Move blocks simply move data from one data item to another. The source data item remains unchanged
- Calc: Calculation blocks provide comprehensive math capabilities. They operate on a variety of data types with either simple arithmetic or transcendental functions such as sine, log, etc.
- Do Loop: Do Loops are a construct that allow blocks to execute more than once during a scan.
- Call: Call blocks call a subchart. Call Blocks transfer execution to the called subchart with specified parameters.
- Wait: Wait blocks insert a wait period in the execution of the flow chart. A Wait block can have a wait period set in milliseconds or hours, minutes, and seconds. If the wait duration is zero, the block defaults to one scan cycle.
- PID: PID blocks perform a mathematical algorithm for process control, which is the Proportional Integral Derivative (PID) algorithm. Think & Do permits up to 64 PID loops in a project.
- SQL: SQL blocks provide a tabbed dialog box that helps you construct SQL queries for database accesses.
- Run: Run Program block lets you start other Windows programs or batch files under the flow chart control.
- Com: Communication blocks permit a flow chart to communicate directly with an external device through a serial port on the PC.
- Motion: Motion blocks provide integrated motion control. Integrated motion control provides convenience and time savings during system design, since you avoid having to link external motion programs with the project.

4.8.4 Types of Branching Blocks

Branching blocks (both Decision and Compare blocks) perform a test and determine which of two paths to take after the test. Each branching block has one entry point and two exit points indicating Yes and No paths from the block. By default, the bottom connector is the Yes path, and the right side is the No path.

A Do Loop is a branching construct that includes a Compare block at the beginning and end of the loop. You place any number of action blocks between the beginning and ending Do Loop blocks. A Do Loop only has one exit. Branches to blocks outside the Do Loop aren't permitted.



The standard orientation of flow chart is top-to-bottom for ease in reading.

Both Decision and Compare blocks appear as diamonds in the flow chart, while the Do Loop appears as a pair of diamonds (see Figure 4-10).

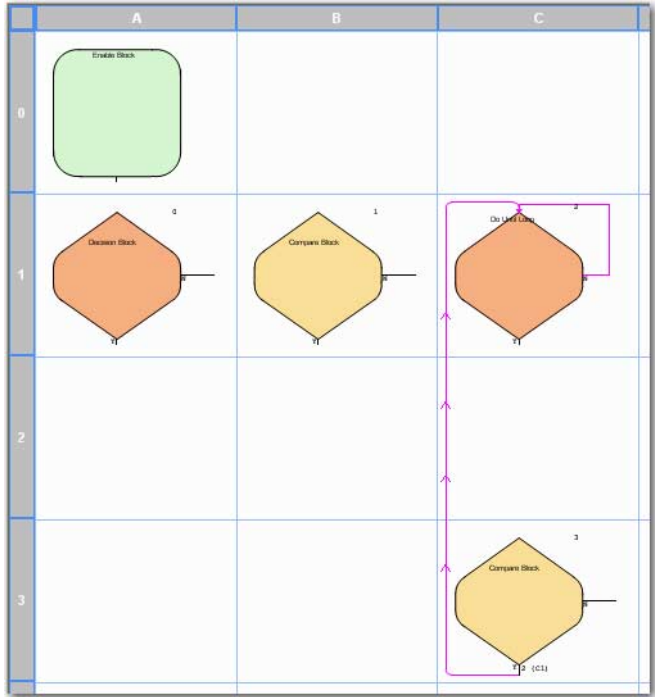


Figure 4-10 Decision, Compare, and Do Loop blocks appear as diamonds

Decision blocks examine the state of a data item. For example, a Decision block can test inputs or outputs to determine if they are ON or OFF or transitioning from one state to the other. A Decision block can have up to seven decision expressions. If there are multiple expressions, an OR or AND condition links them. For example, a Decision block could have the following expressions:

- Input1 ON AND Input2 OFF

Compare blocks compare two data items. For example, a Compare block can compare an integer register value to a constant or another register. A Compare block can compare values using one of the following comparison operators:

- = (Equal To)
- > (Greater Than)
- < (Less Than)
- <> (Not Equal To)
- >= (Greater Than or Equal To)
- <= (Less Than or Equal To)

The Do Loop blocks are similar to Compare blocks. The first one, at the top of the Do Loop provides a test for a maximum loop count. This is primarily to prevent infinite loops and should not be used as a normal exit. The bottom Compare block is a standard Compare block. It is the normal (and only valid) exit point from the loop.



To change the bottom Compare block to a Decision block, right-click on the block, and then select "Change this Compare block to a Decision block" from the pop-up menu.

4.8.5 Notes

Notes don't actually appear in a block. They can appear anywhere on the flow chart page, and don't connect to or affect program flow.

4.9 Connecting Flow Chart Blocks

To draw flow chart block connecting lines, follow these steps:

1. Move the cursor to the bottom (or side for Decision and Compare blocks) connecting point of a block that doesn't have an out-going flow line. The connecting point turns orange (or the specified color) to indicate a potential out-going connector pointer, and the cursor changes to the connector pointer.



If the flow chart editor is inadvertently in connect mode, press the <Esc> key to switch back to select mode.

2. Click the out-going connector point.
3. Move the cursor to the next block's top or left (in-coming) connector point, which will turn green (or the specified color).
4. Click the in-coming connector to complete the line. The cursor changes back to the selector pointer.



FlowView supports automatic routing from one block to another, so it is only necessary to click the out-going connector of one block and the incoming connector of the next to generate the flow line. It is always possible to manually draw the flow line by clicking from point-to-point to route the line.

For example, Figure 4-11 shows a simple project flow chart with all required connecting lines.

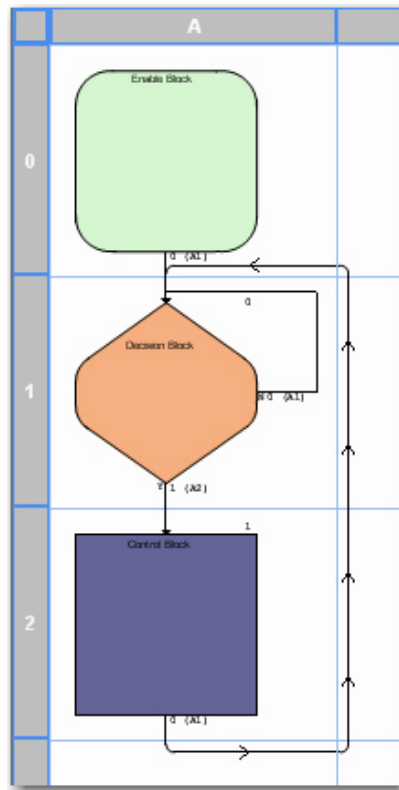


Figure 4-11 Sample flow chart with flow lines connecting all blocks



Unlike CAD programs that permit ending a line where it intersects another line, FlowView requires guiding the pending connection to the common destination.



If the connector appears as a red line, it is not properly connected. This can occur if the connection doesn't meet the input connection point of the block. Try moving the end point of the line until region that represents the block's input connection becomes highlighted.

4.10 Editing Flow Chart Block Expressions

The content of a flow chart block is an expression. To enter and edit block expressions, double-click the block to display the "Configuration" tab of its properties dialog box. For a discussion of each block's "Configuration" tab, see Section 5, "Flow Charts".

4.11 Entering Block Description

Enter a description of a block to summarize actions taken in the block. This is useful to provide an overview version of the flow chart. Specify whether flow charts should display block descriptions or block expressions by toggling the “View... Block Descriptions” menu.

To enter or edit block descriptions:

1. Right-click on the block and select “Block Properties” from the pop-up menu. The “Properties” tab of the dialog box (see Figure 4-12) appears.

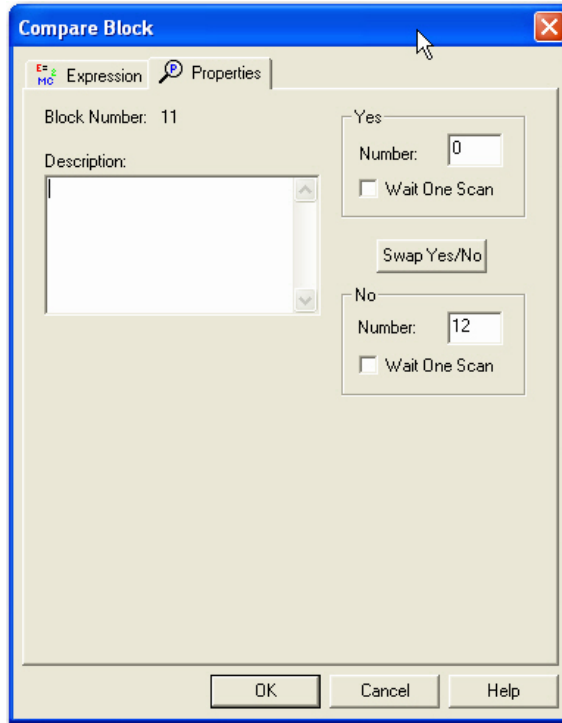


Figure 4-12 “Compare Block... Properties” tab

2. Enter the block “Description”, and then click the “OK” button.



The “Address” fields for “Yes” and “No” in the “Compare” and “Decision” block “Properties” tabs permit entry of a block number without drawing a connection.

3. To view all block descriptions on the flow chart instead of expressions, select the “View... Block Descriptions” menu.

4.12 Operator Screens

Think & Do provides more than just PC-based control with flow charts – it includes a full-featured operator interface or HMI (Human-Machine Interface). Use ScreenView to create operator screens. The remainder of this chapter focuses on the unique features that you need to know to develop HMI screens.



Windows CE target Runtime systems – some devices, such as the WinPLC, do not have direct screen output available to its operating system.

The final output of the development work is a screen or series of screens. These appear in a window on the PC at Runtime. The purpose of a screen is to communicate important information to a machine operator or process engineer. It also provides an opportunity for operator-controlled input. Screens typically include:

- The current state of the machine or process, such as Idle, Run, or Stop.
- Alarm conditions, if they exist
- Production data (such as number of parts, percentage defects, downtime reasons, and trends)
- Diagram of process (on-screen diagram to help orient the operator)

ScreenView provides a full-featured HMI or operator interface design package, which helps to create screens that really communicate. Operator input support includes function keys and data entry.

4.12.1 Screens as Part of a Project

The screen(s) developed using ScreenView become part of a project. Then at Runtime, the screens appear in an independent window on the desktop. Only one screen is visible in the window at a time. You can re-size or minimize screens just as any window. You can also run in full-screen mode, where screen size adjustment isn't available to the operator. Data entry or function key entry requires that the screen be in the active window.



When a project runs, its flow charts—not operator screens—are the highest priority task. During the “Pause” phase of each scan period (see “Building and Running Projects” on page 9-3), the system executes screen tasks and other applications. Adding screens affects system loading, so be sure that the scan time setting is large enough to allow the system to service your operator screen(s).

4.12.2 Creating Operator Screens

To create a new screen in the project, follow these steps:

1. In ProjectCenter, select the “File...New...Screen” menu to display ScreenView and the “Screen Attributes” dialog box (see Figure 4-13).

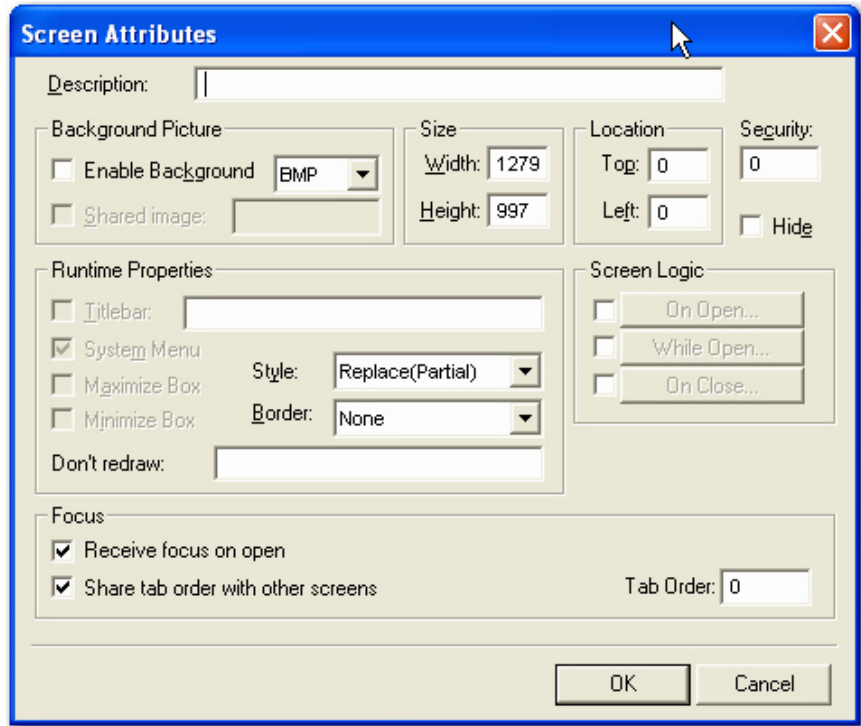


Figure 4-13 Use the “Screen Attributes” dialog box to set up a new screen

2. In the “Screen Attributes” dialog box, type a description of the screen in the “Description” field.
3. Set the “Size” and “Location” for the Runtime screen.
4. Click the “OK” button.

For a discussion of additional options in this dialog box, see “Setting Screen Attributes” on page 6-112.

4.12.3 ScreenView Development Environment

ScreenView uses standard tools and interfaces to make the product user-friendly. ScreenView provides an integrated and unique development environment that gives you fast and easy access to tools and information.

The ScreenView development environment (see Figure 4-14) consists of the following basic areas:

- Title Bar: Indicates the active display or worksheet

- Menu Bar: Contains the main product options and controls, which you can easily access using the cursor or your keyboard
- Toolbars: Provide access to features and tools used in the development environment
- Workspace: Provides tree-view control from which you can access project worksheets and screens
- Screens / Worksheets: Provides an area where you can edit displays and worksheets

The following figure shows the development environment areas and windows in their default position. You can customize this environment by changing the position of the different areas.

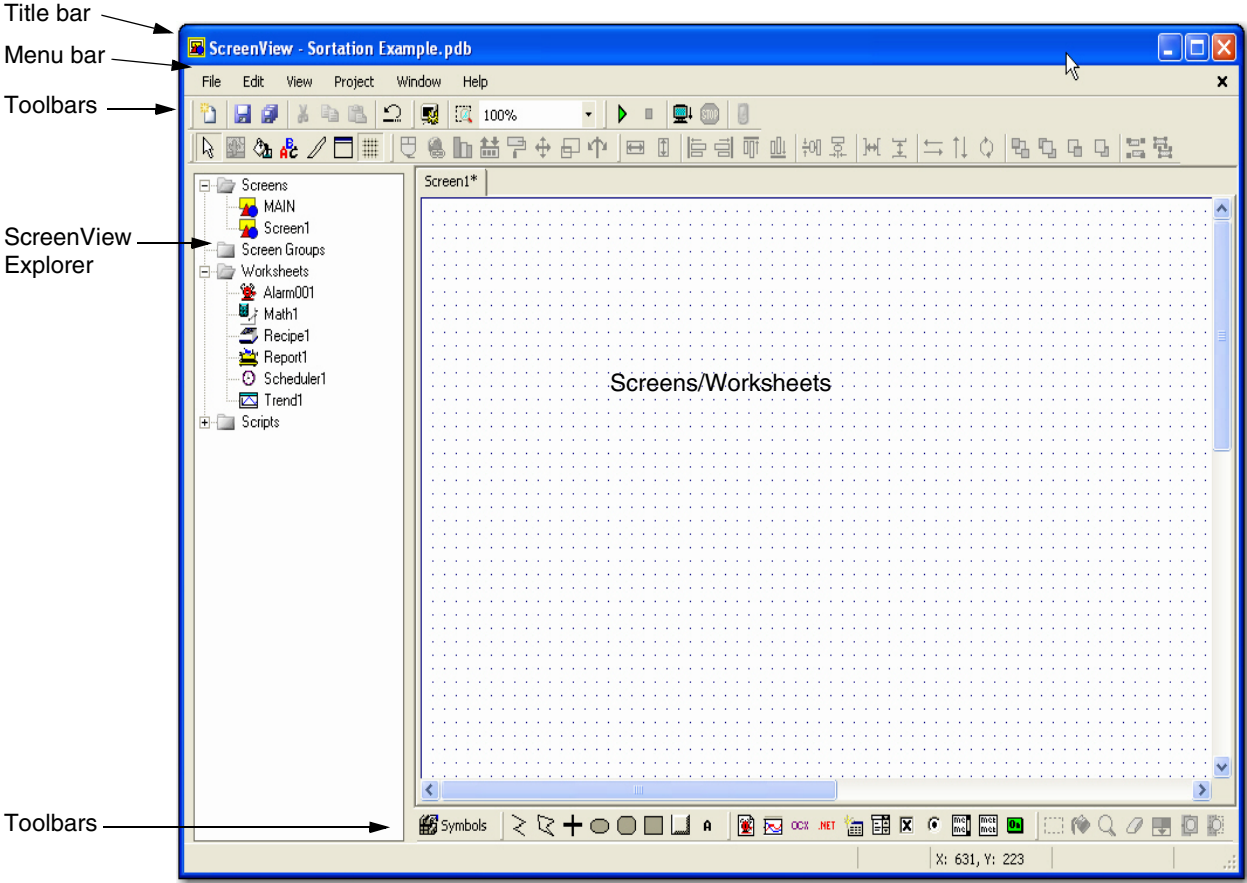


Figure 4-14 ScreenView development environment

ScreenView Explorer

The ScreenView Explorer is a user-friendly interface that lets you quickly find application components (screens or worksheets). The ScreenView Explorer uses a tree-view with two main folders—one for screens and the other for worksheets.

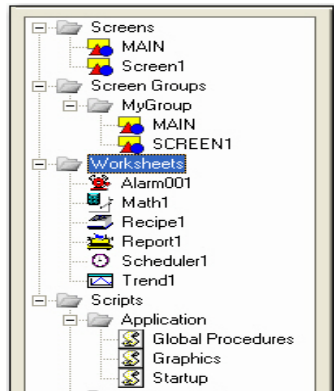


Figure 4-15 ScreenView Explorer

Whenever you create a screen or worksheet, ScreenView adds the new object to the tree. Double-clicking an entry in the ScreenView Explorer displays that object in the ScreenView workspace. Right-clicking an object in the tree (not a folder) displays a pop-up menu with options to “Rename”, “Duplicate”, or “Delete” the selected object.

The “Screens” folder contains screens with finished graphic compilations and working drafts. To view a screen, expand the “Screens” folder and double-click the desired screen to display it in the Workspace.

The “Worksheets” folder contains other worksheet objects that you can create in ScreenView. Among them are alarm (see “Alarm Worksheets” on page 6-115), math (see “Math Worksheets” on page 6-121), and trend (see “Trend Worksheets” on page 6-127) worksheets.

4.12.4 Object Properties Dialog Box

The “Object Properties” dialog box is the interface that allows you to configure the settings for the objects and dynamics designed with ScreenView. It can be launched by double-clicking on the object or by selecting the object (clicking once on it) and pressing <Shift>+<F2>.

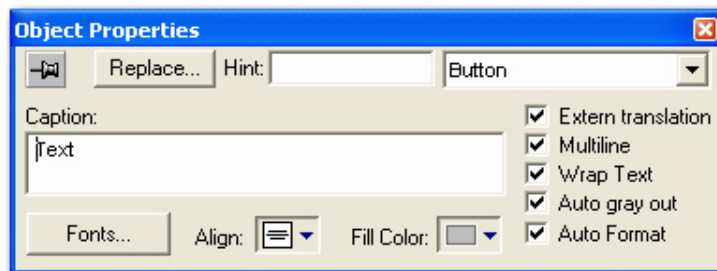



Figure 4-16 The “Object Properties” dialog box

The content of this dialog depends on the object (or group of objects) selected by the user. However, the following interfaces are common for any object (or group of objects):

Table 4-1 “Object Properties” dialog box properties

Field	Remarks	Syntax
(Pin button)	When the pin button is released, focus is passed to the object on the screen as soon as that object is selected. When the pin button is pressed focus remains on the Object Properties window, even when you click the objects on the screen.	Button
“Replace”	Launches the Replace Dialog, where you can replace strings, tags or properties for the selected object (or group of objects).	Button
“Hint”	Tooltip displayed during runtime, when the mouse cursor remains over the object. This option is useful to provide hints to the operator during runtime. The text on the “Hint” field is also written to the Internal Tag Hint during runtime. Therefore, you can trigger actions based on the value of this tag.	Text and/or {Tag} (up to 256 chars)
	The combo-box at the right side of the dialog box lets the user select a specific object of objects that must be edited.	Combo-box



The “Enable Tooltip” option must be enabled (checked) on the “Project Settings” dialog box. Otherwise, the tooltips are NOT displayed during the runtime when the user points the mouse cursor on the objects. You can enable/disable this feature for the local server (select the “Project... Settings” menu).

Clicking the “Replace” button from the “Object Properties” dialog box displays the “Replace” dialog box (see Figure 4-17).

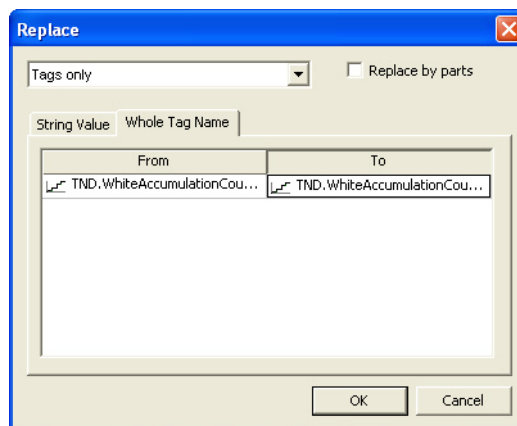



Figure 4-17 The “Replace” dialog box

This dialog provides a fast and simple tool to replace strings (“String Value” tab), tags and/or properties for an object (or group of objects). This dialog display grids where you can edit the “To” column with the string, tag and/or property that must be replaced. The main interfaces in this dialog as described in the following table:

Table 4-2 Object Properties “Replace” dialog box parameters

Field	Remarks	Syntax
	Allows you to filter the type of information that can be replaced: <ul style="list-style-type: none"> – “Tags only”: Displays only the tags configured directly (e.g.: Second, Time, Level, and so forth). – “Tags + Properties”: Displays both the tags configured directly and the properties configured as mnemonics (e.g.: #PumpStatus). – “Custom Properties”: Displays only the mnemonics configured as custom properties (e.g.: #PumpStatus:). 	Combo-box
“Replace by Parts”	When checked allows the user to replace each part of a tag separately: Main Tagname, Array Index, Class Member and Tag Field. It is very useful when you want to replace a specific settings (e.g.: Array Index) for different tags configured in the same Group of Objects.	Button



When pressing the “Replace” button after selecting the “Group of Objects” option in the “Object Properties” dialog box, the replace feature affects all objects and dynamics which are part of the selected Group of Objects. However, when pressing Replace button after selecting a specific object or dynamic (e.g.: Button, Color, etc), the replace feature affects ONLY the selected object or dynamic.

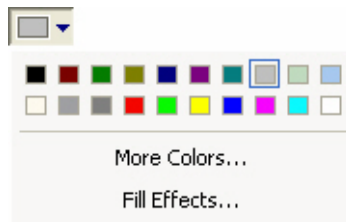
Color Interface

You can edit the color of a component with the Color interface.

1. Click the “Color” icon in the toolbar.
2. Do one of the following:
 - Click the desired color from the twenty that display when the pop-up box opens:



Color Icon



The selected color will be applied to the component that you are editing.

- Click the “More Colors...” link if you want to apply a different color. This displays the “Colors” dialog box, which shows the 143 standard colors from your operating system. If the color you want is available here, you can select a color, and then click the “OK” button.

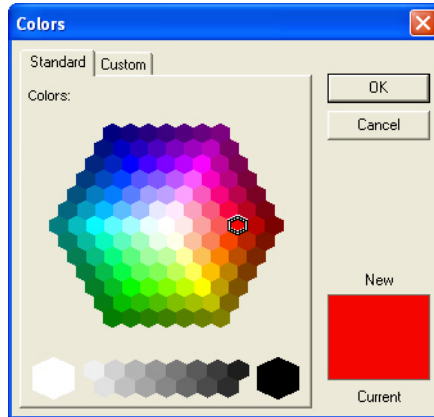


Figure 4-18 The “Colors” dialog box

- To use a color that isn't available on the “Colors... Standard” tab, click the Custom tab. You can then edit the “Hue”, “Sat”, “Lum” or “Red”, “Green”, “Blue” codes of any of the 143 standard colors, creating a custom color.

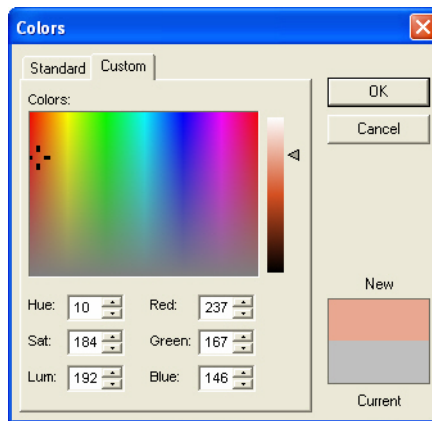


Figure 4-19 The “Colors... Custom” tab

3. Click the “OK” button to apply the selected color to the component that is being edited.

- Depending on the component that you are editing, the “Fill Effects” option may be available from the pop-up interface (see step 2 above). Clicking this link displays the “Fill Effects” dialog box, which lets you apply gradient colors with different styles and variants.

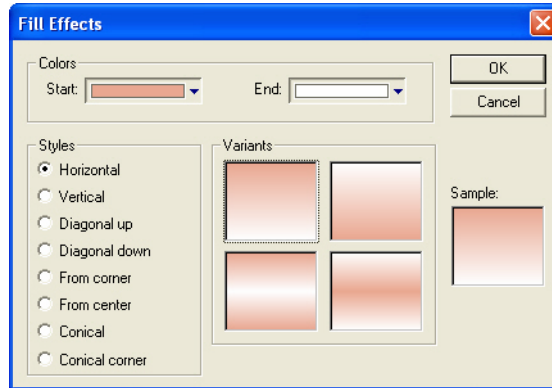


Figure 4-20 The “Fill Effects” dialog box

- Select two colors in the “Start” and “End” fields, select an option from the “Styles” group, and then click the “OK” button to apply the fill effect to the component.



For applications that run on the Windows CE operating system, the Fill Effects interface is available only for the Rectangle object.



Although “Fill Effects” is a useful tool for enhancing the look and feel of your screens, the operating system takes a longer time to fill an object with fill effects than with plain colors. You should develop criteria for using the feature without decreasing the performance of the system, especially under the Windows CE operating system.



Color Icon



The number of colors available when developing the application depends on the color settings configured on the operating system of the development station. The number of colors available when running the application (Runtime) depends on the color settings configured on the operating system of the Runtime station.



Pin Button (Released position)

Configuring the Focus of the Object Properties Dialog Box

Double-clicking any object (or group of objects) in ScreenView displays the “Object Properties” dialog box, which lets you configure the selected object’s settings. The content of this dialog box varies according to the specific object/dynamic selected. However, there is always a pin button in the left upper corner of this dialog box.

When the pin button is released, the focus is passed to the object on the screen as soon as that object is selected. Therefore, we recommend you keep this button released when you want to manipulate (copy, paste, cut or delete) the objects. Although the “Object Properties” dialog box is on the top, the keyboard commands (<Ctrl>+<C>, <Ctrl>+<V>, <Ctrl>+<X> or) are sent directly to the objects.



Pin Button
(Pressed
position)

When the pin button is pressed focus remains on the “Object Properties” dialog box, even when you click objects on the screen. We recommend you keep this button pressed when you want to modify the settings of objects. You can click an object and type the new property value directly in the “Object Properties” dialog box (it is not necessary to click on the window to bring focus to it). Also, when the pin button is pressed, the “Object Properties” dialog box does not automatically close when you click on the screen.

4.12.5 Creating and Using Tags in ScreenView

In ScreenView, you use tags to display state, results of calculations, alarm points, and so forth. In ScreenView, all tags are organized according to their origin— application, internal, or shared. Application and internal tags originate in ScreenView, while shared tags originate in the Think & Do Data Items editor.

The following is a description of the different ScreenView tag types:

- Application tags are user-defined, screen tags created for displays, to read from and write to field equipment, for control, auxiliary tags to perform mathematical calculations, and so forth.
- Internal tags are system tags predefined by ScreenView. Internal tags have predetermined functions (time, date, acknowledge alarms, storage of the logged-on user name and so forth). You cannot delete or modify these tags, but you can access their values from any ScreenView task.
- Shared tags are created in the Data Items editor and imported into the ScreenView environment. You cannot edit shared tags in the ScreenView environment, but you can modify these tags in ProjectCenter. Consequently, you can configure shared tags for any ScreenView task just as any other tag.

Understanding the Screen Tag Syntax

Observe the following guidelines when naming a screen tag:

- You can use letters, numbers, and the character “_” (underscore)
- Do not use the following characters:
` ~ ! @ # \$ % ^ & * () - = \ + \ [] { } < > ?
These characters are parsed by the ScreenView scripting language as logic and arithmetic operators.
- The tag must begin with a letter
- Maximum tag length is 255 characters
- Maximum class member length is 255 characters
- Tag names must be unique. You cannot specify the same name for two tags.
- Tags are not case sensitive (though we recommended using uppercase and lowercase characters to make names more readable. For example, use “TankLevel” instead of “tanklevel”.
- Tag names must be different from internal tag names and math functions.



Use the @ character at the beginning of a tag name to indicate that the tag will be used as an indirect tag in the application.

For additional information, see the ScreenView scripting language in the online help.

Some valid tag examples include:

- Temperature
- pressure1
- count
- x

Using Screen Tag Values

A screen tag value can be one of the following types:

- Boolean is a Boolean or digital variable (0 or 1).
- Integer is a number (positive, negative, or zero). Equivalent to C-type long integer. Examples: 0, 5, -200.
- Real is a number internally stored as a double word. Equivalent to C-type double.
- String (ASCII text) is a character string up to 1024 characters that holds letters, numbers, or special characters. Examples: Recipe product X123, 01/01/90, *** On ***.

4.13 Scripting Languages in ScreenView

ScreenView supports a built-in scripting language and Microsoft Visual Basic Script language (VBScript®). These languages have familiar syntax, operators, and functions. They provide:

- The ability to create new variables and procedures (functions and/or sub-routines)
- Access to properties, methods and/or events from COM objects, including ActiveX controls
- The ability to execute the logic on any platform

For details of both the built-in scripting language and VBScript, see the Think & Do online help.

Section 5

This section informs you about

- Flow charts blocks and how to use them
- Creating and using subcharts

Flow Charts	5-3
5.1 Flow Chart Types and Categories	5-3
5.2 Basic Flow Chart Blocks	5-3
5.2.1 Enable Block.....	5-4
5.2.2 Calculation Block	5-5
5.2.3 Call Block.....	5-7
5.2.4 Note	5-8
5.2.5 Communication Block.....	5-9
5.2.6 Compare Block	5-10
5.2.7 Control Block	5-13
5.2.8 Decision Block	5-15
5.2.9 Do Loop Block	5-17
5.2.10 Motion Block	5-19
5.2.11 Move Block	5-21
5.2.12 PID Block.....	5-23
5.2.13 Run Program Block.....	5-24
5.2.14 SQL Block.....	5-25
5.2.15 Wait Block.....	5-26
5.3 Using Subcharts	5-27
5.3.1 Blocks for Subcharts	5-27
5.3.2 Begin Expression.....	5-28
5.3.3 Call Expression.....	5-29
5.3.4 Interaction of Call and Begin Blocks	5-30
5.3.5 Subchart Design Summary	5-31
5.3.6 Subchart Runtime Characteristics	5-31
5.3.7 When to Use Subcharts	5-32

Think & Do

5 Flow Charts

This section provides an in-depth discussion of the flow charting features of Think & Do.

5.1 Flow Chart Types and Categories

Think & Do has a two-level flow chart hierarchy implemented with two types of flow charts:

- Standard flow charts
- Subcharts

You use standard flow charts to define a project. If your project has only one flow chart, it must be a standard flow chart. Standard flow charts can call subcharts (much like a main program calls subroutines).

Standard and subchart type flow charts can pass variables back and forth. Note that in this manual, we use the term “flow charts” to mean either the standard or subchart type.

5.2 Basic Flow Chart Blocks

The first block in a flow chart is the starting point of execution at runtime. By default, the proper flow chart starting block is at the top of the drawing page when you create a new flow chart:

- Standard flow chart: Enable Block is first
- Subchart: Begin block is first



You can determine the flow chart type at a glance by looking at the first block. Each flow chart can have just one of these two blocks as a starting point. FlowView automatically creates these starting blocks when you create a new flow chart or new subchart.

5.2.1 Enable Block

Each flow chart must have a beginning point, and the Enable block serves that purpose for standard flow charts. The Enable block is in the upper left grid cell in a new flow chart (see Figure 5-1). It has the following characteristics:

- The Enable block is the first block of every standard flow chart. It has a single exit from which flow passes on to the rest of the control program. The Enable block cannot be deleted, copied, or moved on the drawing area.

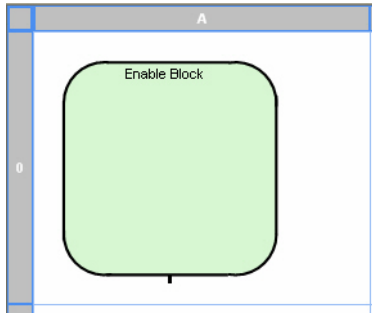


Figure 5-1 Every new flow chart opens with a predefined Enable block

- An empty Enable block (no expression created) causes a flow chart to always run.
- Double-click the block to edit its expression.
- Add statements to the expression using logical operators (see Figure 5-2). The result of the expression is either TRUE or FALSE, enabling or disabling the flow chart respectively.

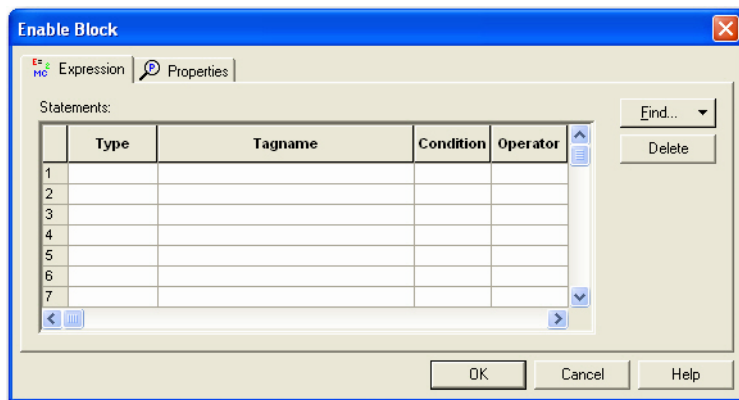


Figure 5-2 The “Enable Block... Expression” tab

- The AND and OR “Operator” in the dialog box provides for additional Boolean conditions in the expression. AND has greater precedence than OR (in other words AND expressions are evaluated before OR expressions, and then they are evaluated from top-down). Think & Do Runtime evaluates the Enable block every scan before executing the flow chart (based on the position of the flow chart in the flow chart list—see “Changing Execution Order” on page 4-6). The flow chart only runs if its expression is TRUE at the start of the scan.

- On the first scan or whenever the Enable block expression transitions from FALSE to TRUE, flow chart execution starts at the Enable block exit. Otherwise, execution starts where it left off the previous scan.

Table 5-1 defines all possible Enable block combinations. This table shows valid data item types and the conditions that appear in the “Condition” drop-down list. For example, it is possible to test to see if an Input is “ON” or “OFF”.

Table 5-1 Valid Enable Block Expressions¹

Data Item Type	ON	OFF
Array	X	X
Comm	X	X
Flag	X	X
Input	X	X
Output	X	X
Timer	X	X

¹ X=valid

5.2.2 Calculation Block

The Calculation block provides comprehensive math capabilities. It operates on a variety of data types with either simple arithmetic or transcendental functions, such as sine, log, etc. By editing the “Calculation Block” dialog box (see Figure 2-2), you select the math operations the block performs, as well as the data types of the two operands and the result.

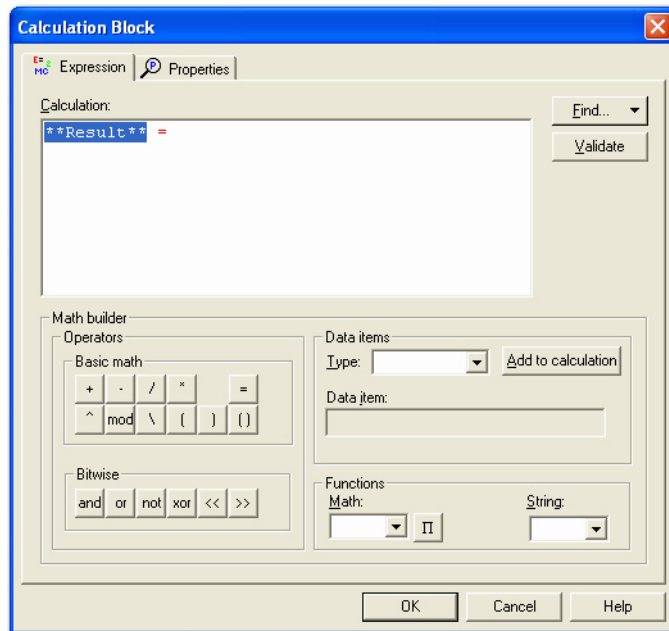
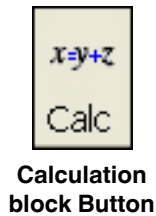


Figure 5-3 The “Calculation Block... Expression” tab

The “Calculation Block... Expression” tab features a “Calculation” field. The “Calculation” field lets you enter an equation producing one result, which can be a function of several variables. The result and variables, which comprise the expression, must be defined in the tag-name database. You may define the tagnames in advance, or just click “Find” drop-down button to go to the Data Item Grid to create and/or select the variable. You can also use the “Data Items” group to enter or select data items. Follow these steps:

1. Select the data item “Type” from the drop-down list.
2. Do one of the following:
 - Enter the data item name in the “Data Item” field. While typing, Think & Do attempts to auto-complete the name.
 - Enter the data item “Index”, for example “01”. When tabbing to the next field, Think & Do displays the data item name associated with the index.
 - Use the “Find...” menu button or double-click the field to display the Data Item Grid.
3. Click the “Add to Calculation” button. If the tagname does not exist, Think & Do prompts you to create the data item.

A single Calculation block produces one result. Connect several Calculation blocks together if you need to produce intermediate results.

The Calculation block includes a Validate function, which checks the syntax of the expression when you click the “OK” button. If the expression is valid, the “Calculation Block” dialog box closes. Otherwise, an error message pops-up. For more information on number ranges and types for math operations, see Section 7, “Math and Data Operations”.

5.2.3 Call Block



Call block
Button

You use the Call block to call a subchart. Just as a program can call a subroutine, the Call block transfers execution to the called subchart. Double-click the Call block to access the “Call Block... Expression” tab shown in Figure 2-3.

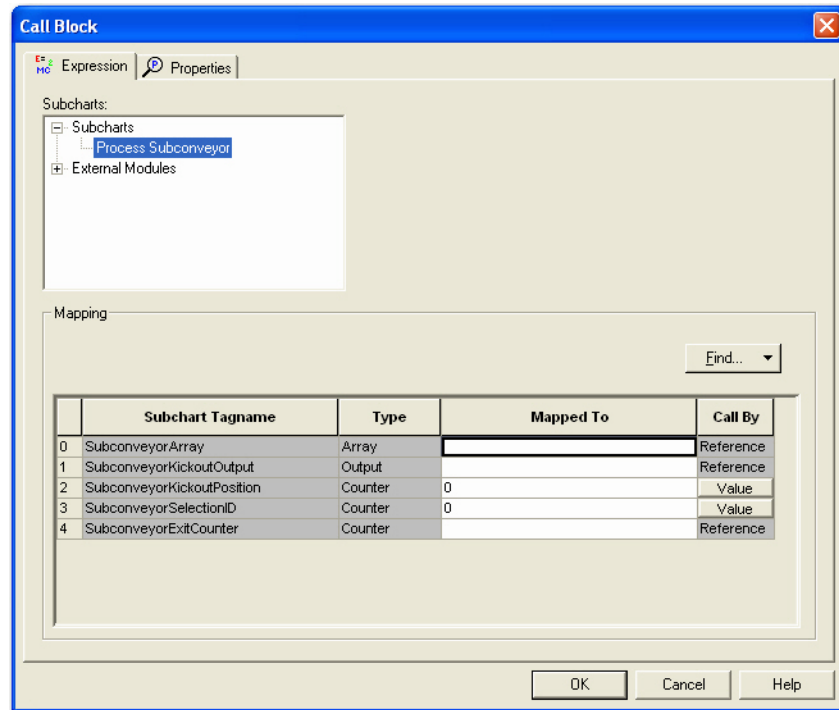


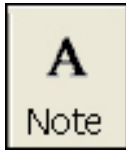
Figure 5-4 The “Call Block... Expression” tab

To use the Call block, select the name of the subchart from the “Subcharts” tree. After selecting a subchart, parameters defined in the subchart appear in the “Mapping” group. This lets you match subchart parameters with those in the calling flow chart. See “Using Subcharts” on page 5-27.



A subchart has to exist before a flow chart can call it. This may sound obvious, but you cannot create the name of a subchart you want to call in the future in the calling flow chart. You must first create a new subchart and save it using the desired name before it appears in the list box in the “Call Block... Expression” tab.

5.2.4 Note



Note Button

Notes are re-sizable text that can appear anywhere on the drawing page. Typically, notes should appear near relevant flow chart blocks. Once created, drag-and-drop or cut-and-paste operations are possible on notes.

To enter a note:

1. Click the “Note” button in the toolbar, and then click the position in the flow chart where the note should appear. This displays the “Note Properties” dialog box (see Figure 5-5).

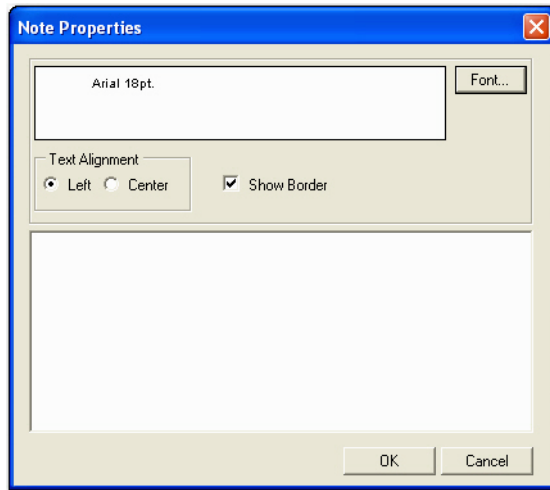


Figure 5-5 “Note Properties” dialog box

2. To change the font, font size, or style, click the “Font...” button to display a standard font selection dialog box.
3. Set the text alignment by selecting the appropriate radio button.
4. Optionally, select the “Show Border” check box to turn on a border for the note.
5. Enter the note text in the text entry field.
6. Click the “OK” button.

To edit a note:

1. Double-click the note to display the “Note Properties” dialog box.
2. Modify the note and/or note properties as desired.
3. Click the “OK” button.

To delete a note, select it and press the <Delete> key.



Communication
block Button

5.2.5 Communication Block

The Communication block allows a flow chart to communicate directly through a serial port on the Runtime with an external device. Each instance of a block handles either a read, write, or control operation. The example shown in Figure 2-5 reads from the serial port into a string named “Drill_Start”. The tagname “COM_1_CNC_Link” is a Comm data item that is mapped to a particular serial port using IOView. The device sends a single byte of data.

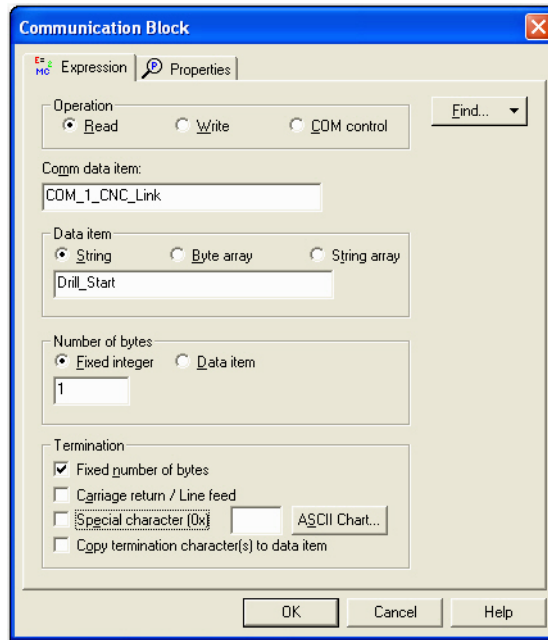


Figure 5-6 The “Communication Block... Expression” tab

Double-click the Communication block to edit the Communication expression. This displays the “Communication Block... Expression” tab. “Read”, “Write”, and “COM control” operations are independently selectable. In the Communications Read block, you must select at least one of the check box options in the “Termination” group for a valid operation. You can also access an ASCII character chart of hex codes, using the “ASCII Chart” button, to select a special character as a terminator.

When you select a “Write” operation, you can optionally choose terminators that you want appended to the data output. Selecting “COM control” provides a choice of “Control Flags” that let you cancel read or write operations, clear buffers, or set modem control signals.

Structured data types

Comm data items are “structured” in that they have associated flags. The purpose of these flags is to indicate transmission/reception completion or an error condition. Each flag is identified in the Data item grid in the “Structured Fields” column. The three available flags are “ReadDone”, “WriteDone”, and “Error”. The data item name includes the flag name as a suffix, separated by a dot (for example: “GetMessage.ReadDone”). The flags are not independent flag data items. You can use them in Decision blocks by referencing their complete name as a structured data item.

5.2.6 Compare Block



**Compare
block Button**

The Compare block can make numerical or string comparisons. Flow follows one of the two exit paths of the Compare block based on the result of the comparison. The type of comparisons available are *less than*, *equal to*, *greater than*, *not equal to*, *greater than or equal to*, and *less than or equal to*. For string comparisons, you can further specify if the first string of the compare is considered a substring of the second with the “Instr” (is contained in) function. Optionally, you can use the “Instr” function to perform case-sensitive comparisons.

Double-click the Compare block to edit its expression. This displays the “Compare Block... Expression” tab (see Figure 5-7). This example compares the register with the name “NoOfUnits” to a “Fixed Integer” constant of “4”.

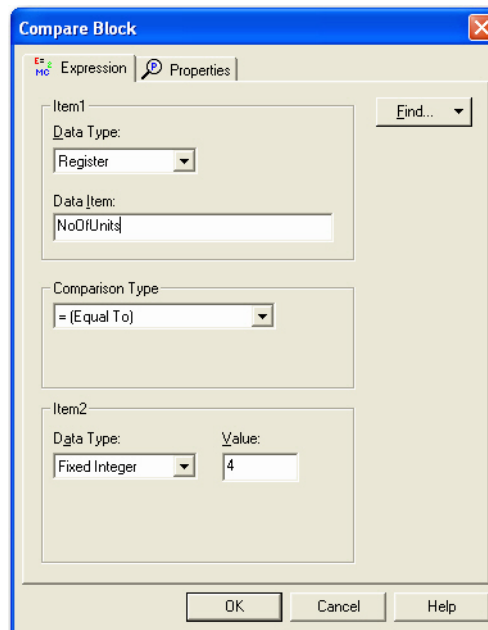


Figure 5-7 The “Compare Block... Expression” tab



When using a constant in the “Compare... Expression” tab, the variable operand must be “Item1” and the constant must be “Item2”.

When Think & Do Runtime executes a Compare block, control will pass through the “Yes” path to the next block in that path if the comparison is TRUE. If FALSE, control passes through the “No” path to the next block in that path.

Editing a Compare Block

To edit a Compare block expression, follow these steps:

1. While in Selection Mode, double-click the Compare block to open the “Compare Block... Configuration” tab (see Figure 5-7).
2. Click the “Data Type” drop-down to select the data item type.
3. Enter the “Data Item” using one of the following techniques:
 - Enter the data item name in the “Data Item” field. While typing, Think & Do attempts to auto-complete the name.

- Enter the data item “Index”, for example “01”. When tabbing to the next field, Think & Do displays the data item name associated with the index.
 - Use the “Find...” menu button or double-click the field to display the Data Item Grid.
4. Use the “Comparison Type” drop-down list to select the desired comparison.
 5. Use the “Item2” group to select a “Data Type” and “Data Item” (or fixed “Value”).
 6. Click the “OK” button.

Yes/No path Selection

By default, the connection that goes out the bottom of the Compare block is the “Yes” path; the exit on the right is the “No” path. The “Compare Block... Properties” tab permits adding a block “Description” and swapping the positions the Yes and No exit paths (another way is to right-click a block, and then select “Swap Yes/No Connectors” from the pop-up menu). Entering the block “Number” in the “Yes” or “No” group explicitly sets or modifies the target block of the “Yes” or “No” connector.

Wait One Scan

Selecting the “Wait One Scan” check box for either the “Yes” or “No” group indicates that the Think & Do Runtime should evaluate the comparison and then wait for the next scan cycle before taking the resulting path.

Compare Block Summary

Table 5-2 defines all possible Compare block combinations. Moves that are valid for individual data items are also valid in arrays. You may also move data to Inputs when the flow chart is a Simulation type.

Table 5-2 Valid Compare Block Comparisons

From... To...	1 Output	N Outputs	1 Flag	N Flags	1 Timer	N Timers	1 Register	N Registers	1 Counter	N Counters	1 Number	N Numbers	1 Byte	N Bytes	System	1 Float	N Floats	1 String	N Strings
1 Float	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Floats	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
Float const.	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
1 Input	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Inputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Output	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Outputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Flag	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Flags	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Timer	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Timers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Register	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Registers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Counter	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Counters	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Number	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Numbers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Byte	✓	8	✓	8	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Bytes	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	✓	=
Fixed Integers	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
Current Date	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓	x
Current Time	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓	x
System	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
1 String	x	x	x	x	✓	x	✓	x	✓	x	✓	x	✓	✓	✓	✓	x	✓	x
N Strings	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
String Const.	x	x	x	x	x	x	x	x	x	x	x	x	✓	x	x	x	✓	x	x
Legend	<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>✓ Valid move</p> <p>x Invalid move</p> </div> <div style="width: 45%;"> <p>= Valid move, if the number of <i>From</i> and <i>To</i> data items are equal</p> <p>2 4 8 16 32 64 Can move up to 2, 4, 8, 13, 32, or 64 data items</p> </div> </div>																		



Control block Button

5.2.7 Control Block

The Control block is the most basic building block of a flow chart. It is one of the action blocks of the Think & Do (see “Types of Action Blocks” on page 4-13). Its job is to do the discrete actions listed in its expression list. A Control block is capable of setting bits ON or OFF (flags or outputs), and doing internal timer or counter operations.

Use the “Control Block” dialog box to select the action(s) for the control block:

- Turn a Flag or Output ON or OFF.
- Pulse an Output (where a pulse turns off after a specified duration).



A Pulse output with a duration of 0 (zero) causes the output to be on for a single scan only.

- Start, stop, reset, or restart a Timer.
- Increment, decrement, or reset a counter.



Unlike ladder logic, these actions are *latched*. In other words, when turning an output on, it stays on until the flow chart (or another flow chart) turns it off. The only exception is the pulse output, which automatically turns off after the specified duration (entered in the Data Item grid).

The Control Block can perform up to ten actions. Double-click the Control block to edit its expression. This displays the “Control Block... Expression” tab (see Figure 5-8).

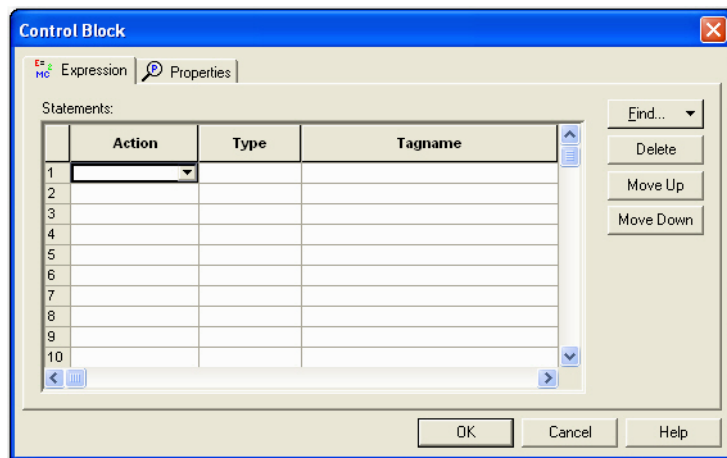


Figure 5-8 The “Control Block... Expression” tab

Since actions in the Control block execute in the order in which they appear, the Control block permits reordering actions. To reorder an action, select it (it appears highlighted), and then use the “Move Up” or “Move Down” buttons to move the selected action. For these buttons to be available, there must be more than one action defined in the block.



The order rarely makes a difference. This feature is primarily to improve readability. It does, however, make a difference when starting a timer, and then resetting that timer, as opposed to the inverse. This is applicable if the timer was previously stopped but did not complete (elapsed time operations).

Editing a Control block

To edit a Control block's expression, follow these steps:

1. Double-click the Control block to open the "Control Block... Expression" tab.
2. Use the "Action" drop-down list to select the action desired.
3. Use the "Type" drop-down list to select a data item type that can perform the selected "Action".
4. Enter the "Tagname" using one of the following techniques:
 - Enter the tagname in the "Tagname" field. While typing, Think & Do attempts to auto-complete the name.
 - Enter the data item "Index", for example "01". When tabbing to the next field, Think & Do displays the data item name associated with the index.
 - Use the "Find..." menu button or double-click the field to display the Data Item Grid.
5. Repeat steps 2 through 4 as required for additional expressions. A block must have at least one expression. After entering more than one expression, the "Move Up" and "Move Down" buttons become available. These buttons permit re-ordering the expression, since the Think & Do Runtime executes them in the order they appear in this dialog box. Select one of the expressions, and then click the desired button to move it in the list.
6. Click the "OK" button.

5.2.8 Decision Block

The Decision block evaluates a Boolean expression as TRUE or FALSE. The expression may consist of one or more bits. The Decision block has one entry point and two exit points (Yes/No), based on the Boolean decision. Use the "Decision Block" dialog box to select the bit(s) to be evaluated, which are of these types:

- Array — when testing a bit in a flag or timer array
- Comm
- Flag
- Input
- Output
- Timer



**Decision
block Button**

The AND and OR “Operator” in the dialog box permits additional Boolean conditions in the expression. AND has greater precedence than OR (in other words AND expressions are evaluated before OR expressions, and then they are evaluated from top-down). Double-click the Decision block to edit its expression. This displays the “Decision Block... Expression” tab (see Figure 5-9).

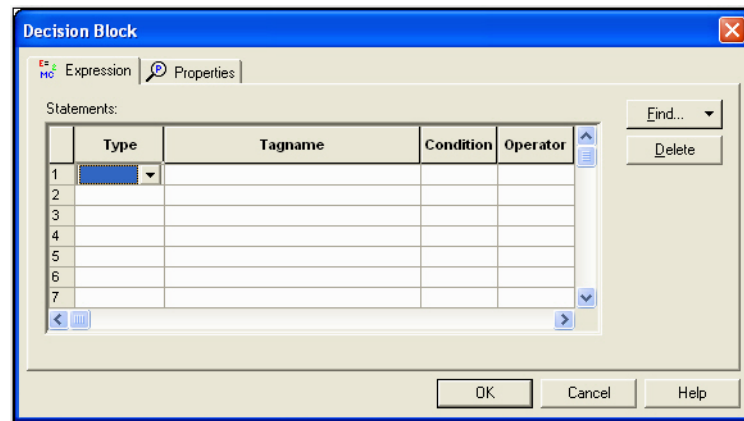


Figure 5-9 The “Decision Block... Expression” tab

Editing a Decision Block

To edit a Decision block expression, follow these steps:

1. While in Selection Mode, double-click the Decision block to open the “Decision Block... Expression” tab.
2. Use the “Type” drop-down list to select the data item type. Only valid data types appear in the list. Available types are Array, Comm, Flag, Input, Output, and Timer.
3. Enter the “Data Item” using one of the following techniques:
 - Enter the tagname in the “Tagname” field. While typing, Think & Do attempts to auto-complete the name.
 - Enter the data item “Index”, for example “01”. When tabbing to the next field, Think & Do displays the data item name associated with the index.
 - Use the “Find...” menu button or double-click the field to display the Data Item Grid.
4. Use the “Condition” drop-down list to set the condition desired. Conditions that appear are: ON or OFF.
5. Click the “OK” button.

Yes/No Path Selection

By default, the connection that goes out the bottom of the Decision block is the “Yes” path; the exit on the right is the “No” path. The “Decision Block... Expression” tab permits adding a block “Description” and swapping the positions the Yes and No exit paths (another way is to right-click a block, and then select “Swap Yes/No Connectors” from the pop-up menu). Entering the block “Number” in the “Yes” or “No” group explicitly sets or modifies the target block of the “Yes” or “No” connector.

Decision Block Summary

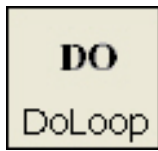
Table 5-3 defines all possible Decision block combinations. Moves that are valid for individual data items are also valid in arrays. You may also move data to Inputs when the flow chart is a Simulation type.

Think & Do

Table 5-3 Valid Decision Block Comparisons

From... To...	1 Output	N Outputs	1 Flag	N Flags	1 Timer	N Timers	1 Register	N Registers	1 Counter	N Counters	1 Number	N Numbers	1 Byte	N Bytes	System	1 Float	N Floats	1 String	N Strings
1 Float	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Floats	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
Float const.	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
1 Input	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Inputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Output	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Outputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Flag	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Flags	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Timer	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Timers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Register	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Registers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Counter	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Counters	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Number	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Numbers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Byte	✓	8	✓	8	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Bytes	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	✓	=
Fixed Integers	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
Current Date	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓
Current Time	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓
System	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
1 String	x	x	x	x	✓	x	✓	x	✓	x	✓	x	✓	✓	✓	✓	x	✓	x
N Strings	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
String Const.	x	x	x	x	x	x	x	x	x	x	x	x	x	✓	x	x	x	✓	x
Legend	<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>✓ Valid move</p> <p>x Invalid move</p> </div> <div style="width: 45%;"> <p>= Valid move, if the number of <i>From</i> and <i>To</i> data items are equal</p> <p>2 4 8 16 32 64 Can move up to 2, 4, 8, 13, 32, or 64 data items</p> </div> </div>																		

5.2.9 Do Loop Block



Do Loop
Block Button

Think & Do provides a Do Loop construct (see Figure 5-10), which you can use within a standard flow chart or subchart. Of course, most flow charts loop back to execute part or all of the chart, but that repetition must wait until the next scan. The Do Loop is the only construct that allows blocks to execute more than once during a scan.

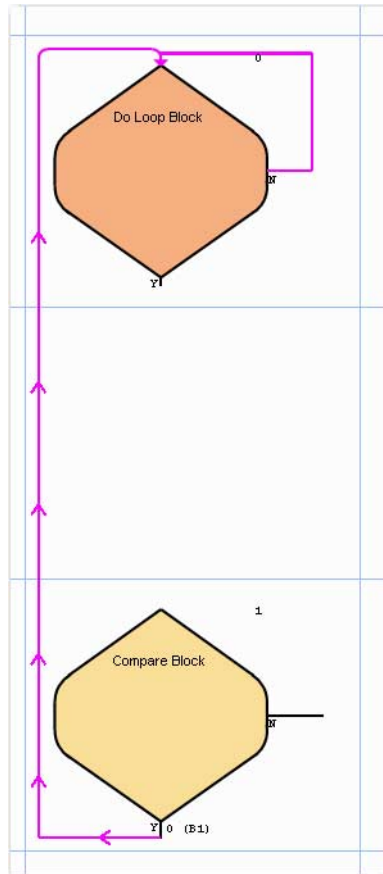


Figure 5-10 A Do Loop consists of Do Loop block linked to a Compare block



The best uses for Do Loops are for data operations such as manipulating arrays and sorting. Remember that I/O read/write operations occur only once per scan, so trying to pulse outputs rapidly from a Do Loop doesn't work.

The Do Loop and Compare blocks shown in Figure 5-10 have a loop-back line from the exit of the Compare to the entry of the Do Loop block. This connecting line will vary in length, allowing you to place several blocks between the Do Loop and Compare blocks.

The Do Loop has the following rules and characteristics:

- The Do loop executes only if logic flow reaches the Do Loop block.
- You specify a maximum loop count, as a safeguard, in the Do Loop block expression.
- The loop executes the blocks you insert within it each scan until the Compare block expression is satisfied (TRUE if the loop exit is “Yes”; FALSE if the loop exit is “No”).

Think & Do

- The block(s) within the Do Loop cannot have an exit point that connects to another point outside the Do Loop. In other words, the Compare block at the bottom of the loop is the only valid loop exit point.
- To change the Compare block to a Decision block, right-click on the Compare block, and then select "Change this Compare block to a Decision block" from the pop-up menu.



Be aware that looping an extremely large number of times can cause logic solve time to exceed the specified scan interval for the project.

The Do Loop has a user-defined maximum loop count (default=1) as a safety exit to prevent an infinite loop at runtime. You should not use it as a normal loop exit method.

To set a maximum loop count:

1. While in Selection Mode, double-click the Do Loop block to open the "Do Loop Block... Expression" tab.
2. Enter the maximum value, in the field provided, and click the "OK" button.

After defining the "Maximum Loop Count", the Do Loop block displays that value as shown in Figure 5-11.

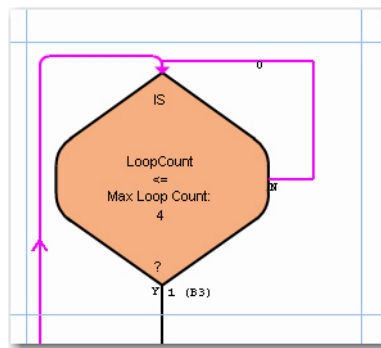
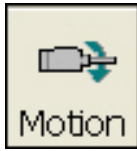


Figure 5-11 Do Loop block showing "Maximum Loop Count"

5.2.10 Motion Block



Motion Block Button

Think & Do's integrated motion control provides a programming interface for motion. Integrated motion control provides convenience and time savings during system design, since you avoid having to link external motion programs with the project. The system block diagram in Figure 5-12 shows a typical I/O subsystem, along with a motion control system design.

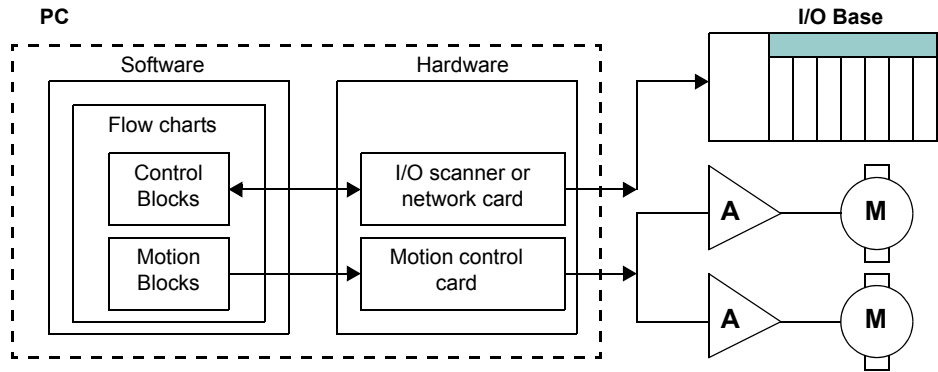


Figure 5-12 Typical I/O subsystem and a motion control systems design

Just as a Control block can set I/O points ON or OFF, a Motion block can issue a command to or request data from a motion control card. In the example “Motion... Expression” tab in Figure 5-13, the command “BeginMoveTo” for “Axis1” has a destination of “Part_Length”, where “Axis1” and “Part_Length” are tagnames.

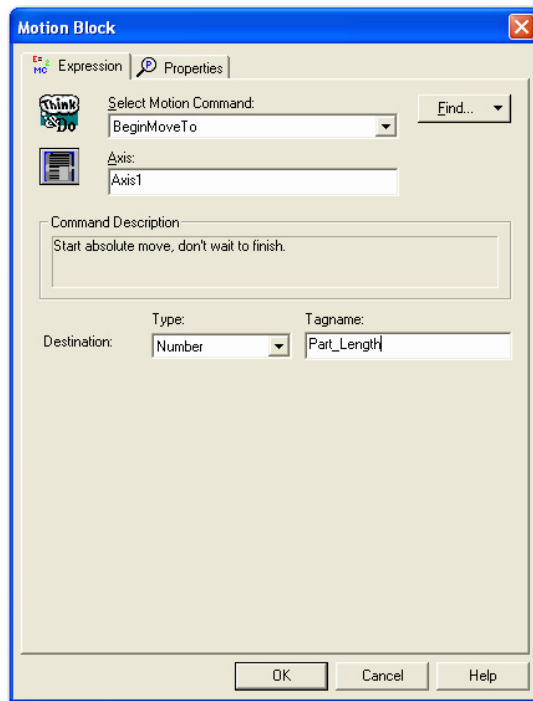


Figure 5-13 The “Motion Block... Expression” tab

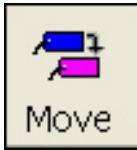
Think & Do

There are generic commands and commands that are specific to particular motion systems. By default, all generic and specific commands appear in the list.

A motion system runs asynchronously to the logic solve scans. This means flow charts need to verify that the requested move or action has completed before requesting the next move or action from the motion card.



Think & Do supports several motion cards that appear in IOView. These including Douloi, Galil, and MEI (see the Drivers menu). Check the Phoenix Contact website at www.phoenixcontact.com for the latest I/O and motion card driver information.



**Move block
Button**

5.2.11 Move Block

The Move block simply moves data from one location to another. All data items, including I/O, exist as data items. Therefore, doing a move from one data item to another simply moves (copies) the data of the first data item into the second data item. The value of the first data item is preserved.

The “Move Block... Expression” tab (see Figure 5-14) permits definition of the “From” and “To” data fields. After selecting the “Data Type” from the drop-down list, the “Starting data item” becomes available. The “From” and “To” variables may be of types listed in the table below. Moves that are valid for individual data items are also valid in arrays. You may also move data to Inputs when the flow chart is a Simulation type.

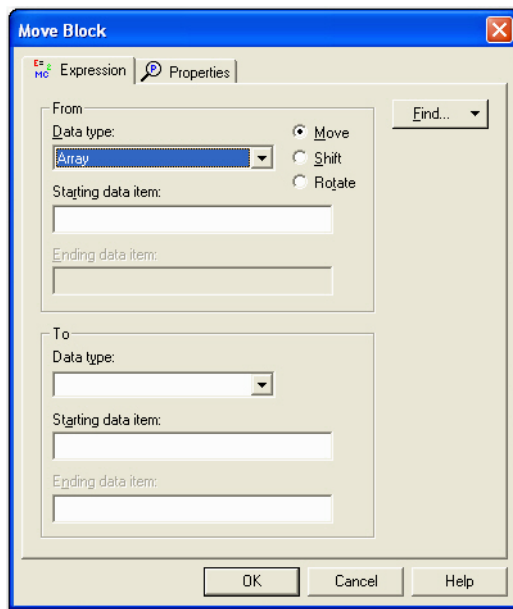


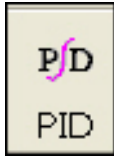
Figure 5-14 The “Move Block... Expression” tab

Think & Do

Table 5-4 Valid Moves

From... To...	1 Output	N Outputs	1 Flag	N Flags	1 Timer	N Timers	1 Register	N Registers	1 Counter	N Counters	1 Number	N Numbers	1 Byte	N Bytes	System	1 Float	N Floats	1 String	N Strings
1 Float	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Floats	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
Float const.	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
1 Input	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Inputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Output	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Outputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Flag	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x
N Flags	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x
1 Timer	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Timers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Register	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Registers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Counter	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Counters	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Number	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Numbers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
1 Byte	✓	8	✓	8	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
N Bytes	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	✓	=
Fixed Integers	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
Current Date	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓	x
Current Time	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓	x
System	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x
1 String	x	x	x	x	✓	x	✓	x	✓	x	✓	x	✓	✓	✓	✓	x	✓	x
N Strings	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=
String Const.	x	x	x	x	x	x	x	x	x	x	x	x	x	✓	x	x	x	✓	x
Legend	<div style="display: flex; justify-content: space-between;"> <div> <p>✓ Valid move</p> <p>x Invalid move</p> </div> <div> <p>= Valid move, if the number of <i>From</i> and <i>To</i> data items are equal</p> <p>2 4 8 16 32 64</p> <p>Can move up to 2, 4, 8, 13, 32, or 64 data items</p> </div> </div>																		

For more information on how the Move block performs powerful data manipulations, refer to “Using the Move Block” on page 7-16.



PID Block Button

5.2.12 PID Block

The PID block performs a mathematical algorithm for process control, which is the Proportional Integral Derivative (PID) algorithm. Think & Do permits up to 64 PID loops in a project.

In the block diagram shown in Figure 5-15, the PID loop has two inputs and one output to the outside world (the process control application). These variables are the Setpoint (SP), the Process Variable (PV), and the Output Variable. These data items also appear in the “PID... Expression” tab (see Figure 5-16).

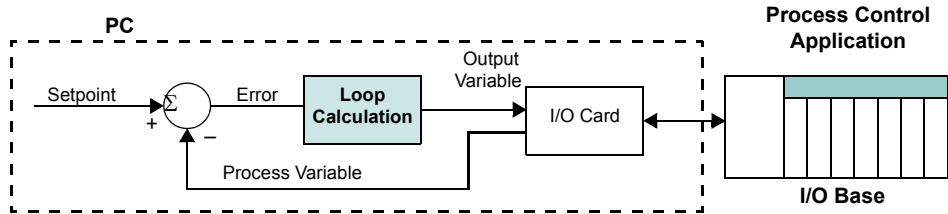


Figure 5-15 PID loop block diagram

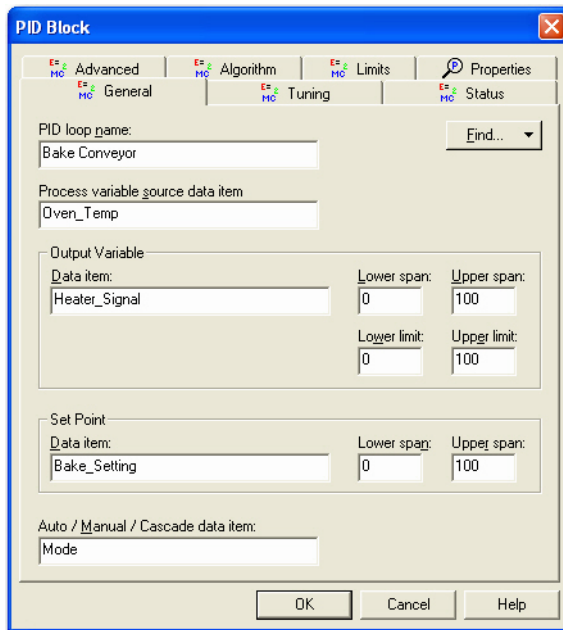


Figure 5-16 The “PID Block... Expression” tab

You must design your flow chart to execute PID Blocks at a frequency suitable for the process control application. This frequency, or loop sample rate, can often be much slower than the normal scan interval. The proper value conserves system processing power while

still guaranteeing loop stability. Therefore, the PID Block must generally be placed inside a timing loop in your flow chart. It is a good idea to put your PID Block in a subchart and call it when needed.



A PID block has a dialog box with several tabs to configure. These are covered in detail in the online help and “PID for Process Control User Manual.” This document is installed as part of the standard Think & Do setup.

5.2.13 Run Program Block



Run Program Block Button

The Run Program block lets you start other Windows programs or batch files under the flow chart control. This is one of the most powerful aspects of having a PC-based control platform. The “Run Program Block... Expression” tab (see Figure 5-17) has parameters that specify the program path, arguments, runtime process parameters, and an error value. You may also specify these parameters indirectly using string data items. The external program runs while the flow chart continues normal execution. The Run Program block can start a custom C++ or Basic program or any “.exe” (executable) file.

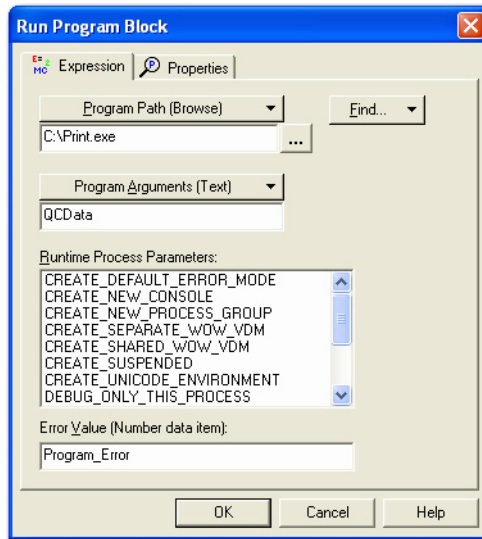


Figure 5-17 The “Run Program Block... Expression” tab



The flow chart design must take into account the execution time of the program it calls. Be aware of potential problems with executing Run Program every scan. It's easy to inadvertently start a program every scan.

5.2.14 SQL Block



**SQL Block
Button**

The SQL block provides a dialog box that helps you construct SQL queries for database access.

An enterprise-enabled Think & Do project is one that integrates factory floor control with enterprise-level operations and databases related to custom orders, inventory, parts procurement, warehousing, shipping, and so on. The SQL block gives Think & Do the ability to communicate in realtime over large corporate networks to enterprise systems.

Double-click a SQL block to display its expression dialog box (see Figure 5-18), which has the following tabs:

- General — configures the request type, record type, and query result data items
- Tag Mappings — map record fields to project tagnames
- Filters (Where) — specify operators and conditions for field names
- Sorting (Order By) — specifies field names and the sorting order
- SQL Statement — displays the resulting SQL statement (cannot be directly edited — use the tabs to change the SQL statement)

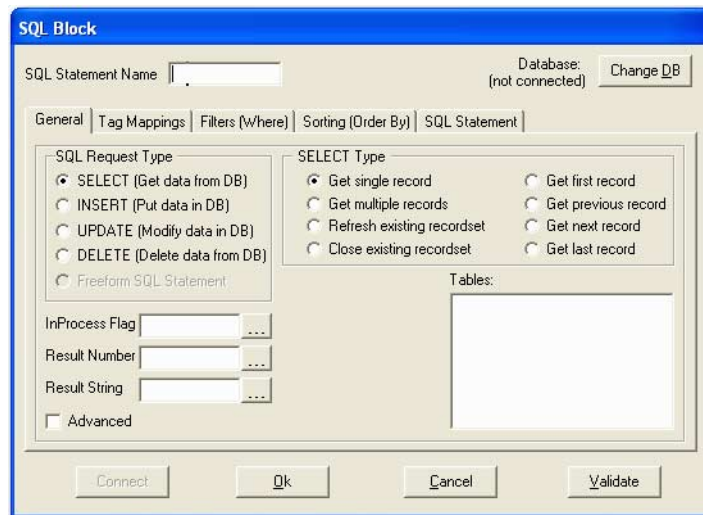


Figure 5-18 The “SQL Block... General” tab

For a full discussion on SQL database operations, see “SQL Database Operations” on page 7-37.

5.2.15 Wait Block



The Wait block inserts a wait period in the execution of the flow chart. Configure the wait period in milliseconds. Select the “One Scan Wait” check box to have the block for exactly one scan cycle. Setting the “Wait Time” to zero automatically sets the “One Scan Cycle” check box.

To configure the wait time:

1. Double-click the Wait block to display the “Wait Block... Expression” tab (see Figure 5-19).

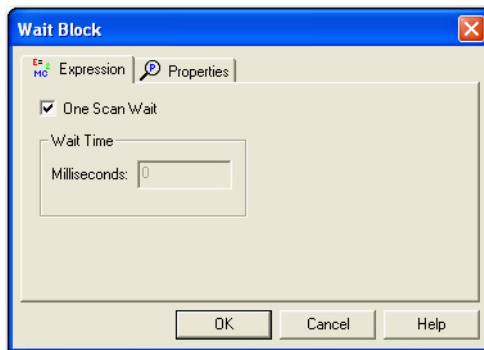


Figure 5-19 The “Wait Block... Expression” tab

2. Do one of the following:
 - To wait for one scan cycle, select the “One Scan Wait” check box.



The duration changes to zero (the next time the “Wait Block” dialog box opens) if it was previously non-zero, and “Wait Time” becomes unavailable.

- To wait for a specified period of time, enter the number of milliseconds to wait in the field.



If the duration is zero, the Wait block defaults to one scan cycle. The Wait block automatically selects the “One Scan Wait” check box upon clicking the “OK” button.

3. Click the “OK” button.

During the wait period, many scan periods may pass. However, the flow chart with the Wait block will not execute further until the wait period ends. On the scan when the wait time expires, flow chart execution resumes at the next block following the Wait block.

The Wait block is easier to use than constructing a timing loop for a delay in the flow chart, since there is no need to create a timer data item or create the flow chart loop. When a Wait block is executing, no other blocks in the current flow chart execute until the wait completes. Other flow charts continue to execute normally as long as they are enabled.

5.3 Using Subcharts

A flow chart can call a subchart, which transfers execution to the subchart. The subchart returns to the original flow chart, transferring execution back again. A subchart can also call another subchart. However, a subchart cannot call itself (recursion is not allowed). Flow charts can nest subchart calls up to 16 levels for tremendous flexibility and support for large projects.

To start a new subchart:

1. Do one of the following:
 - From the ProjectCenter main menu, select the “File... New... Subchart” menu.
 - From the FlowView main menu, select the “File... New... Subchart” menu.
2. Either action displays a dialog that prompts you to name the subchart. After naming it, the subchart appears in the “Subchart” folder in ProjectCenter’s Flow chart explorer, and an empty subchart appears in FlowView.

5.3.1 Blocks for Subcharts

When creating a subchart, the first block must be a Begin block and the last one a Return block. FlowView creates these blocks for you. Naturally, the remainder of the subchart you create must go between and connect to these two blocks. There can be only one Begin and Return block for a subchart (they cannot be copied, pasted, or deleted). The properties of the Begin block determine how the subchart communicates with flow charts that call it.

In discussing subcharts, we use the word parameters in a special sense. Figure 5-20 shows a representation of how data values pass between calling and called programs as parameters. Flow charts call subcharts and pass parameters by reference or value. When passed by reference, the subchart is actually reading and writing the data item used by the calling flow chart. When passed by value, the subchart receives the value of the calling parameter at the time of the call. The subchart uses a local copy that does not directly impact the calling flow chart’s data. When the subchart reaches its first loopback point, its execution pauses for the current scan. The calling flow chart resumes execution when the subchart executes its Return block. Upon return, the calling flow chart is free to use the new values set by the subchart.

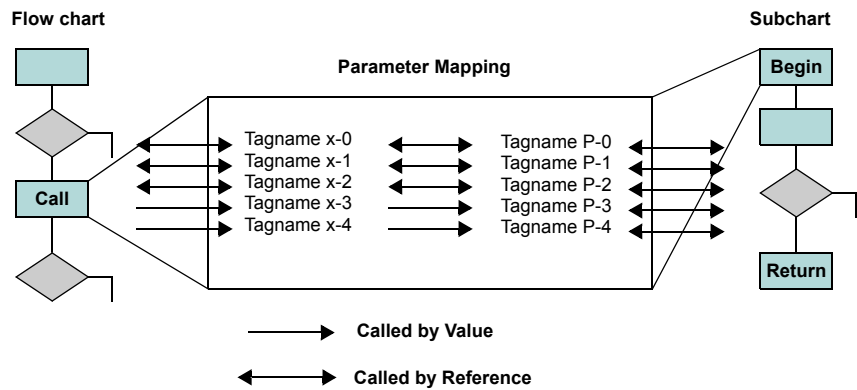


Figure 5-20 Calling flow chart passes values to subcharts via parameters

The benefits of parameter mapping are many. It allows you to create a generic subchart with generic parameter names. When used in a project, many other flow charts can use their own tagnames and call the subchart. As long as the calling flow chart has properly mapped all the parameters defined in the subchart, the subchart has all the data it needs to function.



A subchart has to exist before it can be called. In fact, it's best to completely write the subchart and define its parameters before configuring the call block of the calling flow chart.

5.3.2 Begin Expression

Double-click the Begin block to edit the Begin Expression expression dialog shown in Figure 5-21.

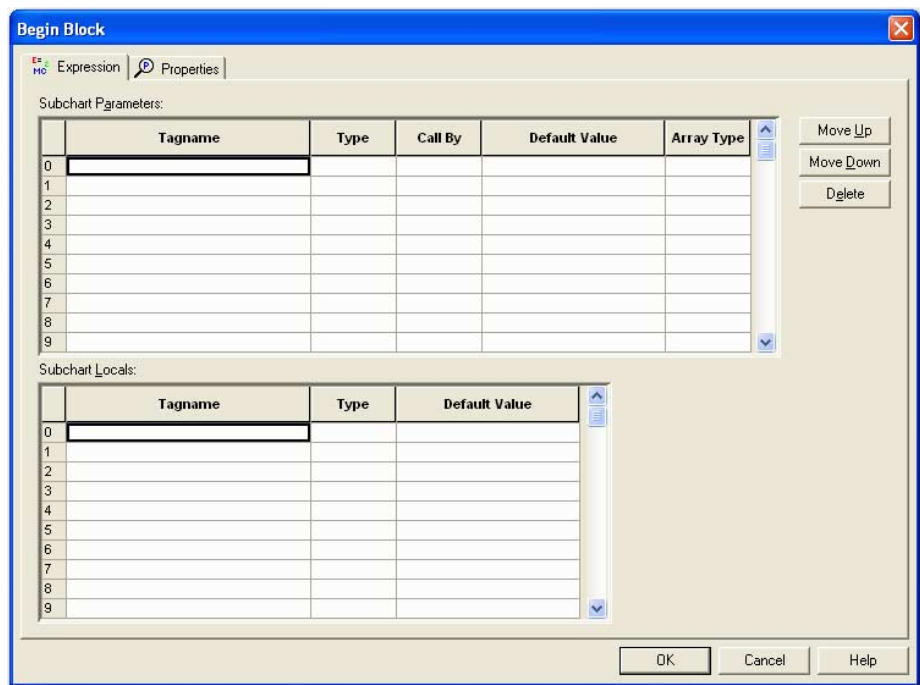


Figure 5-21 The “Begin Block... Expression” tab defines parameters and locals

You’ll notice first that the Begin Block expression dialog has two lists:

- “Subchart Parameters”: These data items map externally to data items of the calling flow chart. Parameter data items are assigned unique names that can be up to 30 characters long.
- “Subchart Locals”: Local data items are only used internally in the subchart. Local data items are assigned unique names that can be up to 30 characters long.

Parameters

All fields for each parameter, except “Default Value”, become drop-down lists when you select in them. In the “Call By” field, you can choose either “Reference” or “Value”. Use call by reference when the calling flow chart needs to see changes made by the subchart. The subchart’s data item references the calling flow chart’s data item, so they change together. Use

call by value when you want the subchart to use the data item as a copy. The “Default Value” entry only applies to data items called by value. For ease of use, Think & Do assigns this default value when the calling flow chart does not assign a value for it in its Call block. The default value field does not apply for call by reference data items. You use the “Array Type” field entry to specify the type of array, when you select “Array” from the “Type” entry and use call by reference.

Locals

Since locals are accessible only within a subchart, there is no call by selection. The default value field assigns a value for the local data item each time the subchart is called. During execution, the subchart can modify this value as necessary.

The “Move Up” and “Move Down” buttons on the right side of the “Begin... Expression” tab provide extra tools to manage subchart parameters and locals. These buttons let you arrange data items in the best order for readability. Parameter order does not affect subchart functionality. You also can select a data item (use the row header) and perform cut-copy-paste operations.

5.3.3 Call Expression

A flow chart (standard or subchart) executes a subchart when the control passes to a Call block. Double-click the Call block to display its expression dialog as shown in Figure 5-22.

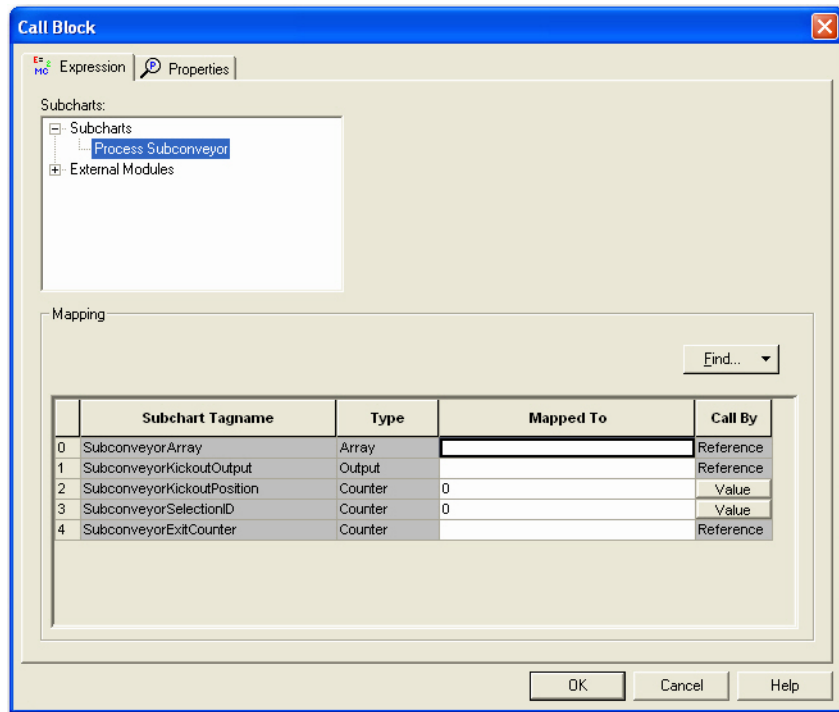


Figure 5-22 The “Call Block... Expression” tab

Subchart Name

Use the “Subcharts” tree-structure list to select the subchart you want to call. You must have previously created the subchart, named it, specified any parameters that it requires, and saved it before it appears in this list box. The list includes all existing subcharts in a project and pre-defined “External Modules”. Standard flow charts in the “Unused” category do not appear in the list.

External Modules

The “External Modules” branch includes the following entries:

- “AutomationDirect.com Ethernet I/O” includes custom commands for the CTRIO module.
- “Cognex” includes the Cognex vision system functions.
- “Interbus Special Functions” includes the PCP functions.
- “Modbus Master” includes custom commands for supporting Modbus Master I/O.
- “Utilities” includes array, string, timer, and inverse trigonometric functions.

For detailed descriptions of these external functions, see the online Help.

Parameter Mapping

When you select the subchart name, the “Mapping” group lists all subchart parameters. All you need to do is use the “Mapped To” field to map (equate) data items of the calling flow chart to those of the subchart. The tagnames you place in the “Mapped To” field are generally different from the subchart parameter names, but a difference is not required.

Call By

This field shows whether the parameter is call by reference or call by value. If call by value, there are two options with “Value” as the default:

- Pass a constant value (enter the value). Click the “Tag” button to enter a value. This toggles the button title to “Value”.
- Pass the value of a tagname. Click the “Value” button to enter a tagname. This toggles the button title to “tag”.

5.3.4 Interaction of Call and Begin Blocks

The selections you make for each data item in a subchart’s Begin block configuration determines the available (and relevant) choices in the calling flow chart’s Call block. This means you need to specify the Begin block configuration before configuring related Call block(s).

Table 5-5 Begin and Call Block Parameter Relationships

Expression choices			
Begin (in Subcharts)			Call (in Flow charts)
Parameters	Pass by Reference	N/A	Call by Reference
	Pass by Value	Default value	Call by Value
			Call by Tag
Locals	N/A	Default value	N/A

An explanation of Table 5-5 (line-by-line) follows:

- A subchart's parameters allow exchange of data between a subchart and the calling flow chart. A subchart's locals are not accessible from any other flow chart. The subchart's locals are not retained (remembered from one call to the next call of the subroutine).
- A parameter passed by reference must be called by reference. This method of mapping allows a flow chart's data item to reference (map) to a different subchart tagname. So, when you change the value of the subchart tagname, it changes the value of the calling flow chart's data item.
- A parameter passed by value is a copy of the value of the original data item. The value of the original data item remains unchanged by the subchart. It may be called by value, or called by tag. When called by value, the flow chart simply uses the value at the time of the call. When called by tag, the flow chart uses a tagname to point to a data item for the subchart to accept as a constant.
- Subchart locals have a default value (value at the time the subchart is called. The subchart can subsequently modify that value, but all this is transparent to the calling flow chart. Therefore, a "Call Block... Expression" tab will not show any locals of the subchart being called.

5.3.5 Subchart Design Summary

The main points of using subcharts in a project are:

- Only a standard flow chart has an Enable block, permitting it to run automatically at run time (if enable conditions are satisfied).
- A subchart only runs if it is called. Either a standard flow chart or another subchart can call a subchart.
- The first block in a subchart is the Begin block, and the last one is the Return block. The subchart program you write goes between these blocks.
- The "Begin Block... Expression" tab defines parameters and locals for the subchart.
- If a subchart operates only on global tagnames (in the project's Data Item grid), you do not have to define parameters or locals in the "Begin Block... Expression" tab. However, this is generally not good design practice, since the subchart will be unsuitable for generic use by several calling flow charts.
- The subchart must exist and be saved to disk before you can edit the Call block parameters in the calling flow chart.
- Parameter mapping lets a subchart create a temporary working copy of the calling flow chart's data items using different local names. The mapping is defined in the "Call Block... Expression" tab of the calling flow chart.

5.3.6 Subchart Runtime Characteristics

The following traits show how subcharts work when you run the project:

- A single subchart can be called by one or more flow charts (standard or subchart) in a project, even at the same time. This multiplies a subchart's capacity to do work in the project.
- Since subcharts can be called from many places, each call is termed a "context." When debugging such a subchart, FlowView prompts you to choose a debug context — asking you which call you want to debug.
- A flow chart or subchart that is executing has an "Active" status.

- A flow chart or subchart that is currently executing a Wait block has a “Waiting” status until the block times out.
- A single call to a subchart does not constitute parallel processing. Specifically, the calling flow chart has an “In Subchart” status until the subchart completes (returns). If the subchart is down two or more levels, all flow charts above it have an “In Subchart” status until it returns.
- A subchart that is not the destination of any Call block will not appear in the Flow Chart Execution Path.
- A subchart that is the destination of a Call block will not execute until the Call block executes. The subchart has an “Out of Scope” status until it is called. If that subchart is called, its status becomes “Active”. If it calls yet another subchart, its status becomes “In Subchart”.

5.3.7 When to Use Subcharts

The real efficiency in using subcharts is their self-duplicating nature at runtime, and your opportunities to re-use them from project to project. Consider writing a generic subchart to do a frequently needed function in your industry. Thus, you save programming time each time you can re-use the subchart in various projects (consider creating a subchart library for use in your company).

One misconception about subcharts suggests a flow chart can call a subchart and then continue doing other things. This is not true. While a subchart executes, the calling flow chart is suspended until the subchart returns. If you need to solve logic in parallel each scan, you must use separate standard flow charts for each logic section.



Think & Do has comprehensive flow chart debugging capabilities. Refer to Section 10, “Debugging Projects” for details.

Section 6

Operator Screen Techniques	6-3
6.1 Target Hardware Considerations	6-3
6.2 Exploring Screens	6-3
6.2.1 Expanding the Explorer Tree	6-4
6.2.2 Setting a Startup Screen	6-4
6.2.3 Renaming, Duplicating, Deleting, or Printing a Screen	6-4
6.2.4 Opening a Screen	6-4
6.3 ScreenView Development Environment	6-5
6.3.1 Creating and Using Tags	6-5
6.3.2 Title Bar	6-10
6.3.3 Menu Bar	6-11
6.3.4 Toolbars	6-11
6.3.5 Using the Standard Toolbar	6-12
6.3.6 Execution Control Toolbar	6-13
6.3.7 Using the Bitmap Toolbar	6-14
6.3.8 Using the Dynamic Objects Toolbar	6-16
6.3.9 Using the Mode Toolbar	6-24
6.3.10 Using the Static Objects Toolbar	6-26
6.3.11 Using the Active Objects Toolbar	6-34
6.3.12 Using the Align and Distribute Toolbar	6-60
6.3.13 Configuring an Alarm	6-68
6.3.14 Configuring an Event	6-74
6.3.15 Trend Control Development Interface	6-74
6.3.16 Trend Control Runtime Interface	6-87
6.3.17 Configuring ActiveX Control Objects	6-89
6.3.18 Configuring .NET Control Objects	6-93
6.3.19 Configuring Grid Objects	6-97
6.3.20 Using and Creating Symbols	6-104
6.4 Setting Screen Attributes	6-112
6.5 Screen Groups	6-114
6.6 Additional Worksheets	6-115
6.6.1 Alarm Worksheets	6-115
6.6.2 Math Worksheets	6-121
6.6.3 Recipe Worksheet	6-122
6.6.4 Report Worksheet	6-124
6.6.5 Scheduler Worksheet	6-126
6.6.6 Trend Worksheets	6-127
6.7 Using Scripts	6-130
6.7.1 Scripts Folder	6-131
6.7.2 Global Procedures	6-131
6.7.3 Graphics Scripts	6-133
6.7.4 Startup Scripts	6-134
6.7.5 Background Scripts	6-134
6.7.6 Screen Scripts	6-135

6 Operator Screen Techniques

This section provides an in-depth discussion of ScreenView features for creating HMI screens. Knowing the capability of ScreenView will help you create useful screens that operators will like to use and engineers will rely on for diagnostics.

ProjectCenter must have a project open before you can use ScreenView to create screens. For information on creating and opening projects, see “Creating a Project” on page 2-3.

6.1 Target Hardware Considerations

Some PCs, such as Windows CE target hardware platforms, do not include a built-in video monitor. However, you can still have screen pages accessed remotely or that may be used later if you re-target the project to a system with a video monitor.

Other HMI options for remote systems include:

- Using the serial port on the CE target device to connect to a small operator interface.
- Using the Tag Link driver allows a local system to access data on the remote system (see “Configuring a Project as an OPC Client” on page 8-14).

6.2 Exploring Screens

In the ProjectCenter, click the “Screens” Explorer bar to see a list of the project’s HMI screens (Figure 6-1).

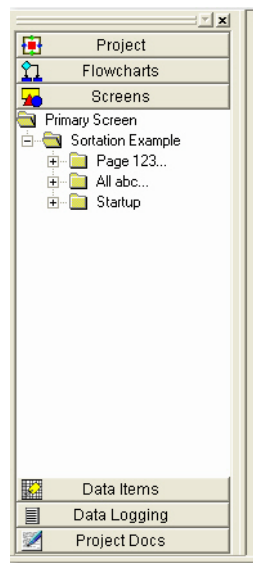


Figure 6-1 The “Screens” Explorer bar in ProjectCenter

The folders in the “Screens” Explorer bar list screens by screen sets. The top-level folder is the “Primary Screens” set. The first folder under that has the name of the project. Within the project screen set, there are categories that appear as sub-folders. The categories are not

necessarily exclusive, meaning that a screen can appear in multiple folders. In addition, you can create screen sets that appear under the project set. The pre-defined screen set categories are:

- “Page 123...” lists all HMI screens in numerical order (order in which they were created).
- “All abc...” lists all HMI screens in alphabetical order.
- “Startup” lists the single screen (user-selected) that the project initially displays at runtime.

6.2.1 Expanding the Explorer Tree

To expand the “Screens” Explorer bar tree, click the “+” beside the folder to expand.

6.2.2 Setting a Startup Screen

To make a screen the startup screen:

1. Select the screen name in the “Screens” Explorer bar.
2. In the main ProjectCenter window, select the “Startup Screen” check box in the “Options” group.

6.2.3 Renaming, Duplicating, Deleting, or Printing a Screen

To rename, duplicate, delete, or print an existing operator screen:

1. Move the cursor over the name of the screen in the “Screens” explorer pane.
2. Right-click and select the desired action from the context menu.



Before creating a new screen in a new project, it's best to name the project first by saving it to a new directory (Think & Do allows only one project per directory).

6.2.4 Opening a Screen

To open an existing screen, double-click the screen icon (graphic) by the name of the screen or the name itself in the “Screens” Explorer bar.

6.3 ScreenView Development Environment

ScreenView appears in an independent window that provides a complete screen development environment. The drawing area is the work space for placing screen objects (Figure 6-2). This section describes the major elements of the development environment.

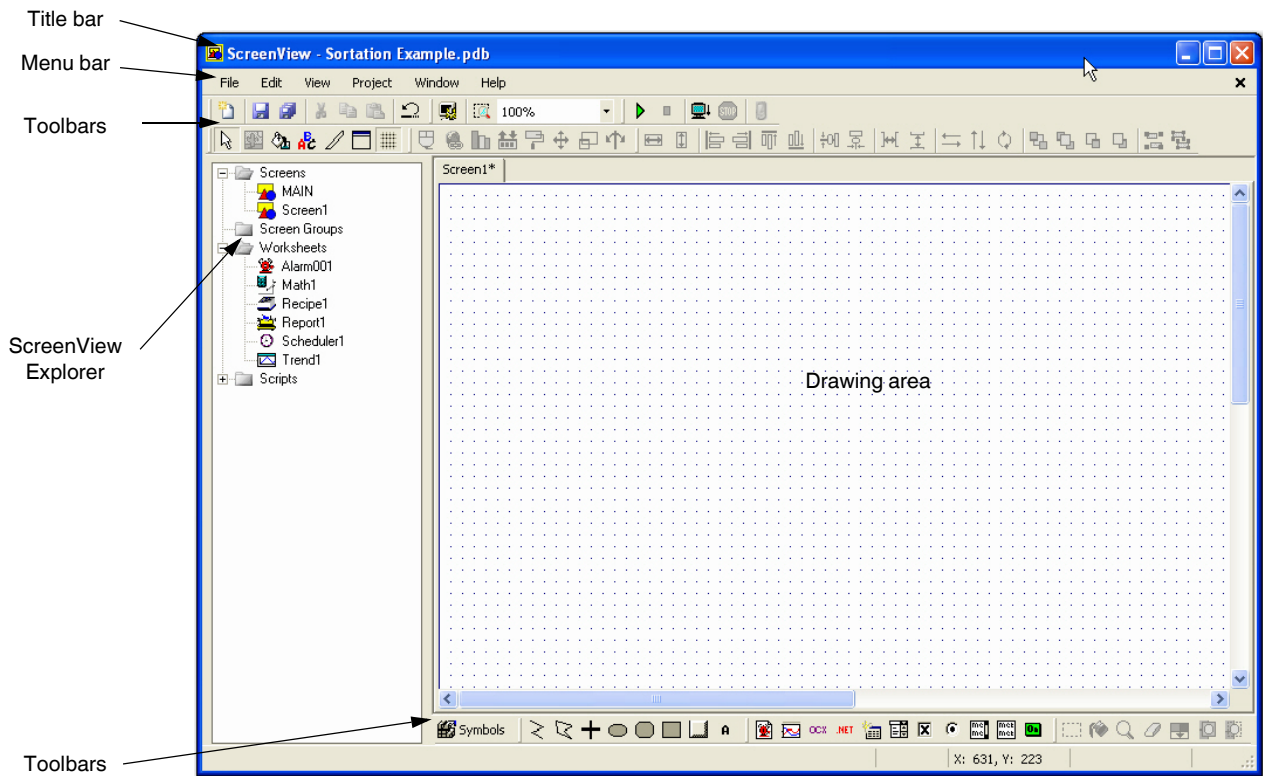


Figure 6-2 Use ScreenView to create and edit HMI screens

6.3.1 Creating and Using Tags

In ScreenView, you use tags to display state, results of calculations, alarm points, and so forth. In ScreenView, all tags are organized according to their origin— application, internal, or shared. Application and internal tags originate in ScreenView, while shared tags originate in the Think & Do Data Items editor.

The following is a description of the different ScreenView tag types:

- Application tags are user-defined, screen tags created for displays, to read from and write to field equipment, for control, auxiliary tags to perform mathematical calculations, and so forth.
- Internal tags are system tags predefined by ScreenView. Internal tags have predetermined functions (time, date, acknowledge alarms, storage of the logged-on user name and so forth). You cannot delete or modify these tags, but you can access their values from any ScreenView task.

- Shared tags are created in the Data Items editor and imported into the ScreenView environment. You cannot edit shared tags in the ScreenView environment, but you can modify these tags in ProjectCenter. Consequently, you can configure shared tags for any ScreenView task just as any other tag.

Understanding the Screen Tag Syntax

Observe the following guidelines when naming a screen tag:

- You can use letters, numbers, and the character "_" (underscore)
- Do not use the following characters:
` ~ ! @ # \$ % ^ & * () - = \ + \ [] { } < > ?
These characters are parsed by the ScreenView scripting language as logic and arithmetic operators.
- The tag must begin with a letter
- Maximum tag length is 255 characters
- Maximum class member length is 255 characters
- Tag names must be unique. You cannot specify the same name for two tags.
- Tags are not case sensitive (though we recommended using uppercase and lowercase characters to make names more readable. For example, use "TankLevel" instead of "tanklevel".
- Tag names must be different from internal tag names and math functions.



Use the @ character at the beginning of a tag name to indicate that the tag will be used as an indirect tag in the application.

For additional information, see the ScreenView scripting language in the online help.

Some valid tag examples include:

- Temperature
- pressure1
- count
- x

Using Screen Tag Values

A screen tag value can be one of the following types:

- Boolean is a Boolean or digital variable (0 or 1).
- Integer is a number (positive, negative, or zero). Equivalent to C-type long integer. Examples: 0, 5, -200.
- Real is a number internally stored as a double word. Equivalent to C-type double.
- String (ASCII text) is a character string up to 1024 characters that holds letters, numbers, or special characters. Examples: Recipe product X123, 01/01/90, *** On ***.

Accessing Tags

ScreenView uses an “Object Properties” dialog box to configure the settings for the objects and dynamics designed with ScreenView. To display an object’s “Object Properties” dialog box, double-click on the object, or select the object (clicking once on it), and then press <Shift>+<F2>. For example, Figure 6-3 shows the “Object Properties” dialog box for a check box object.



Figure 6-3 Object Properties: Check Box

The content of this dialog depends on the object (or group of objects) selected by the user. For detailed description of each “Object Properties” dialog box for each type of object, see “Toolbars” on page 6-11 and its subsections. For now, the key point is the “Tag:” field. This field appears in many “Object Properties” dialog boxes, and elsewhere in ScreenView. Wherever you need to fill in a tagname, you can double-click the field to display the “Object Finder” dialog box (see Figure 6-4).

Object Finder Dialog Box

The “Object Finder” dialog box (see Figure 6-4) lets you choose a tag from one of the three available tag types. The left panel of the dialog box displays a tree of available tag types, and the right panel displays tags of the selected type.

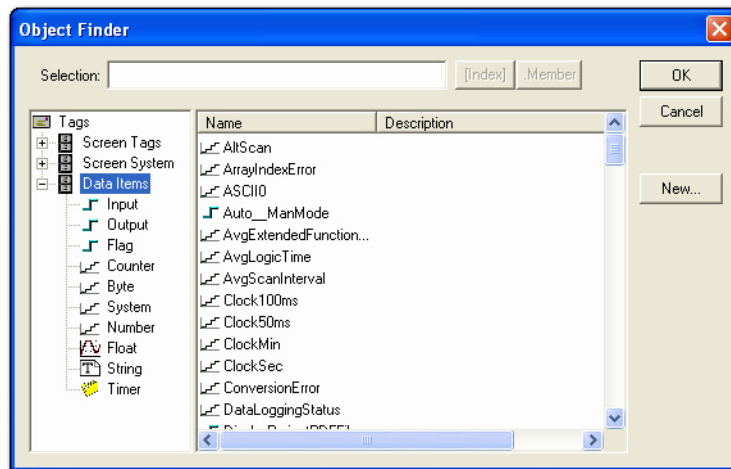


Figure 6-4 “Object Finder” dialog box

The three types that appear under the “Tags” node of the tree are:

- “Screen Tags” are internal, application tags.
- “Screen System” are internal, system tags.
- “Data Items” are Think & Do project tags defined in the Data Items editor.

Selecting one of these tag type nodes displays all tags of that type in the right panel. Double-clicking a node (or clicking the +/- icon in front of the node) toggles display of the data types available for that tag type.

Clicking on one of the tag data types under a tag type shows tags of that type and data type. To select a tag for use, simply double-click it. This closes the “Object Finder” dialog box and displays the selected tag in the field that you originally double-clicked to display the “Object Finder” dialog box.



Selecting a tag in the “Data Items” list automatically adds the “TND.” prefix to the tagname. This prefix is required to access Think & Do project tags in ScreenView.

When you select an array, the “Index” button becomes available, and you must select an index number for the array. Clicking this button displays the “Select Index” dialog box (see Figure 6-5).

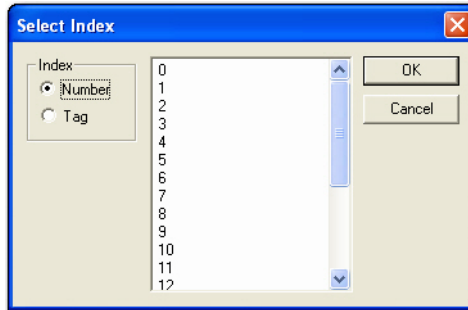


Figure 6-5 The “Select Index” dialog box

You can either select an index number from among the valid indexes displayed in the “Number” list, or you can click the “Tag” radio button to display a list of valid tags (both screen and Think & Do). Select the tag that will contain the index number at runtime.

When you select a timer, the “.Member” button becomes available, and you must select a member of the timer. Clicking this button displays the “Member selection” dialog box (see Figure 6-6). Either double-click the desired member; or select it, and then click the “OK” button.

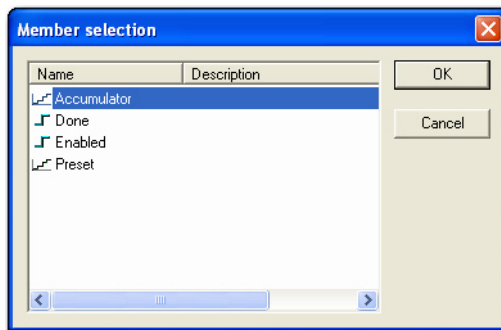


Figure 6-6 The “Member selection” dialog box

Creating ScreenView Tags

With the “Object Finder” dialog box displayed (see Figure 6-4 on page 6-7), click the “New” button to display the “New Tag” dialog box (see Figure 6-7).

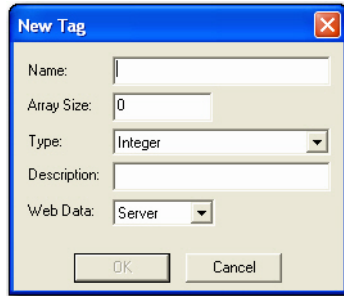


Figure 6-7 The “New Tag” dialog box

The fields in this dialog box are:

- “Name” is the name of the new screen tag.
- “Array Size” is the size of an array. The default value of zero is for tags that are not arrays.
- “Type” is a drop-down list of available internal tag types.
 - Boolean (same as “Flag”)
 - Integer (same as “Number”)
 - Real (same as “Float”)
 - String
- “Description” is any textual description of the tag.
- “Web Data” is not currently implemented.



You can also create screen tags using the “Screen Tags” dialog box (see “Viewing Tags” on page 6-10).

Viewing Tags

In addition to the “Object Finder” dialog box, you can view ScreenView application and system tags in the “Screen Tags” dialog box (see Figure 6-8). To open this dialog box, select the “Project... Screen Tags...” menu.

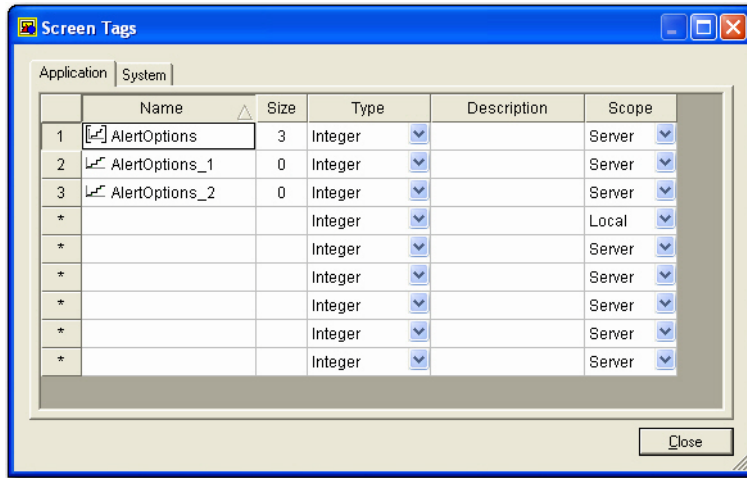


Figure 6-8 The “Screen Tags” dialog box

The “ScreenView Tags... Application” tab displays all screen tags, and the “System” tab displays all systems tags.

With this grid of tags, you can sort tags by any column by right-clicking in the column, and then selecting the “Sort Ascending” or “Sort Descending” menus. You can choose to show fewer or more columns by right-clicking and toggling the column names, or hovering the mouse pointer over “More Columns”, and then select one of the columns in that cascade list.

Other operations available include copy and paste, which lets you create a duplicate tag-name with a sequential number appended automatically. You can change a tagname by clicking in the “Name” field and overtyping the current name, and you can delete tags by right-clicking, and then selecting the “Delete Line” menu.

6.3.2 Title Bar

The title bar (located along the top of the ScreenView window) displays the ScreenView icon, the product name, and the name of the active project (if any).

The title bar also contains the following three buttons (from left to right):

- Minimize button, when clicked, minimizes the window.
- Resize/Maximize button, when clicked, toggles between the following two options:
 - Resize tiles the window
 - Maximize maximizes the window to fill your computer screen
- Exit button, when clicked, automatically saves the database, and then closes ScreenView. If you modified any screens or worksheets, ScreenView prompts you to save your work. This button function is similar to selecting the “File... Exit” menu.

6.3.3 Menu Bar

The ScreenView main menu bar contains the following menus:

- File: Contains options that let you manage application files.
- Edit: Contains options that let you manage displays and worksheets including options to paste copies of symbols (not linked) and copies of images that retain their links.
- View: Contains options that enable you to manage visible tools and provides shortcuts to the dialog boxes you open most frequently.
- Project: Contains options that enable you to execute applications locally and remotely, and provides links used to configure general application settings.
- Window: Contains options that enable you to manage open displays and worksheets.
- Help: Contains options that provide links to information about Think & Do.

For a detailed discussion of each menu option, see the Think & Do online help.

6.3.4 Toolbars

ScreenView provides several toolbars that let you easily perform different actions within the program. This section describes the function and default location of each toolbar.

The following toolbars contain general purpose tools, and they are located across the top of the workspace, just below the menu bar by default:

- Align and Distribute
- Execution
- Dynamic Objects
- Mode
- Standard

The following toolbars contain screen editing tools, and they are located along the bottom of the development environment by default:

- Active Objects
- Bitmap
- Static Objects
- Symbols

The rest of this section describes each of the ScreenView toolbars.

6.3.5 Using the Standard Toolbar

The Standard toolbar provides tools (shortcuts) that duplicate functionality found on the “File”, “Edit”, and “View” menus.



Figure 6-9 The “Standard” toolbar

The tools in the Standard toolbar are described below.



New Screen Tool

New Screen

Click the “New Screen” tool to open the “Screen Attributes” dialog box and create a new screen. Using the “New Screen” tool is the same as selecting the “File... New Screen” menu or typing the <Ctrl>+<N> key combination.



Save Tool

Save

Click the “Save” tool to save any active screens or worksheets. Using the “Save” tool is the same as selecting the “File... Save” menu or typing the <Ctrl>+<S> key combination. Also, the Save function is only available when you modify the active file.



Save All Tool

Save All

Click the “Save All” tool to save all open screens or worksheets. Using the “Save All” tool is the same as selecting the “File... Save All” menu. Also, the “Save All” function is only available when you modify a screen or worksheet.



Cut Tool

Cut

Click the “Cut” tool to remove a selected object from the screen/worksheet and store it on the clipboard, replacing any previously stored selections on the clipboard. You can then use the “Paste” tool to move the cut object to another location on the same screen or to another screen. Using the “Cut” tool is the same as selecting the “Edit... Cut” menu or typing the <Ctrl> + <X> key combination.



Copy Tool

Copy

Click the “Copy” tool to duplicate a selected object and store it on the clipboard. You can then use the Paste tool to move the copied object to another location on the same screen or to another screen. Using the “Copy” tool is the same as selecting the “Edit... Copy” menu or typing the <Ctrl>+<C> key combination.



Paste Tool

Paste

Click the “Paste” tool to place the contents of the clipboard into upper left corner of the active screen. You can Paste a cut or copied object multiple times to a several screens/work-sheets. Using the “Paste” tool is the same as selecting the “Edit... Paste” menu or typing the <Ctrl> + <V> combination.



Undo Tool

Undo

Click the “Undo” tool to cancel the last action performed while working on a screen or worksheet. You can cancel up to 20 actions taken before your last action. The actions in object properties do not increase Undo steps. Using the “Undo” tool is the same as selecting the “Edit... Undo” menu or typing the <Ctrl>+<Z> key combination.



Project Settings Tool

Project Settings

Click the “Project Settings” tool to display the “Project Settings” dialog box. Using the “Project Settings” tool is the same as selecting the “Project... Settings”.



Zoom Tool

Zoom

Click the “Zoom” tool to change the cursor to the zoom mode. Click and drag the mouse on the screen to select the area where you want to zoom. Right-click on the screen to change the cursor to selection mode again. You can also select the zoom scale from the “Zoom” combo-box.



Figure 6-10 Zoom drop-down selection

6.3.6 Execution Control Toolbar

The Execution Control toolbar provides tools (shortcuts) that duplicate functionality found on the “Project” menu.



Figure 6-11 The “Execution” toolbar

The tools in the Execution Control toolbar are described below.

Test Display

Click the “Test Display” tool to run in test display mode, which allows you to configure an application while viewing graphical dynamics on-line in the development environment. Running in test display mode does not enable the Command and Text I/O dynamic or execute worksheets. Using the “Test Display” tool is the same as selecting the “Project... Test Display” menu.



Test Display Tool

Stop Test Display

Click the “Stop Test Display” tool to stop running in test display mode. Using the “Stop Test Display” tool is the same as selecting the “Project... Stop Display Test” menu.



Stop Test Display Tool



Run Application Tool

Run Application

Click the “Run Application” tool to launch any runtime modules specified as Automatic. Using the “Run Application” tool is the same as selecting the “Project... Run” menu.



Stop Application Tool

Stop Application

Click the “Stop Application” tool to stop all runtime tasks. “Stop Application” affects the application on the target station. Be sure you know which target station is configured (local or remote) before executing Stop Application. Using “Stop Application” is the same as selecting the “Project... Stop” menu.

6.3.7 Using the Bitmap Toolbar

Use the Bitmap toolbar to access the Bitmap Screen Editor tools. (This toolbar is available only when the Background Picture layer is active. You can enable the Background Picture layer in the “Screen Attributes” dialog box.)



Figure 6-12 The “Bitmap” toolbar



Select Area Tool

The tools in the Bitmap toolbar are described below.

Select Area

Click the “Select Area” tool to select an area within the Bitmap Screen Editor.



Flood Fill Tool

Flood Fill

Click the “Flood Fill” tool, then click on the screen to paint the surrounding area with the color you specified with the “Fill Color” tool.



Fill Color Tool



Pixel Editing Tool

Pixel Editing

Click the “Pixel Editing” tool to open an “Edit Image” dialog box, where you can draw detailed bitmaps, pixel by pixel.

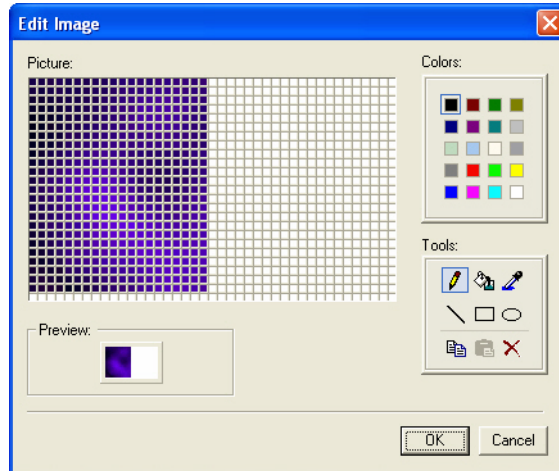


Figure 6-13 The “Edit Image” dialog box



Erase Area Tool

Erase Area

Click the “Erase Area” tool to remove a selected area from the screen.



Change Colors Tool

Change Colors

Click the “Change Colors” tool to change the transparent fill color for a selected area. Before you can use this tool, you should have already specified a fill color (“Fill Color” tool), selected a transparent color (“Select Transparent Color” tool), and defined the area to fill (“Select Area” tool).



Select Transparent Color Tool

Select Transparent Color

Click the “Select Transparent Color” tool to specify a transparent color (referenced by the “Change Colors” tool).



Toggle Transparent Color Tool

Toggle Transparent Color

Click the “Toggle Transparent Color” tool to cause the color selected using the Select Transparent Color tool to become transparent for bitmaps selected in the Bitmap Screen Editor. You can use the Copy (<Ctrl>+<C>) and Paste (<Ctrl>+<V>) commands to exchange bitmap pictures between ScreenView’s Bitmap Screen Editor and any other bitmap editor (for example, Paint Brush).

6.3.8 Using the Dynamic Objects Toolbar

Use the Dynamic Properties toolbar to apply dynamics to objects or a group of objects. Dynamics enable you to modify object properties on the fly (during runtime) according to tag values. Some dynamics also enable you to execute commands or insert values (set points) to the tags.



Figure 6-14 The “Dynamic Properties” toolbar

The tools in the Dynamic Properties toolbar are described below.

Command Property



Command Property Tool

Click the “Command” tool to add the command dynamic to a selected object or group of objects. The command dynamic lets operators click on the object or press a pre-defined key to execute the command at runtime. Double-click on the object to view its object properties.

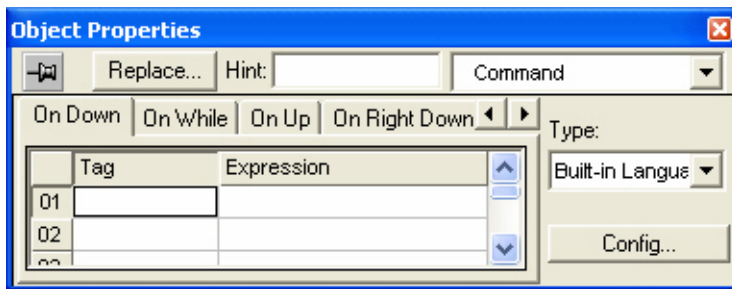


Figure 6-15 Object Properties: Command

The Command dynamic provides one tag for each one of the events supported by it. Notice that more than one event can be configured simultaneously for the same Command dynamic:

- “On Down” executes the command/script once when the user clicks on the object with the left mouse button.
- “On While” keeps executing the command/script continuously while the mouse pointer is pressed on the object. The period (in milliseconds) of execution for the command/script is set in the “Rate” field from the “Configuration” dialog box, except for the VBScript option, which is executed as fast as possible.
- “On Up” executes the command/script once when the user releases the left mouse button on the object.
- “On Right Down” executes the command/script once when the user clicks on the object with the right mouse button.
- “On Right Up” executes the command/script once when the user releases the right mouse button on the object.

- “On Double Click” executes the command/script once when the user double-clicks on the object with the left mouse button.



Think & Do treats the touch-screen actions the same way it treats the mouse pointer actions. In other words, it is transparent for Think & Do if an event was triggered by a touch-screen interface or by a regular mouse pointer.

- The events On Right Down, On Right Up and On Double Click are not supported by CEView applications (running on the Windows CE operating system).
- When creating an application for a touch-screen device, it is important to keep in mind that events On Right Down and On Right Up cannot be triggered on such devices.
- “Key” is the shortcut used to trigger the events On Down, While Down and On Up using a keyboard. This option is especially useful when creating applications for runtime devices that do not provide a mouse or touch-screen interface – the keyboard is the only physical interface available to interact with the application during the runtime.
- “Shift”, “Ctrl”, or “Alt” boxes create a combination key, meaning the <Shift>, <Ctrl>, or <Alt> key must be pressed with the key specified in the drop-down list.
- “Key Modifier” dialog box opens the “Key Modifier” dialog box, which enables you to modify your combination keys. You can choose Left, Right or Left or Right to specify the position on the keyboard of the <Shift>, <Ctrl> or <Alt> key in the combination key. If you choose Left or Right, the command will be executed any time either of these keys is pressed in combination with the key specified in the drop-down list.
- “Config” launches the “Configuration” dialog box, where you configure the dynamic command object.

Hyperlink Property



Hyperlink
Property
Tool

Click the “Hyperlink” tool to add the hyperlink property to a selected object or group of objects. Applying this property allows you to click on the object(s) during execution to launch the default browser and load the specified URL.

Double-click on the object to open the “Object Properties” dialog box.

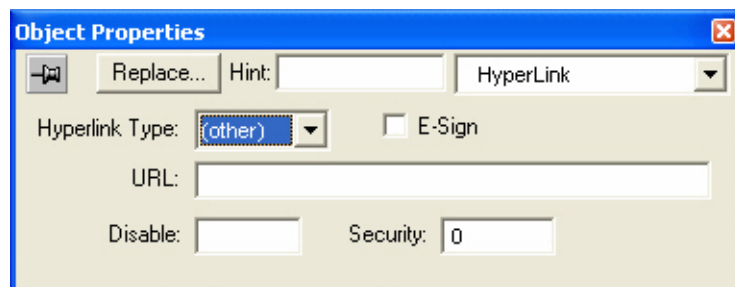


Figure 6-16 Object Properties: Hyperlink

You can use this dialog box to specify the following parameters:

- “Hyperlink Type” combo-box selects a URL protocol from the list. Think & Do uses this protocol when it loads the URL.
- “E-Sign” check box, when checked, prompts the user to enter the Electronic Signature before executing the dynamic.

- "URL" field is the URL address you want to load (for example: "http://www.phoenixcontact.com").

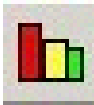


You are not required to enter the protocol type in the URL field. When you select a protocol type from the "Hyperlink Type" list, ScreenView automatically adds the protocol's prefix to the URL address.

- "Disable" field specifies a value (greater than zero) that disables the hyperlink command property for the selected object(s).
- "Security" field specifies a security level for the object(s). If a user logs on, and does not have the required security level, Think & Do disables the hyperlink command for the object(s).

BarGraph Property

Click the "BarGraph" tool to add bar graph properties to a selected object, then double-click on the object to open the "Object Properties" dialog box.



BarGraph Property Tool

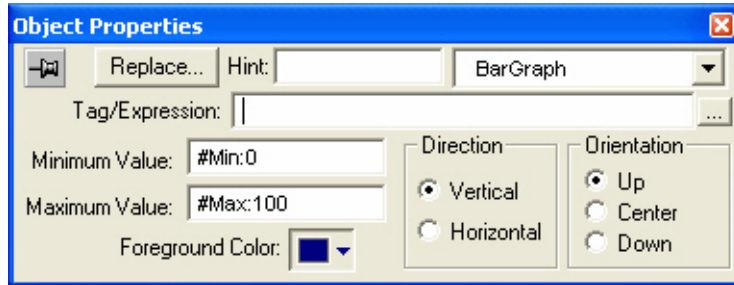


Figure 6-17 Object Properties: BarGraph

Use the "Object Properties" dialog box to specify the following parameters:

- "Tag/Expression" field specifies a tag or expression that evaluates the bar graph level. You also can click the icon to browse your directories for an existing tag or expression.
- "Minimum Value" field specifies a numeric constant or a tag value that defines the minimum value used to calculate the height (if vertical) or width (if horizontal) of the bars.
- "Maximum Value" field specifies a numeric constant or a tag value that defines the maximum value used to calculate the height (if vertical) or width (if horizontal) of the bars.
- If you do not specify a value for this field, ScreenView opens a dialog box requesting you confirm creation of the tag.

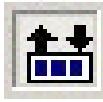


ScreenView also allows you to enter constants in tag/numeric value fields. Constant values (defined by the # character) are equivalent to numeric values, except that constants display in the "Tag Replace" dialog box. You may find constants useful for documentation purposes or for creating generic objects.

For example: #Name:100.

Where the value (100) following the semicolon (;) is the constant, and Name is a constant mnemonic only and not added to database.

- "Foreground" Color specify a fill color for the bars by clicking the combo-box button. When the "Color" dialog box displays, click on a color to select it, and then close the dialog box.
- "Direction" area specifies the "Vertical" or "Horizontal" direction of the bar graph.



Text Property Tool

- “Orientation” area specifies the “Up”, “Center”, or “Down” orientation of the maximum and minimum values when drawing the bars.

Text I/O Property

Click the “Text I/O” tool to add the dynamic input or output text property to a selected Text object. Applying the Text I/O property allows you to insert and display tag values in real time if you are using the keyboard or on-screen keypad to run an application.



You are not required to enter the protocol type in the URL field. When you select a protocol type from the “Hyperlink Type” list, ScreenView automatically adds the protocol’s prefix to the URL address.



You can apply this dynamic property only to text objects containing the “#” character. (Each “#” represents one character.)

Double-click on the object to open the “Object Properties” dialog box.

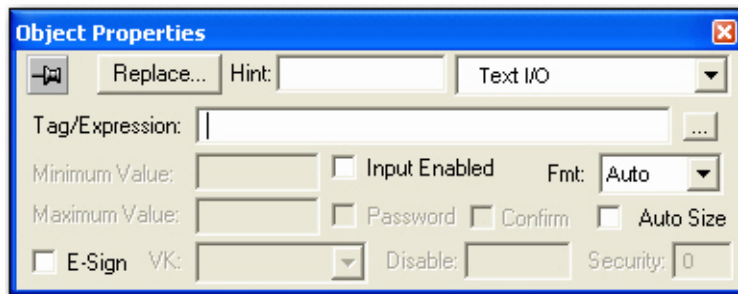
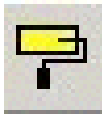


Figure 6-18 Object Properties: Text I/O

Use this dialog box to specify the following parameters:

- “Tag/Expression” text field specifies one of the following:
 - A tag on which to perform an input or output operation
 - An expression on which to perform an output operation only
- You can click the icon to browse your directories for an existing tag or expression.
- “Input Enabled” check box, when selected, allows data entries. Disable (uncheck) the option and this dynamic only executes the data outputs.
- “Confirm” check box, when selected, requires users to confirm any new values set during runtime.
- “Minimum Value” defines a minimum value for the tag associated with this text object. A user will not be permitted to input a number lower than this value.
- “Maximum Value” defines a maximum value for the tag associated with this text object. A user will not be permitted to input a number greater than this value.
 - “Password” check box, when selected, this option to hide password text entries by replacing the text with asterisks (*).
 - “Fmt” combo-box selects a format for the input/output field.
 - “Disable” field disables the tag’s data input property when the value entered is greater than zero.
 - “Security” field specify a value in this field to define the security level for a specific data input object (as defined in the Security section).
- “E-Sign” check box, when selected, prompts the user to enter the Electronic Signature before changing the tag value.

- “VK” select the Virtual Keyboard type used for this object. You need to enable the Virtual Keyboard option “Project Settings... Runtime Desktop” tab before configuring the Virtual Keyboard for this interface.



Colors Property Tool

Colors Property

Click the “Colors” tool to add the color change property to a selected object. The Colors dynamic allows you to modify the color of a static object during the runtime based on the value of a tag or expression.

Double-click on the object to open the “Object Properties” dialog box.

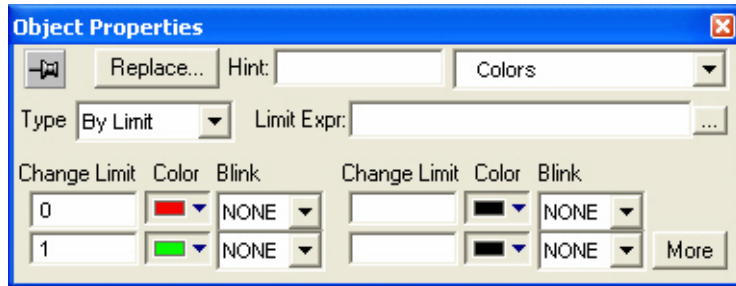


Figure 6-19 Object Properties: Colors

You can use this dialog box to specify the following parameters:

- “Type” determines the mode in which this dynamic works:
 - “By Limit” specifies up to four limits (Change Limit) for this dynamic and a color for each limit. When the value of the tag or expression configured in the “Tag/Expr” field reaches the limits, the color associated with the respective limit is applied to the object.
 - “By Color” specifies the code of the color that must be applied to the object directly in the “Tag/Expr” field. Using this code, you can apply any color supported by your device to the object.



You can configure the “RGBColor()” function in the “Tag/Expr” field when Type = By Color. This allows you to configure the color by its RGB codes. See Color Interface for a table with the codes for the most commonly used colors.

- “Limit Expr” field specifies the name of a tag or expression you want to monitor. When “Type” is “By Limit”, Think & Do compares the result of the tag/expression with the specified Change Limits to determine the proper color for the selected object. When “Type” is “By Color”, the result of this field sets the color that will be applied to the object.

- “Change Limit” field specifies a limit value (a numeric constant or tag) for the color change. The numbers must be configured in ascendant order according to the following sequence of the fields displayed on the “Object Properties” dialog box window: Upper-left, lower-left, upper-right and lower-right field. If you click on the More button, you can configure up to 16 different limits for the color dynamic.

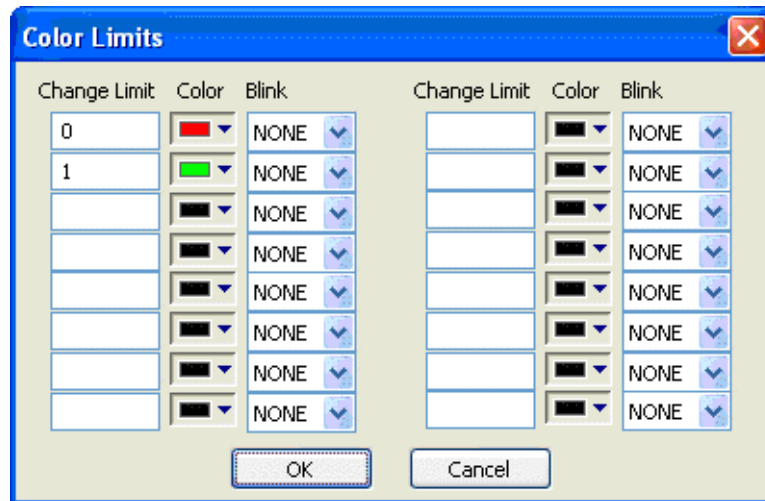


Figure 6-20 The “Color Limits” dialog box

- “Color” combo box associates a color with each color change limit. Clicking this button displays a pop-up with predefined colors. Select a predefined color by clicking it, or click the “More Colors...” to display the “Color” dialog box. When the “Color” dialog box opens, click a color to select it, or use the “Colors... Custom” tab to define a color. Click the “OK” button to close the dialog box and select the color. The color pop-up also has a “Fill Effects...” link that displays the “Fill Effects” dialog box. Use this dialog box to define dynamic fill.
- “Blink” combo-box specifies whether the color change will blink, and how fast it will do so.



The following fields are automatically disabled (grayed out) when “Type” = “By Color”:
“Change Limit”, “Color” and “Blink”.



Position Property Tool

Position Property

Click the “Position” tool to specify when and where to display an object, using the specified tag values.

Double-click on the object to open the “Object Properties” dialog box.

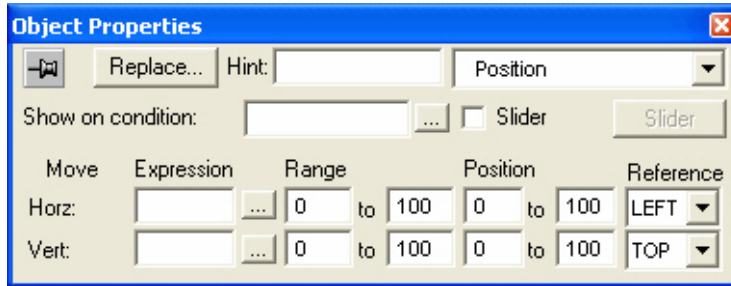
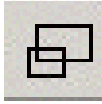


Figure 6-21 Object Properties: Position

You can use this dialog box to specify the following parameters:

- “Show on condition” field:
 - Type a number greater than zero (or leave the field blank) to make the object visible.
 - Type a zero or a negative number to hide the object.
- “Slider” check box enables (checked) the object to act like a slider (which means you can drag the object to apply corresponding values to the tags).
- “Expression” fields specifies the tag associated with the object, allowing the object to move horizontally and vertically throughout the screen.
- “Range” fields specifies upper and lower limits for the tag values, allowing the object to move throughout the screen according to the value of the tag within this range.
- “Position” fields specifies how much change in position (in pixels) you can move an object on the screen according to the established condition. You can enter negative values in the second field (destination position).
- “Reference” combo-boxes select a reference point to be used while moving the object on the screen. Specifying this option is necessary only if you want to resize the object as you move it.
 - “Left” is the left corner of the object
 - “Right” is the right corner of the object
 - “Center” is the center of the object
 - “Top” is the upper corner of the object
 - “Bottom” is the lower corner of the object



Size Property Tool

Size Property

Click the “Size” tool to increase or decrease the size of a selected object or symbol. Double-click on the object/symbol to open the “Object Properties” dialog box.

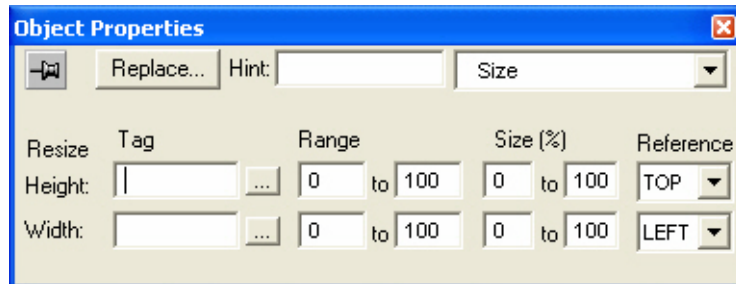


Figure 6-22 Object Properties: Resize

Use the “Object Properties” dialog box to specify the following parameters:

- “Tag” fields specify the “Height” and “Width” fields to increase or decrease the object’s horizontal and vertical size.
- “Range” fields specify upper and lower tag limits, which Think & Do uses to increase and decrease the object size.
- “Size (%)” fields specify a percentage range, which Think & Do uses to increase and decrease the object size.
- “Reference” combo-boxes select one of the following reference points to determine how the object increases size horizontally and vertically.
 - “Left” is the left corner of the object
 - “Right” is the right corner of the object
 - “Center” is the center of the object
 - “Top” is the upper corner of the object
 - “Bottom” is the lower corner of the object



Size Property Tool

Rotation Property

Click the “Rotation” tool to rotate a line. Double-click on the line to open the “Object Properties” dialog box.

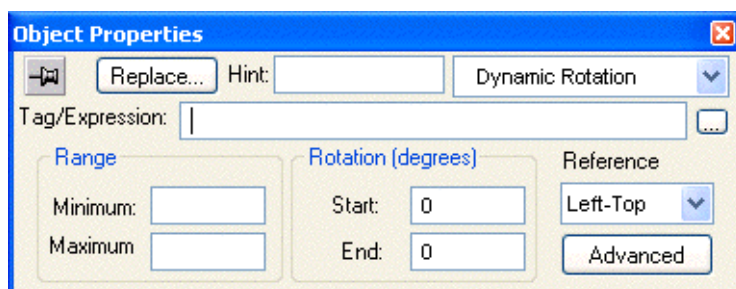


Figure 6-23 Object Properties: Rotation Property

- Use this dialog box to specify the following parameters:
- “Tag/Expression” field specifies a tagname or expression to associate with the selected line. Think & Do reads the value represented on the screen using this variable or expression.
- “Range” area specifies Minimum and Maximum tag values used to move the line throughout the screen according to the established condition.
- “Rotation (degrees)” area specifies starting and ending values to specify how many degrees to rotate a line on the screen (rotation dynamic). You can rotate a line up to 360 degrees.
- “Reference” combo-box specifies one of the following as a reference point on which to rotate the object throughout the screen:
 - “Left-Top” is the upper-left corner of the object
 - “Right-Bottom” is the lower-right corner of the object
 - “Center” is the center of the object.

6.3.9 Using the Mode Toolbar

The Mode toolbar provides tools for general screen editing.

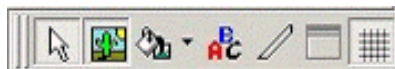


Figure 6-24 The “Mode” toolbar

The tools in the Mode toolbar are described below.



Selection Tool

Selection

Click the “Selection” tool to display a cursor that allows you to select or move objects on the screen.



Bitmap Editor Tool

Bitmap Editor

Click the “Bitmap Editor” tool to switch between the two, basic editing layers:

- Objects layer specifies the layer on which you create the dynamic objects for your screen.
- Background Picture layer is the static background layer of the same screen.

You automatically disable the “Bitmap Editor” tool when you uncheck (disable) the “Enable Background” box (BMP-type only) on the “Screen Attributes” dialog box.



Fill Color Tool

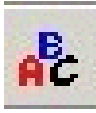
Fill Color

Click the “Fill Color” tool to specify a default fill color for the following objects:

- Closed Polygons
- Ellipses
- Rounded Rectangles
- Rectangles



To save development time, select several objects (of any type specified in the preceding list) and use the “Fill Color” tool to specify a default fill color for all of them at once.



Fonts
Tool

Fonts

Click the “Fonts” tool to specify the font and color for selected Text objects, or to specify a default font and color for new text objects.



To save development time, select several Text objects and use the “Fonts” tool to specify font and color settings for all of the objects at once. (You cannot use this function for grouped Text objects however.)



Line Color
Tool

Line Color

Click the “Line Color” tool to specify a line color for selected objects or to set a default color for new objects, including the following:

- Open Polygons
- Closed Polygons
- Lines
- Ellipses
- Rounded Rectangles
- Rectangles

When you click the “Line Color” tool, the “Line Selection” dialog box (see Figure 6-25) displays. Use this dialog box to specify line styles and color for the selected objects.

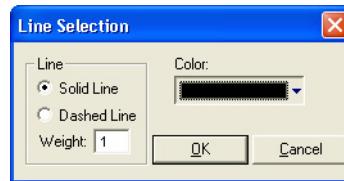
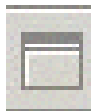


Figure 6-25 The “Line Selection” dialog box



To save development time, you can select several of the preceding objects and use the “Line Color” tool to specify a line color for all of the objects at once.



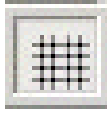
Background
Color
Tool

Background Color

Click the “Background Color” tool to specify a background color for the screen.



You automatically disable this tool when you check the “Enable Background” box on the “Screen Attributes” dialog box.



Grid Tool

Grid

Click the “Grid” tool to specify whether to show or hide the grid on the screen editor.



You can use the “Grid” dialog box to configure the default settings for a grid. To open this dialog box, right-click on the screen and select “Grid Settings” when the pop-up menu displays.

6.3.10 Using the Static Objects Toolbar

The Static Objects toolbar provides the following tools, which you can use to create polygons, rectangles, lines, and other objects for your screen.



Figure 6-26 The “Static Objects” toolbar

The tools in the Static Objects toolbar are described below.

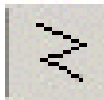
Open Polygon Object

Click the “Open Polygon” tool to draw an open polygon with a border in the specified foreground color.

To draw an open polygon in the drawing area:

1. Click the left mouse button to set the starting point of the polygon.
2. Move the cursor to a new location and click again to place the second vertex.
3. Repeat this process until you create the desired polygon shape.
4. Double-click to stop drawing the polygon.

To view the object properties, double-click on the polygon object and the “Object Properties” dialog box is displays as follows.



Open Polygon Tool

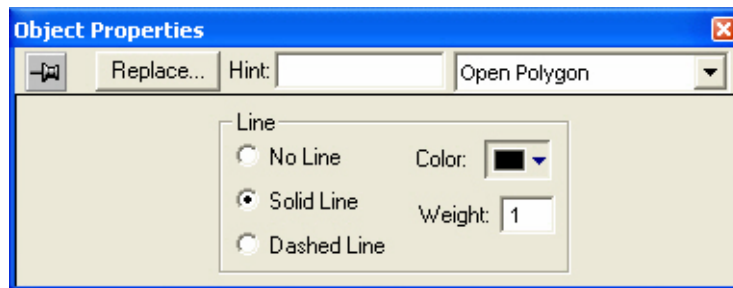


Figure 6-27 Object Properties: Open Polygon

Use the “Object Properties” dialog box to specify the following parameters for the polygon:

- “Line” specifies a border line style by clicking the “No Line”, “Solid Line”, or “Dashed Line” button.
- “Color” specifies a border line color by clicking the “Color” button. When the “Color” dialog box opens, click on a color to select it and then close the dialog box.
- “Weight” specifies the borderline width (in pixels) by typing a number representing the line width into the text box.



Closed Polygon Tool

Closed Polygon Object

Click the “Closed Polygon” tool to draw a closed polygon, using a border in the specified foreground color.

To draw a closed polygon in the drawing area:

1. Click the left mouse button to set the starting point of the polygon.
2. Move the cursor to a new location and click again to place the second point.
3. Repeat this process until you create the desired polygon shape.
4. Double-click or right-click to stop drawing the polygon.
5. To view the object properties, double-click on the polygon object.

The “Object Properties” dialog box is displays as follows.

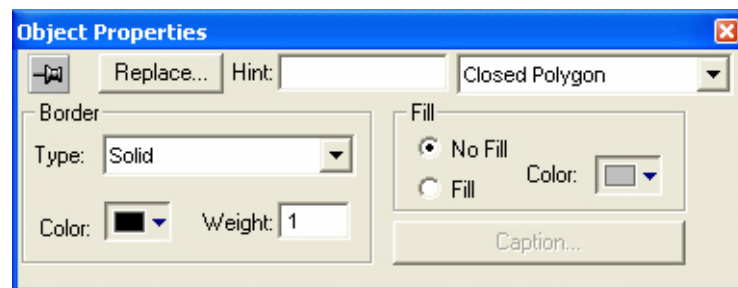


Figure 6-28 Object Properties: Closed Polygon

Use the “Object Properties” dialog box to specify the following parameters for the polygon:

- “Line” specifies a border line style by clicking the “No Line”, “Solid Line”, or “Dashed Line” button.
- “Color” specifies a border line color by clicking the “Color” button. When the “Color” dialog box opens, click a color to select it and then close the dialog box.
- “Weight” specifies the borderline width (in pixels) by typing a number representing the line width into the text box.
- “Fill” specifies whether the polygon is filled, click “No Fill” or “Fill”.

If you enable the “Fill” option, you can specify a fill Color by clicking on the “Color” button. When the “Color” dialog box displays, click a color to select it and close the dialog box.



Line Tool

Line Object

Click the “Line” tool to draw an orthogonal line in the drawing area, as follows:

1. Click the left mouse button to set the starting point of the line.
2. Drag the cursor to adjust the line size.
3. Click again to place the object.

- 4. To view the object properties, double-click on the object.
The “Object Properties” dialog box displays as follows.

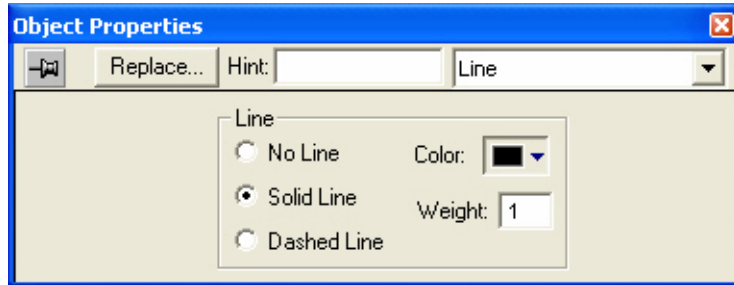


Figure 6-29 Object Properties: Line

Use the “Object Properties” dialog box to specify the following parameters for the orthogonal line:

- “Line” specifies a line style by clicking the “No Line”, “Solid Line”, or “Dashed Line” button.
- “Color” specifies a line color by clicking the “Color” button. When the “Color” dialog box opens, click a color to select it and then close the dialog box.
- “Weight” specifies the line width (in pixels) by typing a number representing the line width into the text box.



Ellipse Tool

Ellipse Object

Click the “Ellipse” tool to draw ellipses, chords, arcs, and rings (see the following figures).

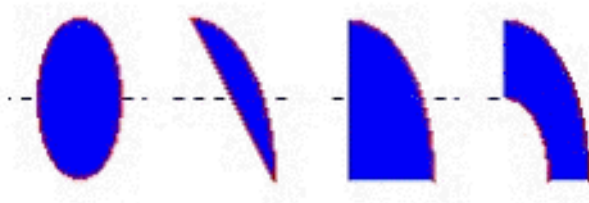


Figure 6-30 Oval, Chord, Arc, and Ring



The Ring style is particularly useful when you are creating plumbing drawings.

To create an ellipse, use the following steps:

1. Click in the drawing area and drag the mouse/cursor to create an oval shape.
2. Release the mouse button to stop drawing the oval.
3. Use the “Object Properties” dialog box to change the shape to a chord, arc, or ring.

4. Double-click on the object to view the “Object Properties” dialog box.

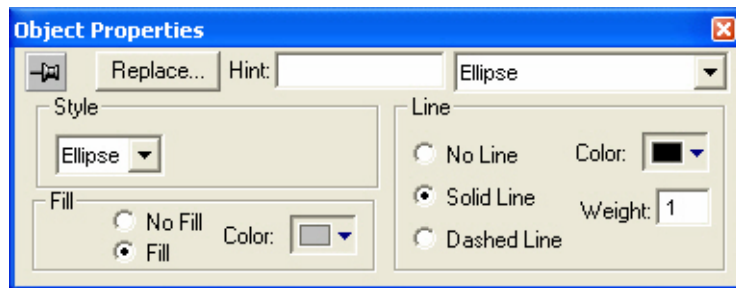


Figure 6-31 Object Properties: Ellipse

Use the “Object Properties” dialog box to specify the following parameters for the ellipse:

- “Style” specifies the object style by selecting “Ellipse”, “Arc”, “Chord”, or “Ring” from the drop-down list. Next, for “Arc”, “Chord”, or “Ring”, select “Left-Bottom”, “Left-Top”, “Right-Bottom”, or “Right-Top” from the “Style” list to choose the quadrant into which the ellipse is drawn.

For example to represent a half-circle pipe, create two Ring objects. Specify one as Left-Bottom and the other as Right-Bottom then join the two objects to create a half-pipe.

- “Fill” specifies whether the ellipse is filled, click “No Fill” or “Fill”.
If you select the “Fill” option, specify a fill color by clicking on the “Color” rectangle. When the “Color” dialog box displays, click on a color to select it and close the dialog box.
- “Line” specifies a line style for the ellipse border by clicking the “No Line”, “Solid Line”, or “Dashed Line” button.
- “Color” specifies the ellipse borderline color by clicking the “Color” button to open the “Color” dialog box. Click the color to select it, then close the dialog box.
- “Weight” specifies a line width for the ellipse border by typing a number representing the line width (in pixels) into the text box provided.



Rounded
Rectangle Tool

Rounded Rectangle Object

Click the “Rounded Rectangle” tool to draw rounded rectangles (empty or filled), as follows:

1. Click in the drawing area and drag the mouse/cursor to create the rectangle.
2. Release the mouse button to stop drawing the object.
3. Double-click on the object to view the “Object Properties” dialog box.



You cannot use the rounded rectangle tool to create a bar graph for Windows CE applications.



A rounded rectangle has one extra tracker in the lower left corner, which enables you to modify the arc angle.



Figure 6-32 Object Properties: Rounded Rectangle

Use the “Object Properties” dialog box to specify the following parameters for the orthogonal line:

- “Line” specifies a borderline style by clicking the “No Line”, “Solid Line”, or “Dashed Line” button.
- “Color” specifies a borderline color by clicking the “Color” button to open the “Color” dialog box. Click the color to select it and then close the dialog box.
- “Weight” specifies a borderline width by typing a number representing the line width (in pixels) into the text box provided.
- “Fill” specifies whether the rectangle is filled by clicking “No Fill” or “Fill”.
If you select the “Fill” option, specify a fill color by clicking on the “Color” button. When the “Color” dialog box displays, click a color to select it and close the dialog box.
- “Color” specifies a fill color by clicking the Color button to open the “Color” dialog box. Click a color to select it, then close the dialog box.
- “Caption” is not enabled for this object.



Button Tool

Button Object

Click the “Button” tool to create custom-sized buttons, as follows:

1. Click in the drawing area and drag the mouse/cursor to create the button shape.
2. Release the mouse button when the button is the size you want.
3. Double-click on the object to view the “Object Properties” dialog box.

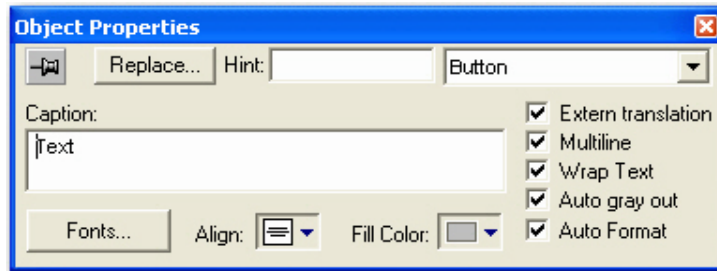


Figure 6-33 Object Properties: Button

Use the “Object Properties” dialog box to specify the following parameters for the button:

- “Caption” specifies a caption by typing the text into the text box.
- “Fonts” specifies a font style for the caption by clicking the “Fonts” button.
 - When the “Fonts” dialog box displays, specify the following parameters:
 - “Font” (typeface)
 - “Font style”
 - “Size”
 - “Effects”
 - “Color”
 - “Script” style
- “Weight” specifies a border line width by typing a number representing the line width (in pixels) into the text box.
- “Extern translation” (optional) enables translation of the button when a translation file is specified via scripting.
- “Align” specifies the alignment for the caption of the button.
- “Fill Color” specifies the color for the button.
- “Multiline” permits the caption of the button to be shown in more than one line, when checked.
- “Wrap Text”, when checked, the object automatically wraps the text when necessary.
- “Auto gray out” turns the caption of the button to gray when the Command dynamic applied to the button is disabled by the “Disable” field.



Text Tool

Text Object

Click the “Text” tool to create text objects, as follows:

1. Click in the drawing area. When a cursor displays, you can type a line of text.
2. After entering the text string, double-click on the new text object to view the “Object Properties” dialog box.

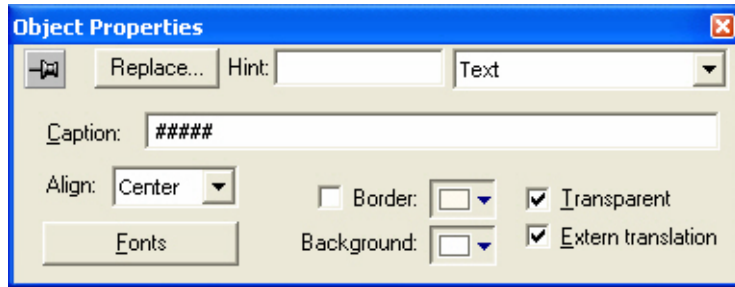
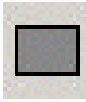


Figure 6-34 Object Properties: Text

Use the “Object Properties” dialog box to specify the following orthogonal line parameters:

- “Caption” specifies a text string by typing a caption in the text box.
- “Align” aligns the text by selecting “Left”, “Center”, or “Right” from the combo-box.
- “Fonts” specifies a font style for the text by clicking the “Fonts” button. When the “Fonts” dialog box displays, you can specify the following parameters:
 - “Font” (typeface)
 - “Font style”
 - “Size”
 - “Effects”
 - “Color”
 - “Script”
 - “Border” specifies a text border by clicking the Border box.To select a border color, click the Color rectangle. When the “Color” dialog box displays, click a color to select it, then close the dialog box.
- “Background” specifies a background color by clicking the “Color” button. When the “Color” dialog box displays, click a color to select it, then close the dialog box.
- “Transparent” check box enables (check) this option to not show the background color. Disable (uncheck) to show the background color.
- “Extern translation” (optional) enables translation of the text when a translation file is specified via scripting.



Rectangle Tool

Rectangle Object

Click the “Rectangle” tool to create rectangles, as follows:

1. Click in the drawing area and drag the mouse/cursor to draw the rectangle.
2. Release the mouse button when the rectangle is the size you want.
3. Double-click on the object to view the “Object Properties” dialog box.



Figure 6-35 Object Properties: Rectangle

Use the “Object Properties” dialog box to specify the following parameters for the orthogonal line:

- “Type” specifies a border line style by clicking on “None”, “Solid”, “Dashed”, “Etched”, “Raised”, or “Sunken”.
- “Color” specifies a border line color by clicking the “Color” button to open the “Color” dialog box. Click the color to select it, and then close the dialog box.
- “Weight” specifies a border line width by typing a number representing the line width (in pixels) into the text box provided.
- “Fill” specifies whether to fill the rectangle by clicking “No Fill” or “Fill”.
- If you select the “Fill” option, specify a fill color by clicking on the Color rectangle. When the “Color” dialog box displays, click a color to select it and close the dialog box.
- “Color” specifies a fill color by clicking the Color button to open the “Color” dialog box. Click a color to select it, then close the dialog box.
- “Caption” opens the “Caption” dialog box. This is where you can edit the text that can be written inside the rectangle object:

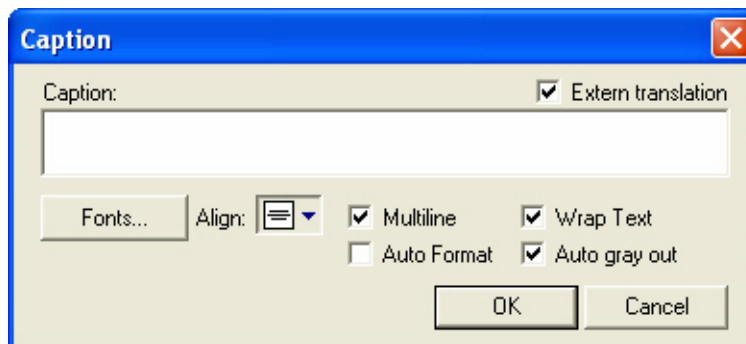


Figure 6-36 The “Caption” dialog box

- “Caption” specifies the text that you want to show inside the rectangle object in this text box.

- “Extern translation” (optional) enables translation of the caption when a translation file is specified via scripting.
- “Fonts” specifies a font style for the caption by clicking the “Fonts” button.
- “Align” specifies the alignment for the caption of the rectangle.
- “Fill Color” specifies the fill color for the rectangle.
- “Multiline” permits the caption of the rectangle to be shown in more than one line, when checked.
- “Wrap Text”, when checked, the object automatically wraps the text when necessary.
- “Auto gray out” turns the caption of the rectangle to gray when the Command dynamic applied to the rectangle is disabled by the “Disable” field or due to the Security System.

6.3.11 Using the Active Objects Toolbar

The Active Objects toolbar provides the following tools, which you can use to create dynamic objects. Active objects typically require more parameters than static objects.

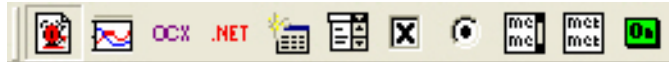


Figure 6-37 The “Active Objects” toolbar



Alarm Event Tool

The tools in the Active Objects toolbar are described below.

Alarm/Event Control Object

Click the “Alarm/Event Control” tool to add an alarm or event control object to your application screen.



When acknowledging an alarm, the Alarm Control object sends a message to the Alarm task with the following information: Tagname, Type, User and Station. This is a solution to control acknowledged alarms from a Web Thin Client.

To create and configure an alarm control object:

1. Click the “Alarm/Event Control” tool.
2. Click in the display, and drag the mouse to create and adjust the object’s shape.
3. Double-click on the object to open the following “Object Properties” dialog box.

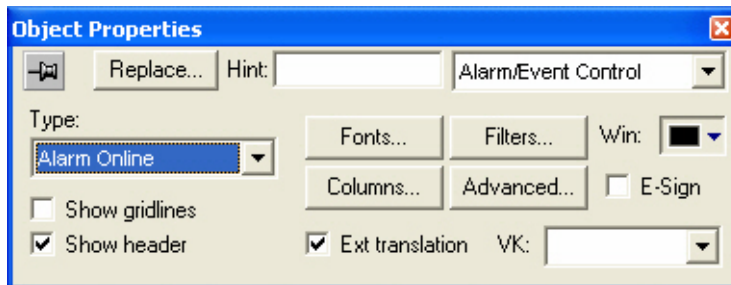


Figure 6-38 Object Properties: Alarm/Event Control

4. Use the “Type” parameter to select an alarm or event. The choices are:
 - “Alarm Online” displays current alarm messages.
 - “Alarm History” displays alarm messages from the history database.
 - “Alarm History+Event” displays alarm messages from the history database and shows the current event.
 - “Event” displays the current event.

Available configuration options depend on whether the “Type” selected is an alarm or event. See “Configuring an Alarm”, below, or “Configuring an Event” on page 6-74.

Trend Control Object



The Trend Control object displays data points (values) from different data sources in a graphic format. Click the “Trend Control” tool to add it to your application screen. Double-click on the object to launch its “Object Properties” dialog box, as follows:

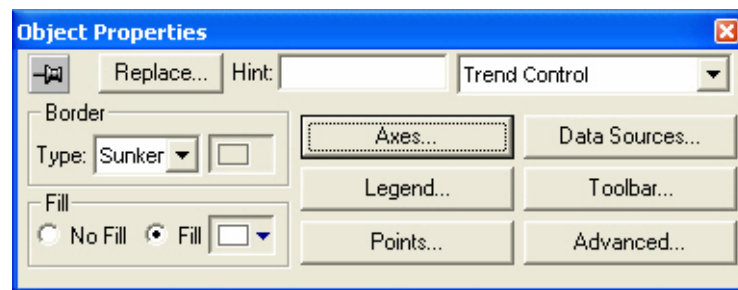


Figure 6-39 Object Properties: Trend Control

The main features provided by the Trend Control object are:

- Display of multiple pens simultaneously
- Support for different Data Sources, such as Tag, Batch, Database and Text File
- Capability to generate X/Y graphs from the configured data sources (please refer to Appendix A for an example of an X/Y chart).
- Simultaneous display of an unlimited number of data points. This feature might be limited by the hardware used since available memory and performance will vary.
- Built-in toolbar, which provides interfaces for the user to interact with the Trend Control object during the runtime
- Built-in legend, which displays the main information associated to each pen linked to the object
- Zooming and auto-scaling tools
- Horizontal and vertical orientation

For information on configuring and using the Trend Control, see “Trend Control Development Interface” on page 6-74 and “Trend Control Runtime Interface” on page 6-87.



ActiveX Control Tool

ActiveX Control Object

Click the “ActiveX Control” tool to open the “Insert ActiveX Control” dialog box.

You use this dialog box place ActiveX components on your screen. When the dialog box opens (see Figure 6-40), it contains a list of all ActiveX components registered on your computer.

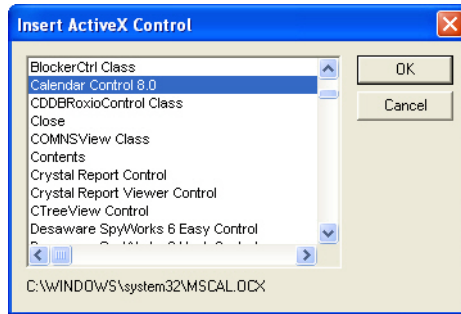


Figure 6-40 The “Insert ActiveX Control” dialog box

ActiveX controls are components designed according to a standard. Because ScreenView is an ActiveX container, you can configure and run ActiveX controls in Think & Do screens. ActiveX controls can provide the following interfaces:

- Properties are variables whose values can be read and/or written for the application (e.g. Object Color, FileName, URL, and so forth)
- Methods are functions from the ActiveX object that can be triggered by the application (e.g. open a dialog box, execute a calculation, and so forth)
- Events are internal messages that can trigger the execution of expressions in the application (e.g. Mouse_Click, Download_Completed, and so forth)

The name of the properties, methods, and events supported by each ActiveX depends on its own implementation.

There are two different ways to interface the application with the ActiveX control:

- By using the ActiveX functions XGet(), XSet() and XRun()
- OR
- By using the “Object Properties” dialog box to configure the object



When using ActiveX controls in your application, make sure the target station (runtime station) has the same ActiveX properly registered. The Think & Do application files include links to the ActiveX controls; however, the installation of these controls on the target station must be executed manually. Consult your ActiveX provider for further information about how to install.

When configuring applications with ActiveX for CEView, make sure that the ActiveX control used in the application is supported on the platform (Windows CE operating system and process type) where you intend to run the application. Consult your ActiveX provider for further information about the supported platforms.

Double-click the ActiveX control to open the “Object Properties” dialog box.

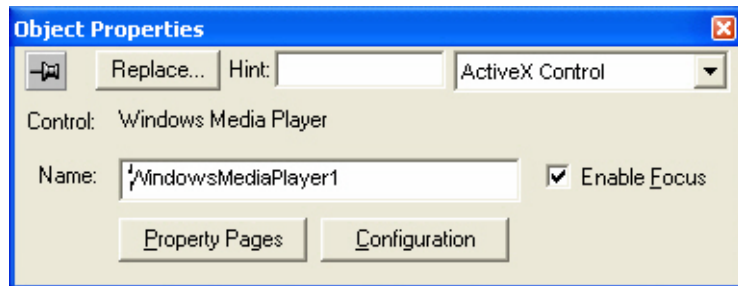


Figure 6-41 Object Properties: ActiveX Control

The “Object Properties” dialog box displays the name of the ActiveX control. Generally, each ActiveX control is either a *.dll or a *.ocx file registered in your local computer. You must assign a name (alias) to the ActiveX control, for the application in the “Name field (e.g. MyControl). This name is used to reference the object when configuring the ActiveX functions in the ScreenView scripting language.



Do not configure two ActiveX controls on the same screen with the same name. For instance, if you insert two “Windows Media Player” ActiveX controls on the same screen, and assign the name MyMP1 to one object (Name field), you cannot assign the same name to the second object on the same screen. You would have to assign the name MyMP2, for example, to the second object.

The “Property Pages” button opens the standard dialog box for configuring the Static Properties (if any). The layout and the options in this dialog box depend on the implementation of each ActiveX control. Use this interface to set properties that should not be changed during the runtime (fixed properties).

The “Configuration” button on the “Object Properties” dialog box opens dialog boxes that allow you to do the following:

- Associate tags to properties of the ActiveX object
- Trigger methods from the ActiveX object based on tag change
- Configure scripts, which are executed when Events from the ActiveX object occur

See “Configuring ActiveX Control Objects” on page 6-89 for information on how to configure these interfaces.



.NET Control Tool

.NET Control Object

Click the “.NET Control Object” tool on the Active Objects toolbar to open the “.NET Framework Components” dialog box, which you can use to place .NET Components on your screen. When the dialog box opens (as in the following figure), it contains a list of all .NET Components registered on your computer.

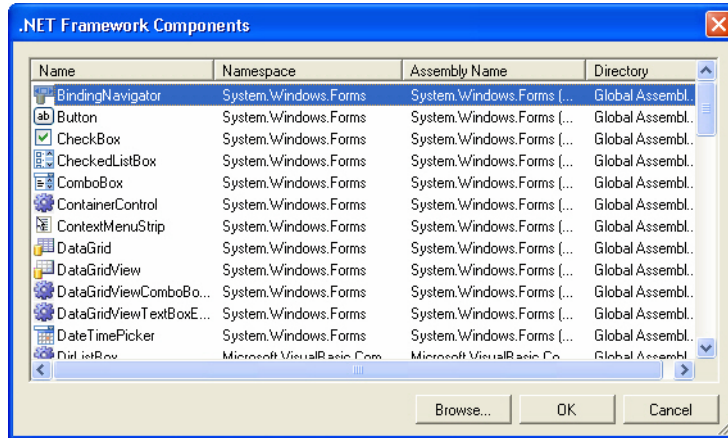


Figure 6-42 The “.NET Framework Components” dialog box

.NET Components are designed according to the Microsoft .NET Framework, which is a standard for modular programming technologies. Because ScreenView is a .NET container, you can configure and run .NET Components in Think & Do screens. .NET Components provide the following interfaces:

- Properties are variables whose values can be read and/or written for the application (e.g. Object Color, FileName, URL, and so forth)
- Methods are functions from the .NET Component that can be triggered by the application (e.g. open a dialog box, execute a calculation, and so forth)
- Events are internal messages that can trigger the execution of expressions in the application (e.g. Mouse_Click, Download_Completed, and so forth)

The name of the properties, methods, and events supported by each .NET Component depends on its own implementation.



When using .NET Components in your application, make sure the target station (runtime station) can support the same components and they are properly installed and registered. The Think & Do application files include links to the .NET Components; however, the installation of these components on the target station must be done separately. Furthermore, when .NET Components are used on screens open in remote Web Thin Clients, the .NET Components must also be manually installed on the Web Thin Client stations. The Microsoft Windows operating system installs a large selection of components by default, but additional components are offered by third-party providers. Consult your .NET Component provider for further information about how to install.

To insert a .NET Control Object, follow these steps:

1. Select a component from the “.NET Framework Components” dialog box.
2. Click the “OK” button to place it in your application screen.

- Once the component (object) is placed, double-click it to open its “Object Properties” dialog box.

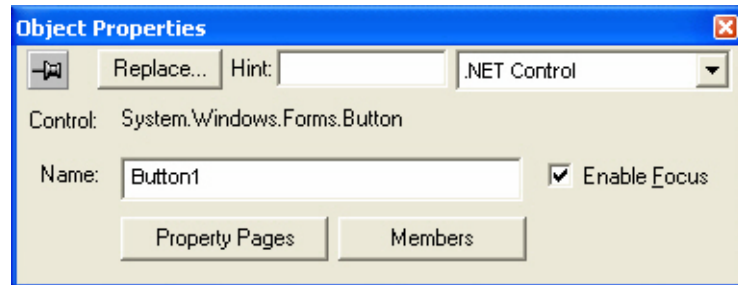


Figure 6-43 Object Properties: .NET Control

The “Object Properties” dialog box displays the name of the .NET Component. You must assign a name (alias) for the component in the “Name” field (e.g. CheckBox1). You use this name to reference the component when using scripting languages (VBScript and built-in ScreenView scripting).



Do not configure two .NET Components on the same screen with the same name. For instance, if you place two CheckBox components on the same screen and assign the name CheckBox1 to one object (Name field), you cannot assign the same name to the second object on the same screen. You would have to assign the name CheckBox2, for example, to the second object.

The “Property Pages” button opens the standard window for configuring the Static Properties (if any). The layout and the options in this dialog box depend on the implementation of each .NET Component. Use this interface to set properties that should not be changed during the runtime (fixed properties).

The “Members” button in the “Object Properties” dialog box opens additional dialog boxes that allow you to do the following:

- Associate tags to properties of the .NET Component (jump)
- Trigger methods from the .NET Component based on tag change (jump)
- Configure scripts, which are executed when Events from the .NET Component occur (jump)

The “Configuring .NET Control Objects” on page 6-93 describes how to configure these interfaces.

Grid Object

The Grid object allows you to read/write data in a tabular format from the data source configured in the object. To draw one, do the following:

- Click the “Grid” tool
- Click on the screen, click the left mouse button, and drag the mouse across the screen to create a box of the desired size (while holding down the mouse button).



Grid Object
Tool

3. Release the mouse button, and the Grid object will display.

ID	Data
1	MMM 1
2	MMM 2
3	MMM 3
4	MMM 4
5	MMM 5
6	MMM 6
7	MMM 7
8	MMM 8
9	MMM 9

Figure 6-44 Sample Grid object

Right-click on the Grid object, and select “Properties” from the menu. This displays the “Object Properties” dialog box.

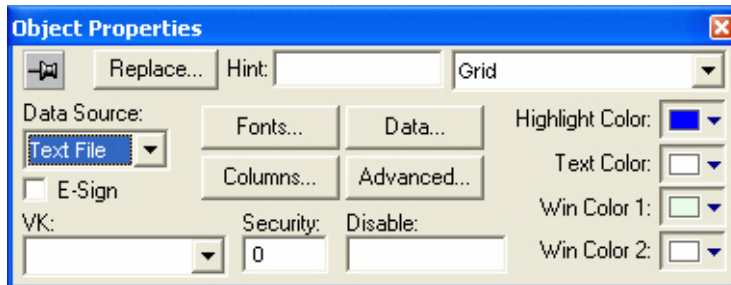


Figure 6-45 Object Properties: Grid

Use the “Object Properties” dialog box to specify the following parameters for the Grid:

- “Data Source” selects the data source type. The object supports three data sources as shown in Table 6-1.

Table 6-1 Available Grid “Data Source” types

Data Source	Description
Text File	Displays data from a text file in the ASCII or Unicode format (e.g. CSV text files).
Class Tag	Displays values from a Class Tag, where the members of the tag are fields (columns) of the Grid object, and each array position is one row of the grid object. Not currently supported.
Database	Displays data from an SQL Relational Database, using ADO (ActiveX Database Object) to exchange data with the database.

- “E-Sign”, when selected, the user will be prompted to enter an electronic signature before entering or modifying data on the object.
- “VK” selects a Virtual Keyboard type used for this object. The option <Use Default> selects the default Virtual Keyboard configured on the Project ? Settings ? Runtime Desktop interface. You can also specify a different virtual keyboard for this Grid object.

- “Security” specifies the security system access level required for the object/dynamic.
- “Disable” specifies an expression in this field to disable data input or action by the user.
- “Highlight Color” selects a background color for the selected row, during the runtime.
- “Text Color” selects a text color for the selected row, during the runtime.
- “Win Color 1” selects a background color for the odd rows.
- “Win Color 2” selects a background color for the even rows.
- “Fonts” launches the “Fonts” dialog box, where you can configure the font settings for the text displayed in the Grid object.
- “Columns” launches the “Columns” dialog box, where you can configure the settings (such as label, column, width, etc.) for the columns of the Grid object (see “Configuring Grid Objects” on page 6-97 for details).
- “Data” launches the “Data” dialog box, where you can specify a data source for the Grid object (see “Configuring Grid Objects” on page 6-97 for details).
- “Advanced” launches the “Advanced” dialog box, where you can configure several settings for the Grid object (see “Configuring Grid Objects” on page 6-97 for details).

Combo Box Object



Combo Box
Tool

Click the “Combo Box” tool to select a single label from a combo-box list of labels. If the list is longer than the space allotted, ScreenView enables a scroll bar for the list. During runtime, if you select a label from the list, the combo-box hides itself and the selected label displays in the combo box.

Double-click on the Combo Box object to open the “Object Properties” dialog box.

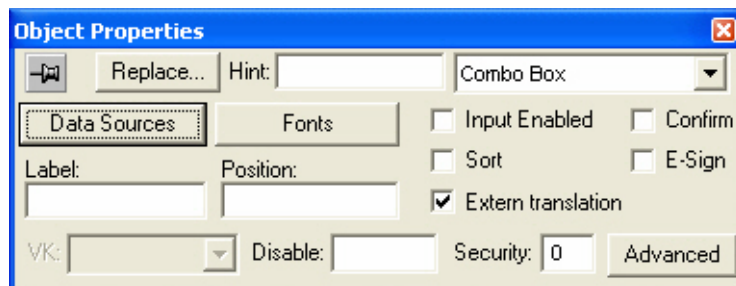


Figure 6-46 Object Properties: Combo Box

You can use this dialog box to set the following parameters:

- “Position” specifies an integer tag, which corresponds to the label currently displayed in the combo box. Changing this tag value changes the label being displayed.
- “Label” specifies a string tag to receive the value of the label currently displayed in the combo box.
- “Input Enabled”, when selected, allows an operator to select a label by typing the contents of that label into a tag in the Label field.
- “Confirm”, when selected, prompts an operator to confirm a command during runtime.
- “VK” specifies a virtual Keyboard type used for this object. You need to enable the Virtual Keyboard option on the “Project... Settings” Runtime Desktop interface before configuring the Virtual Keyboard for this interface.
- “E-Sign”, when selected, the user will be prompted to enter the Electronic Signature before executing the dynamic.

- “Disable” specifies a tag with a nonzero value to disable this combo box. Type a zero, or leave the field blank (default) to enable the command property. If you disable the combo box, it appears grayed out during runtime.
- “Security” specifies a security level for the command (0 to 255). If an operator logs on and does not have the specified security level, the command becomes inactive. If an operator logs on and does have the specified security level, or you leave this field blank, the command property remains active.
- “Fonts” opens the “Font” dialog box. Use this dialog box to change the characteristics of a message font.
- “Data Souces” opens the “Data Sources” dialog box, which allows you to define labels and their order in the combo-box.

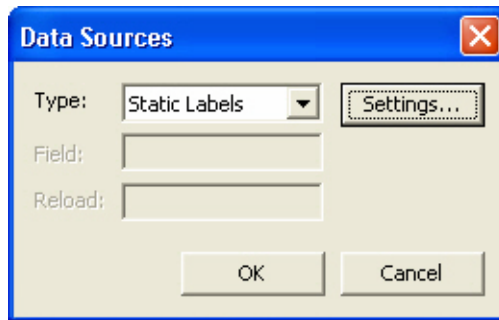


Figure 6-47 The “Data Sources” dialog box

Use the parameters on the “Combo Data” dialog box as follows:

- “Type” permits selection of “Static Labels”, “Text File”, or “Database”.
- “Field” is available for “Text File” and “Database” types. Enter the field number in the text file or database that provides the list of combo-box labels.
- “Reload” specifies a tag. If during execution this tag contains a nonzero value the combo box reloads. Type a zero, or leave the field blank (default) to have the combo-box stay hidden displaying the user-selected label.
- “Settings” displays a dialog box that lets you select a text file, database, or enter static labels.
 - The “Static Labels” dialog box provides a list box for entry of the static labels. Enter labels separated by commas or on separate lines.
 - The “Grid Data - Text File” dialog box lets you select a data file and specify its format (see “Data Source – Text File” on page 6-97).
 - The “Database Configuration” dialog box lets you select the database to use to obtain the labels (see Database Configuration for more information).



Radio Button Object Tool

Radio Button Object

The radio button object is useful to create interfaces where the users can chose one option from multiple options on the display. Follow these steps to use a radio button object:

1. Click the Radio Button tool to create a radio button object on your screen.
2. Click in the drawing area and drag the mouse/cursor to draw the radio button and its label.
3. Release the mouse button when the object is the size you want.

4. Double-click on the object to view the “Object Properties” dialog box.



Figure 6-48 Object Properties: Radio Button

Use the “Object Properties” dialog box to specify the following parameters for the radio button object:

- “Caption” specifies a caption by typing the text into the text box.
- “Fonts” specifies a font style for the caption by clicking the “Fonts” button.
- “E-Sign”, when selected, the user will be prompted to enter the Electronic Signature before executing the command.
- “Confirm”, when selected ensures that Think & Do prompts you to confirm the action at runtime.
- “Key” lets you select a key from the list to associate that keyboard key with the object or group of objects. You can then press this key to check/uncheck the radio button.
 - Click (check) the “Shift”, “Ctrl”, or “Alt” box to create a combination key, meaning the <Shift>, <Ctrl>, or <Alt> key must be pressed with the key specified in the drop-down list.
 - Click the Browse button (...) to open the “Key Modifier” dialog box that lets you modify your combination keys. You can choose “Left”, “Right”, or “Left or Right” to specify the position on the keyboard of the <Shift>, <Ctrl> or <Alt> key in the combination key. If you choose “Left or Right”, the command will be executed any time either of these keys is pressed in combination with the key specified in the drop-down list.
- “Disable” specifies a tag or expression to enable and disable the object. You disable the radio button object when you enter a value different from 0.
- “Security” specifies a value to define a security level for the object, as defined under Security. When a user logs on, and does not have the specified security level, Think & Do disables the object.
- “Tag” specifies a tag to update when the user clicks on the radio button during the runtime. If no “Feedback” was specified, the value of this tag is also used to indicate the current status of the object.
- “True Value” specifies a value that will be used to change the control to TRUE state and to indicate that the control is in TRUE state. For more information about states, please refer to the states table.
- “Advanced” opens the “Advanced” dialog box (see Figure 6-49).

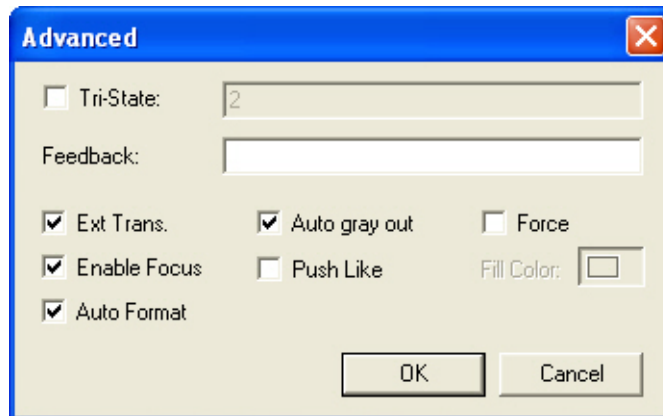


Figure 6-49 The radio button “Advanced” dialog box

- “Tri-State”, if selected, the control has a third state. The third state will be displayed when the tag configured in the “Feedback” field assumes the value specified in the “Tri-State” field. If the “Feedback” field is blank, the third state will be displayed when the tag configured in the “Tag” field assumes the value specified in the “Tri-State” field.
- “Feedback” specifies the value that indicates the state of the object (TRUE, FALSE or Third-State). If the “Feedback” field is blank, the tag configured in the “Tag” field will be used as the “Feedback” tag.
- “Ext Trans.” enables translation of the button caption when a translation file is specified via scripting.
- “Force” selecting this box forces the Tag Database to recognize a tag change when the user clicks on the object, even if the value of the tag in question does not change.
- “Auto gray out” turns the caption of the object to gray when it is disabled by the “Disable” field in the “Object Properties” dialog box or due to the Security System.
- “Enable Focus”, when selected, the object can receive the focus during the runtime by the navigation keys.
- “Push Like”, when selected, the control is displayed as a button, instead of the standard radio button standard shape.
- “Fill Color” specifies the fill color for the button. This option is enabled only when the “Push Like” option is checked.

There are two main modes of operation for this object: Normal Mode and Tri-State Mode. These modes are described below.

Normal Mode

When the “Tri-State” option is unchecked, the object operates in Normal Mode. Therefore, it can assume two states only as shown in Table 6-2.

Table 6-2 Radio Button Normal Mode States

State	Shape	Shape (Push Like)
FALSE	<input type="radio"/> Radio	<input type="button" value="Radio"/>
TRUE	<input checked="" type="radio"/> Radio	<input type="button" value="Radio"/>

When the value of the tag configured in “Feedback” is equal to the value of the tag configured in “True Value” in the “Object Properties” dialog box, the state is set to TRUE. Otherwise, the state is set to FALSE. If the “Feedback” field is blank, the tag configured in the “Tag” field will be used as the “Feedback” tag.

When the user clicks on the object, the tag configured in the “Tag” field is updated with the value configured in the “True Value” field.

Tri-State Mode

When the “Tri-State” option is checked, the object operates in Tri-State Mode. Therefore, it can assume three states as shown in Table 6-3.

Table 6-3 Radio Button Normal Mode States

State	Shape	Shape (Push Like)
FALSE	<input type="radio"/> Radio	<input type="button" value="Radio"/>
TRUE	<input checked="" type="radio"/> Radio	<input type="button" value="Radio"/>
TRI-STATE	<input checked="" type="radio"/> Radio	<input type="button" value="Radio"/>

When the value of the tag configured in “Feedback” is equal to the value of the tag configured in “True Value”, the state is set to TRUE. When the value of the tag configured in “Feedback” is equal to the value of the tag configured in “Tri-State”, the state is set to TRI-STATE. When none of these conditions are satisfied, the state is set to FALSE. If the “Feedback” field is left in blank, the tag configured in the “Tag” field will be used as the “Feedback” tag.



The “Tri-State” field must not be configured with the same value as the “True Value” field nor with an empty string value.

Table 6-4 Radio Button Tri-State Value Returns

Current Status	Value written to the tag configured in the Tag field when the user clicks on the object
FALSE	Value configured in the “True Value” field
TRUE	Value configured in the “Tri-State” field
TRI-STATE	Value configured in the “True Value” field



Check Box Object Tool

Check Box Object

The check box object is useful to create interfaces where the users can chose one option from multiple options on the display. Follow these steps to use a check box object:

1. Click the Check Box tool to create a check box object on your screen.
2. Click in the drawing area and drag the mouse/cursor to draw the check box and its label.
3. Release the mouse button when the object is the size you want.
4. Double-click on the object to view the “Object Properties” dialog box.



Figure 6-50 Object Properties: Check Box

Use the “Object Properties” dialog box to specify the following parameters for the check box object:

- “Caption” specifies a caption by typing the text into the text box.
- “Fonts” specifies a font style for the caption by clicking the “Fonts” button.
- “E-Sign”, when selected, the user will be prompted to enter the Electronic Signature before executing the command.
- ”Confirm” ensures that Think & Do prompts the operator to confirm the action at runtime.
- “Key” use the drop-down list to select a key to associate that keyboard key with the object or group of objects. You can then press this key to check/uncheck the check box.
 - Click (check) the “Shift”, “Ctrl”, or “Alt” box to create a combination key, meaning the <Shift>, <Ctrl>, or <Alt> key must be pressed with the key specified in the drop-down list.
 - Click the Browse button (...) to open the “Key Modifier” dialog box that lets you modify your combination keys. You can choose “Left”, “Right”, or “Left or Right” to specify the position on the keyboard of the <Shift>, <Ctrl>, or <Alt> key in the combination key. If you choose “Left or Right”, the command will be executed any time either of these keys is pressed in combination with the key specified in the drop-down list.
- “Disable” specifies a tag or expression to enable and disable the object. You disable the check box object when you enter a value different from 0.

- “Security” specifies a value to define a security level for the object, as defined under Security. When a user logs on, and does not have the specified security level, Think & Do disables the object.
- “Tag” specifies a tag that updates when the user clicks on the check box during runtime. If no “Feedback” was specified, the value of this tag is also used to indicate the current status of the object.
- “True Value” specifies a value that will be used to change the control to TRUE state and to indicate that the control is in TRUE state. For more information about states, please refer to the states table.
- “Advanced” opens the “Advanced” dialog box.

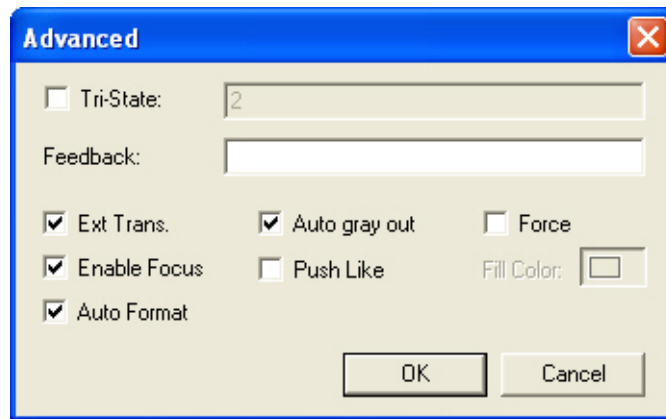


Figure 6-51 The check box “Advanced” dialog box

- “Tri-State” specifies that the control has a third state. The third state will be displayed when the tag configured in the “Feedback” field assumes the value specified in the “Tri-State” field. If the “Feedback” field is blank, the third state will be displayed when the tag configured in the “Tag” field assumes the value specified in the “Tri-State” field.
- “Feedback” specifies a value that indicates the state of the object (TRUE, FALSE or Third-State). If the “Feedback” field is blank, the tag configured in the “Tag” field will be used as the “Feedback” tag.
- “Ext Trans.” enables translation of the caption when a translation file is specified via scripting.
- “Force” forces the Tag Database to recognize a tag change when the user clicks on the object, even if the value of the tag in question does not change.
- “Auto gray out” turns the caption of the object to gray when it is disabled by the “Disable” field in the “Object Properties” dialog box or due to the Security System.
- “Enable Focus”, when selected, the object can receive the focus during the runtime by the navigation keys.
- “Push Like”, when selected the control is displayed as a button, instead of the standard check box standard shape.
- “Fill Color” specifies the fill color for the button. This option is enabled only when the “Push Like” option is checked.

There are two main modes of operation for this object: Normal Mode and Tri-State Mode. These modes are described below.

Normal Mode

When the “Tri-State” option is unchecked, the object operates in Normal Mode. Therefore, it can assume two states only as shown in Table 6-5.

Table 6-5 Check Box Normal Mode States

State	Shape	Shape (Push Like)
FALSE	<input type="checkbox"/> Check	<input type="button" value="Check"/>
TRUE	<input checked="" type="checkbox"/> Check	<input type="button" value="Check"/>

When the value of the tag configured in “Feedback” is equal to the value of the tag configured in “True Value” in the “Object Properties” dialog box, the state is set to TRUE. Otherwise, the state is set to FALSE. If the “Feedback” field is blank, the tag configured in the “Tag” field will be used as the “Feedback” tag.

When the user clicks on the object, the tag configured in the “Tag” field is updated according to Table 6-6.

Table 6-6 Check Box Normal Mode Value Returns

Current Status	Value written to the tag configured in the Tag field when the user clicks on the object
FALSE	Value configured in the “True Value” field
TRUE	“NOT” (Value configured in the “True Value” field)



When the value configured in “True Value” is a string, the tag configured in the “Tag” field will be toggled between an empty string and the “True Value”. If “True Value” is blank, the tag configured in the “Tag” field will be toggled between “UNSELECTED” and an empty string.

Tri-State Mode

When the “Tri-State” option is checked, the object operates in Tri-State Mode. Therefore, it can assume three states as shown in Table 6-7.

Table 6-7 check box Normal Mode States

State	Shape	Shape (Push Like)
FALSE	<input type="checkbox"/> Check	<input type="button" value="Check"/>
TRUE	<input checked="" type="checkbox"/> Check	<input type="button" value="Check"/>
TRI-STATE	<input checked="" type="checkbox"/> Check	<input type="button" value="Check"/>

When the value of the tag configured in “Feedback” is equal to the value of the tag configured in “True Value”, the state is set to TRUE. When the value of the tag configured in “Feedback” is equal to the value of the tag configured in “Tri-State”, the state is set to TRI-STATE. When none of these conditions are satisfied, the state is set to FALSE. If the “Feedback” field is left in blank, the tag configured in the “Tag” field will be used as the “Feedback” tag.



The “Tri-State” field must neither be configured with the same value as the “True Value” field nor with an empty string value.

Table 6-8 Check Box Tri-State Value Returns

Current Status	Value written to the tag configured in the Tag field when the user clicks on the object
FALSE	Value configured in the “True Value” field
TRUE	Value configured in the “Tri-State” field
TRI-STATE	Value configured in the “True Value” field



If NOT (Value configured in the “True Value” Field) is equal to “Tri-State”, the value assigned to the tag configured in the “Tag” field will be the minimum signed integer value different from “True Value”.

When “True Value” is a string, NOT (Value configured in the “True Value” field) will result in an empty string. If “True Value” is an empty string, NOT (Value configured in the “True Value” field) will result in “UNSELECTED”.

List Box Object

Click the “List Box” tool to create a list box object on your screen. Generally, when you execute an application, the active list box object displays a list of messages.

- On a screen containing only one list box object and no text input boxes, the list box object will be active automatically.
- On a screen containing multiple list box objects and text input boxes, you can use a cursor (pointing device) or the <Tab> key to select and activate a list box object.



List Box
Object Tool

You can select a message from the active list box during runtime and write the message value to a tag. (If a list is too long to fit within the viewable area of a list box object, the object provides scroll bars.)

Use the “Enter Reqd” box on the “Object Properties” dialog box to configure selected messages as follows:

- Check (enable) the “Enter Reqd” box and use the keyboard/keypad keys, list control objects from the Library, pointing devices, or user-defined keys containing the Post-Keys() function to scroll through the message list. Then, use the <Enter> key to select the message and write its value to the write tag. You can use the <Esc> and <Tab> keys to return to the previously selected message at any time prior to pressing the Enter key.
- Uncheck (disable) the “Enter Reqd” field to write the value of a selected (highlighted) message the write tag automatically.

To add list box objects to a screen:

1. Click the List Box tool on the Active Objects toolbar.
2. Click in the screen and drag to create/adjust an expanding rectangle.
 - “Height” and the font size determine how many messages are visible.
 - “Width” determines how much of the message length is visible.

After creating a rectangle, you can adjust the size and font characteristics to allow more messages to display in the given space.

3. Double-click on the object to open the “Object Properties” dialog box.

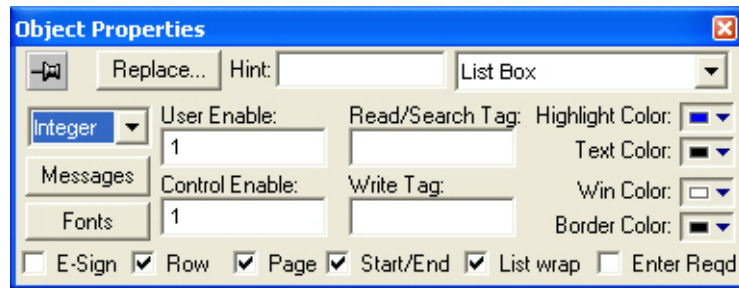


Figure 6-52 Object Properties: List Box



You also can open the “Object Properties” dialog box, by right-clicking on the list box object or by highlighting the object, pressing the <Alt>+<Enter> keys, and selecting “Properties” from the resulting pop-up menu.

You can use this dialog box to specify the following parameters:

- “Value” drop-down list (located below the “Replace” button): Click to select one of the following the tag values used to index the message list.
 - “Boolean”
 - “Integer” (default)
 - “LSB” (least significant bit)



For more information, see the discussion about the “State” field on the “Messages Configuration” dialog box.

- “Messages” opens the “Messages Configuration” dialog box.

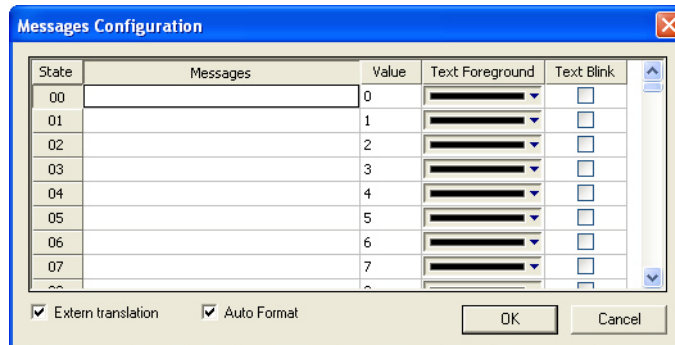


Figure 6-53 The “Messages Configuration” dialog box

Use the parameters on this dialog box as follows:

- “State” (read-only) displays the index of individual messages. ScreenView numbers this field based on the Read/Search Tag type you selected:
 - “Boolean” provides two valid states, labeled 0 and 1
 - “Integer” provides 255 valid states, labeled 1 to 255
 - “LSB” provides 32 valid states (32 bits in an integer value) labeled 0 to 31
 - “Messages” specifies the string value message displayed in the list box object. You can use tags in messages using the {tag} syntax.
 - “Value” specifies a message value matching the specified Read/Search Tag value. (Also, the same value written to the write tag).
 - If you specify “LSB” for the “Value” field, ScreenView uses the value specified in the “State” field for both the Read/Search Tag and the write tag.
 - “Text Foreground” specifies a color for the message text foreground. When the “Color” dialog box displays, click on a color to select it, and close the dialog box.
 - “Text Blink”, when selected, causes the message to blink, once per second, when it displays.
 - “Fonts” opens the “Font” dialog box, which allows you to change the characteristics (style, size, and so forth) of the message font.
 - “User Enable” specifies a tag, expression, or a (nonzero) number to select a message in the runtime application. The default is 1 (true, enabled).
 - “Control Enable” specifies a tag, expression, or a (nonzero) number to select a message in the runtime application — depending on the current value of the Read/Search Tag. The default is 1 (true, enabled).
- ScreenView bases this parameter on the “Value” field (“Messages Configuration” dialog box) you associate with the selected message. Enabling this field allows tag changes triggered by the process to affect which messages you can select.
- “Read/Search Tag” specifies an integer or a Boolean tag to point to a selected message based on the message “Value” field (“Messages Configuration” dialog box). You can use the Control Enable and User Enable fields to control whether the operator or a process can alter this tag.
 - “Write Tag” (optional) specifies a string tag to receive the “Message” value of the last-selected message. When you close and reopen the screen containing a list box object, ScreenView uses this tag value to determine the last message selected in the list box.

- “E-Sign”, when selected, the user will be prompted to enter the Electronic Signature before executing the dynamic.
- “Row”, when selected, include set up and set down arrows in the list box object scroll bar.
- “Page”, when selected, include page up and page down arrows in the list box object scroll bar.
- “Start/End”, when selected, include home and end arrows in the list box object scroll bar.
- “List wrap”, when selected, continue displaying and scrolling the message list (starting at the opposite end) after you scroll to the beginning or end of the list.
- “Enter Req’d”, when selected, allows users to select messages using the Enter key only. It prevents the Tab key from selecting messages.
- Color boxes, when selected, opens the “Color” dialog box or the 16-color “Color Selection” dialog box. Either dialog box allows you to specify or change colors for the list box object. Click a color to select it and then click the “OK” button to close the dialog box.
 - “Highlight Color” specifies a color for highlighting messages (default is blue).
 - “Text Color” specifies a color for highlighting message text (default is black).
 - “Win Color” specifies a color for the list box background (default is white).
 - “Border Color” specifies a color for the list box border (default is black).



Smart
Message
Object Tool

Smart Message Object

Click the “Smart Messages” tool to create one or more smart message objects, which you can use to display messages and graphics based on tag values when you execute the application. ScreenView provides the following smart message object types:

- Message Display displays any one of multiple messages within a single screen object.
- Multistate Indicator permits display of any one of multiple messages within a single screen object, and also has the ability to display bitmap images with the messages.
- Multistate Pushbutton permits display of messages and bitmap images. This object also resembles a multi-position switch in that it allows you to switch (toggle between) messages by clicking on the object during the runtime.

These smart message object types vary in their ability to display messages and graphics, write to a tag, and control how many messages and graphics display on the screen. However, all of the object types can receive process input (Read Tag value) to determine which message to display.

To add a smart message object to the screen:

1. Click the Smart Message button and position the mouse on the screen.
2. Click and drag to create (and adjust the size of) a rectangle.
3. You use the rectangle’s size and font size to determine how much text and how large a bitmap image you can display on the screen. Later, you can change the rectangle’s size and font characteristics to allow longer messages to display in a given space.

- Double-click on the object to open the “Object Properties” dialog box.

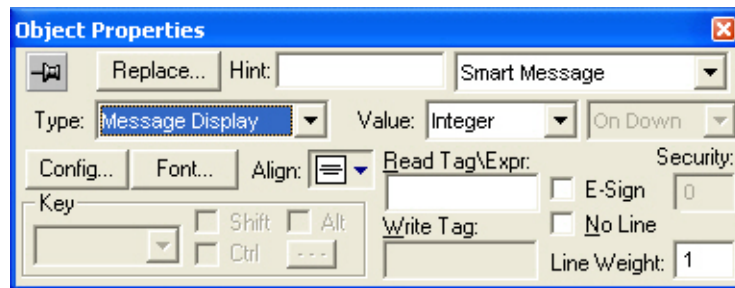


Figure 6-54 Object Properties: Smart Message

You can use this dialog box to specify the following parameters:

- “Type”, when selected, selects the smart message object type. The object type sets the behavior of the object during the runtime and the features supported by it:
 - “Message Display” (default)
 - “Multistate Indicator”
 - “Multistate Pushbutton”
 - “Value” selects the type of values used to index the message list:
 - “Boolean” provides two valid states. Use this selection when you want to display either one of two different messages, based on a boolean value (0 or 1).
 - “Integer” (default) provides 500 valid states. Use this selection when you want to display different messages based on specific values from an Integer tag.
 - “LSB” (least significant bit) provides 32 valid states (32 bits in an integer value). Use this selection when you want to display different messages based on which bit from an integer tag is set. If more than one bit from the Integer tag is set simultaneously, the message associated with the least significant bit that is set (value 1) will be displayed.



If “Multistate Pushbutton” is the Smart Message type, only 16 different messages can be associated with the object, even for “Integer” or “LSB Value” types.

- “Read Tag/Expr” specifies the name of an integer or a Boolean tag. The value of this tag will determine which message will displayed by the object during the runtime.
- “Write Tag” (optional and available for “Multistate Pushbutton” only) specifies the name of an integer or a Boolean tag. The value associated with the message currently displayed by the object is written to this tag.
- “Align” selects the alignment of the text displayed by the Smart Message object.
- “Key” (optional and available for “Multistate Pushbutton” only) defines a shortcut used to go to the next message (step) using a keyboard when the “Multistate Pushbutton” type is selected. This option is especially useful when creating applications for runtime devices that do not provide a mouse or touch-screen interface, when the keyboard is the only physical interface available to interact with the application during the runtime.
- “Event” (available for “Multistate Pushbutton” only) selects the option to use to change the message:
 - “On Down” switches to the next message when you click on the object (default).
 - “While Down” switches to the next message continuously while you hold the mouse button down on the object.

- “On Up” switches to the next message when you release the mouse button on the object.
- “E-Sign” (available for “Multistate Pushbutton” only), when selected, the user will be prompted to enter the Electronic Signature before executing the dynamic.
- “Security” (available for “Multistate Pushbutton” only) defines System Access Level required for the object/dynamic.
- “No line”, when selected, the line border of the object is not visible.
- “Line Weight” defines the thickness of the line drawn around the object (the border).
- “Fonts” launches the “Fonts” dialog box, where you can configure the font settings for the text displayed in the object.
- “Config...” launches the “Configuration” dialog box, where you can configure the messages for the object.

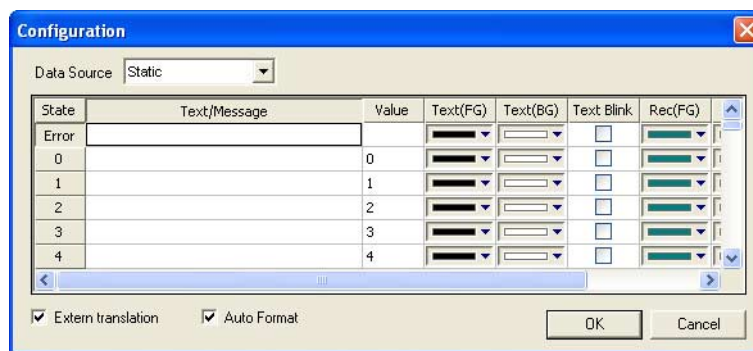


Figure 6-55 The smart message “Configuration” dialog box

- “Data Source” defines where the messages displayed comes from. They can either be configured directly on the object (“Data Source” is “Static”) or read from an external text file (“Data Source” is “Text File”). When the “Data Source” is “Static”, the “Configuration” dialog box is displayed as pictured above, and you can configure all the settings on the grid. When the “Data Source” is “Text File”, the “Configuration” dialog box displays a field for entering the path and name of the file from which the messages will be read (the source file). See Source File Format for further details about the format of the text file supported by the Smart Message object when the “Data Source” is “Text File”.
- “Extern translation” enables translation of the message when a translation file is specified via scripting.
- The following table describes the meaning of the properties associated with each message, regardless of the Data Source:
 - “Text/Message” defines the message (text) that will be displayed when selected during the runtime. You can concatenate tag values to the message in this field by configuring the tag between curly brackets. For example: The level value is {TagLevel}.
 - “Value” specifies a unique value for each message. During the runtime, the object will display the message associated with the value that matches the value of the tag configured in the “Read Tag” field. If there is no such message, the message configured in the first row (“State” is “Error”) displays during the runtime. When the object Type is set as “Multistate Pushbutton”, the value associated with the current message is also written to the tag configured in the “Write Tag” field (if any).

- “Text (FG)” is the foreground color for the messages displayed during the runtime.
- “Text (BG)” is the background color for the messages displayed during the runtime.
- “Text Blink” specifies that the message text will blink during the runtime.
- “Rec (FG)” specifies line color (Border) for the rectangle behind the message.
- “Rec (BG)” specifies the background (Fill) color for the rectangle behind the message.
- “Rec Blink” specifies that the rectangle behind the message will blink during the runtime.
- “Graphic File” specifies the path and name of the bitmap file (*.BMP) (if any) that will be displayed when the message associated with it is selected during the runtime. If you do not specify the path, the bitmap file must be stored in the application’s directory.
- “Transparent” specifies the color that will be transparent in the graphic file, if the En. “Transparent” check-box is checked.
- “En. Transparent”, if selected, the color selected in the “Transparent” field will be set as transparent in the graphic file.



The properties “Graphic File”, “Transparent” and “En. Transparent” are not available for the Message Display type.



You can copy data from this dialog box and paste it into an Excel worksheet, and vice versa.

Source File Format

This section describes the format of the text file supported by the Smart Message object when the “Data Source” is “Text File”. The main advantage of using an external text file instead of static values is that it gives you the flexibility to change the messages during the runtime, by pointing to a different text file, or even by changing the content of the text file dynamically.

The text file must be created in the CSV format (comma separated values), where the comma character (,) is used to divide the columns (data) in each line (row) of the file. Therefore, you can use any CSV editor such as Microsoft Notepad and Microsoft Excel to create the CSV file with the messages and their properties for the Smart Message object.

The description of each property associated with the messages is provided in the Smart Message section. The order of the data in the CSV file is described in Table 6-9.

Table 6-9 Smart Message CSV File Format

Column #	Property	Default Value
1	Text/Message	-
2	Value	-
3	Text (FG)	0
4	Text (BG)	16777215
5	Text Blink	0
6	Rec (FG)	8421376
7	Rec (BG)	16777215
8	Rec Blink	0

Table 6-9 Smart Message CSV File Format

Column #	Property	Default Value
9	Graphic File	-
10	Transparent	0
11	En. Transparent	0

When configuring text messages that have the comma character as part of the message, you must configure the whole message between quotes (e.g. "Warning, Turn the motor Off"); otherwise, the comma will be interpreted as a data separator instead of as part of the message.

- The first line of this file is equivalent to the "State" is "Error". In other words, if there is no message associated with the current value of the tag configured in the "Read Tag" field, the message configured in the first row ("State" is "Error") is displayed during the run-time.
- The data configured in the "Value" column of the first row from this file is irrelevant. This row must always be configured, regardless of the object type (even for "Multistate Pushbutton").
- Only the "Text/Message" and "Value" columns are mandatory. The other columns are optional, and the default values will be used if you do not specify any value for them (see table).
- The fields "Text(FG)", "Text(BG)", "Rec(FG)", "Rec(BG)", and "Transparent" can be configured with the code of the color associated with it. The code can be entered directly in decimal format (e.g. 255) or in hexadecimal format using the syntax #value (e.g. #0000FF).
- The fields "Text Blink", "Rec Blink", and "En. Transparent" can be configured with Boolean values "0" or "1" (0 = Unchecked; 1 = Checked), or with the keywords "FALSE" or "TRUE" (FALSE = Unchecked; TRUE = Checked).

Example:

```
Error Message,,0,16777215,1,8421376,16777215,1,error.bmp,0,0
Message Zero,0,0,16777215,0,8421376,16777215,0,open.bmp,65280,1
Message Ten,10,0,16777215,0,8421376,16777215,0,closed.bmp,65280,1
Message Twenty,20,0,16777215,0,8421376,16777215,0,,0,0
Message Thirty,30,0,16777215,0,8421376,16777215,0,,0,0
```



Use the Smart Message editor ("Data Source" is "Static") to configure the messages, values and colors. To do so, select the configuration, copy it and paste it into an Excel worksheet. Then, you can save the Excel worksheet as a CSV file (using the "File... Save As" menu). This procedure provides you with a user friendly interface for configuring the color codes.



Smart Message Object Tool

Pushbutton Object

Click the "Pushbutton" tool to create a pushbutton object using the Command dynamic object property with an object or pre-configured pushbuttons. ScreenView provides the following pre-configured button types, all of which mimic the standard panel buttons of the same name:

- "Momentary" (default) buttons change state (Open or Closed) when pressed and reverts to its initial state when released. This button type always displays in its normal position when the screen opens.

- “Maintained” buttons change state (Open or Closed) when pressed but does not revert to its initial state when released. The operator must press the button again to change its present state. This button type maintains its state across screen changes.
- “Latched” buttons change state (Open or Closed) when pressed and remains in this state until released by changing the “Reset” tag.

ScreenView also provides the following button styles:

- Rectangular with a faceplate and indicator light
- Rectangular without a faceplate or indicator light (default)
- Rectangular with a 3-D
- Rectangular with a floating appearance

To add one or more pre-configured buttons to a screen:

1. Click the Pushbutton tool, and position the mouse (pointer) on the screen.
2. Click and drag to create/adjust the size of the rectangular button.
3. The button size and text font characteristics determine how much text you can display and how much area you can touch on a touch screen. You can resize the button and change the font characteristics later to permit longer messages to be shown in a given space.
4. Double-click on the object to open the “Object Properties” dialog box.



Alternatively, you can right-click on the pushbutton object or highlight the object, press the <Alt>+<Enter> keys, and select Properties from the resulting pop-up menu to open the “Object Properties” dialog box.

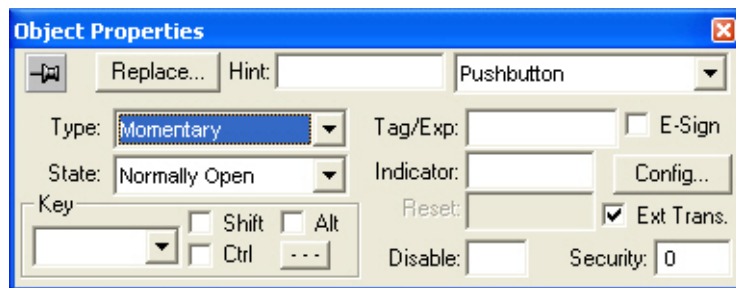


Figure 6-56 Object Properties: Pushbutton

You can use this dialog box to specify the following parameters:

- “Type” specifies the pushbutton type (“Momentary” (default), “Maintained”, or “Latched”).
- “State” specifies a default state for the pushbutton (“Normally Open” (default) or “Normally Closed”).

Click the button to toggle between its default and non-default state (according to its specified “Type”). For example, in the button’s initial state, it may conform to characteristics specified in the “Open” area of the “Configuration” dialog box (see below). Click the button again to toggle to the opposite state, which in this example is “Closed”, and conform to characteristics specified in the “Closed” area.

- “Tag/Exp” specifies a tag or an expression to accomplish the following:
 - Type in a tag to receive the “Write Value” from the appropriate state (Open or Closed) area in the “Configuration” dialog box.

- Type an expression to execute “On Down”, when you press the pushbutton down.



ScreenView does not write the result of any expression in the “Tag/Exp” field into a tag.

- “Indicator” specifies a tag to define an indicator that causes the button to change to a specified color when the tag value matches one of two specified values. You must define both the colors and tag values in the “Configuration” dialog box. If you leave this field blank, the indicator changes color automatically when you press the button.
- “E-Sign”, when selected, the user will be prompted to enter the Electronic Signature before executing the dynamic.
- “Reset” (active for “Latched” pushbutton type only) specifies a tag to control the button’s latched state, as follows:
 - Type a zero and the button will remain in a latched state after you press it.
 - Type a nonzero value and a latched button will become unlatched after you press it. You must reset the tag value to zero before you can press the button again.
- “Key” enter a key in the text box or select a non-alphanumeric key from the drop-down list. Enter a single character or key only. Numbers are not valid entries for this field.
- Select (check) the “Shift”, “Ctrl”, or “Alt” box to create a combination key, meaning the <Shift>, <Ctrl>, or <Alt> key must be pressed with the key specified in the drop-down list.
- Click the Browse button (...) to open the “Key Modifier” dialog box, which lets you modify combination keys. You can choose “Left”, “Right”, or “Left or Right” to specify the position on the keyboard of the <Shift>, <Ctrl>, or <Alt> key in the combination key. If you choose “Left or Right”, the command will be executed any time either of these keys is pressed in combination with the key specified in the drop-down list.
- “Disable” specifies a tag that contains a nonzero value when you want to disable this pushbutton. When disabled, pressing the button has no effect. This box is empty by default, which also enables the command property.
- “Ext Trans.”, when selected, translate the text automatically when a translation file is specified via scripting.
- “Security” specifies a value to define a security level (0 to 255) for this button. If the user does not have the specified security level, the button becomes inactive. If the user has the appropriate security level, or you leave this field blank, the button remains active.

- “Config” opens the “Configuration” dialog box, which allows you to specify style and state parameters for the pushbutton.

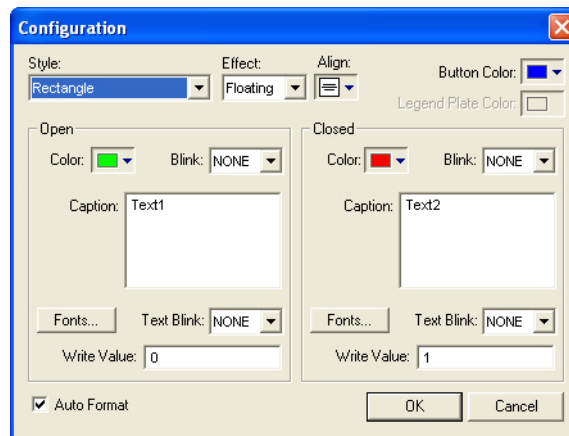


Figure 6-57 The pushbutton “Configuration” dialog box

This dialog box provides the following parameters:

- “Style” specifies a pushbutton style (“Rectangle” (default) or “Rectangle with Indicator”).
- “Effect” specifies a 3-D effect for the pushbutton. The choices are:
 - “Floating” (default) specifies buttons that resemble a flat object with a shadow
 - “3D” specifies buttons that have beveled edges and appear to “depress” into the screen when pressed.
- “Align” specifies the alignment for the caption of the pushbutton.
- “Caption” specifies the caption of the button. Alternatively, if the button style includes an indicator, the legend plate.

You can use the “Style” and “Effect” parameters in combination to create four different buttons, as shown in the following figures.

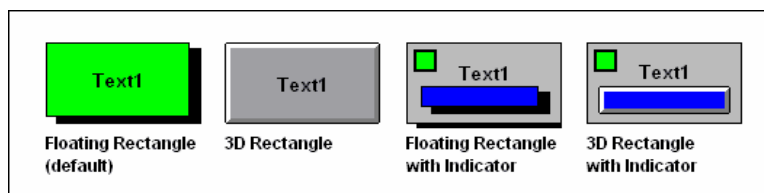


Figure 6-58 Pushbutton styles

- “Button Color” specifies a default color for the button area of a pushbutton object that includes an indicator and a faceplate. When the “Color” dialog box displays, click on a color to select it, and close the dialog box.
- “Legend Plate Color” specifies or changes a default color for the legend plate area of a pushbutton object that includes an indicator. When the “Color” dialog box displays, click on a color to select it, and close the dialog box.
- A legend plate encloses a button and indicator light. This field becomes inactive if the pushbutton Style does not include an indicator.

- “Open” and “Closed” areas configure the appearance of a pushbutton object in its open and closed states.
 - “Color” specifies a default color for an indicator in each State. When the “Color” dialog box displays, click on a color to select it, and close the dialog box.
 - If you selected a pushbutton style that does not include an indicator, you can use this field to specify a button color for each State.
 - “Blink” specifies whether the color you specified in the “Color” box blinks and how fast it blinks for each state (“None” (no blinking, default), “Slow”, and “Fast”).
 - If you set the color to blink, it alternates between the color specified in the “Color” box and the “Legend Plate Color” (if an indicator) or the “Button Color” (if a button).
 - “Caption” specifies text for the top, middle, and bottom lines (by position) of the button. Alternatively, if the button style includes an indicator, the legend plate.
 - “Fonts” opens the “Font” dialog box, which you can use to specify or change the message font characteristics for each state.
 - “Text Blink” specifies whether the text you specified blinks and how fast it blinks for each state (“None” (no blinking, default), “Slow”, and “Fast”). Unlike a blinking color, blinking text appears and disappears.
 - “Write Value” fields specify a value. When the pushbutton is in the appropriate state (“Open” or “Closed”), ScreenView writes this value to the tag specified in the “Tag/Exp” field (“Object Properties” dialog box).

6.3.12 Using the Align and Distribute Toolbar

The Align and Distribute toolbar provides tools that allow you to edit screen objects.

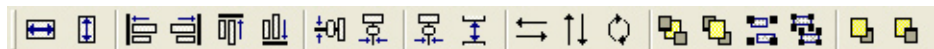
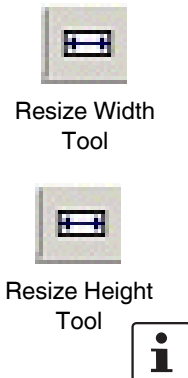


Figure 6-59 The “Align and Distribute” toolbar



Resize Tools

Use the following toolbar options for resizing:

- Click the “Resize width” tool to set the width of all selected objects to the width of the last object selected (the object with the filled handles). You can use the “Resize width” tool to resize one selected object by setting its width equal to its height.
- Click the “Resize height” tool to set the height of all selected objects to the height of the last object selected (the object with the filled handles). You can use the “Resize height” tool to resize one object by setting its height equal to its width.

Use the “Resize width” and “Resize height” tools to create circles from an ellipse or squares from rectangles. Select only one object before using these tools.

You also can use your cursor, mouse, and keyboard arrow buttons to resize objects on your screen. When you select an object (or group of objects) with the cursor, selection handles (black squares) display at each corner and at the midpoint of each side. You can use these handles as follows:

- To enlarge an object, click on a handle and drag your cursor (or pointer) in the direction indicated by the arrows that display. Clicking and dragging a corner resizes the entire object (height and width, while clicking on a side resizes the object in one direction only (height only or width only).

- To enlarge an object with finer resizing control, click on a handle and do not release the left mouse button. Click the arrow keys to resize the object (in the direction indicated by the resizing arrows) one pixel at a time. Release the mouse button when you are finished resizing the object.
- To select and resize an open or closed polygon, draw a selection box around the polygon and group it. You can then click on a handle and drag it to resize the object.
- To change the shape of an open or closed polygon, click on a handle and a boxed square displays at the base of your cursor. Drag the handle to move its position and change the shape of the polygon.



All objects with dynamic properties and Group of Symbols objects (including most symbols and library objects) have multiple “Object Properties” dialog boxes and properties. Use the drop-down list on the “Object Properties” dialog box (select the “View... Properties” menu) to access these different dialog boxes and properties.

Also, if you resize a symbol or group of objects, ScreenView resizes all objects within the symbol or group accordingly.

Align, Center and Distribute Tools

When you select a series of objects (two or more), you can align those objects based on the location of the last object selected. As you select objects, solid handles display on the last object selected, and the handles on all previously selected objects become empty (unfilled) boxes.



In all of the figures provided, the rectangle represents the last object selected.

Use the following alignment tools to align a series of objects.

- Click the “Align left” tool to align all selected objects to the left edge of the last object selected. For an example, see the following figure:



Align Left Tool

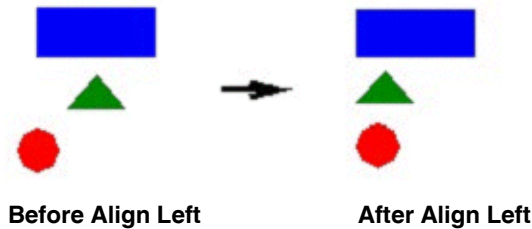
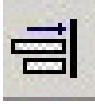


Figure 6-60 Aligning objects left

Think & Do



Align Right
Tool

- Click the "Align right" tool to align all selected objects to the right edge of the last object selected. For an example, see the following figure:

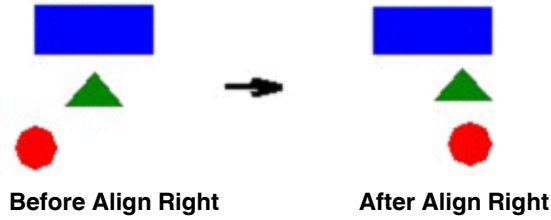


Figure 6-61 Aligning objects right



Align Top
Tool

- Click the "Align top" tool to align all selected objects to the top edge of the last object selected. For an example, see the following figure:

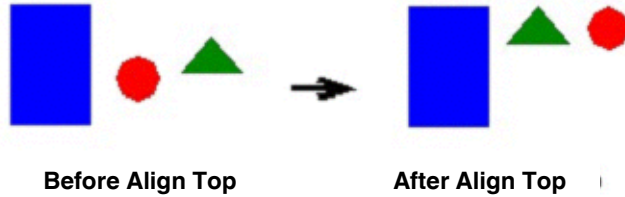
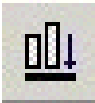


Figure 6-62 Aligning objects top



Align Bottom
Tool

- Click the "Align bottom" tool to align all selected objects to the bottom edge of the last object selected. For an example, see the following figure:

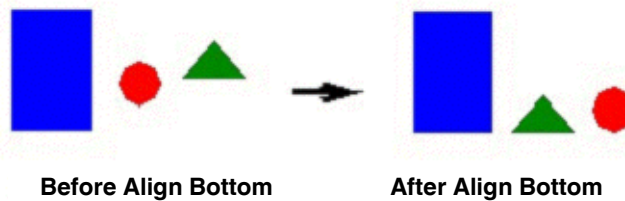
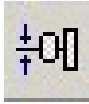


Figure 6-63 Aligning objects bottom



Center Vertically Tool

- Click the "Center Vertically" tool to align all selected objects to the vertical center of the last object selected. For an example, see the following figure:

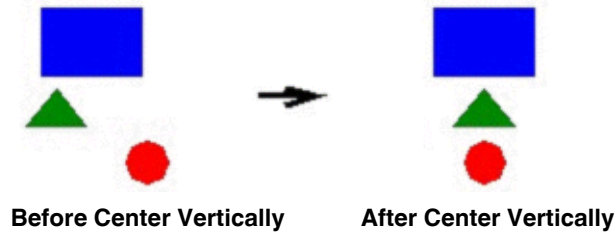


Figure 6-64 Centering objects vertically



Center Horizontally Tool

- Click the "Center Horizontally" tool to align all selected objects to the horizontal center of the last object selected. For an example, see the following figure:



Figure 6-65 Centering objects horizontally



Evenly Distribute Horizontally Tool

- Click the "Evenly distribute horizontally" tool to put an equal amount of horizontal space between a series of objects (two or more). For an example, see the following figure:



Figure 6-66 Distributing objects horizontally

Think & Do



Evenly Distribute Horizontally Tool

- Click the “Evenly distribute vertically” tool to put an equal amount of vertical space between a series of objects (two or more). For an example, see the following figure:

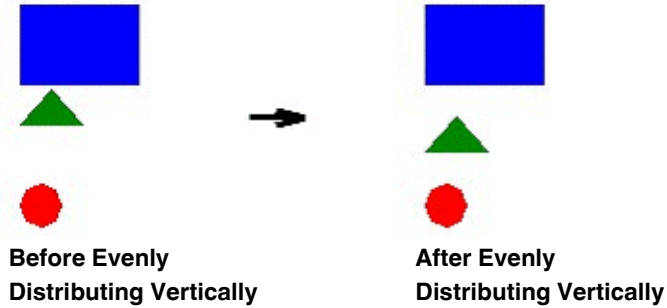


Figure 6-67 Distributing objects vertically



The distribution tools may move the last object selected (with solid handles) by no more than a few pixels to equally space all of the objects.



Flip Horizontally Tool

Flip Horizontally Tool

Click the “Flip Horizontally” tool to invert the selected object horizontally. The object rotates around an imaginary line through its horizontal center until it is a mirror image of the original object. For example, see the following figure:

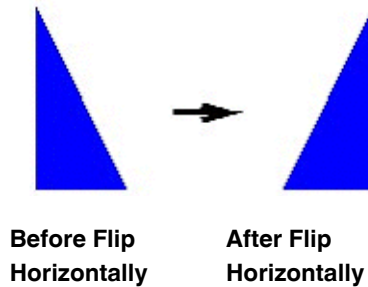


Figure 6-68 Flipping objects horizontally



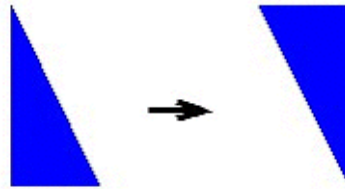
Use this tool only with a single selected object or grouped object. You cannot use this tool with multiple objects selected.



Flip Vertically Tool

Flip Vertically Tool

Click the “Flip Vertically” tool to invert the selected object vertically. The object rotates around an imaginary line through its vertical center until it is a mirror image of the original object. For an example, see the following figure:



Before Flip Vertically

After Flip Vertically

Figure 6-69 Flipping objects vertically



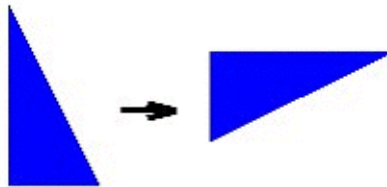
Use this tool only with a single selected object or grouped object. You cannot use this tool with multiple objects selected.



Rotate Tool

Rotate Tool

Click the “Rotate” tool to rotate the selected object 90 degrees (a quarter turn) clockwise.



Before Rotate

After Rotate

Figure 6-70 Rotating objects



Use this tool only with a single selected object or grouped object. You cannot use this tool with multiple objects selected.

Move to Front and Back Tools

ScreenView assigns a unique identification number (ID#) to every object on the screen. These ID#s always start at zero and range up to the total number of objects on the screen. You can click on an object to display its ID# in the status bar.

ScreenView uses ID#s to determine whether an object displays in front of, or behind, another object on the screen. Objects with lower ID#s display behind objects with higher ID#s.

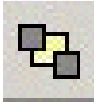
Use the following object layer tools to move selected object(s) behind or in front of another screen object(s).



Use this tool only with a single selected object or grouped object. You cannot use this tool with multiple objects selected.

Also, if you select a group of objects and move them behind or in front of another object, the selected group of objects maintain their original display order.

- Click the "Move to back" tool to move a selected object or objects behind all other objects on the screen. ScreenView assigns the object the lowest ID# and moves that object behind all other objects on the screen.



Move to Back Tool

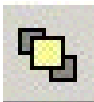


Figure 6-71 Moving objects to back



Alternatively, right-click on an object and select "Move to back" from the object's pop-up menu.

- Click the "Move to front" tool to move a selected object or objects in front of all other objects on the screen. ScreenView assigns the object the highest ID# and moves that object behind all other objects on the screen.



Move to Front Tool



Figure 6-72 Moving objects to front



Alternatively, right-click on an object and select "Move to front" from the object's pop-up menu.

Group and Ungroup Tools

Use the following tools to group and ungroup two or more selected objects.



All objects with dynamic properties and Group of Symbols objects (which includes most symbols and library objects) have multiple "Object Properties" dialog boxes and properties. You can use the drop-down list on the "Object Properties" dialog box (select the "View... Properties" menu) to access these different dialog boxes and properties.



Group Tool

- Click the “Group” tool to combine multiple objects into a single object to facilitate object selection and manipulation. (You can access each part of the group in the “Object Properties” dialog box.)



Alternatively, you can right-click on an object and select “Group” from the object’s pop-up menu.

- Click the “Ungroup” tool to separate a grouped object into its individual components.



Ungroup Tool



Alternatively, you can right-click on an object and select “Ungroup” from the object’s pop-up menu.



A complex grouped object can consist of several sets of grouped objects (known as sub-groups). Consequently, you may find it necessary to ungroup all of the subgroups to completely ungroup a complex object.

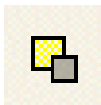
Move Backward and Forward Tools

ScreenView assigns a unique identification number (ID#) to every object on the screen. These ID#s always start at zero and range up to the total number of objects on the screen. You can click on an object to display its ID# in the status bar.

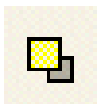
ScreenView uses ID#s to determine whether an object displays in front of, or behind, another object on the screen. Objects with lower ID#s display behind objects with higher ID#s. Use the following object layer tools to move selected object(s) behind or in front of another screen object(s).



Use these tools only with a single selected object or group of objects. You cannot use these tools with multiple objects selected. Also, if you select a group of objects and move them the behind or in front of another object, the selected group of objects maintain their original display order.

Move
backward
Tool

- Click the “Move backward” tool to move the selected object or group one layer below the next object on the screen. (Alternatively, right-click on the object and select “Move backward” from the pop-up menu.) ScreenView assigns the selected object the next available ID# less than the object behind which it was moved.

Move Forward
Tool

- Click the “Move forward” tool to move the selected object or group one layer above the next object on the screen. (Alternatively, right-click on the object and select “Move forward” from the pop-up menu.) ScreenView assigns the selected object the first available ID# greater than the object in front of which it was moved.

6.3.13 Configuring an Alarm

After creating an Alarm/Event, double-click on the object to open the Alarm/Event Control “Object Properties” dialog box.

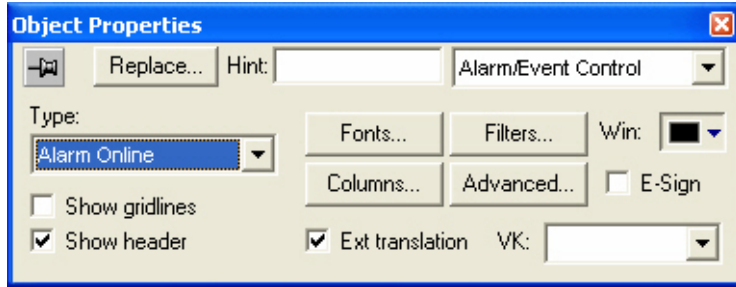


Figure 6-73 Object Properties: Alarm/Event Control

Available configuration options in this dialog box for alarm events are:

- “Show gridlines” displays gridlines in the object.

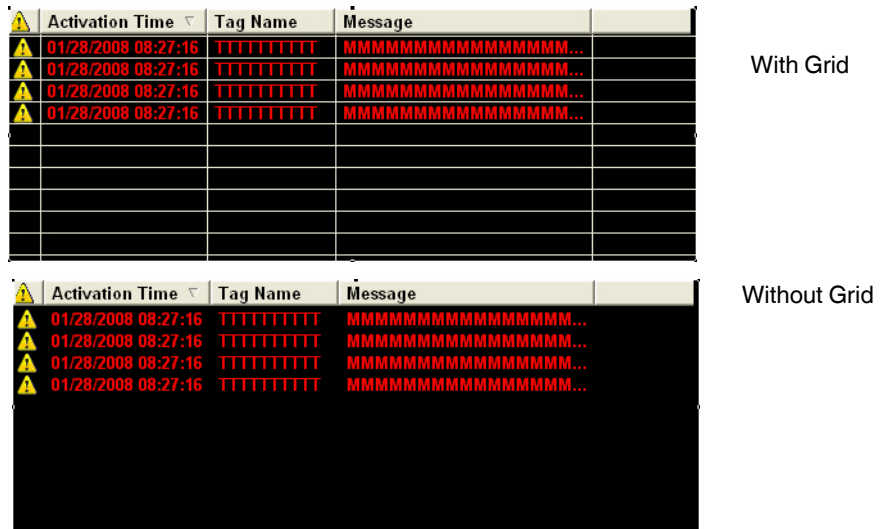


Figure 6-74 Displaying a Grid on an Alarm/Event Control

- “Show Header” displays a header on the object.

Activation Time	Tag Name	Message
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...

With Header

01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...
01/28/2008 08:27:16	TTTTTTTTT	MMMMMMMMMMMMMMMMM...

Without Header

Figure 6-75 Displaying a Header on an Alarm/Event Control

- “Win” selects a background color for the alarm control object. Click the color box to open the color palette pop-up, then simply click a color to select it.
- “Ext translation” enables the external translation of alarm messages when a translation file is specified via scripting
- “E-Sign” prompts the user to enter the Electronic Signature before executing the dynamic.
- “VK” defines a Virtual Keyboard type used for this object. You need to enable the Virtual Keyboard option using the “Project... Settings... Runtime Desktop” interface before configuring the Virtual Keyboard for this interface.
- Click the “Fonts” button to open a standard “Fonts” dialog box, where you can specify display properties for the alarm text.
- Click the “Columns” button to open the “Columns” dialog box where you can specify display properties for columns in the alarm control object (see “Columns” on page 6-70).
- To filter alarm messages during runtime, click the “Filters” button. The “Filters” dialog box displays so you can specify filtering parameters for the alarm control object (see “Filters” on page 6-71).
- Click the “Advanced” button to open the “Advanced” dialog box where you can specify advanced properties for the alarm control object (see “Advanced” on page 6-72).

Columns

You access the “Columns” dialog box from the Alarm/Event Control “Object Properties” dialog box when you click the “Columns” button..

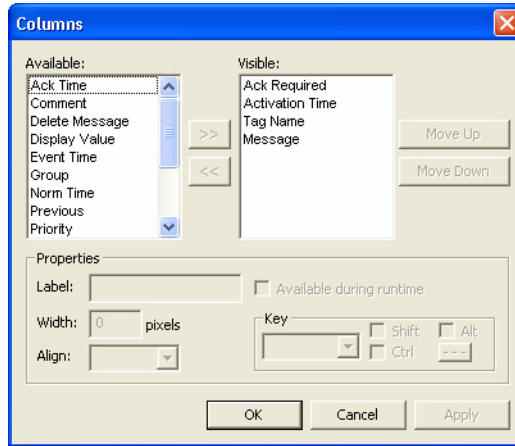


Figure 6-76 The Alarm/Event Control “Columns” dialog box

The parameters available in this dialog box are:

- The “Available” list contains all of the column types available for this object. The “Visible” list contains all of the column types currently in use for the object. Click the “>>” and “<<” buttons to move selections between the two lists.



Use the “Columns” dialog box to display the most recently replaced value with the new value. To do so, move both “Value” and “Previous” from the “Available” list to the “Visible” list.

Click the “Move Up” or “Move Down” buttons to rearrange the order of columns in the Visible list.

- “Label” and “Width” fields change the default column labels and widths at runtime.
- “Align” specifies alignment (“Left”, “Center”, or “Right”) for the alarm message text within a specified column.
- “Available during runtime”, when selected, allows the user to add selected columns to the visible list during runtime.
- “Key” assigns a shortcut to each column. This allows operators to sort the information on the Alarm Control object by any column, using keyboard keys instead of the mouse cursor.

When you are finished, click the “OK” button to close the “Columns” dialog box.

Filters

You access the “Filters” dialog box from the Alarm/Event Control “Object Properties” dialog box when you click the “Filters” button..

Figure 6-77 The Alarm/Event Control “Filters” dialog box

- Use the “Group” field to filter alarm messages by one or more user group(s). Type the Group number into the text field (for example, 1). You also can use a comma or a dash to specify more than one group (for example, 1,3,5-6).
- Use the “Selection” field to filter alarm messages by the Selection text configured on the Alarm worksheet.
- Use the “From” and “To” parameters in the Priority group to filter alarm messages based on priority. Type values into the text fields to delimit the priority range.
- Use the “Tagname”, “Message”, and/or “Username” text fields in the “Search in columns” group to specify criteria for filtering alarm messages. Type a tagname, message, and/or user name into the text field for which you want Think & Do to search.
- Use the parameters in the “Interval” group to filter alarm messages by the last x number of messages (“Latest”) or based on a period of time (“Period”).
- Use the parameters in the “Initial Sort” group to set the default sorting order. Select a column type from the “Column” combo boxes, and select “Asc” or “Desc” from the drop-downs to sort in ascending or descending order. Click the “Allow sort in runtime” check box to enable the sort to occur during runtime.



- Configure tagnames (string tags) between curly brackets { } in the “Group”, “Selection”, “Tagname”, “Message”, and “Username” fields to modify the filtering options during runtime.
- Configure integer tagnames for the fields in the “Priority” group and/or on the latest field from the “Interval” group to modify these values during runtime.
- Configure string tagnames for the Period fields in the “Interval” group to modify those values during runtime.
- Use wildcards (* and ?) when specifying values for the “Selection”, “Tagname”, “Message”, and “Username” fields.

Advanced

You access the “Advanced” dialog box from the Alarm/Event Control “Object Properties” dialog box when you click the “Advanced” button.

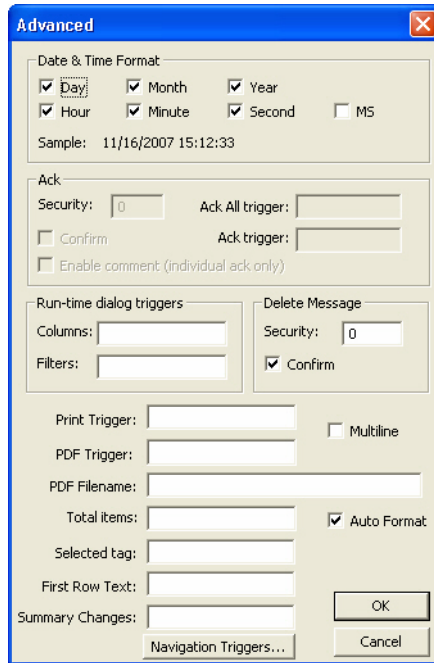


Figure 6-78 The Alarm/Event Control “Advanced” dialog box

- Use the parameters in the “Date & Time Format” group to control which date and time information displays in the alarm message. Click (enable) a check box to include that element in the display. Note: MS stands for milliseconds.



Watch the Sample text to preview how the information will look in the alarm message.

- Use the parameters in the “Ack” group to control how alarms are acknowledged.
 - “Security” field specifies a numeric value to specify which security level can acknowledge an alarm message. Only those users with the specified level can respond.
 - “Ack All trigger” field specifies a tag to receive a value. When the tag changes value, it indicates that all messages in the alarm object have been acknowledged.
 - “Ack trigger” field specifies a tag to receive a value. When the tag changes value, it indicates that the message at the top of the alarm object has been acknowledged.
 - “Confirm” check box, when selected, displays a confirmation dialog box when the user tries to acknowledge a single alarm.
 - “Enable comment” (individual ack only) check box, when selected allows the user to enter comments about the alarm, just after acknowledging it.
- Use the parameters in the “Run-time dialog box triggers” group to control
 - “Columns” field specifies a tag to receive a value. When the tag changes value, it opens a dialog box allowing the user to customize the columns visible in the object.

- “Filters” field specifies a tag to receive a value. When the tag changes value, it opens a dialog box allowing the user to filter the columns visible in the object.
- Use the parameters in the “Delete Message” group to control who can delete alarm messages from the Alarm History:
 - “Security” specifies which security level can delete alarm messages. Only those users with the specified security level will be allowed to delete an alarm message.
 - “Confirm”, when selected, requires the user to confirm a message deletion before Think & Do actually deletes the selected alarm message.
- “Print trigger” field specifies a tag in this field to print an alarm summary from your default printer when this tag changes value.
- “PDF Trigger” field specifies a Tag in this field. When the value of the Tag changes, the data currently filtered in the Alarm/Event Control is distilled to a PDF file and saved to the path specified in the PDF Filename field below.
- “PDF Filename” field specifies a complete file path and name where the PDF file is to be saved. You can also enter a tagname using the {tag} syntax.



PDF Trigger and PDF Filename are not supported in applications running on Windows CE or Web Thin Client.

- “Total items” field specifies an integer tag to see how many alarms remain after Think & Do filters the alarm object using parameters specified on the “Filters” dialog box.
- “Selected tag” field specifies a string tag to enable the end user to click on an alarm message to see the name of the tag associated with that alarm event.
- “First Row Text” field specifies a string tag. This tag will receive the text of all fields from the first row of the Alarm/Event Control. The fields are tab delimited. Whenever the first row changes — either due to a new Alarm/Event, or simply because the rows are reordered — the value of the configured tag is updated.
- “Summary Changes” field specifies an integer tag. This tag will receive a running count of the number of changes in the Alarm/Event Control. For example, when a new Alarm occurs or when an Alarm is acknowledged, the value of the configured tag will be incremented. Reordering the rows is not counted as a change.
- Click the “Navigation Triggers” button to open the following dialog box.

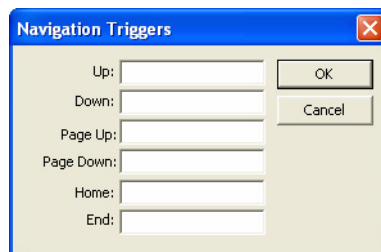


Figure 6-79 The Alarm/Event Control “Navigation Triggers” dialog box

Make the on-screen Alarm Control object scroll up, scroll down, page up, page down, go to home (beginning) of page, or go to end of page by configuring tags in the corresponding fields. Whenever the values of the configured tags change, the Alarm Control object will navigate that way. This is useful for adding navigation controls to the screen; for example, if you configure the same tag to the Up field in this dialog box and a Pushbutton object, then the Alarm Control object will scroll up whenever the Pushbutton object is pressed.

When finished, click the “OK” button to close the “Advanced” dialog box.

6.3.14 Configuring an Event

Use the Alarm/Event Control screen object to view a list of all logged events. Use the following steps to configure this object:

1. Click the Alarm/Event Control tool to add an alarm/event control object to your application screen.
2. Double-click on the new object to open an “Object Properties” dialog box.

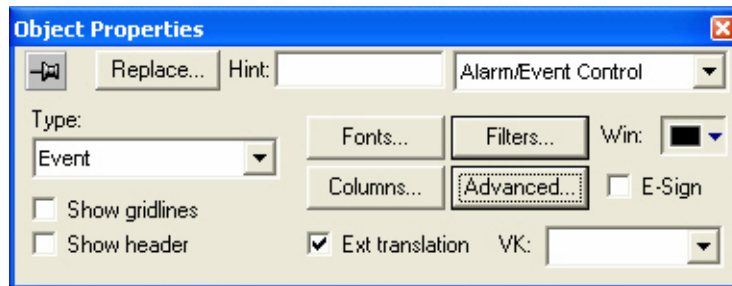


Figure 6-80 Object Properties: Alarm/Event Control

3. Click the “Type” combo-box drop-down arrow and select “Event” or “Alarm History + Event” from the list.

Use most of the same parameters to configure Event objects and Alarm History objects. The only difference is that the Selection, Group, and Priority filters are disabled for the Event-type object. For details, see “Configuring an Alarm” on page 6-68.

6.3.15 Trend Control Development Interface

This section describes the development interface and all the settings available to you as you configure the Trend Control object on the screen.

Although the Trend Control object supports flexible configurations to meet the specific needs of your application, most of the settings are set by defaults based on the most common interfaces. Therefore, in many cases, you will only configure data points (displayed during the runtime), which can be done easily by clicking the “Points” button from the “Object Property” dialog box.

Trend Control
Tool

Click the Trend Control tool to add it to your application screen. Double-click on the object to launch its “Object Properties” dialog box.

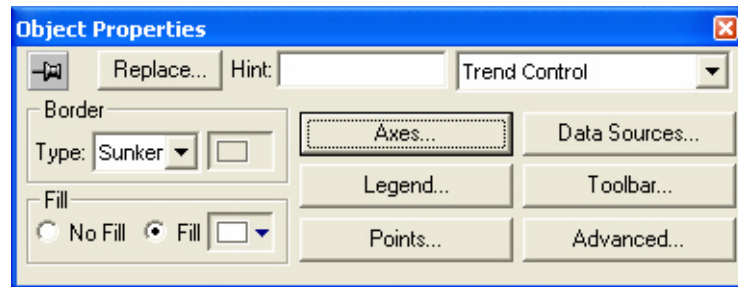


Figure 6-81 Object Properties: Trend Control

This “Object Properties” dialog box provides the following parameters:

- “Border” group specifies a border line “Type” (style) by selecting “None”, “Solid”, “Dashed”, “Etched”, “Raised”, or “Sunken”. You can also select the color of the border line with the color box to the right of the “Type” field.
- “Fill” group lets you can choose a background color for the Trend Control object by selecting it from the color box to the right of the “Fill” radio button. If you select “No Fill”, the background of the Trend Control object will remain transparent.

The buttons on this dialog box launch other dialog boxes for configuring specific settings for the Trend Control object. They are:

- Data Sources
- Points
- Axes
- Legend
- Toolbar
- Advanced

Data Sources

The “Data Sources” button on the Trend Control “Object Properties” dialog box launches the “Data Sources” dialog box.

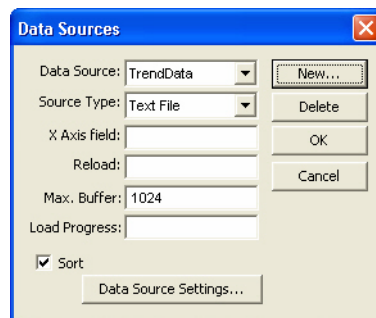


Figure 6-82 The Trend Control “Data Sources” dialog box

The data source defines the location of the values from the data point(s) associated with it. Many points can share the same data source – you do not need to create one data source for each data point.

The data source tag is available by default to the Trend Control object. You can add additional data sources with the “New” button. The name you enter will be used as an alias to link the data points to this new data source.

The other fields in this dialog box allow you to edit the data source settings:

- “Source Type” specifies the source type of the location of the data point values.
- “X-Axis field” specifies the name of the field (column) from the data source that holds the X axis data.
- “Max. Buffer” specifies the maximum amount of data (in bytes) that will be held in runtime memory.
- “Load Progress” specifies the tag to receive a real value (0-100) that represents the percentage of the Data Source load progress.
- “Sort”, when selected, sorts the data and shows the Cursor column value until the Max. Buffer is filled. When disabled (unchecked), the data are not sorted and the Cursor column value is not shown.
- “Data Source Settings”, when selected defines the settings for the “Source Type”. The settings for each “Source Type” are shown in Table 6-10.

Table 6-10 Data Source Settings, “Source Type” Actions

Data Source Type	Description	X-Axis field	Data Source Settings
Batch	Batch generated by the Trend task of Think & Do	Disabled. The X-Axis data will be retrieved automatically on the correct position from the proprietary Batch file generated by Think & Do.	Use the displayed dialog box to enter the data point values’ “Batch Name” for their retrieval. You can configure a tag between curly brackets in this field to change this setting dynamically during the runtime.
Database	SQL Relational Database	Field name that contains the X axis data	Configure the settings to link this Data Source to the SQL Relational Database that holds the data point values. See Think & Do online help for more information on the “Database Configuration” dialog box.
Text File	Text file (e.g. CSV file) with data point values separated by a specific delimiter	Number of the column that holds the X-Axis data. The number 0 refers to the first column, 1 refers to the second column, and so on.	Use the “Grid Data - Text File” dialog box to enter the name of the text file that holds the data points (see “Data Source – Text File” on page 6-97).

Points

The “Points” button on the Trend Control “Object Properties” dialog box launches the “Points” dialog box.

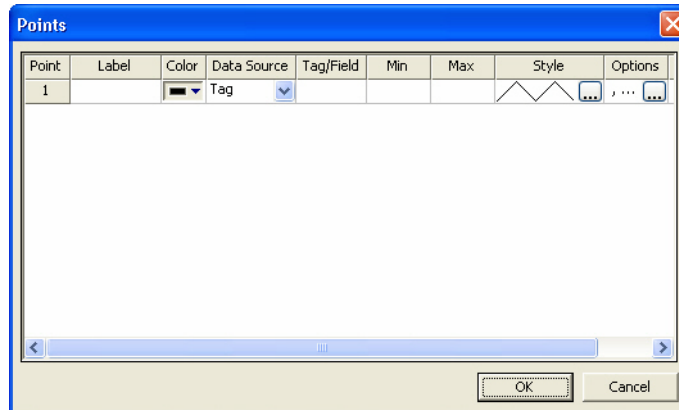


Figure 6-83 The Trend Control “Points” dialog box

The value of each data point can be represented in the Trend Control object as a pen, during the runtime. You can select which data points will be visible during the runtime (add/remove pens to the chart), regardless of how many data points you associate with the Trend Control object.

The properties of each data point are:

- “Point” specifies the Data Point ID. Each data Point has a unique ID, which is assigned automatically when the data point is created in this interface.
- “Label” specifies the label associated with the Data Point can be displayed on the Legend, during the runtime, providing a short reference to the user for each data point.
- “Color” specifies the color of the pen used to draw the values of the Data Point on the Trend Control object
- “Data Source” specifies the data source that holds the values for the data point. The Data Source Tag is available by default. See Data Sources for further information about how you can make additional data sources available for the object.
- “Tag/Field” value depends on the “Data Source” type associated with the data point:
 - “Tag” specifies the name of the tag with values to display. If the tag is configured in the Trend task, the history data is automatically retrieved; otherwise, only the online values are displayed.
 - “Batch” specifies the name of the tag with values to be retrieved from the Batch History file generated by the Trend task, and displayed on the object.
 - “Database” specifies the name of the field (column) in the SQL Relational Database that holds the data point values.
 - “Text File” specifies the number of the column that holds the data point values. The number 0 refers to the first column, 1 refers to the second column, and so on.
- “Min” is the minimum value displayed in the Y scale for the data point
- “Max” is the maximum value displayed in the Y scale for the data point
- “Style” configures the style for the pen (color, type, state, etc.). See “Pen Style” dialog box for more information.

- “Options” configures optional settings for each data point. You can use the dialog box to configure these settings or type their values directly in the “Options” field, using the comma character as the delimiter. See “Options” dialog box for more information about these settings.
- “Hide”, when the tag specified has the value 0, the pen associated with the point is displayed on the object; otherwise, it is hidden.

Pen Style

The “Pen Style” dialog box allows you to configure the style of the pen used to draw the data point values on the object during the runtime. Also, it can be launched during the runtime, allowing the user to customize these settings on-the-fly.

You have the option of defining a Hi Limit and a Lo Limit for each data point, with the “Options” dialog box. The “Pen Style” dialog box allows you to configure different settings for the pen (e.g. color), both when its values are within the limits (Normal State) and not within the limits (Out of Limits state).

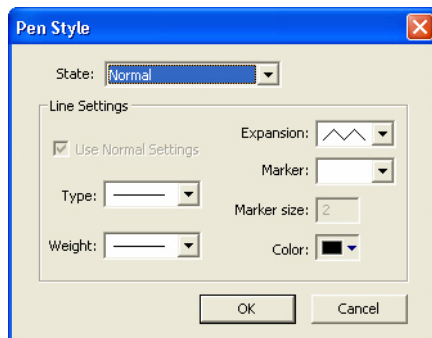




Figure 6-84 The Trend Control “Pen Style” dialog box

After you select a “State” (“Normal” or “Out of Limits”), you can configure its pen style. The properties are:

- “Use Normal Settings” is available only for the “Out of Limits” state. When checked, the pen will always be displayed with the settings for the “Normal” state, even if the data point values are not within the limits configured for it.
- “Type” specifies the type of the line used to draw the pen.
- “Weight” specifies the weight (thickness) of the line used to draw the pen.
- “Expansion” specifies the algorithm used to link the points, as follows:
 -  specifies that consecutive points are interpolated directly to each other using a line. This option is suitable for analog values.
 -  specifies that consecutive points are linked through vertical and horizontal lines only (steps). This option is suitable for Boolean values.
- “Marker” specifies the type of marker (if any) that must be displayed on each specific sample retrieved from the data source and displayed on the object.
- “Color” specifies the color of the marker (if any) and line used to draw the pen on the object.
- “Marker Size” specifies the size of the marker (if any).



When running the application under the Windows CE operating system or on the Web Thin Client (any OS), the “Pen Style” dialog box – available during the runtime – allows the user to change the pen color only.

Options

Use this dialog box to configure optional settings for each data point.

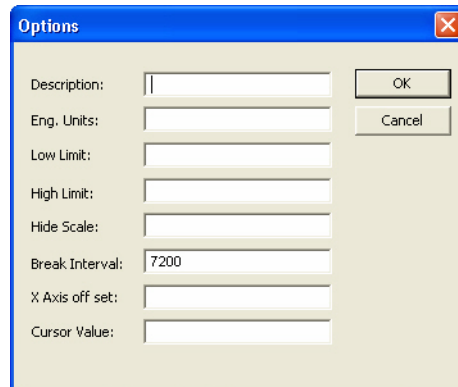


Figure 6-85 The Trend Control “Options” dialog box

- “Description” specifies the text that can be displayed in the legend, providing a short description about the data point, during the runtime. When tags are used, the default description is the one configured for the tag.
- “Eng. Unit” specifies the text that can be displayed in the legend, providing the Engineering Unit associated with the data point, during the runtime. When tags are used, the default units are the ones configured for the tag.
- “Lo Limit”, when the data point value is below this limit, specifies a different pen style (e.g. color) to use. See “Pen Style” dialog box for further information. When tags are used, the default Low Limit is the Low Alarm value configured for the tag.
- “Hi Limit”, when the data point value is above this limit, specifies a different pen style (e.g. color) to use. See “Pen Style” dialog box for further information. When tags are used, the default High Limit is the High Alarm value configured for the tag.
- “Hide Scale” specifies a tag in this field to control the visibility of the scale (Y axis) associated with this pen during the runtime by changing the value of this tag (0=Show ; 1=Hide).
- “Break Interval” specifies a maximum interval between two consecutive points. If the time between two consecutive samples is higher than this number (in seconds), the Trend Control assumes that there was no data collection in this period and does not draw a line linking both samples. When the X Axis is configured as numeric, the value on this field represents a numeric scalar value. If the X Axis is configured as date/time, the value for this field is given in seconds.
This field has some special values:
 - -1 : Do not connect the points.
 - -2 : Connect only points in ascending order.
- “X Axes off-set” specifies the off-set for this data point from the X-Axis scale configured for the object. This option is useful when you want to display data from two or more data points using a different X scale (period of time/value) for each one, so you can compare them. When the X Axis is configured as numeric, the value on this field represents a numeric scalar value. If the X Axis is configured as date/time, the value for this field is given in seconds.

Think & Do

- “Cursor Value” specify a tag in this field. During the runtime, the Trend Cursor object updates the value of this tag with the value of the intersection between the data point pen and the vertical cursor (if any).

Axes

The “Axes” button on the Trend Control “Object Properties” dialog box launches the “Axes” dialog box.

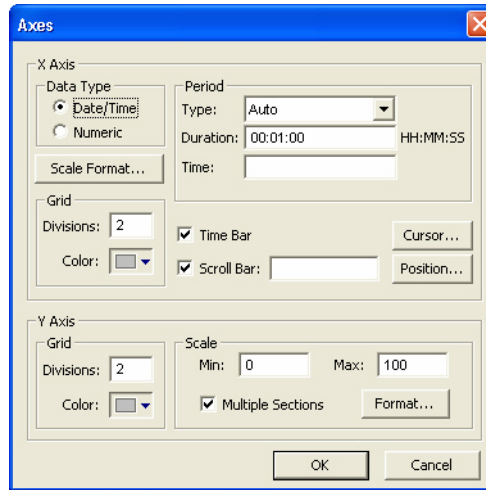
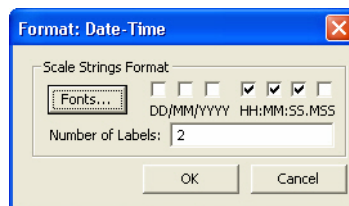


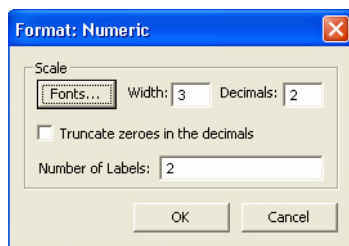
Figure 6-86 The Trend Control “Axes” dialog box

This dialog box lets you configure the settings for the X and Y axis.

- “Data Type” specifies that the X axis can display either Date/Time values or numeric values, according to this setting.
 - For “Date/Time”, the “Scale Fromat” button displays the following dialog box:



- For “Numeric”, the “Scale Fromat” button displays the following dialog box:





The number of decimal points for the X or Y scale (Decimals) can be configured with a tag. Therefore, this setting can be modified dynamically during the runtime.

- “Period” depends on the “Data Type” configured for the X axis, as shown in Table 6-11.

Table 6-11 “Period” Properties Depend on “Data Type”

Data Type	Property	Description
Date/Time (Period)	Type	<ul style="list-style-type: none"> – “Start Date/Time”, when selected, the value of the tag configured in the Time field defines the starting Date/Time for the data displayed on the object. – “Time Before Now”, when selected, the value of the tag configured in the Time field defines the amount of time before the current Date/Time, which will be used as the starting Date/Time for the data displayed on the object. – “Auto”, when selected, the Trend Control object works with “Start Date/Time” when it is triggered to “Pause Mode”, and it works with “Time Before Now” when it is triggered to “Play Mode”.
	Duration	Defines the “Period” of data displayed on the object. You can configure a string tag in this field, so you can change the duration dynamically during the runtime by changing the value of this tag. The format of the value supported by this property is HH:MM:SS. For example, 36:00:00 (thirty six hours).
	Time	<p>This field is optional. The value of the tag configured in this field represents a period of time, rather than a specific date or time. The meaning of this value depends on the option set for the Type property.</p> <ul style="list-style-type: none"> – When the “Type” is set as “Start Date/Time”, the value of the tag configured in this field must comply with the format Date Time. For example, 02/10/2005 18:30:00. – When the “Type” is set as “Time Before Now”, the value of the tag configured in this field must comply with one of the following formats: <ul style="list-style-type: none"> • Time (string value). For example, 48:00:00 (forty eight hours). • Number of hours (real value). For example, 2.5 (two hours and thirty minutes).
Numeric	Min	Minimum value displayed on the X-axis.
	Max	Maximum value displayed on the X-axis.



The tags configured in the Period/Range fields are automatically updated when the user changes the X scale dynamically during the runtime, using the Time bar embedded in the object.

If the Time field is left blank (or if the tag configured in this field has the value 0), the object displays data up to the current Date/Time.

- “Grid” (X axis or Y axis) configures the number of divisions (vertical or horizontal lines) drawn on the object for the X and/or Y axis respectively, as well as the color of these lines.

- “Time bar”, when selected, the Time bar is displayed below the X axis during the runtime; otherwise, it is hidden. The time bar is a standard interface that can be used by the operator to change the X axis scale during the runtime.
- “Scroll bar”, when selected, the Scroll bar is displayed below the X axis during the runtime; otherwise, it is hidden. The time bar is a standard interface that can be used by the operator to navigate through the X axis scale during the runtime. Optionally, you can configure a tag in the Scroll bar field, which defines the period for the scroll bar. If this field is left empty, the period is equal to the current value for Duration of the X axis.
- “Cursor” specifies an optional ruler orthogonal to the X axis, which can be used during the runtime to obtain the value of any pen at a specific point (intersection of the pen with the cursor). When you click this button, the “Cursor” dialog box launches, where you can configure the settings for the optional vertical cursor.

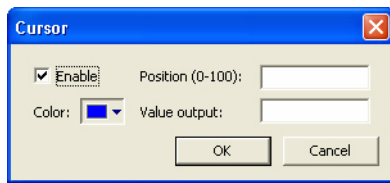


Figure 6-87 The Trend Control “Cursor” dialog box

- “Enable”, when selected, the vertical cursor is visible during the runtime.
 - “Color” specifies the color of the line drawn for the cursor.
 - “Position (0100)” You can configure a numeric tag in this field, which is proportional to the position of the cursor on the X axis, from 0 to 100%. When this value is changed, the position of the cursor is automatically modified.
 - “Value Output” specifies a string tag that returns the value of the X axis in which the cursor is currently positioned.
- Position defines the position of the X-axis, as well as its direction and orientation.

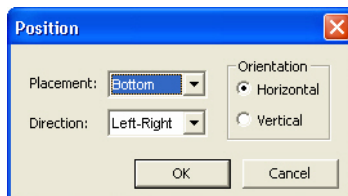


Figure 6-88 The Trend Control “Position” dialog box

Available parameters in the “Position” dialog box are:

- “Placement” specifies where the X axis will be placed.
 - “Direction” specifies the direction of the X-axis.
 - “Orientation” specifies the orientation of the X-axis.
- Scale defines the properties of the Y axis, as follows:
 - “Min/Max” specify default minimum and maximum values displayed in the Y axis. Used when more than one pen shares the same scale (Multiple Sections disabled), and/or for the points whose Min and Max fields are not configured (left blank).
 - “Auto Scaling”, when checked, the Y scale is dynamically and automatically adjusted during the runtime so the values currently displayed fit it.

- “Multiple Selections”, when selected, the Y scale is divided automatically into one section for each pen; otherwise, all pens share the same Y scale.
- “Format” launches a dialog box for configuring the format of the labels displayed by the Y axis.

Legend

The “Legend” button on the Trend Control “Object Properties” dialog box displays the “Legend” dialog box.

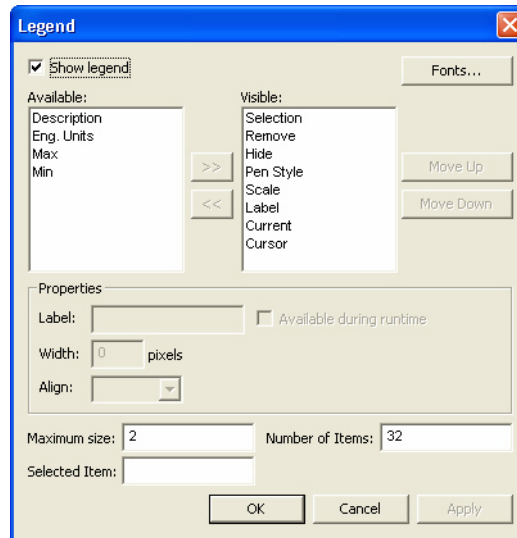


Figure 6-89 The Trend Control “Legend” dialog box

- “Show legend”, when selected, the embedded legend is displayed during the runtime. This interface provides useful information associated with the pens currently linked to the object.
- “Available” / “Visible” specifies fields displayed in the legend during the runtime. You can add fields to and remove them from the Visible box using the >> and << buttons respectively. Moreover, you can use the Move Up and Move Down buttons to change the order in which the fields are displayed in the legend during the runtime.
- “Properties” specifies properties for the field highlighted in the Available or Visible box:
 - “Label” specifies a label for the field displayed during the runtime
 - “Width” specifies the width for the field (in pixels) during the runtime.
 - “Align” specifies the alignment of the data displayed in the field
 - “Available during runtime”, when selected, the user can show or hide the field during runtime.
- “Maximum size” defines the size of the legend in terms of number of rows. For instance, the user might have 8 points being displayed in the trend object, if the maximum size is set to two, the legend will have a scroll bar to allow the user to scroll to the other points.
- “Number of items” specifies the number of points (default) displayed on the legend. You can allow the user to add/remove points during the runtime regardless of the value in this field.
- “Selected Item” specify a numeric tag. The object writes in this tag the number of the selected row. In addition, you can select different rows by writing their values in this tag.

- “Fonts” sets the font for the text displayed in the legend.

Toolbar

The “Toolbar” button on the Trend Control “Object Properties” dialog box displays the “Toolbar” dialog box.

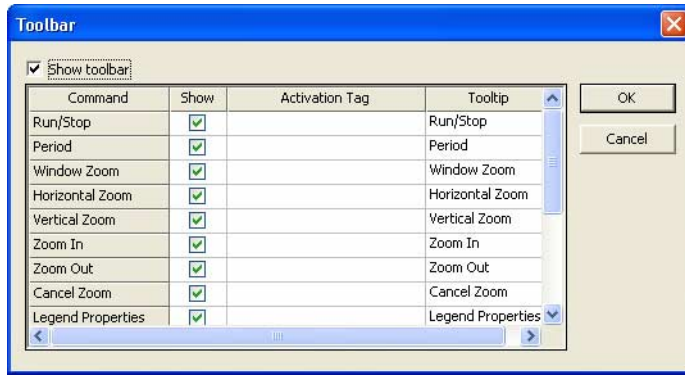


Figure 6-90 The Trend Control “Toolbar” dialog box

“Show toolbar”, when selected, the embedded toolbar is displayed during the runtime. This interface provides useful buttons to trigger actions associated with the object.

Configure the following settings for each Command (button) available for the toolbar:

- “Show”, when selected, the button is displayed on the toolbar embedded on the Trend during the runtime.
- “Activation Tag” specify a tag in this field (optional). When the tag changes value, it triggers the respective command. This option is useful when you want to create customized interfaces to trigger the commands, instead of (or redundant with) the embedded toolbar.
- “Tooltip” specifies the text to display as a tooltip during the runtime when the mouse cursor is on the icon of the toolbar.

Advanced

The “Advanced” button on the Trend Control “Object Properties” dialog box displays the “Advanced” dialog box.

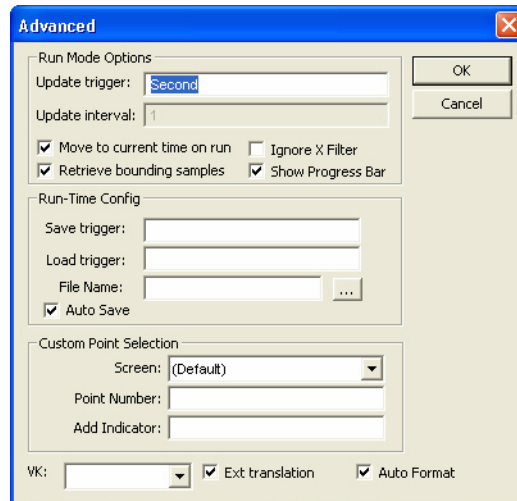


Figure 6-91 The Trend Control “Advanced” dialog box

- “Run Mode Options” define the behavior of the trend when in run mode:
 - “Update trigger”, when the tag configured in this field changes value, the trend object is updated (refreshed).
 - “Update interval”, when the update trigger is issued and the X Axis if of type numeric, the value on this field will be added to the minimum and maximum values of the X Axis.
 - “Move to current time on run”, when selected, X axis shifts to the current time automatically when the object is triggered to Play mode, during the runtime.
 - “Retrieve bounding samples”, when selected, the object retrieve the data out-bound the object (first points only). Uncheck this option can improve the performance, since the points outbound the object will not be retrieved from the history. On the other hand, the object will not draw lines linking the first and last samples to the extremities of the object.
 - “Ignore X Filter”, when selected, the X Filter is ignored to avoid adding the WHERE or querying clause to the Data Sources.
 - “Show Progress Bar”, when selected, a progress bar will appear during the trend load.
- “Run-Time Config” specifies how the settings of the Trend object modified during runtime can be saved in temporary files. This option can be used to:
 - Keep the settings consistent, so when the user closes the screen and opens it again, or re-starts the application, the settings configured during the runtime are not lost.
 - Create standard settings for different scenarios and load the appropriate configuration during the runtime, based on a pre-defined condition or based on the user-selection.

- The properties of this frame are described in the following table:
 - “Save trigger” specifies the tag that changes value (e.g. toggles) to cause the current settings of the Trend object to be saved in the temporary file. This command is not available for the Web Thin Client.
 - “Load trigger”, when the tag configured in this field changes value (e.g. toggles), the settings from the temporary file are loaded and applied to the Trend object during the runtime.
 - “File Name” specifies a file name. If this field is left blank, the temporary file is saved in the application “\Web” sub-directory with the syntax “<Screen-Name><ObjectID>TrendControl.stmp” (e.g. “MyScreen10TrendControl.stmp”). The Web Thin Client station saves/loads the temporary file in the standard Temp directory of the operating system (e.g. “\Documents” and “Settings\<CurrentUser>\Local Settings\Temp”).

You can configure a customized file name for the temporary file in this field or even configure a string tag between curly brackets, so the user can change the name of the configuration file dynamically during the runtime by changing the value of this tag. If you do not specify any path, the file is saved in the application “\Web” sub-directory by default.
 - “Auto Save”, when selected, the current settings of the Trend are automatically saved in the temporary file when the screen where the Trend is configured is closed during the runtime. If the box is not checked, the settings are saved only when the Save trigger command is executed.



After the screen where the Trend object is configured is saved, the settings are not automatically loaded from the temporary file when the screen is opened again, unless the Load trigger command is executed before the screen is closed.

- “Custom Point Selection”, lets you create a custom dialog box to modify or insert pens to the object, as follows:
 - “Screen” specifies the name of the screen which must be launched when the user triggers a command to modify or insert a new pen to the object during the runtime.
 - “Point Number” specifies the point number (from the “Points” dialog box), indicating the point associated with the pen that will be inserted or modified during runtime.
- “Add Indicator” indicates that the user triggered an action to insert a new pen (value 1) instead of modifying a pen that is already been visualized (value 0).
- “VK” specifies the Virtual Keyboard type used for this object.
 - “Ext Translation” enables translation of the text when a translation file is specified via scripting.

6.3.16 Trend Control Runtime Interface

When enabled, some embedded interfaces can help the user to interact with the Trend Control during the runtime. This section describes these interfaces:

- “Toolbar” specifies commands available in the embedded Toolbar. Available commands are described in Table 6-12.

Table 6-12 ScreenView Toolbar Icons






















Command	Icon	Description	Activation Tag
Run		Set the Trend to the Play Mode. In this mode, the X axis is continuously updated (Online Mode). This option is disabled (grayed out) when the trend is already in Play Mode.	0 = Play Mode on 1 = Play Mode off
Stop		Set the Trend to the Stop Mode. In this mode, the X axis is not continuously updated (History Mode), so the user can visualize history data in a frozen period of time. This option is disabled (grayed out) when the trend is already in Stop Mode.	0 = Stop Mode on 1 = Stop Mode off
Period		Launches an embedded dialog box, where the user can modify the X axis scale main settings	When the Activation Tag changes value (e.g. toggles), this command is executed.
Window Zoom		Allows the user to click on the Trend area and drag the cursor to select the area that must be visible when the cursor is released. This option is disabled (grayed out) when the Multiple Section option (for the Y scale) is active.	
Horizontal Zoom		Allows the user to click on two points on the Trend area, defining the Horizontal scale that must be available	
Vertical Zoom		Allows the user to click on two points on the Trend area, defining the Vertical scale that must be available. This option is disabled (grayed out) when the Multiple Section option (for the Y scale) is active.	
Zoom In		Allows the user to zoom in (display half of the current X and Y scales) each time they click on the Trend area.	
Zoom Out		Allows the user to zoom out each time they click on the Trend area.	

Table 6-12 ScreenView Toolbar Icons

Command	Icon	Description	Activation Tag
Cancel Zoom		Cancel the Zoom selection	When the Activation Tag changes value (e.g. toggles), this command is executed.
Legend Properties		Launches an embedded dialog box, where the user can modify the Legend main settings	
Pen Style		Launches an embedded dialog box, where the user can modify the style of the selected pen.	
Add Pen		Launches a dialog box, where the user can add a new pen to the Trend object	
Remove Pen		Removes the selected pen from the Trend object	
Multiple Sections		Switches the Y scale to Multiple Sections (a section for each pen) or Single Section (all pens share the same Y scale section).	0 = Multiple Sections on 1 = Multiple Sections off
Cursor		Turns the cursor (ruler) to visible or hidden	0 = Cursor on 1 = Cursor off
Auto Scale		Changes the Y axis scale to fit all values from the pens that are currently being monitored.	When the Activation Tag changes value (e.g. toggles), this command is executed.

- “Legend” specifies the commands available in the embedded Legend. Available commands are described in Table 6-13.

Table 6-13 ScreenView Legend Icons

Command	Icon	Description
Selection		Launches a dialog box, where the user can replace the data point associated with the selected pen on the legend
Remove		Removes the selected pen from the Trend object
Hide		When checked, the selected pen is visible; otherwise, it is hidden.
Pen Style		Launches an embedded dialog box, where the user can modify the style of the selected pen.
Scale		When this box is checked, the Y axis scale is visible; otherwise, it is hidden. The scale can be hidden only when the Multiple Sections option is off.

- “Scroll bar” specifies that the user can slide through the X axis values, according to the period configured for this scale.
- “Time bar” specifies that the user can modify the Duration, as well as the Start Date/Time and/or the End Date/Time, for the data displayed on the object. Changing these values will affect the tags associated with the X axis scale (if any).

6.3.17 Configuring ActiveX Control Objects

After inserting the ActiveX control (see “ActiveX Control Object” on page 6-36), double-click the ActiveX control to open the “Object Properties” dialog box.

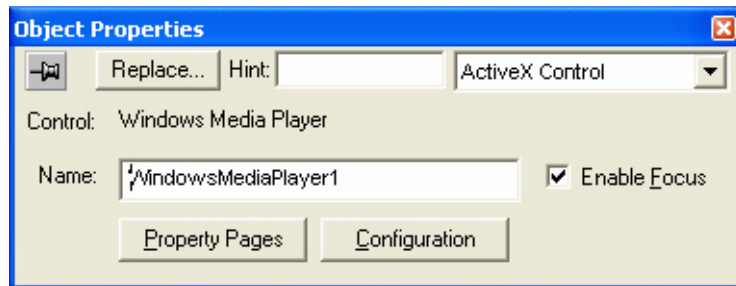


Figure 6-92 Object Properties: ActiveX Control

Click the “Configuration” button to configure the ActiveX control. This displays the “Configuration... Properties” tab shown below.

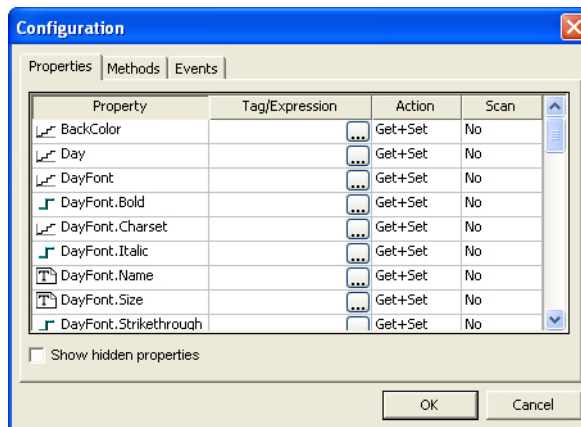


Figure 6-93 The ActiveX Control “Configuration... Properties” tab



Although the “Configuration” dialog box displays the list of all properties, methods and events, you only have to configure the items that you need for your project.







The screen shots used in the following sections depict the Windows Media Player ActiveX control. Although the name of the properties, methods and events varies depending on each ActiveX control, the configuration interface is the same for any ActiveX control. The concepts described here apply to all of them.

The columns in this tab are:

- "Property" lists all properties available from the ActiveX object, and indicate their types with an icon (see Table 6-14).

Table 6-14 ActiveX Property Icons

Property Icon	Property Type
	Boolean
	Integer
	Real
	String

- "Tag/Expression" specifies the tag associated with the respective property of the ActiveX object. The "Action" column defines whether the value of this tag will be written to the ActiveX property, or if the value of the ActiveX property will be written to this tag (or both).



Configure an expression in this field if you want to write the result of an expression to the property of the ActiveX object. However, in this case, the value of the property cannot be read back to one tag (unless you use the XGet() function). Therefore, an expression is configured in this field, the Scan field is automatically set to Set.

- Action defines the direction of the interface between the tag or expression configured in the Tag/Expression field and the ActiveX property, according to Table 6-15.

Table 6-15 ActiveX Actions

Action	Description
Get	Read the value of the ActiveX property and write it to the tag configured in the Tag/Expression field.
Set	Write the value from the tag or expression configured in the Tag/Expression field into the ActiveX property.
Get+Set	Executes both actions (Get and Set). However, when opening a screen with the ActiveX object, Think & Do executes the Get command before executing any Set command. That is, the tag configured in the Tag/Expression field is updated with the value of the ActiveX property when Think & Do opens the screen where the ActiveX is configured.
Set+Get	Executes both actions (Get and Set). However, when opening a screen with the ActiveX object, Think & Do executes the Set command before executing any Get command. That is, the ActiveX property is updated with the value of the tag configured in the Tag/Expression field when Think & Do opens the screen where the ActiveX is configured.



When the value of the property is "Read-only" (cannot be overwritten by the application), the Action field is automatically set to Get.

- Scan defines the polling method to get values from the ActiveX properties, according to Table 6-16.

Table 6-16 ActiveX Scan Polling Methods

Scan	Description
No	The value of the ActiveX property is read and written to the tag configured in the Tag/Expression field, only when the screen with the ActiveX object is open, and when the ActiveX object sends a message to Think & Do to update this tag.
Always	Think & Do keeps polling the value of the ActiveX property and updating the tag configured in the Tag/Expression field with this value.



Some ActiveX controls are designed to send messages to their containers (application) indicating that a property changed value and the new value should be read (Get) again. However, other ActiveX controls do not implement this algorithm. In this case, the only way to get the updated values of the ActiveX properties is to keep polling these values from the ActiveX control (Scan=Always).

Configuring Methods

Select the “Methods” tab to configure the ActiveX methods.

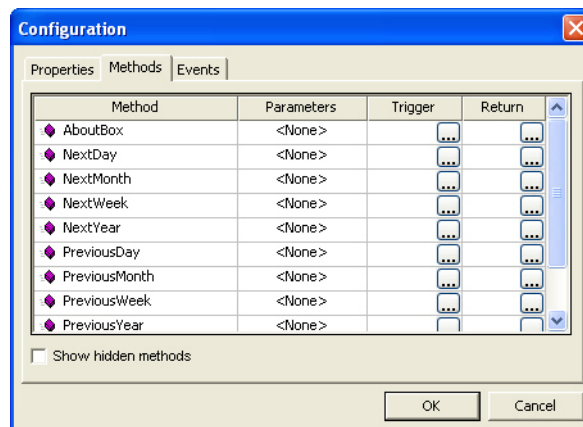


Figure 6-94 The ActiveX Control “Configuration... Methods” tab

The columns in this tab are:

- “Method” lists all methods available from the ActiveX object.
- “Parameters” specifies tags associated with the parameters of the method of the corresponding ActiveX object. If the method does not support any parameter, the fixed text <None> is displayed in the Parameters field. Otherwise, you can type the tags associated in the parameters of the ActiveX object. When the method has more than one parameter, you can type one tag for each parameter, separating them by a comma (.). For

example, TagA , TagB , TagC. When the method is executed, either the value of the tags are written to the parameters of the method (input parameters), or, after the method is executed, the ActiveX writes the value of the parameters to the tags (output parameters).



When you click the Browse button (...), it will display the list of parameters supported by the method, allowing you to associate one tag with each parameter.

- “Trigger”, when the tag configured in this field changes value, the respective method of the ActiveX control is executed.
- “Return” specifies the tag that receives the value returned by the method (if any).

Configuring Events

Select the “Events” tab to configure the ActiveX Events.

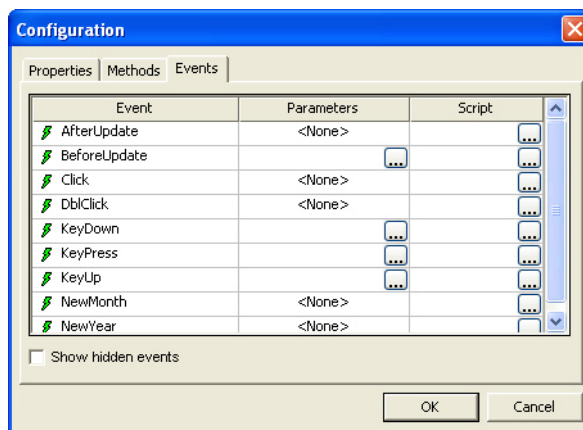


Figure 6-95 The ActiveX Control “Configuration... Events” tab

The columns in this tab are:

“Event” lists all events available from the ActiveX object.

- “Parameters” specifies tags associated with the parameters of the event of the corresponding ActiveX object. If the event does not support any parameter, the fixed text <None> is displayed in the Parameters field. Otherwise, you can type the tags associated with the parameters of the ActiveX object. When the event has more than one parameter, you can type one tag for each parameter, separating them by a comma (.). For example, TagA , TagB , TagC. When the event is generated, either the value of the tags are written to the parameters of the event (input parameters), or the parameter values are written to the tags (output parameters).



When you click the Browse button (...), it will display the list of parameters supported by the event, allowing you to associate one tag with each parameter.

- Script specifies the script that will be executed when the event is triggered by the ActiveX control.



When you click the Browse button (...), it will display a dialog box with the complete script associated with the event. The main dialog box displays only the expression configured in the first line of the script.

6.3.18 Configuring .NET Control Objects

After inserting the .NET Framework control (see “.NET Control Object” on page 6-38), double-click the .NET control to open the “Object Properties” dialog box.

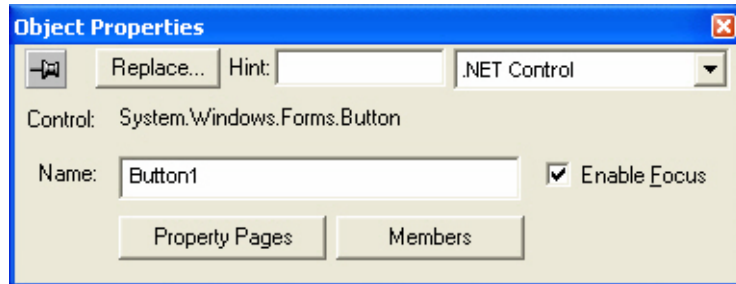


Figure 6-96 Object Properties: .NET Control

Click the “Members” button to configure the .NET Framework control. This displays the “Members... Properties” tab shown below.

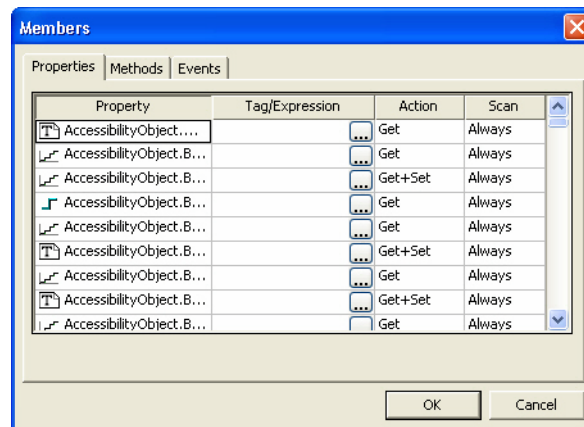


Figure 6-97 The .NET Control “Members... Properties” tab



Although the “Members” dialog box displays the list of all properties, methods and events, you only have to configure the items that you need for your project.







The screen shots used in the following sections depict the CheckBox component. Although the names of properties, methods and events varies by component, the configuration interface is the same for any .NET Component. The concepts described here apply to all of them.

The “Properties” tab provides a grid with the following fields:

- “Property” list all properties available from the .NET Component, and indicate their types as shown in Table 6-17.

Table 6-17 .NET Framework Property Icons

Property Icon	Property Type
	Boolean
	Integer
	Real
	String

- “Tag/Expression” specifies the tag associated with the respective property of the .NET Component. The “Action” column will define whether the value of this tag will be written to the property, or if the value of the property will be written to this tag (or both).
- “Action” defines the direction of the interface between the tag or expression configured in the “Tag/Expression” field and the .NET property, according to Table 6-18.

Table 6-18 .NET Framework Actions

Action	Description
Get	Read the value of the property and write it to the tag configured in the Tag/Expression field.
Set	Write the value from the tag or expression configured in the Tag/Expression field into the property.
Get+Set	Executes both actions (Get and Set). However, when opening a screen with the .NET Component, Think & Do executes the Get command before executing any Set command. That is, the tag configured in the Tag/Expression field is updated with the value of the property when Think & Do opens the screen where the .NET Component is configured.
Set+Get	Executes both actions (Get and Set). However, when opening a screen with the .NET Component, Think & Do executes the Set command before executing any Get command. That is, the property is updated with the value of the tag configured in the Tag/Expression field when Think & Do opens the screen where the .NET Component is configured.



When the value of the property is “Read-only” (cannot be overwritten by the application), the Action field is automatically set to Get.

- “Scan” defines the polling method to get values from the properties. For .NET Components, all properties scan Always by default. That is, Think & Do keeps polling the value of the property and updating the tag configured in the Tag/Expression field with this value.

Configuring Methods

Select the “Methods” tab to configure .NET Framework objects.

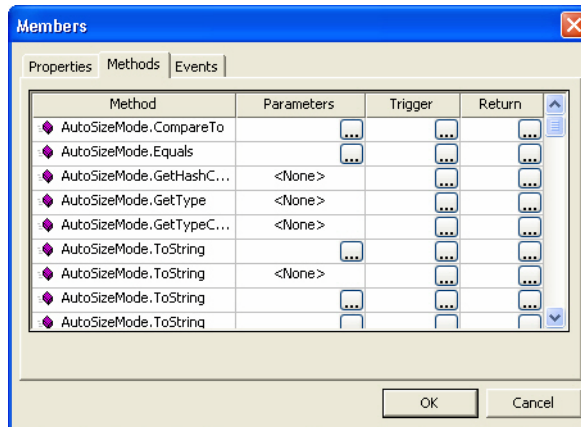


Figure 6-98 The .NET Control “Members... Methods” tab

The “Methods” tab provides a grid with the following fields:

- “Method” lists all methods available from the .NET Component.
- “Parameters” specifies the tags associated with the corresponding method. If the method does not support any parameter, then the fixed text <None> is displayed. Otherwise, you can enter the tags that you want to associate with the parameter. When the method has more than one parameter, you can enter one tag for each parameter, separating them by a comma (.). For example, TagA , TagB , TagC.



When you click the Browse button (...), it will display the list of parameters supported by the method, allowing you to associate one tag with each parameter.

- When the method is executed, either the value of the tags are written to the parameters of the method (input parameters), or, after the method is executed, the .NET Component writes the value of the parameters to the tags (output parameters).
- “Trigger”, when the tag configured in this field changes value, the respective method of the .NET Component is executed.
- “Return” specifies the tag that receives the value returned by the method (if any).

Configuring Events

Select the “Events” tab to configure the .NET Framework objects.

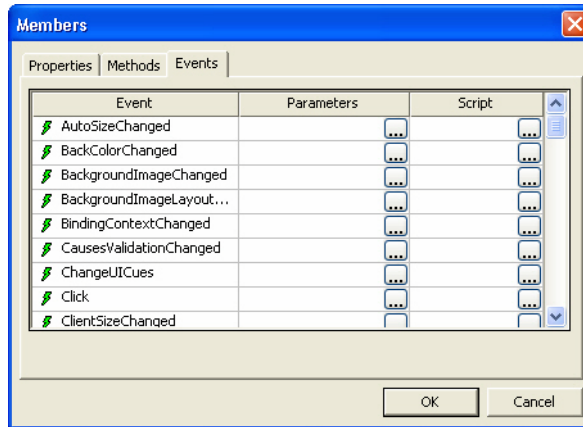


Figure 6-99 The .NET Control “Members... Events” tab

- The “Events” tab provides a grid with the following fields:
- “Event” lists all events available from the .NET Component.
- “Parameters” specifies tags associated with the corresponding event. If the event does not support any parameter, then the fixed text <None> is displayed. Otherwise, you can enter the tags that you want to associate with the parameter. When the event has more than one parameter, you can enter one tag for each parameter, separating them by a comma (.). For example, TagA , TagB , TagC.



When you click the Browse button (...), it will display the list of parameters supported by the event, allowing you to associate one tag with each parameter.

When the event occurs, either the value of the tags are written to the parameters of the method (input parameters), or, after the event occurs, the .NET Component writes the value of the parameters to the tags (output parameters).

“Script” specifies the script that will be executed when the event is triggered by the .NET Component.



When you click the Browse button (...), it will display a dialog box with the complete script associated with the event. The main dialog box displays only the expression configured in the first line of the script.

6.3.19 Configuring Grid Objects

After inserting Grid object (see “Grid Object” on page 6-39), double-click the Grid object to open the “Object Properties” dialog box.

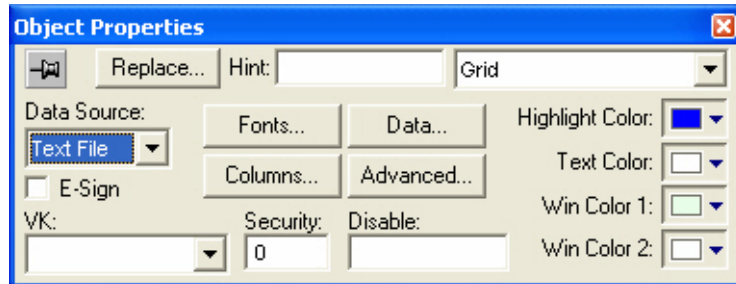


Figure 6-100 Object Properties: Grid

Select a “Data Source” from the drop-down list, click the “Data” button to configure the data used in the grid.

Data Source – Text File

Clicking the “Data” button with “Text File” as the “Data Source” displays the “Grid Data - Text File” dialog box shown below.

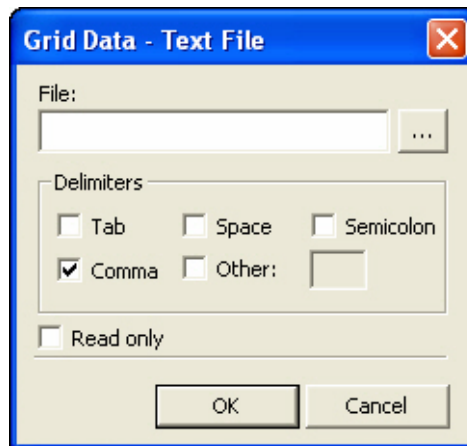


Figure 6-101 The “Grid Data - Text File” dialog box

The parameters in this dialog box are:

- “File” specifies the name of the text file source. You can either type the file name and its path or click the browse (...) button to browse for it. (If the file is stored in the application folder, you can omit the path in the name.)

Think & Do

- “Delimiters” specifies the delimiter(s) used in the data source file. For instance, if the data will be read from a CSV (comma separated values) file, you would select the Comma option. You can even choose a custom delimiter by checking the Other option and typing the custom delimiter in the field beside it.



Configure tagnames between curly brackets {TagName} in the File field.

- “Read Only”, when selected, makes the grid read only.

Data Source – Class Tag

Clicking the “Data” button with “Class Tag” as the “Data Source” displays the “Grid Data - Class Tag” dialog box shown below.

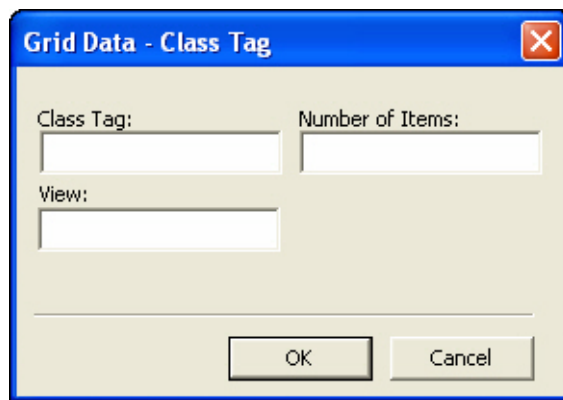


Figure 6-102 The “Grid Data - Class Tag” dialog box

- The parameters in this dialog box are:
- “Class Tag” enter the name of the main class tag source. (Do not specify a specific member of the class tag.) You can specify the initial array position in this field (e.g. Mytag[10]); otherwise, 0 (zero) will be used as the initial position by default.
- “Number of Items” enter the number of array positions from the Class Tag that should be displayed.
- “View”, when the tag configured in the optional field changes value (e.g. toggles) during the runtime, the Grid object launches a dialog box that lets the user show/hide each column or modify their positions.

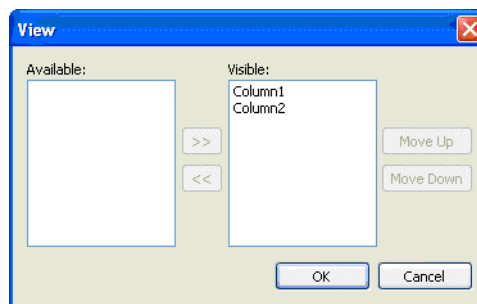


Figure 6-103 The Runtime “View” pop-up dialog box

Data Source – Database

Clicking the “Data” button with “Database” as the “Data Source” displays the “Grid Data - Database” dialog box shown below.

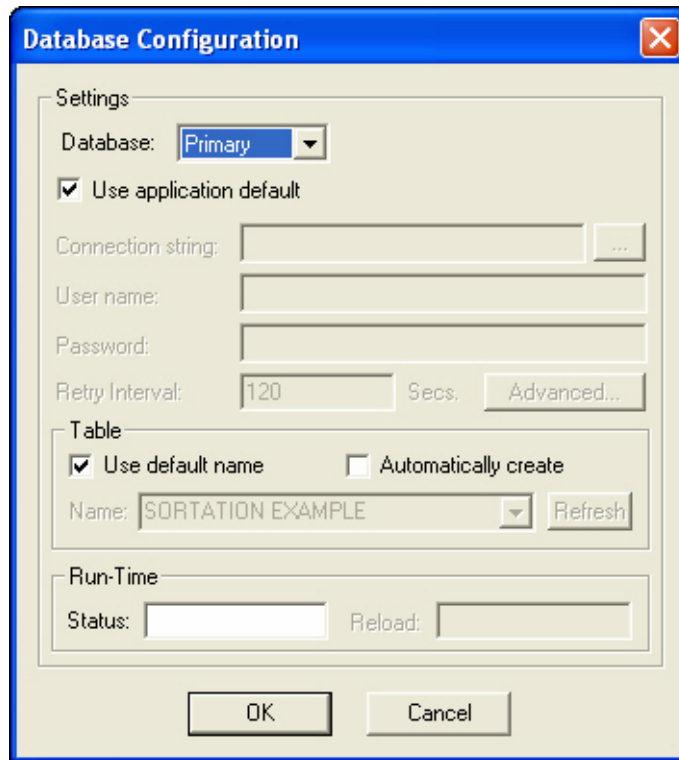


Figure 6-104 The “Grid Data - Class Tag” dialog box

See the “Database Configuration Dialog Box” topic in the Think & Do online Help for more information about this dialog box.

Configuring Columns

Configure the settings for each column displayed by the Grid object during the runtime by clicking the “Columns” button in the Grid “Object Properties” dialog box. This displays the following dialog box.

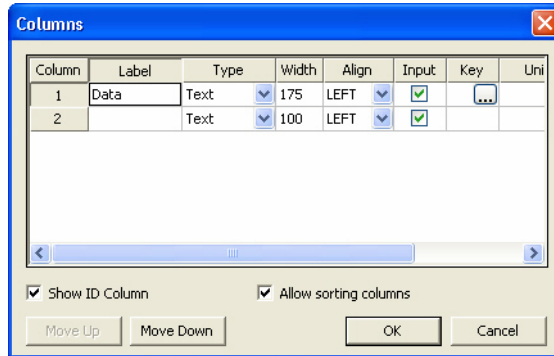


Figure 6-105 The Grid “Columns” dialog box

The columns in this dialog box are:

- “Column” specifies the ID Number that defines the position of the column in the table.
- “Label” enter a Title for each column, which will display as the heading (first) row of the Grid object.



Configure tags between curly brackets in the Label field to modify it dynamically during the runtime. When the label is blank (e.g.: “”), then the width of the column is set to 0 during the runtime. This option is useful to hide columns during the runtime.

- “Field” enter the name of the field (column) in the SQL Relational Database that the Grid object is linked to. If this field is left in blank, the text configured in the Label field will be used as the Field name. (This setting is available only when the Data Source type is set to Database.)
- “Type” select the Type of interface that will be used in the column. The options are shown in Table 6-19.

Table 6-19 Grid Column Types

Type	Description
Text	Displays alphanumeric values
Numeric	Displays numeric values
Picture	Displays the picture (*.bmp or *.ico format) from the data source. For instance, if the value from the data source is MyFile.bmp, the Grid object will display the picture from the file MyFile.bmp stored in the application’s folder. The picture will be automatically resized to fit the cell of the Grid object. The picture file(s) must be stored in the \Web sub-folder of the application to support this feature on the Web Thin Client stations. CEView applications support pictures in bitmap format (*.bmp), but not in icon format (*.ico).

Table 6-19 Grid Column Types

Type	Description
Check-box	Displays check-box interfaces. The check-box will be unchecked if the value read from the file is 0, <NULL> or "FALSE"; otherwise, the check-box will be checked. By default, Think & Do will use the value 0 for unchecked and the value 1 for checked.
Time	Displays the value in the Time format (e.g. HH:MM:SS). This setting is available only when the "Data Source" type is set to "Database".
Date	Displays the value in the Date format (e.g. MM/DD/YYYY). This setting is available only when the "Data Source" type is set to "Database".
Date/Time	Displays the value in the Date/Time format (e.g. MM/DD/YYYY HH:MM:SS). This setting is available only when the "Data Source" type is set to "Database".



Notes:

- When the Data Source type is set to Database, it is important to make sure that the Type for each column configured in the object matches the Type of the respective field in the database.
- When the Data Source type is set to Database, you can configure valid SQL statements, directly in the Field (e.g. List(DISTINCT [Cell_Name]) AS [Cell Name]). You can also configure tagnames between curly brackets to modify this setting during the runtime (e.g. {MyFieldName}).



If Picture is the column type, the Grid object displays a default icon () if the picture file is not found during the runtime. You can configure a different picture to be displayed when the file is not found by copying the picture file to the Web sub-folder of the application and configuring its name on the <ApplicationName>.APP file, as follows:

[Objects]

GridDefaultPicture=<PictureFileName>

- "Width" enter a width of the column, in pixels.
- "Align" select an Alignment for the data shown in the column. There are three options: Left, Right or Center.
- "Input", when selected, allows the user to enter data in this column during the runtime.
- "Key" designates a shortcut for sorting the values in this field. A shortcut is a combination of keys pressed on a keyboard at one time (e.g. <Ctrl>+<C>, <Ctrl>+<V>, etc.). This option is especially useful when creating applications for runtime devices that do not provide a mouse or touch-screen interface and only have a keyboard for interacting with the application during the runtime.



When the "Data Source" type is set to "Class Tag", and the "Columns" dialog box is left blank, the object displays the values from all members of the class tag with the following default column settings:

Label = <Name of the Member from the Class tag>

Type = Text

Width = <Minimum size to display the name of the member from the class tag on the header of the Grid object>

Align = Center

Input = Enabled (checked)

Key = <None>

- “Show ID Column”, when selected, automatically displays the number of each row.
- “Allow sorting columns”, when selected, lets the user sort the values in the columns during the runtime, either by clicking on the label or by using the shortcut configured for each column. This option is disabled if the “Show Header” option from the “Advanced” dialog box is not checked.



Use the Move Up and Move Down buttons to reorder the display of the columns.

Configuring Grid Advanced Parameters

Configure advanced grid settings by clicking the “Advanced” button in the Grid “Object Properties” dialog box. This displays the following dialog box.

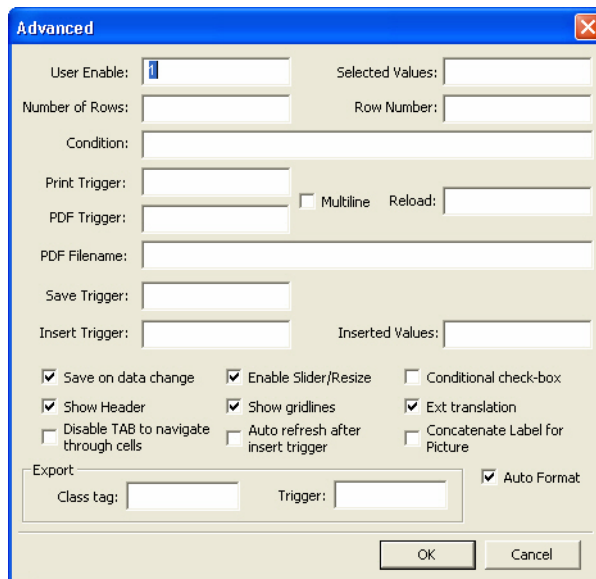


Figure 6-106 The Grid “Advanced” dialog box

This dialog box allows you to configure the advanced settings, as follows:

- “User Enable”, if the value of this tag is TRUE (different from 0), the user can select different rows of the object by clicking on them during the runtime. This field can be configured with a tag or with a numeric value.
- “Selected Values” specifies the array tag used to write values from each column of the selected row to each position of the array. Moreover, you can modify the value of the cells currently selected in the Grid object by changing the value of array tag configured in this field. The initial array position (offset) can be configured in this field.
- “Number of Rows” specifies the tag the receives the number of rows currently available in the Grid object.
- “Row Number” specifies the tag used to write the number of the row currently selected during runtime. In addition, you can select different rows by writing their values in this tag.

- “Condition” specifies the data filter expression. This expression must follow the basic syntax of <ColumnName> <Comparison Operator> <Value> (e.g. ColumnX > 200). When using Text File or Class Tag for Data Sources, the <ColumnName> is the value specified in the Label. When using Database for the Data Source, the column is the value specified in the Field. (In this case, if the Field is left blank, the column value specified is the Label.)



Tips:

- You can combine several conditions simultaneously in the “Condition” field, using the logic operators AND, OR, and NOT. For example, ColumnAge>‘10’ OR ColumnName=‘John’ AND ColumnDate>‘05/20/2003’.
 - You can use wildcards (* and ?) in the Condition field to filter data.
 - You can configure tags between curly brackets {TagName} in the Condition field to change the filtering condition during the runtime.
- “Print Trigger” specifies the tag that when toggled, the data currently filtered in the object is sent to the default printer.
 - “Reload” specifies the tag that when toggled, the object reloads the data from the data source and displays it.
 - “PDF Trigger” specifies a Tag that when toggled, the data currently filtered in the Alarm/Event Control is distilled to a PDF file and saved to the path specified in the PDF Filename field below.
 - “PDF Filename” specifies a complete file path and name where the PDF file is to be saved. You can also enter a tagname using the {tag} syntax.



PDF Trigger and PDF Filename are not supported in applications running on Windows CE.

- “Number of Rows” specifies a tag that receives the number of rows currently available in the Grid object to the tag configured in this field.
- “Save Trigger”, when the tag configured in this field is toggled, the data source (Text File or Database) is updated with the current values of the Grid object. This field is not available when the Data Source type is Class Tag, because the values are automatically updated in the tags as you change a cell in the grid.
- “Insert Trigger”, when Auto refresh after insert trigger is enabled (checked), the tag configured in this field is used as a trigger to refresh the database table. Whenever the value of the tag changes, a new row is added to the table and the values of the array configured in the Inserted Values field are automatically inserted.
- “Inserted Values”, if the Insert Trigger is being used, then the array tag configured in this field provides the values that will be inserted. This field must only contain an array tag, although it can be of any size.
- “Save on data change”, when selected, the values are updated on the data source (Text File or Database) as soon as the user enters a new value on the grid, during the runtime. This option is disabled when the Data Source type is Class Tag, because the values are automatically updated in the tags as the user changes the value of the cells in the grid.
- “Enable Slider/Resize”, if this box is not checked, the user is unable to scroll the list by dragging the slider button, or to change the cell’s size during the runtime.
- “Conditional Check-box”, when selected, the user cannot uncheck a check-box on the Grid during the runtime, unless all preceding check-boxes in the same column are already unchecked. This option is especially useful when you want to oblige the user to follow a pre-defined sequence. This field is not available when the Data Source type is Class Tag.

- “Show Header”, when selected, the header of the Grid object is visible during the runtime, displaying the label of each column.
- “Show gridlines”, when selected, the gridlines of the Grid object are visible during the runtime.
- “Ext. translation”, when selected, the text displayed by the Grid object will be translated when a translation file is specified via scripting.
- “Disable TAB to navigate through cells”, when selected, the user can only navigate through the cells of the Grid Object with the arrow keys, rather than the Tab key. You should disable the Tab key for navigation if you want it to be used for switching to the next object that supports focus on the screen.
- “Concatenate Label for picture”, when selected, the reference name for the picture is the result of the concatenation of the name in the Field column with the value of the Label column. The result will be <Label name>_<Field value>.
- “Export” exports the data from the Grid object to a class-array tag, regardless of the Data Source selected for the object. The fields shown in Table 6-20 must be specified to support this feature.

Table 6-20 Grid Advanced Dialog Box Export Fields

Field	Description
Class tag	Type the main tagname of the class-array tag that will receive the exported values. Each row from the Grid object will be exported to one array position of the array tag, by matching column labels. The initial array position can be configured in this field; 0 is the default.
Trigger	When the tag configured in this field changes value (e.g. toggles), the data is exported from the Grid object to the class-array tag configured in the Class tag field.



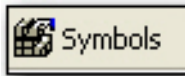
The Export feature is an easy and powerful tool to transfer data from different data sources to tags. After exporting the data to tags, you can use different tasks to manipulate the data, such as the FileWrite() function, or the Recipe or Report tasks to save the data in text files (e.g. CSV files).

6.3.20 Using and Creating Symbols

A symbol is an object (or group of objects) that is saved to the ScreenView “Symbol” folder, so that you can reuse it again and again in your project.

Every time you reuse a symbol, you actually make a copy that is linked to the Master Symbol file in the “Symbol” folder. (These linked copies are also called instances of the symbol.) Thereafter, if you make any changes to the Master Symbol, then those changes automatically propagate to every linked copy in every project.

You can customize each linked copy of the Master Symbol by defining custom properties. For example, when you create a gauge that displays tank levels and then save that gauge as a Master Symbol, you can define custom properties on the symbol that will allow each linked copy to display the level of a different tank.



Symbols
Toolbar Button

To display available symbols, click the “Symbols” button at the left side of the bottom toolbar. This displays the Symbol Library—a dual-pane panel that appears in the ScreenView workspace. Figure 6-107 shows the Symbol Library with the two top-level folders:

- “Application Symbols” provides a listing of symbol folders with symbols used in the current application.
- “System Symbols” provides pre-defined symbol folders with symbols that are available for use in any application.

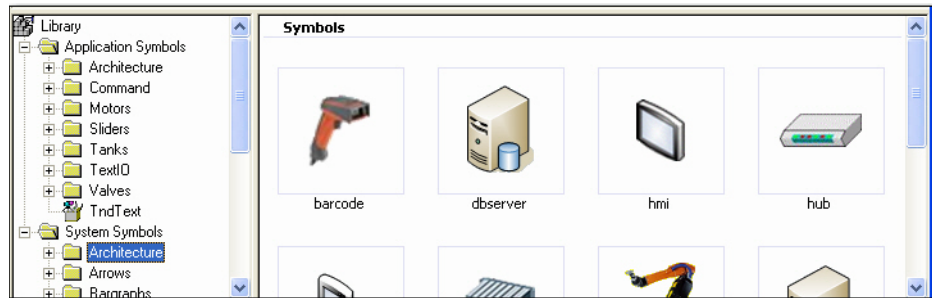


Figure 6-107 Symbol Library

In this example, “Application Symbols” and “System Symbols” folders are open and their sub-folders closed. In addition, the image shows the “System Symbols... Architecture” folder selected and the symbols available in that folder.

Inserting a Symbol

To insert a symbol in a screen follow these steps:

1. Open the desired screen or create a new screen.
2. Open the Symbol Library by clicking the “Symbols” button in the bottom toolbar.
3. Navigate the folders in the left pane of the Symbol Library to find the desired symbol.
4. Select it in the right pane.
5. Click the location on the screen where you want the symbol positioned.

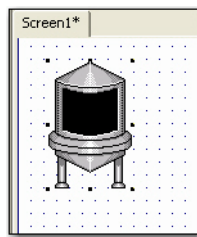


Figure 6-108 Symbol placed on a screen



You can also insert symbols using the “Edit... Paste From” menu. However, when using this menu option, the inserted symbol does not keep any link with the Master Symbol — it is just a simple copy of the object (or group of objects) used to create the Master Symbol. Another technique that preserves the linked symbol is to right-click in the screen workspace, and then select “Insert Linked Symbol...” from the context menu.



Both the “Edit... Paste From” and “Insert Linked Symbol” menus begin navigation with the “Application Symbol” folder (see Figure 6-107 on page 6-105). To reach the “System Symbols” folder, navigate to the “{drive}:Program Files\Phoenix Contact\ThinkNDo\HMI\Symbol” folder.

Once the symbol is inserted, you can manipulate it like any other object in the screen. You can align and distribute it with other objects, and you can apply dynamic properties to it. However, the first thing to do is complete the custom properties for this instance of the Master Symbol.

Specifying Custom Properties

To specify custom properties for a symbol, follow these steps:

1. Double-click the symbol to open the “Object Properties” dialog box for the symbol.

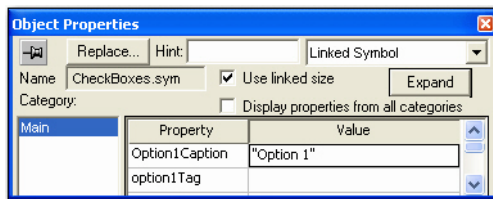


Figure 6-109 Object Properties: Linked Symbol

2. Click the “Expand” button to open the “Symbol Properties” dialog box. The dialog box that appears displays all available custom properties for the symbol.

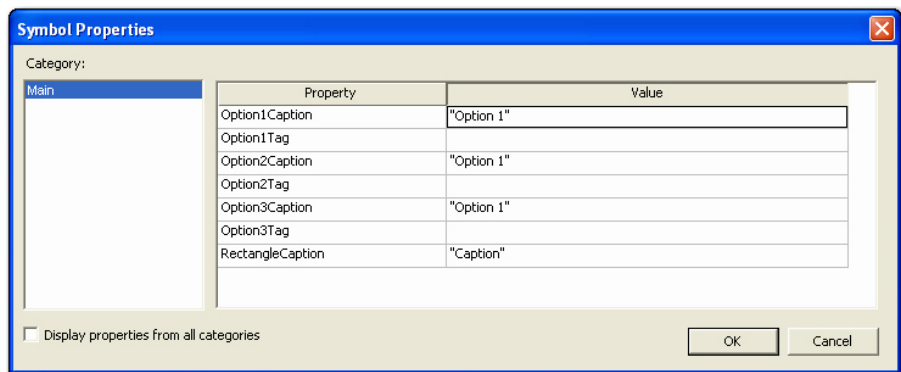


Figure 6-110 “Symbol Properties” dialog box for the linked symbol

In this example, the three check boxes are used to determine whether to alert Tom, Dick and/or Harry. The captions are updated accordingly, and the check box tags are configured with the first three indices of a Think & Do flag array called “TND.AlertOptions”.

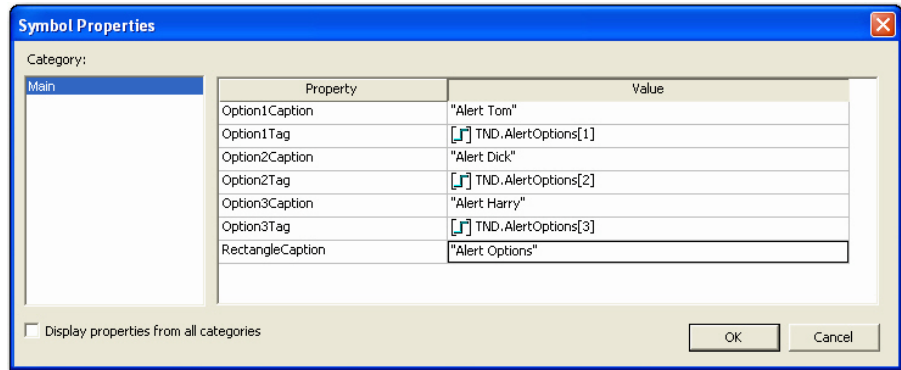


Figure 6-111 “Symbol Properties” dialog box with custom properties specified

3. Click the “OK” button to close the “Symbol Properties” dialog box, and then close the “Object Properties” dialog box.
4. ScreenView resolves custom properties during runtime.



Remember, the completed custom properties on each instance of a symbol are independent from every other instance of that symbol. If you make any changes to the Master Symbol file, then those changes automatically propagate to each instance when you re-save a screen with one or more copies of the Master Symbol.

Creating Symbols

To create a Master Symbol and save it to the “Symbol” folder, follow these steps:

1. Design your symbol just as you would normally draw an application screen, using any combination of static and active objects. For example, three check boxes in a rectangular pane would look like Figure 6-112.

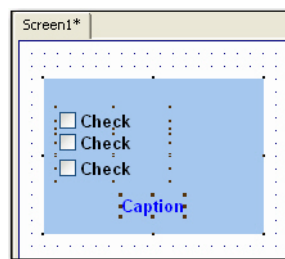


Figure 6-112 Creating a Master Symbol

2. Select the object(s) or group that you want to save as a symbol.



It is not necessary to make a group out of two or more objects before saving them as a symbol. Saving the objects together as a symbol effectively groups them as well.

3. Right-click on the selected object(s) and choose “Create Linked Symbol” from the context menu.

4. ScreenView displays a standard “Save As” dialog box. Use the dialog box to select a folder and give the new symbol a file name. Symbol files (“.sym”) are saved in the “HM\Symbol” directory of your project:
“{drive}:\Program Files\PhoenixContact\ThinkNDo\Projects\{project name}”.
5. Click the “Save” button to save the file. The symbol appears in the Symbol Library under “Application Symbols”.

The symbol is ready to be reused in your project. Every copy will have identical properties. You must now define custom properties for the Master Symbol — that is, the properties you want to be able to customize each time you reuse the Master Symbol.

Editing Master Symbols

Edit a Master Symbol to add or delete objects in the symbol or to define custom properties. Any modifications you make to the Master Symbol automatically propagate to every linked copy in every application project.

To edit a Master Symbol:

1. Right-click an instance of the Master Symbol that is placed on a screen, and then select the “Edit Linked Symbol” context menu. The symbol file opens for editing in its own tab in the ScreenView workspace (see Figure 6-113). Use the Symbol Editor as you would for any screen, except that every object in the window is part of the Master Symbol. If you add, move or delete objects in the Symbol Editor, then you may change the size or shape of the symbol and disrupt the layout of any screens where it is used.

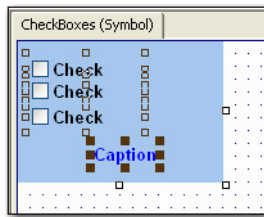


Figure 6-113 Symbol in Symbol Editor

In addition to adding, moving or deleting objects in the symbol, you can also edit the Object Properties as you normally would. You may want some properties to be the same in every instance of the Symbol, but other properties need to be customized according to where and how the symbol is used. In this example, you probably want to customize the captions of the three check boxes and the tags that the check boxes enable/disable, as well as the caption of the pane itself.

2. Select the first object in the Master Symbol and open its “Object Properties” dialog box. For example, Figure 6-114 shows the first check box “Object Properties” dialog box.

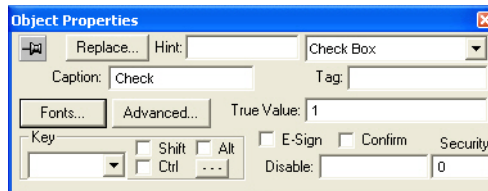


Figure 6-114 Object Properties: Check Box

- In any field where you would normally configure a tag, expression, or value, you can instead define a custom property using the following syntax:

`#<Label>:<Default>`

where <Label> is a name to identify the property, when you are completing the properties on an instance of the symbol, and <Default> is an optional default value for the property.



All standard syntax applies to <Default>, including tagnames, indirect tags, arrays, strings, numerical and boolean values, and scripting expressions. Also, even if you do not want to assign a default value to the custom property, you must type the colon character (:) after the <Label>.

- In this example, we want to be able to customize which tag the check box will enable/disable when it is clicked; in the Tag field, type “#Option1Tag” as shown in Figure 6-115.

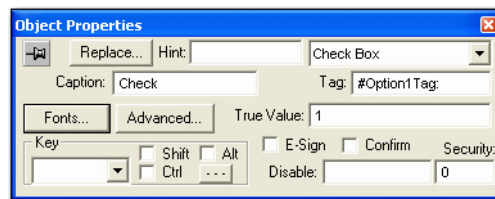


Figure 6-115 Object Properties: Check Box with a custom property specified

When completing the properties of an instance of the symbol, the “Option1Tag” property appears like Figure 6-116 for an instance of the symbol..

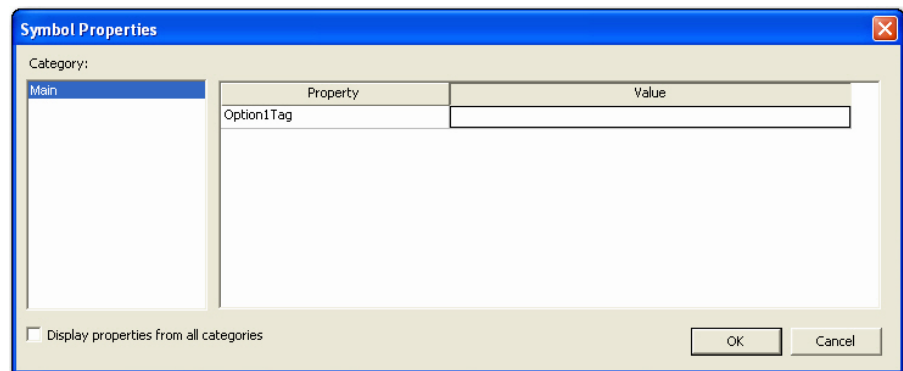


Figure 6-116 Symbol Properties showing a custom property to define

- Depending on the context, some object properties require a specific type of value like a string, boolean or numerical value. For these properties, you must enclose the custom property definition in curly brackets “{ }”. In this example, the “Caption” field requires a string, so type ‘{#Option1Caption:"Option 1"}’ as shown in Figure 6-117.

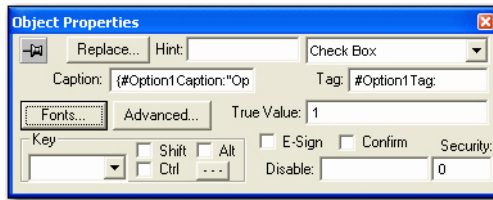


Figure 6-117 Object Properties: Check Box with a custom property specified

When completing the properties of an instance of the symbol, the “Option1Caption” property appears like Figure 6-118.

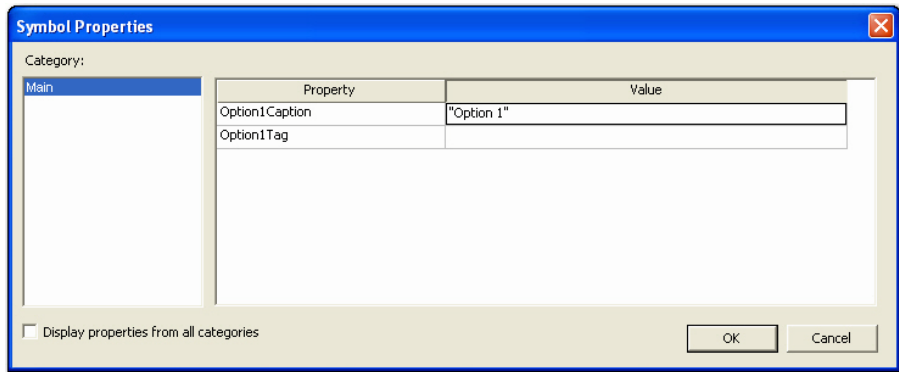


Figure 6-118 A default custom property in the “Symbol Properties” dialog box

- Repeat steps 2 through 4 as needed, to define the rest of the custom properties for the symbol. In this example, the finished symbol has all of the properties as shown in Figure 6-119.

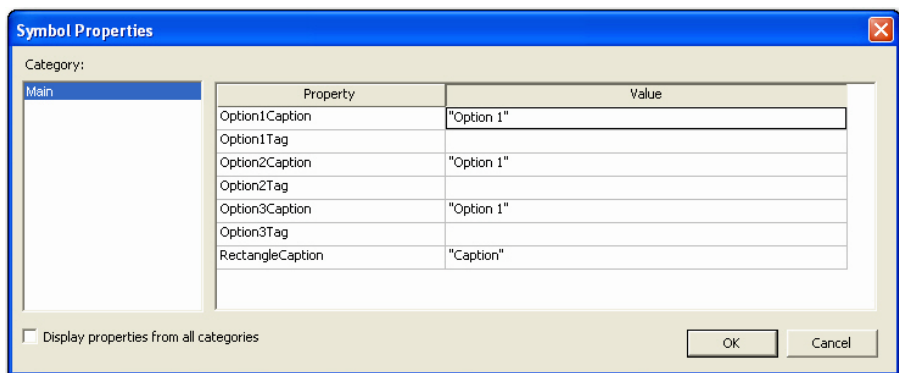


Figure 6-119 Object Properties: Check Box with all custom properties specified

- Save the symbol by selecting the “File... Save Checkboxes (Symbol)” menu, or right-click on the Symbol Editor tab, and then select “Save Checkboxes (Symbol)” from the context menu.



“Checkboxes” is the name of the symbol in this example.

- Close the Symbol Editor by selecting the “File... Close Checkboxes (Symbol)” menu, or right-click on the Symbol Editor tab, and then select “Close Checkboxes (Symbol)” from the context menu.
- If an open screen has an instance of the symbol, you must save the screen to update the instance of the symbol.

Adding Tooltips

You can configure a description for each custom property available in the symbol. After creating a symbol, open it with the Symbol Editor, right-click in the Symbol Editor (not on the symbol itself), and then choose “Edit Symbol Properties” from the context menu. This displays the “Symbol Properties” dialog box with a “Description” column (see Figure 6-120).

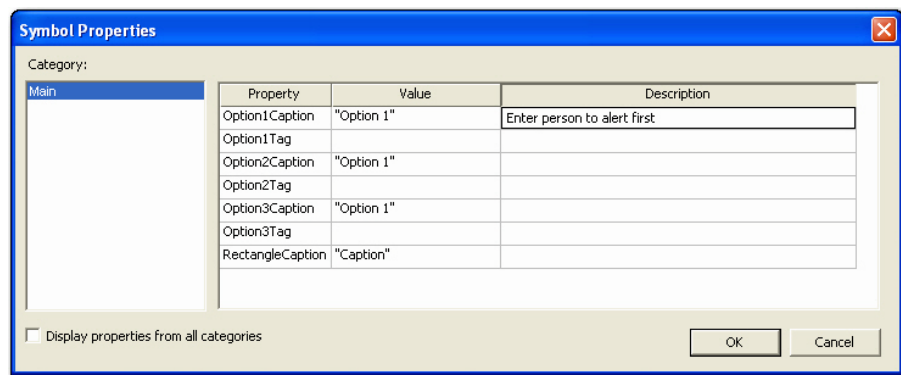


Figure 6-120 The “Symbol Properties” dialog box for editing tooltips

When assigning values to the custom properties of the symbol on the screens, the user can read the description as tooltips just by moving the mouse cursor on the property name, as illustrated in Figure 6-121.

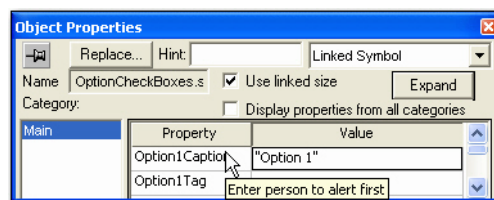


Figure 6-121 The symbol “Object Properties” dialog box showing a tooltip

6.4 Setting Screen Attributes

Use the “Screen Attributes” dialog box to set Runtime characteristics of the screen. To set screen attributes, do one of the following:

- Creating a new screen opens the “Screen Attributes” dialog box (see “Creating Operator Screens” on page 4-19).
- For a screen that you’ve already created, follow these steps:
 1. In the ScreenView Explorer, expand the “Screens” folder.
 2. Double-click the desired screen to display it in the Workspace.
 3. Right-click on a blank area of the screen, and select “Screen Attributes” from the pop-up menu.

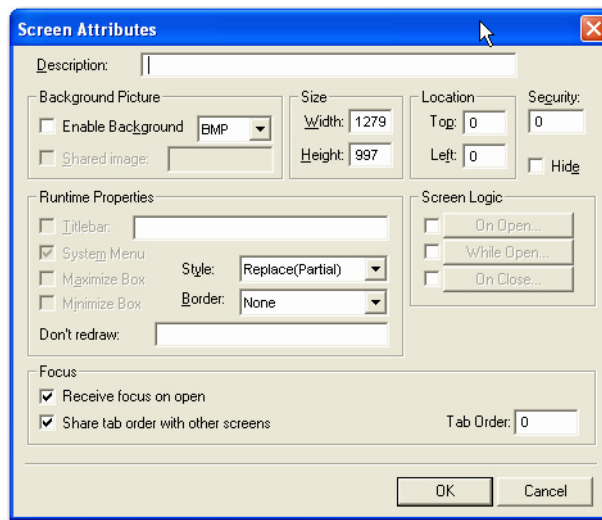


Figure 6-122 The “Screen Attributes” dialog box

Use the parameters on the “Screen Attributes” dialog box as follows:

- “Description” specifies a description of the screen attribute for documentation purposes. The text you enter in this field displays in the status bar (by default) located at the bottom, left of the screen when you are in Run Application mode.
- “Background Picture” area specifies the following parameters for the background.
 - “Enable background” check box, when selected enables the use of background bit-maps. The default is unselected.
 - “Enable Background” combo-box lets you select one of the following Windows 2K/XP/Vista background options:

• BMP	• JPG
• DXF	• PCD
• EPS	• PNG
• FAX	• TIF
• FMF	• TGA
• FPX	• WMF
• IMG	• WPG



Windows CE supports .BMPs only.

- “Size” area allows specification of an integer number in the “Width” and/or “Height” boxes to specify the size (in pixels) of the selected window.
- “Location” area allows specification of an integer number in the “Top” and/or “Left” fields to specify the location of the window (in pixels) in relation to the current screen.
- “Security” specifies the security level for the screen.
- “Hide”, when selected, keeps the screen loaded in memory after calling it for the first time, which facilitates faster loading when you open the screen. Think & Do executes Screen Logics normally.
Enabling this feature (default is disabled) causes a high use of GDI resources, consequently we recommend that during development, you monitor these resources using the “InfoResources” function.
- “Runtime Properties” area specifies the following parameters to define the window properties when you are running in Run Application mode:
 - “Titlebar” specifies a name to display in the viewing screen’s title bar during Run Application mode. Click (check) to activate or (uncheck) to deactivate the title bar.
 - “System Menu”, when selected, enables the system menu.
 - “Minimize Box”, when selected, activates the Minimize button.
 - “Maximize Box”, when selected, activates the Maximize button.
 - “Style” selects a window style (default is “ Replace(Partial)”).
 - “Overlapped” opens a window without closing any other window.
 - “Popup” opens a window that remains in front of other windows, but leaves other windows enabled.
 - “dialog box” opens a window that remains in front of other windows, but disables the other windows until you close the opened window.
 - “Replace(Complete)” opens a window and closes any other “Replace” and “Popup” style windows.
 - “Border” selects a border style. Choose:
 - “None” (default) specifies no border and does not allow a title bar or resizing
 - “Thin” specifies a thin border window that cannot be resized during runtime
 - “Resizing” specifies a normal border that can be resized during runtime
 - “Don’t Redraw” specifies a tag or value to control how screen dynamics are refreshed. Specifying a value higher than zero disables all screen dynamics.
- “Screen Logic” area, when selected, executes mathematical functions in one or more of these events: “On Open”, “While Open”, “On Close”.
After enabling an event, click on the corresponding button to open a dialog box where you can enter the following information:
 - “Tag” specifies a tagname to receive a return value from the Expression column.
 - “Expression” specifies a mathematical expression or function to be performed. The return value is applied to the Tagname field.
 - “Trigger (While Open dialog box only)”. Type a tag to work as a trigger (any value change) to execute this worksheet. If you leave this field blank, Think & Do executes the worksheet in the minimum time slice your system can perform.

6.5 Screen Groups

The "Screen Groups" folder combines individual display screens from the "Screens" folder into groups that appear concurrently on the display screen at runtime. This lets you create a display that uses common components such as a master header, navigation bar, and status bar.



To get screens to appear properly in a group, they must not overlap. So, for example, to have a 1024x100 header on the screen, set its size to exactly 1024x100 and its starting location to 0,0. Change the primary screen to 1024x668 instead of 1024x768 and set its starting location to 100, 0 instead of 0, 0.

To create a new screen group:

1. Right-click on the "Screen Groups" folder.
2. Select the "New Screen Group..." menu to display the "Insert Screen Group" dialog box (see Figure 6-123).

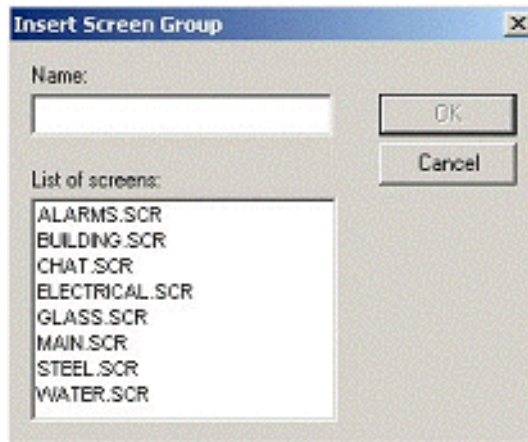


Figure 6-123 "Insert Screen Group" dialog box

3. Type a name for the new folder into the "Name" field.
4. Create a group of screens for this folder by selecting screens from the "List of screens" list. To select multiple screens press the <Ctrl> key as you click on the screen names. Release the <Ctrl> key when you finish.
This list contains only those screens currently located in "Screens" folder.
5. Click the "OK" button to close the "Insert Screen Group" dialog box and create the group.

To remove a specific screen group, right-click on its subfolder, and then select "Delete" from the context menu.

6.6 Additional Worksheets

The "Worksheets" folder contains an entry for each worksheet created in the project. Available worksheets are:

- Alarm
- Math
- Recipe
- Report
- Scheduler
- Trend

To create a new worksheet, do one of the following:

- Right-click on the "Worksheets" folder and select the new worksheet desired from the pop-up menu.
- Select the "File... New... Worksheet..." menu, and then select the worksheet desired from the cascade menu.

To open a worksheet, double-click the entry in the tree.

To delete a worksheet, right-click its entry in the tree, and then select "Delete" from the pop-up menu.

6.6.1 Alarm Worksheets

Alarm worksheets let you configure alarm groups and tags related to each group. The Alarm task defines the alarm messages generated by Think & Do. The primary purpose of an alarm is to inform the operator of any problems or abnormal conditions during the process so he can take corrective action(s).

Use one of the following methods to create a new Alarm worksheet:

- Right-click on the "Worksheets" folder, and then select "New Alarm" from the pop-up menu.
- Select the "File... New... Worksheet... Alarm" menu.

A new Alarm worksheet displays as shown in Figure 6-124.

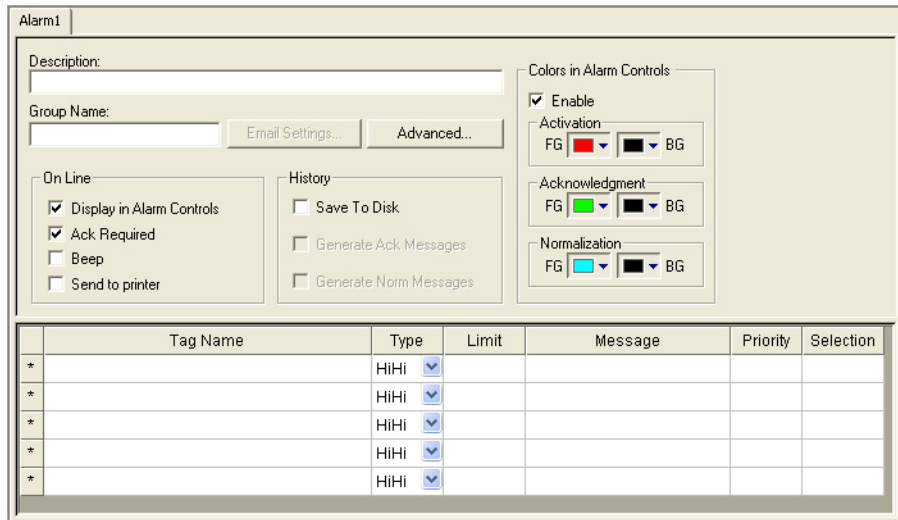


Figure 6-124 Alarm worksheet

The Alarm task is executed by the Background task module (BGTASK). The Alarm task handles the status of all alarms and save the alarm messages to the history, if configured to do so. However, the Alarm task does not display the alarm messages to the operator. The Alarm/Events control object, available on the Active Objects toolbar from the screen editor, must be created and configured in a screen in order to display the alarms to the operator.

Each Alarm worksheet is composed of two areas:

- Header specifies settings applied to all tags and alarms configured in the same alarm group. These settings allow you to configure the formatting of the message and the actions that must be triggered based on alarm events (e.g.: print alarms, send alarms by email, and so forth). For more information, see “Header Settings”, below.
- Body configures alarm messages and associate them to conditions linked to tags. For more information, see “Body Settings” on page 6-119.



The Alarm task avoids automatically acknowledging alarms by another alarm. For example, the Hi (Lo) alarm should not be automatically acknowledged when the HiHi (LoLo) alarm becomes active.



Notice that each tag/type cannot be available in more than one Alarm group simultaneously because the Alarm Group is a property associated to each Tag/Alarm Type (e.g.: Tag: Level; Alarm Type: Hi; Alarm Group: 2).

Header Settings

The header settings are at the top portion of the Alarm worksheet (see Figure 6-125).

Figure 6-125 Alarm heading

Table 6-21 describes the header settings on an Alarm worksheet.

Table 6-21 Alarm Worksheet Header Settings

Field	Remarks	Syntax
Description	Description of the alarm group. It is displayed on the workspace. This field is used for documentation only.	Text (up to 80 chars)
Group Name	Name of the Alarm group. During the runtime, the operator can filter alarms based on the Group Name by the built-in "Filters" dialog box of the Alarm/Event control object.	Text (up to 32 chars)
Advanced	Launches the "Advanced Settings" dialog box (see "Advanced Settings" on page 6-120), where you configure additional settings for alarm conditions.	Button
On Line > Display in Alarm Controls	When checked, the alarms are available to be displayed on the Alarm/Event Control object (see "Alarm/Event Control Object" on page 6-34).	Checkbox
On Line > Ack Required	When checked, the alarms require acknowledgment. In this case, the alarms are displayed on the Alarm/Event Control object (Online mode) until they are acknowledged AND normalized (see "Alarm/Event Control Object" on page 6-34).	Checkbox
On Line > Beep	When checked, the computer keeps beeping while there are alarm(s) to be acknowledged, currently active.	Checkbox

Table 6-21 Alarm Worksheet Header Settings

Field	Remarks	Syntax
On Line > Send to Printer	When checked, the alarm messages are sent to the printer as soon as the alarm event occurs. When using this option, you must use a matrix printer (instead of DeskJet or LaserJet) in order to print the message(s) and feed just one line – otherwise, each alarm will be printed in a different sheet of paper. The alarms will be printed in the default printer. If you want to send alarms to a printer different from the default printer, you can specify the printer path/name, editing the following parameter in the <ApplicationName>.APP file: [AlarmLog] Device=<PrinterPath/PrinterName>	Checkbox
History > Save to Disk	When checked, the alarm messages are stored in the history log when they become active.	Checkbox
History > Generate Ack Messages	When checked, the alarm messages are stored in the history log when they are acknowledged.	Checkbox
History > Generate Norm Messages	When checked, the alarm messages are stored in the history log when they become normalized.	Checkbox
Colors in Alarm Controls > Enable	When checked, the alarms configured in this group will be displayed with the colors assigned to each alarm state (Activation, Acknowledgement or Normalization), according to the colors configured in the Alarm Group.	Color
Colors in Alarm Controls > FG and BG	You can configure the text foreground color (FG) and background color (BG) for the alarms displayed on the Alarms/Events Control object (see “Alarm/Event Control Object” on page 6-34). Each alarm state can be displayed with a different color schema: <ul style="list-style-type: none"> – Activation specifies that the alarm is active and not acknowledged – Acknowledgement specifies that the alarm is active and acknowledged – Normalization specifies that the alarm is no longer active and not acknowledged. 	Color

Body Settings

The body settings are at the lower portion of the Alarm worksheet (see Figure 6-126).

	Tag Name	Type	Limit	Message	Priority	Selection
*		HiHi				
*		HiHi				
*		HiHi				
*		HiHi				
*		HiHi				

Figure 6-126 Alarm body

Table 6-22 describes the body settings on an Alarm worksheet.

Table 6-22 Alarm Worksheet Body Settings

Field	Remarks	Syntax
Tag Name	Name of the tag associated with the alarm. Double-click the field (or right-click and then select "Insert Tag") to display the "Object Finder" dialog box (see "Object Finder Dialog Box" on page 6-7). Typing a new name gives you the opportunity to create the internal tag (see "Creating ScreenView Tags" on page 6-9).	Tag
Type	Type of the alarm: <ul style="list-style-type: none"> – HiHi: Activates the alarm if the tag value is equal or higher than the limit. – Hi: Activates the alarm if the tag value is equal or higher than the limit. – Lo: Activates the alarm if the tag value is equal or lower than the limit. – LoLo: Activates the alarm if the tag value is equal or lower than the limit. – Rate: Activates the alarm if the tag value varies faster than the rate specified to the alarm. – DevP: Activates the alarm if the tag value is equal or higher than the Set Point tag plus the limit. – DevM: Activates the alarm if the tag value is equal or lower than the Set Point tag minus the limit. <p>When using the types Rate, DevP and DevM, it is necessary to configure additional settings by the "Tag Properties" dialog box, launched by the Tag Properties toolbar.</p>	Combo-box
Limit	Limit associated with each alarm. The limits can be modified dynamically during the runtime, by the tag fields HiHiLimit, HiLimit, LoLimit, LoLoLimit, Rate, DevP and DevM (e.g.: TagLevel->HiLimit).	Number

Table 6-22 Alarm Worksheet Body Settings

Field	Remarks	Syntax
Message	Message associated to the alarm. The message can be displayed on the Alarm/Event Control object and/or stored in the Alarm History and/or sent by Email, depending on the settings configured in the header of the Alarm group.	Text and/or {Tag} (up to 256 chars)
Priority	Priority number associated to the alarm. When displaying alarms on the Alarm/Event Control object, the operator can filter and/or sort the alarms by priority.	Number (from 0 up to 255)
Selection	Alias associated to the alarm (e.g.: AreaA, AreaB, etc). When displaying alarms on the Alarm/Event Control object, the operator can filter and/or sort the alarms by their selection value.	Text (up to 7 characters)

Advanced Settings

The “Advanced Settings” dialog box (see Figure 6-127) is where you configure additional settings for alarm conditions. Access this by clicking the “Advanced” button in the alarm header settings section of the Alarm worksheet.

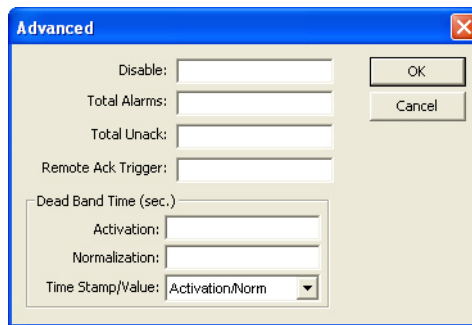


Figure 6-127 Alarm worksheet, “Advanced” settings dialog box

Table 6-23 describes the advanced settings on an Alarm worksheet.

Table 6-23 Alarm Worksheet Advanced Settings

Field	Remarks	Syntax
Disable	When the value of the tag configured in this is TRUE, all alarms configured in this group are temporarily disabled. This option is useful to disable alarms under special conditions (e.g.: during maintenance).	Tag
Total Alarms	The tag configured in this field, if any, is updated with the number of alarms from this group, which are currently active.	Tag
Total Unack	The tag configured in this field, if any, is updated with the number of alarms from this group, which are currently active AND have not been acknowledged yet.	Tag

Table 6-23 Alarm Worksheet Advanced Settings

Field	Remarks	Syntax
Remote Ack Trigger	When the tag configured in this field change of value, all active alarms from this group are acknowledged. This option can be used to acknowledge alarms regardless of any action from the operator.	Tag
Dead Band Time > Activation	Each alarm must remain continuously in its alarm condition for the period of time specified in this field before becoming active. This option is useful to avoid generating alarms on intermittent conditions (e.g.: noise). If this field is left in blank, the alarm becomes active as soon as its condition is true.	Tag or Number
Dead Band Time > Normalization	Each alarm must remain continuously out from its alarm condition for the period of time specified in this field before becoming normalized. This option is useful to avoid normalizing alarms on intermittent conditions (e.g.: noise). If this field is left in blank, the alarm become normalized as soon as its condition is no longer true.	Tag or Number
Dead Band Time > Time Stamp/Value	Each alarm maintains a time stamp of the last significant activity, along with the value of the tag at that time. You can select the type of activity that updates the time stamp: <ul style="list-style-type: none"> – Activation/Norm (default): The time when the dead band ended — that is, when the alarm becomes activated or normalized. – Last Tag Change: The time when the value of the tag last changed during the dead band. – Start Condition: The time when the dead band started. 	Combo

6.6.2 Math Worksheets

Math worksheets let you implement additional routines to work with the basic functions of different ScreenView tasks. A Math worksheet contains a group of programming lines that Think & Do executes as a Background task during runtime. You can configure the Math worksheet to provide free environments for logical routines and mathematical calculations needed by the project. For these purposes, the ScreenView scripting language is very simple and easy to use.



ScreenView sequentially increments the number that identifies the Math worksheet for each newly created worksheet.

Use one of the following methods to create a new Math worksheet:

- Right-click on the "Worksheets" folder, and then select "New Math" from the pop-up menu.
- Select the "File... New... Worksheet... Math" menu.

A new Math worksheet displays as shown in Figure 6-128.

	Tag Name	Expression
*		
*		
*		
*		
*		

Figure 6-128 Math worksheet

The Math worksheet is divided into two areas:

- Header area (top section), which contains information for the whole group
- Body area (bottom section), where you define each tag, expression, and Programming Lines (logical routines and mathematical calculations through functions and logical operations) in the group.

Use the header parameters on this worksheet as follows:

- "Description" specifies a description of the worksheet for documentation purposes.
- "Execution" specifies an expression, a single tag, or a constant value to determine when the worksheet should execute.



Think & Do executes the worksheet only when the "Execution" field result is not zero. If you always want the worksheet to execute, type a 1 (constant value) in the "Execution" field.

Use the body parameters on this worksheet as follows:

- "Tagname" specifies a tag to receive a return value from the specified calculation in the "Expression" column. Double-click the field (or right-click and then select "Insert Tag") to display the "Object Finder" dialog box (see "Object Finder Dialog Box" on page 6-7). Typing a new name gives you the opportunity to create the internal tag (see "Creating ScreenView Tags" on page 6-9).
- "Expression" specifies an expression to return a return value to the specified tag in the "Tagname" column.

6.6.3 Recipe Worksheet

Recipe worksheets let you configure data interchange between the application database and disk files in ASCII or DBF format; transferring values between files and realtime memory.

You typically use a Recipe worksheet to store process recipes, but you can store any type of information (such as operation logs, passwords, and so forth) in these files. The recipes task reads and writes tag values into files, and it transfers tag values from the application to a file or from a file to the application.



ScreenView sequentially increments the number that identifies the Recipe worksheet for each newly created worksheet.

Use one of the following methods to create a new Recipe worksheet:

- Right-click on the "Worksheets" folder, and then select "New Recipe" from the pop-up menu.
- Select the "File... New... Worksheet... Recipe" menu.

A new Recipe worksheet displays as shown in Figure 6-129.

Figure 6-129 Recipe worksheet

The Recipe worksheet is divided into two areas:

- Header area (top section), which contains information for the whole group
- Body area (bottom section), where you define each tag in the group.

Use the header parameters on this worksheet as follows:

- "Description" specifies a description of the worksheet for documentation purposes.
- "Save As XML" select (check) to save information in XML format, or (uncheck) to save in the DAT format.



You can load information in a ".DAT" file into different tags using a second Recipe worksheet, but you must load information in an ".XML" file into tags with the same name as the tag from which the data originated.



As with HTML pages, you must be running the Web server to view XML data from the Web. Unlike the HTML pages in the runtime system, XML pages do not need to have the application running to view the XML data. (You must be running Internet Explorer version 5.0 or higher to view XML data.)

- "File Name" specifies a file name related to the recipe group. Use static text ("File1") or a dynamic tag value ({FileNameTag}).
- "Register Number" specifies a tag to define the register number to be read or written into a DBF file.

- "Unicode" select (enable) to save the recipe in UNICODE format (two bytes per character) or (disable) to save the recipe in ANSI format (one byte per character).



When saving a worksheet, you can save it using any name you choose (you are not required to use a predefined file name). A configuration file using the default extension ".RCP" (or ".XSL" if you specify "Save As XML") contains the recipe configuration and the "File Name" field contains the data file name to be read or written.

Use the body parameters on this worksheet as follows:

- "Tag Name" field specifies tagnames to update with file contents or with values to write to a file. If the tag is an array, you must specify the first position to use. Double-click the field (or right-click and then select "Insert Tag") to display the "Object Finder" dialog box (see "Object Finder Dialog Box" on page 6-7). Typing a new name gives you the opportunity to create the internal tag (see "Creating ScreenView Tags" on page 6-9).
If the tag is an Array or a Class (or both), then Think & Do automatically enables every array position and class member by default. To configure a specific array position and/or class member, type it in the Tag Name field as normal.
For example, "TND.level[3].member".
- "Number of Elements" specifies how many positions of the array tag are in use.



When you define an array tag, its initial position is zero.



You can configure a tagname between curly brackets {TagName} in this field, allowing the user to change the Number of Elements configured in the recipe for each array tag dynamically, during runtime.

To read or write a recipe group, use the ScreenView scripting language function.

6.6.4 Report Worksheet

The Report worksheet contains a definition of reports (text type) to be sent to a printer or disk. The Reports task allows you to configure your own report (text type) with data from the system. The main purpose of this task is to make report creation easier and more efficient.



ScreenView sequentially increments the number that identifies the Report worksheet for each newly created worksheet.

To print a report, use a ScreenView scripting language function anywhere an expression is allowed.

Use one of the following methods to insert a new Report worksheet:

- Right-click on the "Worksheets" folder, and then select "New Report" from the pop-up menu.
- Select the "File... New... Worksheet... Report" menu.

A new Report worksheet displays as shown in Figure 6-130.

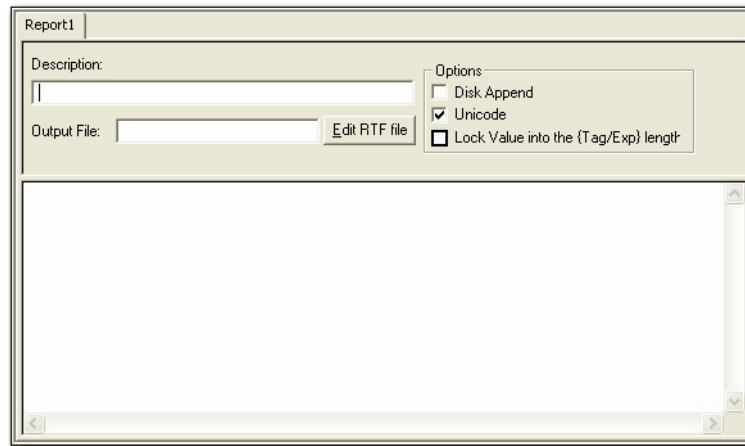


Figure 6-130 Report worksheet

The Report worksheet is divided into two areas:

- Header area (top section), which contains information for the whole group
- Body area (bottom section), where you define each tag in the group.

Use the header parameters on this worksheet as follows:

- "Description" specifies a description of the worksheet for documentation purposes.
- "Disk Append" used when printing a file:
 - Check the box to add (append) the new report to the end of an existing file.
 - Uncheck the box to replace the existing report in that file with the new report.
- "Unicode" select (enable) to save the report in UNICODE format (two bytes per character) or (disable) to save the report in ANSI format (one byte per character).
- "Lock Value into the {Tag/Exp} length" select (enable) to automatically truncate the values of Tags/Expressions in the report to fit between the curly brackets, as they are positioned in the body of the report (see below). This helps preserve the layout of the report. If this option is left unchecked, then the full values of Tags/Expressions in the report will be displayed.
- "Output File" specifies a string or tagname that contains the filename for printing to a file. The string may include a tagname that contains part of the filename (using the "{tag}" syntax).
For example: "report{day}.out"
Where the generated file might be "report1.out", "report2.out"... and so on, according to the value of the "day" tag.



A report configuration file uses ".RCP" as the default extension. The "Output File" field is the file where data is stored.

Double-click the field (or right-click and then select "Insert Tag") to display the "Object Finder" dialog box (see "Object Finder Dialog Box" on page 6-7). Typing a new name gives you the opportunity to create the internal tag (see "Creating ScreenView Tags" on page 6-9).

- "Edit RTF file" button accesses the report as an RTF file, which you can edit.

Use the body portion of this worksheet for report formatting. You can configure a report using data in the system and indicating where to print the tag values. Each tagname will replace the "{tag}" tagname. For float tags, use the following syntax: "{tag n}", where "n" is the number of decimal characters you want printed.



If you are using the standard report editor (text only: ASCII or UNICODE), then the number of characters reserved for the tag value will be equal to the number of characters used to type the tagname (including the two "curly" brackets). For example, configuring "{TND.TagA}" in the report body reserves ten characters for the tag value in the report file. This behavior is not valid for reports in RTF format.

6.6.5 Scheduler Worksheet

The Scheduler worksheet generates events with defined mathematical expressions to be executed according to the time, date, or any monitored event.



ScreenView sequentially increments the number that identifies the Scheduler worksheet for each newly created worksheet. Different scheduler groups have only organizational purposes.

Use one of the following methods to insert a new Scheduler worksheet:

- Right-click on the "Worksheets" folder, and then select "New Scheduler" from the pop-up menu.
- Select the "File... New... Worksheet... Scheduler" menu.

A new Scheduler worksheet displays as shown in Figure 6-131.

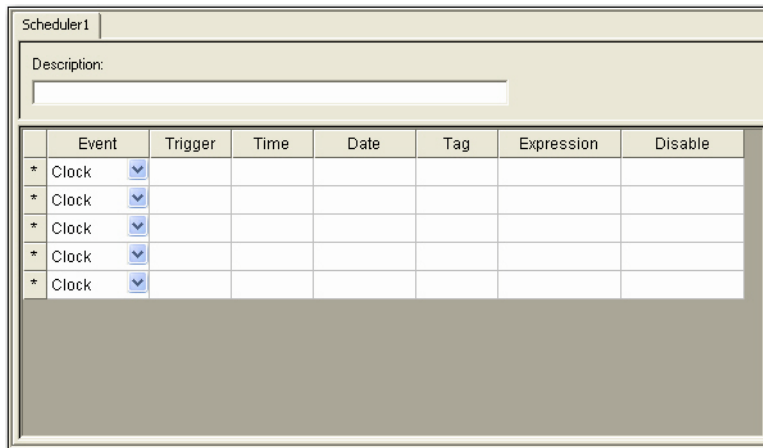


Figure 6-131 Scheduler worksheet

The Scheduler worksheet is divided into two areas:

- Header area (top section), which contains information for the whole group.
- Body area (bottom section), where you define each tag, expression, and condition for the group.

Use the header parameter on this worksheet as follows:

- "Description" specifies a description of the worksheet for documentation purposes.

Use the body parameters on this worksheet as follows:

- "Event" selects an event type from the following:
 - "Calendar" generates time bases greater than 24 hours. For example, you can define an event that prints a report every Friday at a specific time.



Be sure to complete the "Date" field if you want a specific date for event execution.

- "Clock" generates time bases smaller than 24 hours (intervals in minutes or seconds). This function is frequently used with trend graphics. For example, you can define a tag that will be incremented each hour.
- "Change" relates to a change in value of a tag in the Trigger field.
- "Trigger" specifies a tag that starts a change event related to a change in tag values. When the Trigger tag changes, Think & Do returns the value specified in the Value field to the tag. This field is used only by the change event.
- "Time" specifies a time interval in which an event must occur when used by the clock – hours (0 to 23), minutes (0 to 59), and seconds (0 to 59). You also can use this field to specify a specific time to be used by calendar events.
- "Date" specifies a specific date on which a calendar event must occur – day (1 to 31), month (1 to 12), and year (1900 to 2099). If you leave the field blank, the event occurs daily. This field is used only by the calendar event.
- "Tag" specifies a tag to receive a new value or expression return in the event.
- "Expression" specifies an expression with return value that is set to the tag. This field is used by all events.
- "Disable" specifies a disable condition for the specified function. Leave this field blank or use an expression value equal to zero to execute the function. Use an expression value equal to one and the function will not execute (Disable = 1).



In the fields that permit a tagname ("Trigger", "Tag", "Expression", and "Disable"), double-click the field (or right-click and then select "Insert Tag") to display the "Object Finder" dialog box (see "Object Finder Dialog Box" on page 6-7). Typing a new name gives you the opportunity to create the internal tag (see "Creating ScreenView Tags" on page 6-9).

6.6.6 Trend Worksheets

Trend worksheets let you configure history groups that store trend curves. Use the Trend task to declare which tags must have their values stored on disk, and to create history files for trend graphs. Think & Do stores the samples in a binary history file (*.hst), and shows both history and on-line samples in a screen trend graph.

To show a trend graph on the screen, you must click the Trend tool on the Active Objects toolbar to create a trend object.



ScreenView sequentially increments the number identifying the Trend worksheet for each newly created worksheet using four bytes to save date and time information, and eight bytes per variable in each sampling.

Use one of the following methods to create a new Trend worksheet:

- Right-click on the "Worksheets" folder, and then select "New Trend" from the pop-up menu.
- Select the "File... New... Worksheet... Trend" menu.

A new Trend worksheet displays as shown in Figure 6-132.

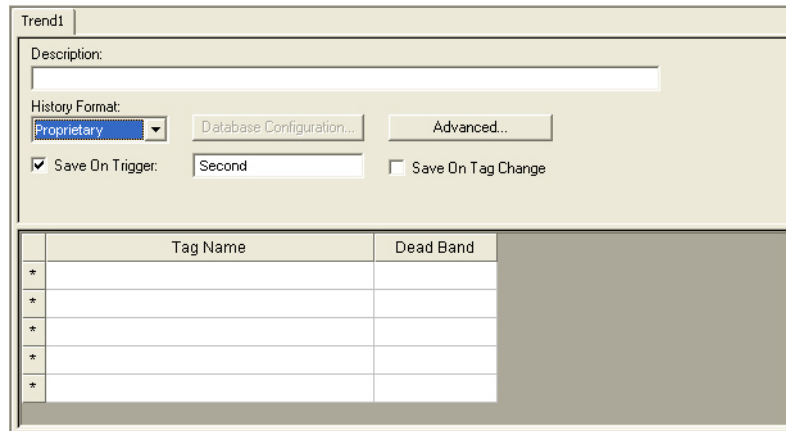


Figure 6-132 Trend worksheet

The Trend worksheet is divided into two areas:

- Header area (top section), which contains information for the whole group
- Body area (bottom section), where you define each tag in the group. This section contains several columns (only two are shown in the preceding figure).

Use the header parameters on this worksheet as follows:

- “Description” specifies a description of the worksheet for documentation purposes.
- “History Format” specifies a trend type from the list. The following options are available:
 - “Proprietary” selects the Think & Do proprietary format for saving history data.
 - File Format: Binary
 - Default path: “<Application Path>\Hst\GGYYDDMM.HST”, where:
 - YY = Two last digits of the year
 - MM = Month
 - DD = Day



Think & Do provides the “HST2TXT.EXE” and “TXT2HST.EXE” programs, which enable you to convert trend history files from binary (“.hst”) to text (“.txt”) and vice versa. For more information about these programs, see the “Converting Trend History Files from Binary to Text” and “Converting Trend History Files from Text to Binary” topics in the online help.

- “Database” enables the “Database Configuration” button, which displays the “Database Configuration” dialog box.
 - Database type specified in the “Database Configuration” dialog box.
 - Default table name: TRENDGGG, where GGG is the Trend worksheet number, for example TREND001 for Trend worksheet 001.
- “Database Configuration” opens the “Database Configuration” dialog box, where you can enter the requisite settings to link Think & Do to an external SQL relational database for the purpose of saving the trend history.
- “Save On Trigger” check box and field, when selected and a tagname is specified, saves trend samples when the specified tag changes. (Tag change can be an event from the Scheduler.)
- “Save On Tag Change” check box, when selected, always saves the trend sample when a value change occurs in any of the tags from that group.

- “Advanced” displays the “Trend Advanced Settings” dialog box. For information about completing the fields in this window, see the “Creating Batch History” topic in the online help.

Use the body parameters on this worksheet as follows:

- “Tagname” specifies the tagname to be saved in the history file. Double-click the field (or right-click and then select “Insert Tag”) to display the “Object Finder” dialog box (see “Object Finder Dialog Box” on page 6-7). Typing a new name gives you the opportunity to create the internal tag (see “Creating ScreenView Tags” on page 6-9).



After adding or removing tags from a Trend worksheet, any history files (“*.hst”) you previously created will no longer be compatible with the new setting. Consequently, the data from those history files will no longer be displayed by the Trend object.

- “Dead Band” specifies a value to filter acceptable changes when “Save on Tag Change” is used. For example, “Dead Band” has value = 5. If the tag value is 50 and changes to 52, the system will not register this variation in the database, because it is less than 5. If the change is equal to or greater than 5, the new value will be saved to the history file.
- “Field” specifies the name of the field in the database where the tag will be stored. If this field is left blank, the name of the tag will be used as the field name. Array tags and classes will have the characters “[”, “]” and “.” replaced by “_”. For examples, see Table 6-24.

Table 6-24 Default Database Field Name Examples

Tagname	Default Field
TND.MyArray[1]	TND_MyArray_1
TND.MyClass.Member1	TND_MyClass_Member1
TND.MyClass[3].Member2	TND_MyClass_3_Member2



The Trend task can accept only up to 240 tags in a single worksheet. If you manually configure more than 240 tags in the same worksheet, then the Trend task will generate an error when you start the finished application.

Setting the Trend Database

Follow this procedure to set the trend database:

1. In the ScreenView Explorer, double-click the applicable trend to display the Trend worksheet (see Figure 6-133).

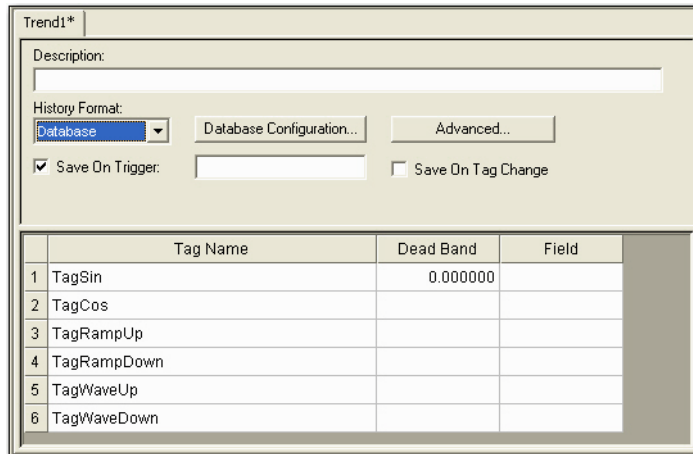


Figure 6-133 Trend worksheet

2. Use the “History Format” drop-down list to select “Database”.
3. Click the “Database Configuration” button. This opens the “Database Configuration” dialog box.
4. Enter applicable data in this window, and then click the “OK” button when you are finished.
5. Select the “File... Save” menu to save the changes to the Trend.

When using the trend table with a relational database, the fields saved in the History Trend are described in Table 6-25.

Table 6-25 Trend History Fields

Field Name	Data Type	Remarks
Time_Stamp	TimeStamp	TimeStamp (Date and Time) when the data was saved.
<Tagname>	Integer or Real (depending on the tag type)	ScreenView creates one field (column) in the database for each tag configured in the Trend worksheet.

6.7 Using Scripts

Code configured in script groups execute as a Background task. ScreenView supports the following types of scripts:

- Global Procedures are available for call by any other script.
- Graphics scripts are special pre-defined subroutines. There are subroutines that execute:
 - Once when a graphics module starts
 - Continuously while running
 - When the graphics module closes
- Startup scripts execute just once when the Background task module (BGTask) starts.
- Background scripts execute when the condition specified in their "Execution" field is TRUE (non-zero). Think & Do scans background script groups sequentially (based on the number of the group).
- Screen scripts are special pre-defined subroutines. There are subroutines that execute:
 - Once when a screen opens
 - Continuously while the screen is open
 - When the screen closes

6.7.1 Scripts Folder

There are two sub-folders under the "Scripts" folder.

- The "Application" folder only includes pre-defined script types.
- The "Background" folder includes any number of user-defined scripts that execute when enabled.

To edit a script in the "Application" folder, double-click its icon in the "Scripts... Application" folder. This opens the script in a tab in the ScreenView workspace.

To create a new background script group, do one of the following:

- Right-click the "Scripts... Background" folder, and then select "New Background Script" from the pop-up menu.
- Select the "File... New... Background Script" menu.

To edit an existing background script, double-click its icon in the "Scripts... Background" folder in the ScreenView Explorer. This opens the script in a tab in the ScreenView workspace.

To save or close a script, right-click its tab, and select the appropriate menu.

6.7.2 Global Procedures

Use global procedures to declare and implement procedures that can be called by any other VBScript in the project. You can declare local variables within each procedure declared in this interface (local scope within each procedure); however, you cannot declare global variables in this interface.

Think & Do

To create or edit global procedures, double-click the "Global Procedures" icon in the "Scripts... Application" folder. This opens the script in a tab in the ScreenView workspace (see Figure 6-134).

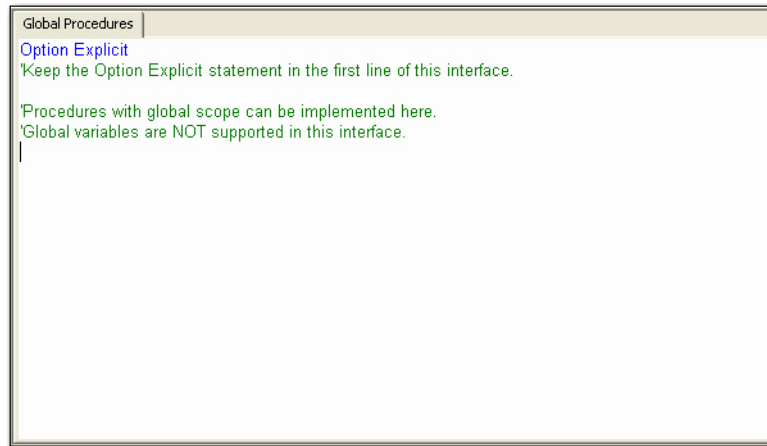


Figure 6-134 Global Procedure script page



An asterisk (*) following the tab name indicates that the page has changed and requires saving.

The procedures implemented in this interface are only executed by Think & Do when they are called from another VBScript in one of the other script groups (Graphics, Startup, or one of the Background scripts).

For an example, see the "Scripts... Global Procedure" topic in the Think & Do help.

6.7.3 Graphics Scripts

To create or edit graphic scripts, double-click the "Graphics" icon in the "Scripts... Application" folder. This opens the script in a tab in the ScreenView workspace (see Figure 6-135).

```

Graphics*
Variables with local scope can be declared and initialized here.

Procedures with local scope can be implemented here.

This procedure is executed just once when the graphic module is started.
Sub Graphics_OnStart()
End Sub

This procedure is executed continuously while the graphic module is running.
Sub Graphics_WhileRunning()
End Sub

This procedure is executed just once when the graphic module is closed.
Sub Graphics_OnEnd()
End Sub

```

Figure 6-135 Graphics script page



An asterisk (*) following the tab name indicates that the page has changed and requires saving.

Use graphics scripts based on pre-configured subroutines:

- Graphics_OnStart(): The code configured within this subroutine automatically executes just once when the graphic module starts. This interface is useful for initializing variables or executing logic that must be implemented when starting the application.
- Graphics_WhileRunning(): The code configured within this subroutine executes continuously while the graphic module is running. The rate at which this subroutine is called depends on the performance of the platform where the application is running.
- Graphics_OnEnd(): The code configured within this subroutine automatically executes just once when the graphic module closes.



Do NOT change the name of the pre-configured subroutines. If you do, the system will be unable to call them.

On the system that is running Think & Do:

- The graphic module is the Viewer task.
- The Graphics_OnStart() subroutine executes once on the system when the Viewer task launches.
- The Graphics_WhileRunning() subroutine executes while the Viewer task is running.
- The Graphics_OnEnd() subroutine executes once on the system when the Viewer task shuts down.

The variables and procedures declared in this interface are NOT available for any other VB-Script interface – they have local scope only.



The procedures and/or variables created in this interface have local scope—they can be accessed only from the graphic script of the local station where they are being executed.

For an example, see the “Scripts... Graphics” topic in the Think & Do help.

6.7.4 Startup Scripts

The code on the "Startup" page executes once when the Background task module (BG-Task) starts. This is useful for initializing variables or executing logic that must be implemented when the application starts. You can use startup scripts to declare and initialize variables and implement procedures. However, variables or procedures declared on this page are not available to graphics scripts.

To create or edit startup scripts, double-click the “Startup” icon in the "Scripts... Application" folder. This opens the script in a tab in the ScreenView workspace (see Figure 6-136).

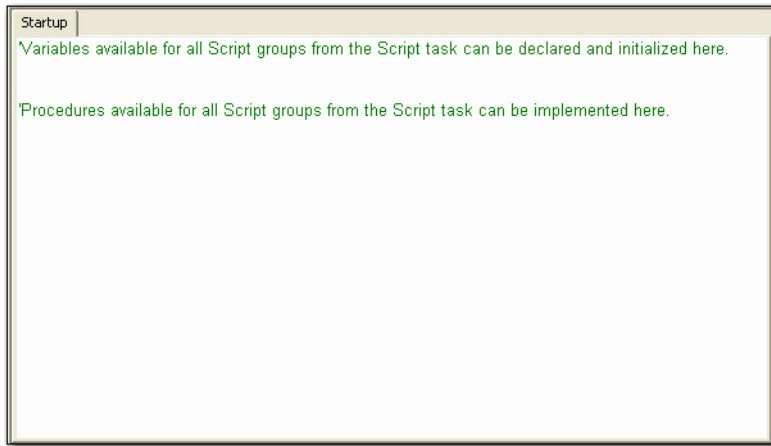


Figure 6-136 Startup script page



An asterisk (*) following the tab name indicates that the page has changed and requires saving.

For an example, see the “Scripts... Startup” topic in the Think & Do help.

6.7.5 Background Scripts

Background scripts execute when the condition specified in their "Execution" field is TRUE (non-zero). Think & Do scans background script groups sequentially (based on the number of the group).



You must use the syntax supported by the ScreenView scripting language in the "Execution" field. Only the body of the script group supports VBScript language.

To create a new background script, do one of the following:

- Right-click the "Scripts... Background" folder, and then select "New Background Script" from the pop-up menu.
- Select the "File... New... Background Script" menu.

To edit an existing background script, double-click its icon in the "Scripts... Background" folder in the ScreenView Explorer. This opens the script in a tab in the ScreenView workspace (see Figure 6-137).

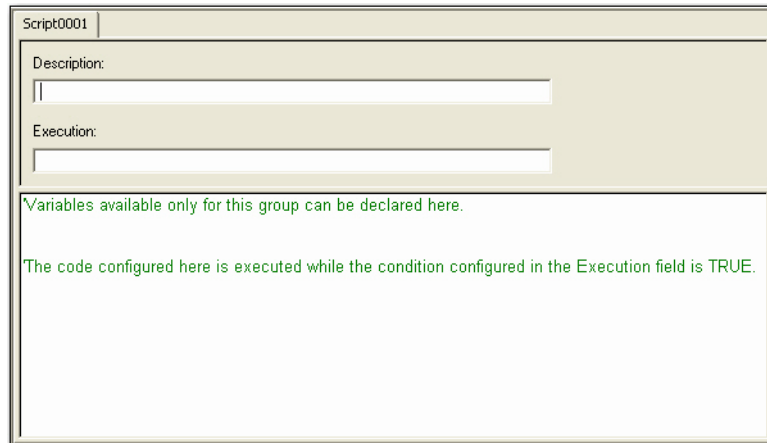


Figure 6-137 Background script page



An asterisk (*) following the tab name indicates that the page has changed and requires saving.

To save or close a script, right-click its tab, and select the appropriate menu.



When any background script is saved during runtime, the startup script interface executes again, and the current value of the local variables of any script group is reset.

Variables declared in a background script have local scope for that specific script only. They are not available for any other script. From any background pages, you can call procedures implemented in the global procedures or in the startup pages.

For an example, see the "Scripts... Background" topic in the Think & Do help.

6.7.6 Screen Scripts

Screen scripts are special pre-defined subroutines that are directly related to specific screens. To edit the screen script for a screen, right-click on the screen, and then select the "Screen Script" menu to display the screen script in a tab in the ScreenView workspace.

This interface can be used to execute logic on the following events, based on pre-configured subroutines:

- Screen_OnOpen(): The code configured within this sub-routine is automatically executed just once when its screen is open.
- Screen_WhileOpen(): The code configured within this sub-routine is automatically executed continuously while its screen is open. The rate in which this sub-routine is called depends on the performance of the platform where the application is running.

Think & Do

- Screen_OnClose():The code configured within this sub-routine is automatically executed just once when the screen is closed.

The variables and procedures declared in this interface are available for the VBScript interfaces of the screen where the screen script is configured.



Do NOT change the name of the pre-configured sub-routines. If you do, the system will be unable to call them automatically.



The procedures and/or variables created in this interface have local scope—they can be accessed only from the specific screen where they are implemented.

For an example, see the “VBScript... Screen Scripts” topic in the Think & Do help.

Section 7

This section informs you about:

- How to use variables and constants in Think & Do
- Details on math functions and string handling
- How to use SQL and PID blocks

Math and Data Operations	7-3
7.1 Exploring Data Items	7-3
7.1.1 Initial and Retentive Values.....	7-5
7.2 Tagname Data Types and Formats	7-6
7.2.1 Data Type Descriptions	7-7
7.3 Understanding Arrays.....	7-8
7.3.1 Defining Arrays	7-9
7.3.2 Initializing Arrays.....	7-10
7.4 Think & Do System Data Items	7-11
7.4.1 Date and Time	7-11
7.4.2 Fixed Timers (or Clocks).....	7-12
7.4.3 Scan Clocks.....	7-12
7.4.4 Timer Data Items	7-13
7.5 Counting with Number Data Items	7-15
7.6 Using the Move Block	7-17
7.6.1 The Move Expression	7-17
7.6.2 Moving Between Same Data Types.....	7-18
7.6.3 Moving Multiple Data Item Values.....	7-18
7.6.4 Move Summary	7-23
7.6.5 Shifting and Rotating Arrays	7-23
7.6.6 Moving Array Elements	7-25
7.7 Using the Calculation Block	7-26
7.7.1 Building a Calculation Expression.....	7-26
7.7.2 Expression Display Conventions	7-27
7.7.3 Runtime Math Errors.....	7-27
7.7.4 Basic Math Operators.....	7-28
7.7.5 Bitwise Operators	7-29
7.7.6 Math Functions	7-29
7.8 Handling Analog Values	7-30
7.8.1 Mapping analog data	7-31
7.8.2 Scaling Analog Data	7-32
7.9 String Handling.....	7-35
7.9.1 Using Move Blocks with Strings.....	7-35
7.9.2 Exchanging Strings With Other Types	7-36
7.9.3 Using the Calculation Block with Strings.....	7-36
7.9.4 Comparing and Sorting Strings	7-38
7.10 SQL Database Operations	7-39
7.10.1 SQL Block Overview.....	7-39

Think & Do

7.10.2	General Tab.....	7-40
7.10.3	Tag Mapping Tab.....	7-44
7.10.4	Filters (Where) Tab.....	7-45
7.10.5	Sorting (Order By) Tab.....	7-47
7.10.6	SQL Statement Tab.....	7-48
7.11	PID Block for Process Control.....	7-49

7 Math and Data Operations

This chapter discusses how to use variables and constants in Think & Do. It begins with an overview of how to view data items, and includes details on math functions and string handling. It also includes information on how to use SQL and PID blocks.

7.1 Exploring Data Items

Data items are a central part of every project – they are the variables and constants that contain project data. Data items have tagnames, which are names you assign as part of a project. ProjectCenter contains the tools for viewing all parts of the project, including the Data Item Explorer.

To explore or create data items:

1. Use the “Project Explorer” bars in the left pane of ProjectCenter.
2. Click the “Data Items” Explorer bar. The “Data Items” grid appears in the main document window.

The “Data Items” grid displays a list of one data type at a time.

To selecting a data type to explore:

1. In the “Project Explorer” pane, click the “Data Items” Explorer bar.
2. Under “Data Types”, click the folder that corresponds to the data type desired. If necessary, use the scroll bar to view the entire list of data types.

With the exception of “System” data items, a new project has blank grids for each data type until you define tags for that data type.

Figure 7-1 shows the main features of the “Data Items” grid. Each data type appears in the Data Item grid as individual spreadsheets, with their own unique lists of data items. Each project uses its own subset of data types according to its needs.

Index	Tagname	Initial	Retentive	Description	Mapped to...
C 1	WhiteAccumulationCounter	0	<input type="checkbox"/>		
C 2	GreenAccumulationCounter	0	<input type="checkbox"/>		
C 3	RedAccumulationCounter	0	<input type="checkbox"/>		
C 4	YellowAccumulationCounter	0	<input type="checkbox"/>		
C 6	WhiteDiverterPosition	7	<input type="checkbox"/>		
C 7	GreenDiverterPosition	12	<input type="checkbox"/>		
C 8	RedDiverterPosition	18	<input type="checkbox"/>		
C 10	WhitePackageID	1	<input type="checkbox"/>		
C 11	GreenPackageID	2	<input type="checkbox"/>		
C 12	RedPackageID	3	<input type="checkbox"/>		
C 23	YellowPackageID	4	<input type="checkbox"/>		
C					

Figure 7-1 The Data Item grid displays all tags of a particular data type

Selecting a Data Item

To select a data item, follow these steps:

1. Move the cursor over the letter in the first column, such as “N.” The cursor changes to an arrow.
2. Click to select the row of the Data Item grid, highlighting the row.

Adding Data Items

To add data items with specific tagnames, follow these steps:

1. Click the empty cell (in the “Tagname” column) and type the tagname.
2. Press <Enter> or click the cursor in another cell, and Think & Do creates the new item.

To add data items with default tagnames:

1. Click a cell in the row for the new data item.
2. Right-click to access the pop-up menu.
3. Click “Add” to add a new data item at the bottom of the list, or “Insert” to insert a new row above the selected row.

To add or insert multiple data items:

1. Click a cell in the row where the first of the new data items will go.
2. Right-click to access the pop-up menu.
3. Click “Add Many...” or “Insert Many...” to get the “Add (or Insert) Multiple Data Items” dialog box.
4. Enter the number of data items to add or insert, and click the “OK” button.

New data items appear with default names in a sequence, such as “Number5,” “Number6,” “Number7,” and so on.



When defining several data items for a new project or temporary ones for testing, using default tagnames can be a time saver. Type over the tagname later with a specific, meaningful name as the data items become a permanent part of the project. Flow charts, screens, and I/O mappings will automatically use the new tagnames.

Editing Data Item Grid Entries

The Data Item grid has full editing capability on data items with Cut, Copy, and Paste functions.

To delete data item(s), follow these steps:

1. Select a row by clicking its row header number. Select multiple rows by doing a click-and-drag from the first to last row of the group to be selected. The highlighted rows are ready for deleting.
2. Press the <Delete> key.

To edit tagnames with copy-paste functions, follow these steps:

1. Select the text in the tagname cell.
2. Then click the “Copy” toolbar icon or right-click to access the pop-up menu, and select “Copy” the button.
3. Select the destination tagname cell.
4. Click the “Paste” toolbar button, or right-click and select “Paste” from the pop-up menu.



Data item tagnames are automatically forced to be unique within a project. When pasting an existing tagname to a new cell, Think & Do automatically appends a numeric suffix to the name of the new cell. You can then edit the name or number.

Finding and Replacing Tagnames in the Data Item Grid

To find data items, follow these steps:

1. On the ProjectCenter menu bar, select the “Edit... Find” menu.
2. In the “Find” dialog box, enter the tagname as the search string.
3. Specify the search direction, and select “Match case” if required.
4. Click the “Find Next” button. Press the <F3> key to Find Again, using the same search string and settings.

To find and replace tagnames in the Data Item grid, follow these steps:

1. On the ProjectCenter menu bar, select the “Edit... Replace” menu.
2. In the “Replace” dialog box, enter the tagname as the search string in the “Find What” field.
3. Enter the new text in the “Replace with” field.
4. Select “Match case” if required.
5. Click “Find Next”, “Replace” or “Replace All” as needed.

7.1.1 Initial and Retentive Values

The “Initial” column in the Data Item grid applies to most data types. At startup, the Runtime project initializes all data items to the initial value in the tagname database (except for data items marked as retentive). The **retentive status has priority over any initial value**, except for the very first time a project is run.

Data item values may be configured as retentive or non-retentive. The term “retentive” describes saving the value to disk whenever the Runtime project closes, or prior to a power-down if a UPS is used (see Note below). Retentive data items have initial values equal to their previous value before shutdown.

To configure an initial value:

1. Click the cell in the “Initial” column for the desired data item to have a non-zero starting value at Runtime. To initialize individual array items, click the drop-down arrow in the “Initial” column. This specifies an initial value for each element of the array. For more information on initializing array elements, see “Initializing Arrays” on page 7-10.
2. Enter the initial value in the cell. Make sure the data matches the data type. (For example, a Number type cannot have a starting value of 12.57, while a Float type can).

To configure a retentive data item:

1. Select the check box in the “Retentive” column for the appropriate data item rows. Only data items that can be retentive have the “Retentive” column present in the Data Item grid.



When making a timer retentive, only the accumulator and preset are retentive.

2. Click the check box again to deselect the retentive setting (toggles between retentive/non-retentive).

7.1.1.1 Inputs

It's very important to think about I/O in the context of retentive memory. Since the state of inputs originates externally, the Data Item grid does not present a retentive option for inputs. At Runtime, the first read of inputs establishes their state. This occurs before the first logic solve of the flow charts.

7.1.1.2 Outputs

Outputs can be retentive, since flow charts in Think & Do turn them ON or OFF. Remember that, when a Control block turns an output (or any bit) ON or OFF, this is a permanent or latched event until another control block changes the bit. With the retentive setting, you can have the outputs power up in the last state of the most recent Runtime session.



A flow chart can periodically write to data item SYS-43 to save retentive data item values to disk. This can be used with or without a UPS.



In order for retentive data items to be truly retentive during an unexpected power loss, the PC system must have an Uninterruptible Power Supply (UPS) and power-monitoring software. When external power is lost, the power-monitoring software must specifically shut-down the Think & Do Runtime project. This saves retentive data items to disk while the PC is still on backup power.

To pulse output, specify a “Pulse Value” for an output tag. Then use the “Pulse Output” action in a Control block. When pulsed, the output turns ON for the time specified in the “Pulse Value” parameter. After the specified time, the output automatically turns off.

7.2 Tagname Data Types and Formats

Table 7-1 lists the various data types supported by Think & Do. Registers are BCD, Bytes are 8-bit values, Strings are NUL-terminated strings, Floats are double precision 64-bit floating-point numbers.

Table 7-1 Think & Do Data Types

Data Type	Prefix	Minimum Value	Maximum Value	Number of Bits	Read/Write	Maximum Data Items	Can Be Retentive?
Array	Arr				R/W	65535	Yes
Axis	Axis				R/W	65535	No
Byte	B	0	255	8	R/W	65535	Yes
Comm	COM				R/W	255	No
Counter	C	-32768	32767	16	R/W	65535	Yes
Flag	F	0	1	1	R/W	65535	Yes
Float	FP	2.225e-308	1.797e+308	64-bit IEEE	R/W	65535	Yes
Input	I	0	1	1	Read*2	65535	No
Number	N	-2147483648	2147483647	32	R/W	65535	Yes
Output	O	0	1	1	R/W	65535	Yes
String	Str		255 char. max		R/W	65535	Yes
System	SYS		2147483647	32	R/W	8 (fixed)	No
Timer	T	0	4294967295	32	R/W	4096	Yes



Data items that belong to an array have the properties of their data type. See the appropriate row on the chart for the array’s data items.



To enable counters, select the “Project... Settings” menu in ProjectCenter, and then check the “Enable Counter data type” check box.



You can write to inputs in simulation mode and with AppTracker.

7.2.1 Data Type Descriptions

The following list describes the data types in the table above. Knowing this information is important in choosing the best data type for each tagname in the project. A general guideline is to choose the data type that preserves the accuracy or format of the data it contains.

Array

The “Array” data item specifies an indexed range of data items of type Byte, Flag, Float, Number, String, or Timer. The elements in an array are unique to that array, and are **not** members of the standard data types by the same name. Array elements are only accessible to flow charts and screens via the array’s tagname and index number of the element, for example, “Array[Index]”, where “Index” is an integer constant or number.

Axis

Axis data items represent motors (axes) to which motion commands are applied. They must be mapped to physical I/O in IOView. If they are not mapped, the specified motion command does not operate at project Runtime.

Byte

Byte data items represent 8-bit binary values. These are convenient for I/O-related bit patterns or for ASCII character codes in communications. Entries can be specified in decimal, hexadecimal (0x prefix), binary (b suffix), or a character (ASCII).

Comm

Comm data items represent serial port devices that communicate with the Runtime. Through IOView, they map to the physical COM ports of the target.

Flag

Flags are single-bit variables that have a value of either “0” or “1”. These are for general use to store a Boolean (TRUE/FALSE) variable. Although these are single-bit variables, they are not associated with discrete I/O points.

Float

Float (floating point) data items use IEEE 64-bit format. They are useful when you need to represent decimal fractions. Their range provides for extremely small and extremely large numbers: $-2.225073858507201e-308$ to $1.7976931348623158e+308$. The number of significant digits is 17.

Input

An Input data item is a single-bit (“0” or “1”) for use with discrete input points for the I/O subsystem. A maximum of 65535 input points are available. Note that an input data item only needs a tagname in the Data Item grid to exist. The mapping of an Input data item to physical I/O is a separate action performed in IOView. You can map input data items in any order to actual physical inputs.

Number

The Number data type provides signed 32-bit integers, which range from -2147483648 to +2147483647.

Output

An Output data item is a single-bit (“0” or “1”) for use with discrete output points for the I/O sub-system. A maximum of 65535 output points are available. An Output data item only needs a tagname in the Data Item grid. The mapping of an Output data item to physical I/O is a separate action performed in IOView. You can map output data items in any order to actual physical outputs.

String

String data items can hold up to 255 UNICODE characters. Strings are NULL-terminated, which means the display will terminate when the string reaches the first NULL character. They are useful in communicating with serial or parallel port devices, such as alarming systems, bar code readers, instrumentation devices, printers, and other devices that communicate with the PC through serial or parallel ports.

System

System data items contain specialized information available to flow charts and screens at Runtime. For more information, see Appendix A 1, “System Data Items”.

Timer

Timer data items count upward once per millisecond. Their timer value is accessible to flow charts, and have a range of 0 to +4294967295 milliseconds.

7.3 Understanding Arrays

In mathematical terms, arrays are a contiguous series of like things, individually identifiable by their index numbers alone. Computers benefit from using arrays, simplifying sorting, finding minimum and maximum values, etc. In Think & Do, brackets [] denote array indices.

Arrays in Think & Do have either a number value or a fixed integer tag that serves as the index. The data types available to form an array are:

- Byte
- Flag
- Float
- Number
- String
- Timer

A single array must consist of data items of only one type. The array itself is a single data item of the “Array” data type. The Data Item grid array definitions in ProjectCenter determines how you reference array elements, such as “Level[0]”, “Level[1]”, etc. For example,

by defining the data type as “Float”, the array contains only Float-type data items. Finally, the array size determines the number of elements) in the array. Since the first index of an array is 0, the maximum index number is (size -1).



If the project needs to access or display name labels that correspond to an array of scalar values, consider creating a string array of the same length as the values array. Then use corresponding array index numbers to use the appropriate name labels in conjunction with the values (see Figure 7-2).

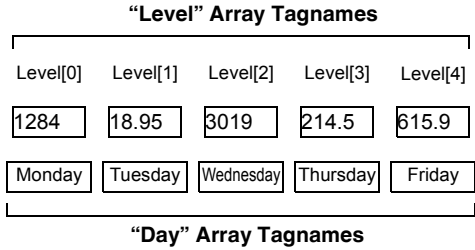


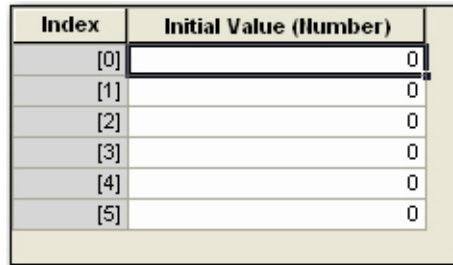
Figure 7-2 Use a label array with string values that correspond to data values

7.3.1 Defining Arrays

To define an array, follow these steps:

1. In ProjectCenter, click the “Data Item” Explorer bar and then click (highlight the “Array” folder) to display the project’s array definition grid.
2. In the “Tagname” field, type the array’s tagname.
3. Enter a description.
4. In the “Array Type” drop-down list, select the data type of the elements of the array, for example “Number”.
5. In the “Index” column, the first array has a default index = 0 (Arr-0). Optionally you can change the index to another value.
6. In the “Size” column, enter the number of elements. Think & Do automatically adds square brackets around the size when exiting the field. For string arrays, enter a size (number of elements) followed by the size of each element. The default element size of 255 appears if only a single size value is entered. To specify the size of the string elements, enter the size of the array in square brackets followed by the length of the string elements, also in square brackets (for example, [5] [20] creates an array of five string elements, each 20 characters long).

7. In the “Initial” column, enter an initial value for all elements, or use the down-arrow to enter a unique value for each element (Figure 7-3). For more information on initializing arrays, see “Initializing Arrays” below.



Index	Initial Value (Number)
[0]	0
[1]	0
[2]	0
[3]	0
[4]	0
[5]	0

Figure 7-3 The drop-down arrow displays a table with initial values

8. Use the “Retentive” check box to define a retentive value (“Initial and Retentive Values” on page 7-5).
Up to 65535 arrays can be defined.

7.3.2 Initializing Arrays

The Data Item grid in ProjectCenter allows setting initial values for any array type. You can set a single value that applies to all array elements or you can define individual values for each element. Other features permit copying initial values, filling a series of elements with the same value, and creating an automatic sequence of values.

To set a single value for all array elements – enter the value in the “Initial” field.

To set unique values by element – follow these steps:

1. Click the down-arrow next to the “Initial” field to display a table of indexes and initial values (see Figure 7-3).
2. Enter a value for each element.

To copy and paste element values – follow these steps:

1. Click the down-arrow next to the “Initial” field to display a table of indexes and initial values (see Figure 7-3).
2. Right-click the element you want to copy, and choose “Copy” from the pop-up menu.
3. Select the element you want to paste, and choose “Paste” from the pop-up menu.

To restore default values (0, 0.0, or null) – follow these steps:

1. Click the down-arrow next to the “Initial” field to display a table of indexes and initial values (see Figure 7-3).
2. Single or multiple select (using <Ctrl> or <Shift> plus left-mouse click, or drag select) elements that you want to restore to their default value.
3. Right-click and choose “Restore Defaults” from the pop-up menu.

To fill values – follow these steps:

1. Click the down-arrow next to the “Initial” field to display a table of indexes and initial values (see Figure 7-3).
2. Single or multiple select (using <Ctrl> or <Shift> plus left-mouse click, or drag select) elements that you want to fill.
3. Right-click and choose “Fill” from the pop-up menu.
4. Choose “Down,” “Up” or “Series...” from the cascade menu to do one of the following:
 - “Down”: copies the top-most element value to all selected elements.
 - “Up”: copies the bottom-most element value to all selected elements.
 - “Series...”: displays a dialog box to choose either a “Linear” or “Growth” series and enter a “Step Value”. A linear series adds the step value to each successive selected element. A growth series multiplies the step value times each successive selected element.

7.4 Think & Do System Data Items

The Think & Do Runtime system provides system information to flow charts and screens, via System-type data items. System data types use data item ID numbers SYS-0 to SYS-79. You can view a listing of these in the data table by selecting “System” in the Data Item Explorer. Using these data items in the project for example, can:

- Display the current system scan time
- Use a bit that is on every other scan
- Monitor I/O scan statistics
- Work with retentive data
- Check for Runtime errors

For a complete listing of System data items, Appendix A 1, “System Data Items”.

7.4.1 Date and Time

Although not a listed System data type, the real-time clock in the Runtime target is accessible to projects. The names “Year”, “Month”, and “Day” are reserved names for the real-time values that you can access in the following ways:

- Flow chart Move block expression
- Flow chart Calculation block expression

To use the Runtime target’s date in a Move block, follow these steps:

1. Create three consecutive Number type data items named CurrentYear, CurrentMonth, and CurrentDay (example tagnames).
2. Create three more consecutive Number type data items named CurrentHour, CurrentMinute, and CurrentSecond.
3. In FlowView, use the “File” menu to access a new or existing flow chart.
4. Place a Move block onto the drawing area (see “Adding Flow Chart Blocks to a Flow Chart” on page 4-9).
5. Double-click the Move block to access the “Move Block... Expression” tab.
6. In the “Move Block... Expression” tab, select the “Current Date” data type in the “From” field.

7. In the “To” group’s “Data type” field, select “Number.” In the “Starting data item” field, enter “CurrentYear”. In the “Ending data item” field, enter “CurrentDay”.
8. Repeat the above steps for a second Move Block that moves from “Current Time” to “CurrentHour” (starting) and “CurrentSecond” (ending).

7.4.2 Fixed Timers (or Clocks)

Some of the most useful System data items are the clocks with a fixed time base and 50% duty cycle. They are great for creating flashing indicators in HMI screens, etc.

Table 7-2 Clock Tagname and ID

ID	Tagname	OFF time	ON time
SYS-32	ClockMin	30 sec.	30 sec.
SYS-33	ClockSec	0.5 sec.	0.5 sec.
SYS-34	Clock100ms	50 ms	50 ms
SYS-35	Clock50ms	25 ms	25 ms

All System data items are (32-bit) type, but these timers have a 1-bit value. The LSB of the double word alternates at the corresponding frequency for the timer.

To access system timer bits do one of the following:

- Use a Compare block to compare their value with a fixed integer constant “0”. Use the Yes/No exit paths from the Compare block to take different actions with Control blocks.
- Use a Move block to move the system timer bit directly to an output. This is an easy way to create an oscillator.

7.4.3 Scan Clocks

The scan clock System data items are synchronized with the Runtime scan of the flow chart logic.

Table 7-3 Scan Clock Tagname and ID

ID	Tagname	Description
SYS-36	FirstScan	ON for the first scan, OFF from then on
SYS-37	AltScan	ON during every other scan

FirstScan

The “FirstScan” data item is TRUE (1) only during the first scan at Runtime. It is then FALSE (0) for the remainder of that Runtime session. This data item is particularly useful for power-up initialization flow charts which need only to run one time.

AltScan

The “AltScan” data item alternates between 0 and 1 each scan. It is useful to perform a task every other scan.

7.4.4 Timer Data Items

Think & Do provides up to 65535 timers for use in the project. Timers have their own Timer data type, with a timing range of 0 to 4,294,967,295. Each timer is actually a collection of related variables, organized in a **structure** that is accessible in various project tools. Timer variables include the Accumulator (timer value), Preset Value, Timer Done status bit, and Timer Enabled status bit.

Timer Structures

The timer tagname always includes a suffix that specifies a particular timer variable. The timer structure syntax is: <Timer tagname>.<Timer field>. For example, a tank mixing flow chart may use a mixing timer with the tagname "MixTimer". The timer's preset value is specified with the following syntax: "MixTimer.Preset".

The timer structures include:

- <TimerTagname>.Accumulator
- <TimerTagname>.Preset
- <TimerTagname>.Done
- <TimerTagname>.Enabled

The main characteristics of programmable timer variables are:

- Accumulator: The "Timer.Accumulator" value has a range of 0 to 4,294,967,295, with a 1 ms resolution (each count = 1 ms). The timing range encompasses approximately 49.7 days.
- Preset Value: "Timer.Preset" is a user-defined value that the timer compares with its accumulator. When the accumulator reaches the preset value, the timer stops and the "Timer Done" status bit becomes true. At Runtime, the initial preset value for each timer originates in the Data Item grid definition for each timer (in ms units). The default preset is zero. Flow charts can read/write the preset value, even while the timer is running.
- Done: The "Timer.Done" status is a single-bit value — 0=FALSE, 1=TRUE. "Timer.Done" becomes TRUE when the accumulator reaches the preset value, staying TRUE until the timer is reset.
- Enabled: The "Timer.Enabled" status is a single-bit value — 0=FALSE, 1=TRUE. "Timer.Enabled" is TRUE only when the timer is running. It becomes FALSE when the timer stops or reaches the preset value.

The following block diagram shows the four Timer data items. Various blocks in a flow chart have access to the timer's value, controls and outputs. The timer's preset value initially comes from the setup in the Data Item grid at Runtime.

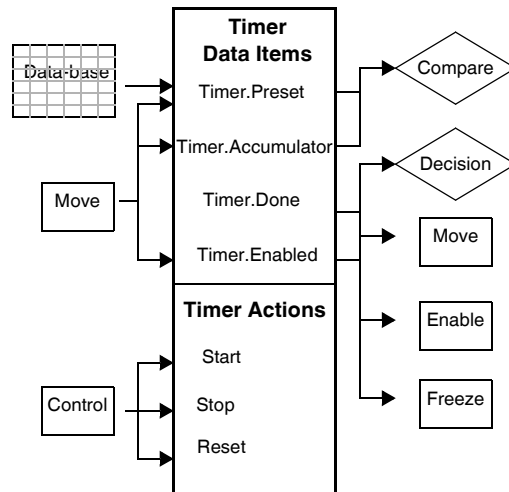


Figure 7-4 Timer data items are available to different blocks

The diagram shows that flow chart Control blocks can perform the actions “Start”, “Reset”, and “Stop” on timers. Do not confuse these actions with the four variables in the timer’s structure. Timer actions are described below.

Timer Actions

- Start: A Timer Start causes the timer’s accumulator to begin timing. It also turns ON the “Timer.Enabled” status bit.
- Reset: A Timer Reset sets the timer’s accumulator to 0 ms and turns the Timer Done bit OFF. It also turns OFF the “Timer.Enabled” status bit. After a timer expires, you must reset it before you can start it again.
- Stop: A Timer Stop causes the accumulator to hold its current value. The “Timer.Enabled” bit turns OFF, but the “Timer.Done” bit remains OFF.

Using Timers

Ways in which project components can interact with a timer include:

- A Control block can start, stop, or reset a timer.
- A Decision block can use the “Timer.Enabled” status bit or the “Timer.Done” status bit in a Boolean expression.
- A Compare block can compare the “Timer.Accumulator” or “Preset” with another value.
- A Move Block can move a value to (overwrite) the “Timer.Accumulator” value (timing continues if the timer was enabled). It can move the value from the Timer Accumulator to read the value.
- The “Timer.Done” status bit is read-only. Writes to it are ignored.
- A Move block can move a value to the “Timer.Enabled” bit. This overrides the current status of the timer’s “Start”, “Stop”, and “Reset” action controls.
- A screen object configured for data input can write to the same timer variables as a move block with the same response from the timer.
- A screen object configured for data output can read any of the timer variables.

During project development you can monitor or modify the variables in timer structures.

To monitor a timer in AppTracker at Runtime:

1. In AppTracker's Data Item explorer, find "Timer" in the "Data Items" category.
2. Click a "Watch" tab near the bottom of the "AppTracker" window.
3. Drag the timer data item from the upper right pane to the "Watch Window" area and release. The "Watch Window" displays the current timer value.
4. To watch the variables in the Timer's structure, click the "+" by the timer in the watch window to expand the display (see Figure 7-5).

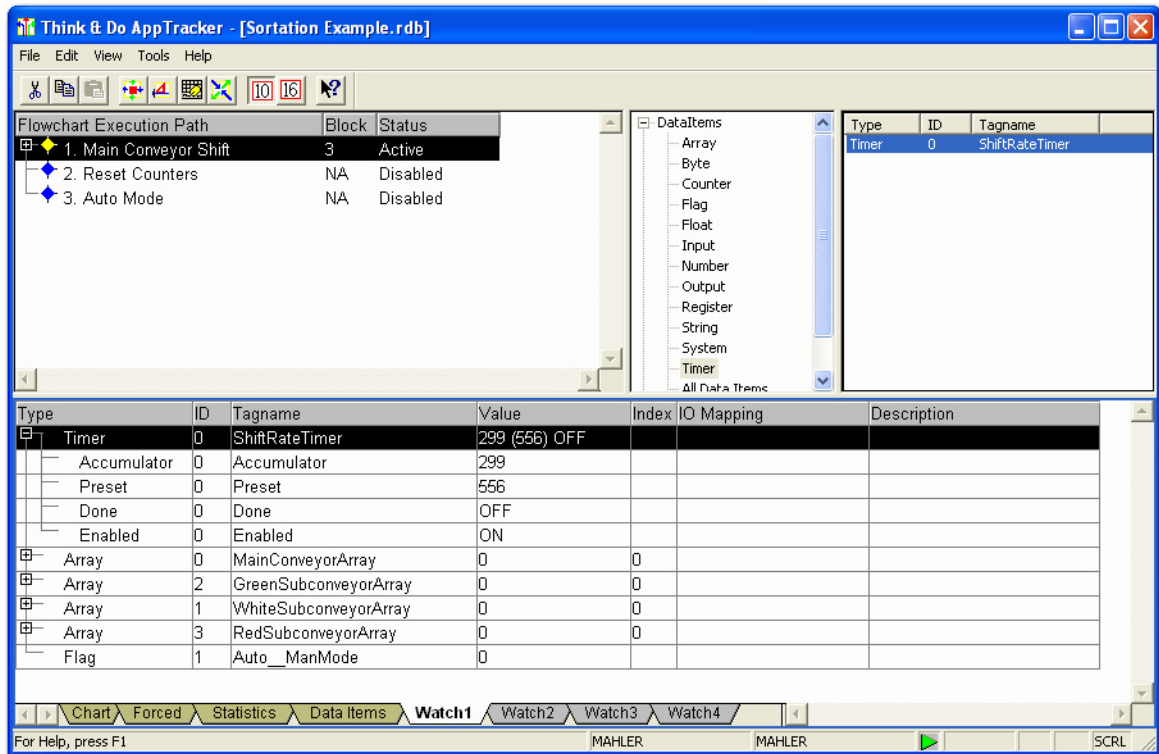


Figure 7-5 AppTracker watch window with timer variables visible

7.5 Counting with Number Data Items

Think & Do provides up to 65535 Number type data items for use in the project. Each number is a signed integer. Numbers are 32-bit values, having a range of $-2,147,483,648$ to $2,147,483,647$. Also, numbers have "Tagname", "Initial Value" and "Description" fields in the Data Items grid. The main characteristics of number data items are:

- Numbers can count upward (increment) and downward (decrement).
- An "Initial" value can be specified for each number in the Data Item grid. The initial value is the number's beginning value at Runtime.
- The number's "Clear" control, accessible from a flow chart Control block, sets the number's value to 0.

- The number's "Increment" control adds one to the current count.
- The number's "Decrement" control subtracts one from the current count.
- A number can count both in the positive and negative number ranges. Accordingly, you can specify a negative initial value.



A number can increment or decrement each scan, if a Control block performs that action. Counting is not transition-based, as in other types of controllers.

Ways in which project components can interact with a number include:

- A Control block can increment, decrement, or clear a number.
- A Compare block can compare the number's current value with another variable or constant.
- A Move block can replace the current value with a new value or read the current value.
- A Calculation block can use a number as an operand or result in an expression.
- IOView can use a number data item to map analog I/O values (for more details on analog I/O, see "Handling Analog Values" on page 7-30).

7.6 Using the Move Block

The purpose of the Move block in a flow chart is to write data from one variable to another. Mathematically, a move data operation is the same as an assignment statement:

MOVE from Data Item A to Data Item B, or
Let B = A

It is important to see the move as an assignment operation or copy because the move is non-destructive. That is, moving data from A to B does not clear A. A still has its original value, but B has a new value equivalent to A.



For readers who are more accustomed to using Load and Out instructions in accumulator-based controllers, think of the Move block as the Load-Out combination.

7.6.1 The Move Expression

The Move block is a specialized control block in flow chart design because it does a specific kind of action. Double-click the block (or right-click and select “Expression”) to display the “Move Block... Expression” tab (Figure 7-6).

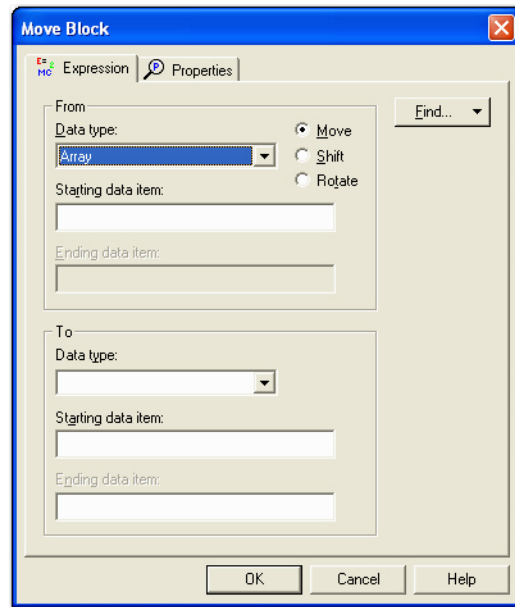
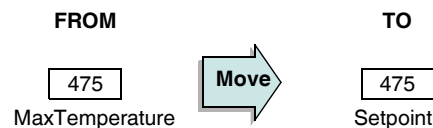


Figure 7-6 The “Move Block... Expression” tab

The “Move Block... Expression” tab has “From” and “To” groups. You must complete the “From” data fields before completing the “To” data fields. Think & Do automatically presents only the valid data types in the “To” group based on the data type selected in the “From” group. If the data types are not the same, the Move block performs the necessary data format conversion.

7.6.2 Moving Between Same Data Types

The simplest case is moving data from one data item to another of the same type. The following example uses two data items of the Number type, making the tagname “Setpoint” equal to “MaxTemperature”.



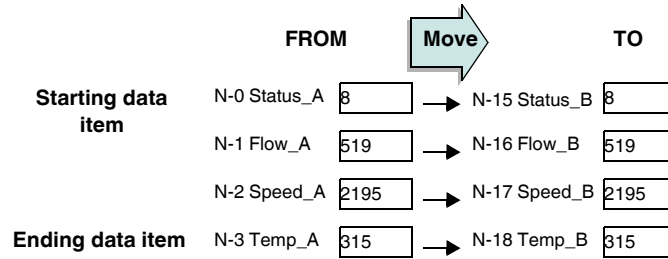
To perform this type of move, follow these steps:

1. Double-click the “Move” block to display the “Move Expression” dialog box.
2. Select the “Data type” in the “From” group.
3. Select the “Starting” field and click the “Find” button to select the tagname from the data item list. Select the tag and click the “OK” button.

4. Select the "Data type" in the "To" group.
5. Select the "Starting data item" in the "To" group, and click the "Find" button. Select the tag and click the "OK" button.

7.6.3 Moving Multiple Data Item Values

You can take advantage of the Move block's ability to move multiple data values in a single block. For example, the illustration below shows a move of a series of four numbers.

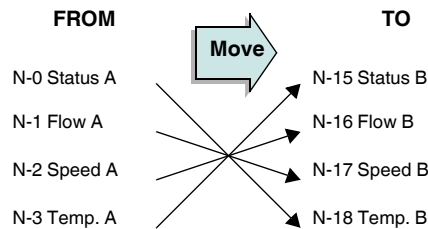


The "Starting data item" and "Ending data item" fields define the ranges in the "From" and "To" groups. All items whose indexes fall between the index of the items in the starting and ending fields belong to the group. Their position in the group is by increasing or decreasing index values (for example, type: N; index: 0, 1, 2, and 3). Indexes are set in the Data Item grid. Only the first and last items are named in the starting and ending tagname fields. The starting item of a move series is defined from either low or high index number. Make sure that both "From" and "To" groups use "Starting data item" and "Ending data item" in the same way.

When specifying the same data types for the "From" and "To" groups, you must use the same number of data items in each range. If not, an error dialog appears.

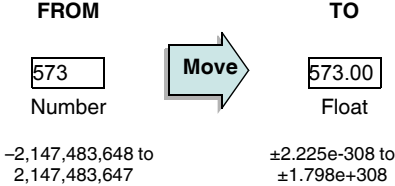
Moves With Inverted Order

Sometimes you may need to move a series of data item values, inverting their order as you do. The effect is pictured below. To do this, swap the tagnames in the "Starting data item" and "Ending data item" for either the "From" group or "To" group in the "Move Block... Expression" tab.

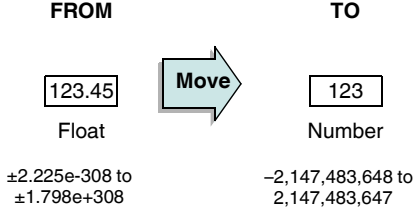


Moving Between Different Data Types

This section discusses the basic principles of how a Move block works when it needs to perform data conversion. In the example below, the move is from a Number to a Float data type. The number set for type Float contains all possible integers in type Number, so each number will convert.

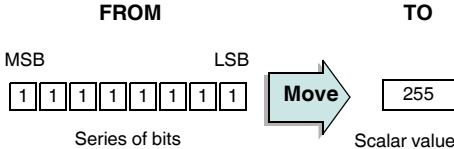


The next example presents the converse of the previous one; moving a Float data type to a Number type. In this case, the decimal fraction part of the Float number is truncated. Only the “123” moves to type Number. Notice the difference in magnitudes of the ranges. Data type Float can store much larger and much smaller numbers than the data type Number.



Moving Between Bits and Scalar Values

In the following example, a series of one-bit data items is moved to a scalar value data item.



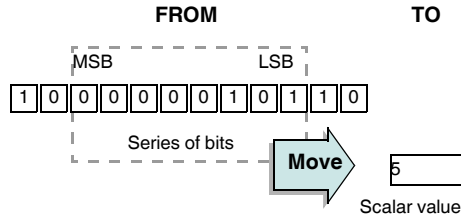
In this example, Think & Do moves eight bits (it doesn't have to be just eight) into a single number. The individual bits represent a range of eight adjacent data items of the same data type. These could be flags or I/O points, etc. The Move block action assigns the proper binary value to each bit when it moves the data. It is not necessary for the series of bits to have a particular binary weighting (that is, 2^x, where x is the position number of the bit counting from the right and starting with zero). The Move applies the weighting to the result when the data moves.



See “Move Summary” on page 7-23 for the number of data items that can be moved between various data types.

Moving Discrete Inputs to a Number

In the following example, the MSB and LSB notations define a range of bits (data items) within the entire range of data items of that type. This range does not have to begin/end on byte or word boundaries, so a lot of flexibility exists. The Move extracts the data items within the specified range (00000101), assigns the binary weighting, and produces the correct result = 5.



Suppose you have an 8-bit speed sensor input on discrete inputs I-0 to I-7. You will need to bring the bits in as an 8-bit number. The “Move Block... Expression” tab in Figure 7-7 does this. The Starting data item is the MSB. The Ending data item is the LSB. When moving bits to a number, start with the MSB.

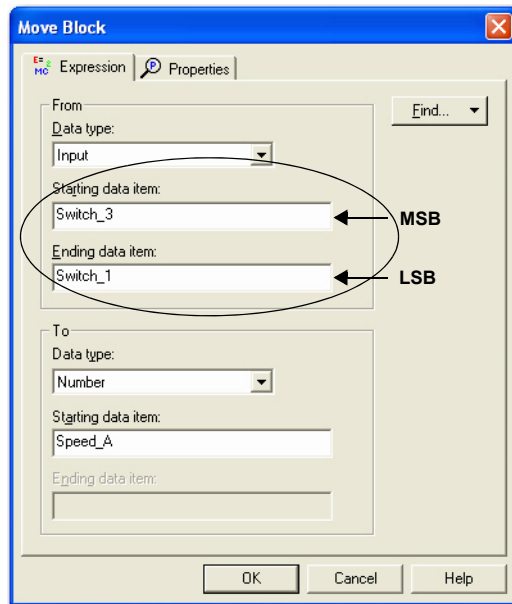
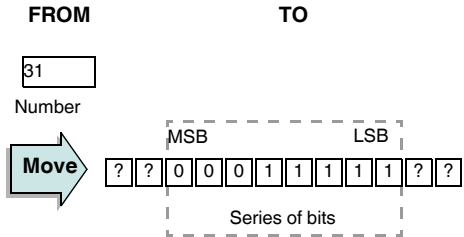


Figure 7-7 The MSB and LSB are specified in the range fields

Moving a Number to Discrete Outputs

The following example illustrates how to move a Number data type to a series of discrete output bits. The Move block automatically moves the LSB of the counter to the LSB of the specified output range. Since a 16-bit value is moving to eight 1-bit destinations, the Move block ignores the upper 16 bits of the number.



Suppose an LED display needs eight bits of data. The “Move Block... Expression” tab shown moves the counter value to the eight discrete outputs O-0 to O-7. When moving a scalar value to discrete bits, start with the MSB. The resulting “Move Block... Expression” tab is in Figure 7-8.

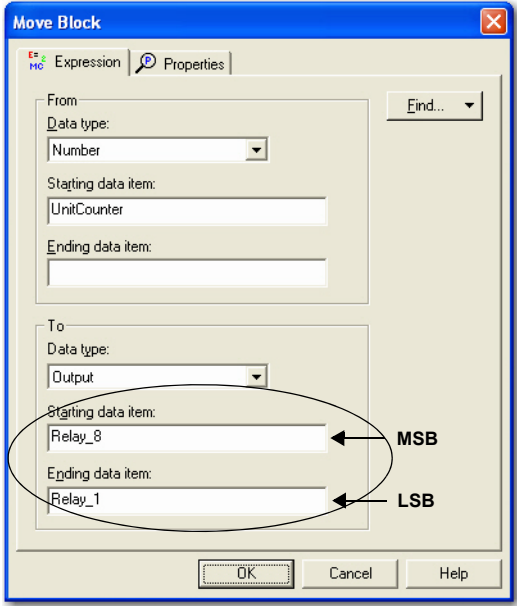


Figure 7-8 To move a number to a series of output bits, specify MSB/LSB



The “Starting data item” is the MSB. The “Ending data item” is the LSB.

Moving Analog Values

Analog values begin as numbers (signed or unsigned) by the proper mapping in IOView. An analog channel may have 8, 12, or 16 data bits, with or without a sign bit. Bipolar voltage inputs typically use sign bits. Once converted to a number, analog data can be handled as any other number. Floating point representation is especially useful when representing data in real-world engineering units for operator interface and data entry.



Converting and scaling raw analog data to floating point data items in engineering units is very useful. See “Handling Analog Values” on page 7-30 for details on how to map, scale, and convert analog values.

7.6.4 Move Summary

The following table defines all possible Move Block combinations. Moves that are valid for individual data items are also valid in arrays. You may also move data to Inputs when the flow chart is a Simulation type.

Table 7-4 Move Block summary

From... To...	1 Output	N Outputs	1 Flag	N Flags	1 Timer	N Timers	1 Register	N Registers	1 Counter	N Counters	1 Number	N Numbers	1 Byte	N Bytes	System	1 Float	N Floats	1 String	N Strings													
1 Float	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
N Floats	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=													
Float const.	✓	✓	✓	✓	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
1 Input	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x													
N Inputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x													
1 Output	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x													
N Outputs	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x													
1 Flag	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	x	x													
N Flags	x	=	x	=	32	x	16	x	16	x	32	x	8	x	32	64	x	x	x													
1 Timer	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
N Timers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=													
1 Register	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
N Registers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=													
1 Counter	✓	16	✓	16	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
N Counters	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=													
1 Number	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
N Numbers	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=													
1 Byte	✓	8	✓	8	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
N Bytes	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	✓	=													
Fixed Integers	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
Current Date	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓	x													
Current Time	x	x	x	x	x	x	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	x	✓	x													
System	✓	32	✓	32	✓	x	✓	x	✓	x	✓	x	✓	x	✓	✓	x	✓	x													
1 String	x	x	x	x	✓	x	✓	x	✓	x	✓	x	✓	✓	✓	✓	x	✓	x													
N Strings	x	x	x	x	x	=	x	=	x	=	x	=	x	=	x	x	=	x	=													
String Const.	x	x	x	x	x	x	x	x	x	x	x	x	✓	x	x	x	x	✓	x													
Legend	<table border="0"> <tr> <td>✓</td><td>Valid move</td> <td>=</td><td>Valid move, if the number of <i>From</i> and <i>To</i> data items are equal</td> </tr> <tr> <td>x</td><td>Invalid move</td> <td>2</td><td>4</td><td>8</td><td>16</td><td>32</td><td>64</td><td>Can move up to 2, 4, 8, 13, 32, or 64 data items</td> </tr> </table>																			✓	Valid move	=	Valid move, if the number of <i>From</i> and <i>To</i> data items are equal	x	Invalid move	2	4	8	16	32	64	Can move up to 2, 4, 8, 13, 32, or 64 data items
✓	Valid move	=	Valid move, if the number of <i>From</i> and <i>To</i> data items are equal																													
x	Invalid move	2	4	8	16	32	64	Can move up to 2, 4, 8, 13, 32, or 64 data items																								

7.6.5 Shifting and Rotating Arrays

The discussion of the Move block to this point has shown how to move a data item or range of items to another data item or range of data items. The Array data type includes unique elements of the subtypes: Byte, Flag, Float, Number, String, and Timer. The Move block is even more powerful when used to manipulate arrays. In addition to moving data to or from arrays, you can manipulate data items within an array with the “Shift” and “Rotate” functions.

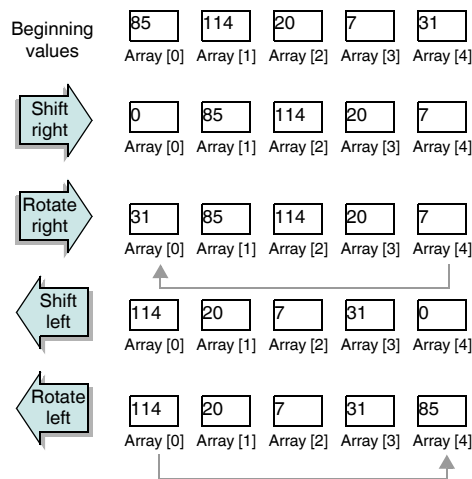
Shift

The Shift function moves each value to the adjacent data item toward a specified end (left or right) of the array. The value of the data item at the opposite end of the array becomes zero.

Rotate

The Rotate function moves each value to the adjacent data toward a specified end (left or right) of the array. In this case, the value at the opposite end of the array is replaced with the data item where the rotation began.

The figure below depicts the directional convention of “Shift”, “Rotate”, “Left”, and “Right” for an example array of five data items.



The examples show a shift or rotation by one position, assuming that each operation shown starts with the beginning values. The Move block lets you shift or rotate by multiple positions to make data manipulation even faster and easier.

The Move block makes the “Shift” and “Rotate” functions available to the Array data type.

To configure a shift or rotate operation:

1. Place a Move block onto the drawing area, and double-click the block to edit its expression.
2. In the “From” group, use the “Data type” drop-down list to select “Array”. This displays “Move,” “Shift” and “Rotate” option buttons.
3. Click the “Shift” or “Rotate” option as needed.
4. In the “Starting data item” field, enter the array name with no index number (such as “Levels[]”), or use the “Find...” button to select the array tagname in the project’s data item table.
5. Click the shift/rotate direction “Left” or “Right” radio button.

- 6. In the "Amount" field, enter the amount of shift or rotate positions for the move, and click the "OK" button.

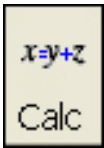
7.6.6 Moving Array Elements

The Move Block can move elements of an array to another array or to a range of data items.

To move element(s) of an array:

- 1. In the "Move Block" expression dialog box, select the array in the "From" group.
- 2. Select the "Move" option.
- 3. Double-click the "Starting data item" field to display the Data Item grid array list.
- 4. Click the array tagname in the grid and click the "OK" button in the "Select Data Item" dialog box. This inserts the array tagname in the "Move Block" dialog box.
- 5. Enter a positive integer or a Number tagname in the brackets (such as "[5]" or "[ArrayNum]"). Double-click the brackets to find a Number in ProjectCenter's Data Item grid.
- 6. If moving more than one array element, repeat steps 4 and 5 in the "Ending data item" field.
- 7. In the "To" group, complete the "Data type" and "Starting data item" fields. Complete the "Ending data item" field if moving multiple elements.

7.7 Using the Calculation Block



Calculation Block Button

The Calculation block in a flow chart lets flow charts perform math and data operations. Each Calculation block produces one result, but its expression may be complex. Figure 7-9 shows the "Calculation Block... Expression" tab.

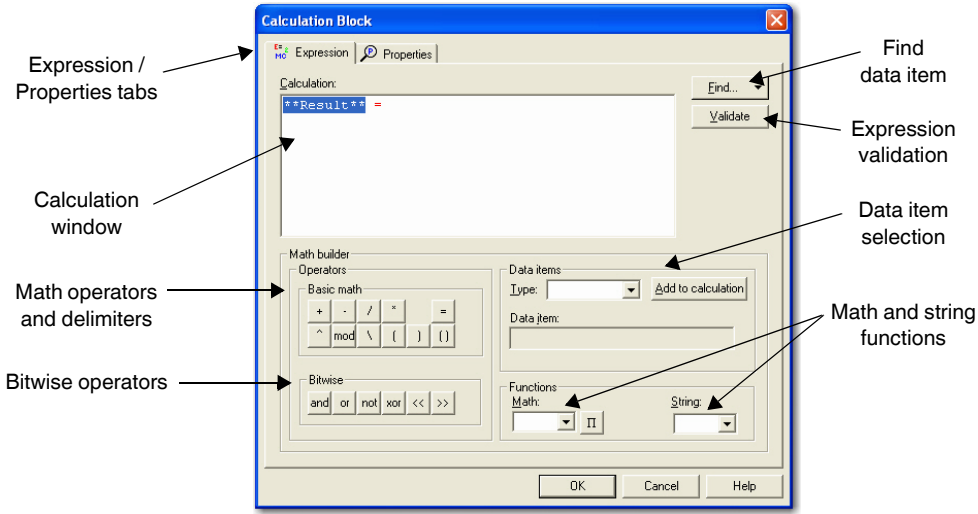


Figure 7-9 The "Calculation Block... Expression" tab

7.7.1 Building a Calculation Expression

The default “Calculation Block... Expression” tab (Figure 7-9) is ready for an expression entry. It lets you build calculation expressions with virtually no typing. Typically, you begin by selecting a data item to contain the calculation result.

To specify a calculation expression result, follow these steps:

1. Select the default “**Result**” text (will be highlighted when selected).
2. In the “Data Items” group, choose the data item type for the calculation result from the drop-down list.
3. In the “Data Item” field, do one of the following:
 - Type the tagname reference.
 - Double-click the field and use the Data Item grid to select a tag.
 - Select the field, and then click the “Find...” menu button. Select either “Find Global Data Item” or “Find Local Data Item”. These selections display the Data Item grid with the relevant tags. Use the Data Item grid to select a tag.
4. Click the “Add to Calculation” button. This inserts the tagname into the expression, replacing any selected text with the tagname.

To complete and validate an expression, follow these steps:

1. Move the text cursor to the right of the equal sign (=).
2. Use the “Data Items” group to select and add the data item.
3. Click the math operator or type it.
4. Type in constants directly.
5. Click the “Validate” button when you have completed the expression to verify that the expression’s syntax is correct.

Complete the “Calculation” field by using the tools in the “Math Builder” group, “Data Items” group, or by manually typing the constants.

Some rules for creating math expressions:

- Use double quote characters around string constants, for example, `String0 = "Hello, world"`
- When a string has an embedded quote character, you must place a backslash character before that quote. For example: `String2 = "He says \"Hello\" to you."`
- Highlight a range of text in the expression to replace it with the next selected data item.
- Expressions may contain up to 50 terms, or 1024 characters (including the result).
- The expression window may contain several lines of text (long expressions wrap to new lines).
- Validate each expression before closing the “Calculation Block” dialog box.

7.7.2 Expression Display Conventions

The expression editor applies color to text components to display how it is interpreting the expression as you create the expression. It also makes use of various parentheses, brackets, etc., to ensure certain parts of the equation are solved before other parts are solved.

The conventions are:

- Tagnames appear in blue. The exception is an array index, which must be enclosed in brackets (black).
- Operators and functions appear in red.

- String constants must be enclosed in double quotation marks and appear in dark magenta.
- Parentheses, constants, and commas appear in black.
- Unrecognized text appears in green. Text appears in green until sufficient characters are added to allow the system to recognize the entry.

7.7.3 Runtime Math Errors

The “Validate” button in the “Calculation Block... Expression” tab finds syntax errors in the equation. Overflow, index out of range, and other errors can occur when running the project. Think & Do Runtime flags these errors using System data items as the Runtime detects them. The errors detected are:

Table 7-5 Runtime Math Errors

ID	Tagname	Description
SYS-39	Math Error	an overflow has occurred on the right side of the equation, such as a divide by zero
SYS-42	Conversion Error	a data conversion from one data type to another exceeded the minimum or maximum value allowed for the receiving data item type
SYS-41	String Error	an error has occurred in the handling of string data item(s)

To check for an overflow condition or error, use a Compare block in the flow chart. Place it immediately after the Calculation block being checked, since the execution of other Calculation blocks overwrites math error system data items.

7.7.4 Basic Math Operators

Table 7-6 lists the basic math operators available in Think & Do:

Table 7-6 Math Operators

Operator	Function
+	Addition or String Concatenation
–	Subtraction
/	Float Division
*	Multiplication
\	Integer Division
^	Exponentiation
mod	Modulus (Remainder after division)
(Open Parenthesis
)	Close Parenthesis
()	Parenthesis
=	Equal



Think & Do formats results to fit the result data type specified in the “Calculation Block... Expression” tab. If the result data item is not a Float, Think & Do truncates any fractional part of the answer. For integer division, Think & Do converts operands to an integer before executing the divide operation.



The mod operation may have variations in its usage. The mod operation returns the remainder after division. Some examples:
 $17 \bmod 5 = 2$
 $-17 \bmod 5 = -2$
 $-17 \bmod -5 = 2$
 $17 \bmod -5 = -2$

7.7.5 Bitwise Operators

The following table lists bitwise operators. Use these operators to perform Boolean operations on two data items to produce a (third item) result, or to manipulate bits within a single data item.

Table 7-7 Bitwise Operators

Operator	Function
and	Bitwise AND – evaluates the state of corresponding bits in two data items. If both bits = 1, then it sets the corresponding bit in the result data item = 1 (otherwise 0).
or	Bitwise OR – evaluates the state of corresponding bits in two data items. If either bit = 1, then it sets the corresponding bit in the result data item = 1 (otherwise 0).
not	Bitwise NOT – evaluates the state of each bit in a data item, producing a result which has the opposite state for each bit (substitutes 1 for 0, and substitutes 0 for 1).
xor	Bitwise Exclusive OR – evaluates the state of corresponding bits in two data items. If either, but not both, bit = 1, then it sets the corresponding bit in the result = 1 (otherwise 0).
<<	Bitwise Shift Left – shifts the bits in a data item (named on the left side of the operator) by the number of positions defined on the right side of the operator, from the LSB towards the MSB.
>>	Bitwise Shift Right – shifts the bits in a data item (named on the left side of the operator) by the number of positions defined on the right side of the operator, from the MSB towards the LSB.

7.7.6 Math Functions

The following table lists the math functions available in the Math Builder group. These include trigonometric and transcendental functions. You can have a data item or expression within the parentheses: Result = Function(Expression)

Table 7-8 Math Functions

Operator	Function
abs	The absolute value of a number is its unsigned magnitude. For example, ABS(-1) and ABS(1) both return 1.
cos	Returns a value which is the cosine of an angle. The value of the number or expression must be valid for an angle in radians.
exp	Returns the base of natural logarithms, e, raised to a power. The constant e is approximately equal to 2.71828. This function returns e raised to the power of x, where x is the value of the expression.
log	Returns a value that is the natural logarithm of a number. The value of the expression must be greater than zero.
log10	Returns a value that is the log base 10 of a number. The value of the expression must be greater than zero.
round	Normalizes a floating point number or expression result to the nearest integer value, eliminating the fractional part. Fractional parts equal to 0.5 or greater round upward.
sin	Returns a value which is the sine of an angle. The value of the expression must be valid for angle in radians.
sqr	Returns the square root of the value specified.
tan	Returns a value which is the tangent of an angle. The value of the expression must be valid for an angle in radians.
pi	The value of π .
trunc	Discards the fractional part of a floating point number or expression result, leaving the integer portion.

Derived Functions

The math functions (Table 7-8) can be used to compute other related math functions. In the following example, (x) represents a data item or constant in radians.

- cosecant(x) = ((sin(x)^(-1))
- cotangent(x) = ((tan(x)^(-1))
- secant(x) = ((cos(x)^(-1))



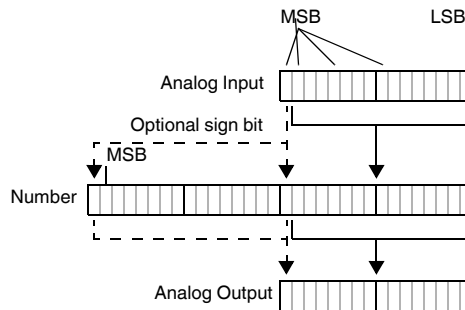
The input range for the trigonometric functions (cos, sin, tan) is -1e+7 to 1e+7 radians. Numbers outside this range will generate a Math Error, SYS-39 and have unexpected results.

7.8 Handling Analog Values

Think & Do has a powerful set of I/O drivers that understand the difference between discrete (individual) I/O points and I/O data representing numerical values. The current set of drivers handle analog values up to 16 bits wide or 15 bits data plus a sign bit (see “Moving Analog Values” on page 7-22). Some controllers impose tedious programming tasks, such as handling sign bits separately from values, and time de-multiplexing the values for 2 to 8 channels from a single data word. Think & Do takes the usual work out of handling analog data.

7.8.1 Mapping analog data

Analog values typically have 8, 12, or 16 data bits. Moving analog input values from the I/O system to data items occurs automatically during each scan. But, you must properly configure (map) analog channels in IOView. Typically, each analog channel is mapped to a Number data item. While a number does have 32 bits, analog input data transfers to the least significant 16 bits of a number. The 16th can optionally be a sign bit for bipolar inputs.



The I/O Mapping tab in IOView includes a “Sign Extend” option for bipolar inputs. Checking the “Sign Extend” option (see Figure 7-10) causes the I/O mapping to transfer the analog input’s sign bit to the 32-bit number’s sign bit.

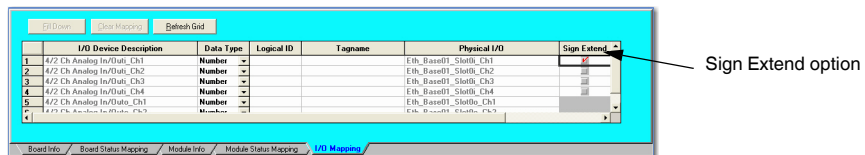
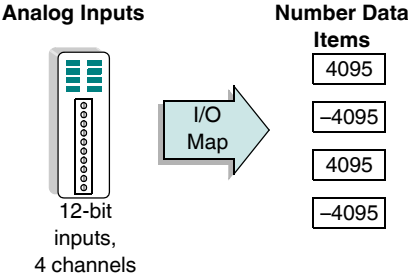


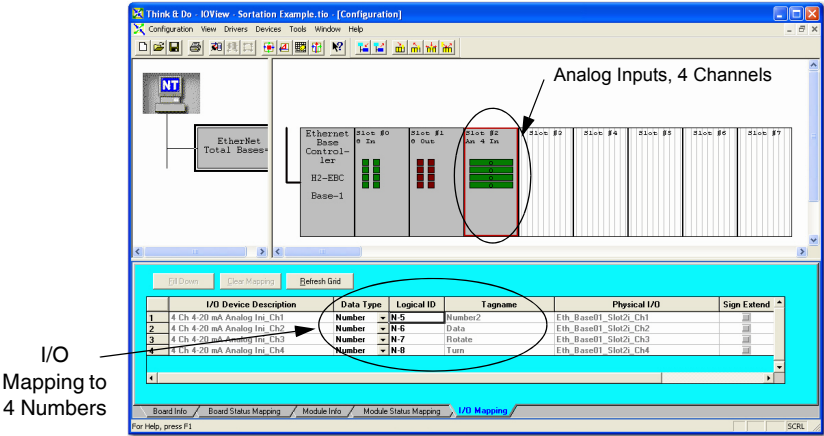
Figure 7-10 IOView includes a Sign Extend option

Sign bits for analog outputs need no configuration. If a number is negative, the MSB of the analog data is set=1.

An analog module with multiple channels will map to one Number data item for each channel. As shown below, 12-bit binary data is already right-justified in the decimal values. After the sign bits for a bipolar input maps to its Number data item's sign bit, the number can be handled just as any other signed number.



In IOView, modules can be added in the base (such as DL405) or some I/O families (such as DL205). There are some I/O families that simply scan the base to add modules. Figure 7-11 shows a 4-channel analog input module is in Slot #2. To map the analog values to number data items, select the module graphic and click the "I/O Mapping" tab. In the Logical ID column, type the number to map or double-click the field to go to the Data Item grid in ProjectCenter to select the data item and return.



Connect Button



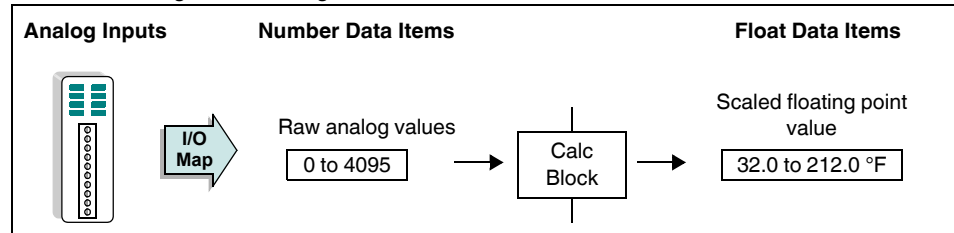
Scan Button

- To see the analog input values in real time:
1. Ensure the analog module exists on the I/O network, which must be powered on and connected to the PC.
 2. Click the "Connect" button on the IOView main toolbar.
 3. Click the "Scan" button on the main toolbar. When IOView establishes the scan to the analog module, the on-screen bar graph depicts the current analog value.

Analog output module on-screen graphics can write to outputs as well. To view a data sheet of the module's specs, wiring, and jumper settings, right-click the module and select the part number from the menu list.

7.8.2 Scaling Analog Data

Getting analog values to/from number data items is easy enough. In many cases, the raw (binary) analog values must be scaled to match the engineering units preferred in the flow chart blocks. For operator screens, values displayed in engineering units (such as degrees, PSI, or gallons per minute) have much more meaning. The Calculation block is the perfect tool to do analog value scaling.



The diagram above shows the data path from the analog input to the scaled value. (Analog outputs work the same way, but in reverse.) Use Float (floating point) data items to store scaled values because they preserve fractional numbers and can represent extremely large or small quantities. Note that data items, including Floats, do not “store” in engineering units... that’s simply a result of having a quantity that relates to standardized units.

Scaling can be done in a single Calculation block using the equation for a line $y = mx + b$, where:

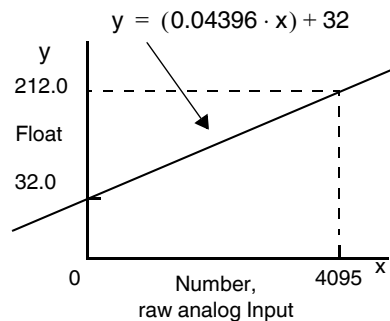
- x is the raw analog value
- y is the scaled value
- m is the slope of the line
- b is the offset, or y-intercept point

In the example above, you compute the slope (m) by evaluating:

$$m = \frac{\Delta y}{\Delta x} = \frac{212 - 32}{4095 - 0} = 0.04396$$

The y-intercept point (b) is 32.

The graph below plots the line equation:



Plug the equation into a Calculation block as shown in Figure 7-12.

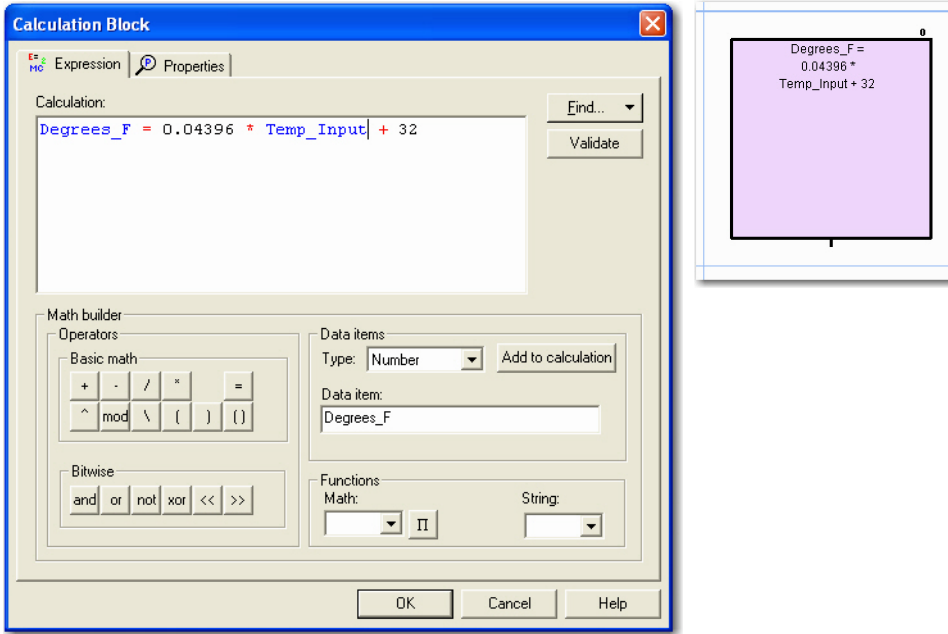
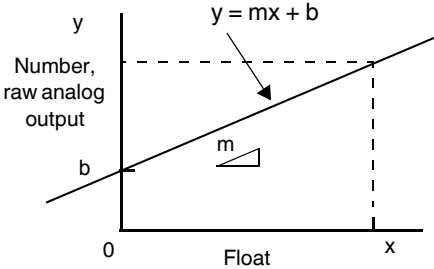


Figure 7-12 Scaling equation in the “Calculation Block... Expression” tab

To scale Float values for analog outputs:

1. Derive the equation for the line in the figure below (the reverse of the prior graph). The scaled (Float) analog value is the x-axis and the converted raw analog value (Number) is the y-axis.
2. For the number (y), use the one mapped to the analog output. In this way, the Calculation block can directly update the analog output value.
3. Update the analog output as often as the application will require it at Runtime.



Using the expression with the standard scaling equation, the project needs just one Calculation block per input or output channel.

7.9 String Handling

This section discusses string type data items and flow chart blocks that handle them. String data items contain UNICODE characters, or text. While text messages in HMI screens are for display and data entry, the embedding of text characters in string data items give you the ability to parse, sort, concatenate, and other typical string handling tasks. Both Move and Calculation blocks are useful in manipulating string data. The Comm block is also useful in collecting string data via a serial port (see “Communications” on page 8-3).

NULL Termination

The internal structure of strings uses the NULL character for string termination. Think & Do ignores characters in a string beyond the first NULL character.

Characters in string → X X X X X X NUL

7.9.1 Using Move Blocks with Strings

String data items contain one or several UNICODE characters. The Move block is the main way to get the characters into a string data item.

Moving Text into a String Data Item

To move text into a String data item:

1. In ProjectCenter, access the project’s Data Item grid and explore the String data type.
2. Type a tagname for a new string.
3. In the Initial column, enter the beginning value for the string at Runtime (optional).
4. Define the length of the string in the “Length” column. The default is 80 characters and the maximum is 255 characters.
5. In FlowView, place a Move block in the drawing area for the flow chart.
6. In the “Move Block...Expression” tab, select “String Constant” for the “From Data” type.
7. In the “String value” field, enter the new text.
8. In the “To” group, specify “String” as the data type and enter the destination string tagname in the “Starting data item” field. Then click the “OK” button.

Moving a String Data Item to Another String Data Item

To move the contents of one String data item into another:

1. Repeat the steps from section “Moving Text into a String Data Item”, above, but choose “String” for both data types
2. Specify the two strings in the “From” and “To” group’s settings.



All characters in the first string can move if the length of the second string is the same or greater. If the length of the second string is less, the move will truncate the right-most part that does not fit.

Moving Ranges of String Data Items

One of the best uses of the Move block for string handling is to move a range of string data items to another set of data items. If these two groups of string data items are in Arrays (of type String), one array can move into the other array, as long as both arrays have the same number of elements. For each string moved, truncation only occurs if the destination string data item is shorter than the source string.

7.9.2 Exchanging Strings With Other Types

To use a text string as a number, UNICODE data must be converted into numerical digit values. When using the Move Block to move a string data item into a number data item, Think & Do does this conversion automatically. For example, the string "1659" can be converted to the number 1659.

Conversely, a number can be converted to a representative text string by using a Move block. In the "From" group, define the tagname and data type for the number. In the "To" group, define the destination string data item.

7.9.3 Using the Calculation Block with Strings

The Calculation block is very powerful, and includes a set of functions for manipulating string data items and values:

Table 7-9 Calculation Block Functions

Function	Description
asc	returns the value of the first character of a string
chr	sets a string to the character whose value is x
find	returns the location of a string's position within another string
mid	returns a substring from within a string
left	returns the left-most N characters of a string
right	returns the right-most N characters of a string
len	returns the number of characters in a string
str	returns a string representation of a number
val	returns the value of characters in a string as a numeric value of the appropriate type

In addition to the functions above, the Calculation block can concatenate strings (join end-to-end) by using the addition "+" function.



Complex nesting of string operations may produce build errors. If this occurs, separate the expression into multiple Calculation blocks.

Find Function

The Find function determines the location of a string within another string data item. The result returns an integer, representing the zero-based index (location) of the first occurrence of the substring. If the substring is not found, Find returns a (-1).

The format of the expression using the Find function in the Calculation field is:

****Result**** = Find (source,substring), where:
 *****Result***** is the number data item that receives the location
 “source” is the string to be searched
 “substring” is the string to be found

Mid Function

The Mid function extracts and returns a substring from within a string. Certain parameters can be specified that allow the search to find the substring.

The format of the expression in the Calculation field is:

****Result**** = Mid(source,start location,len), where:
 *****Result***** is the string data item that receives the substring
 “source” is the string to be searched
 “start location” is the zero-based position in the source string where the substring begins
 “len” is a fixed integer or number data item that determines the number of characters in the substring result



If “len” is less than or equal to zero, it defaults to the remaining length of the source string. If the “start location” is greater than the length of the source string, the function returns a NULL string.

The “start location” and “len” parameters may (independently) be either a number, a constant, or a function entered in the expression.

Left and Right Functions

The Left and Right functions extract and return the N left-most or right-most characters from a string as a substring. The number of characters “N” is specified in the substring.

The format of the expression in the Calculation field is:

****Result**** = Left(source,num), or
****Result**** = Right(source,num), where:
 *****Result***** is the string data item that receives the substring found
 “source” is the string to be searched
 “num” determines the number of characters in the substring result

Len Function

The Len function returns an integer equal to the length (number of characters) of a string data item.

The format of the expression in the Calculation field is:

****Result**** = Len(source), where:
 “Result” is the number data item that receives the length
 “source” is the string data item or expression to be measured

7.9.4 Comparing and Sorting Strings

The Compare Block is very useful for comparing and sorting strings.

To compare two string data items:

1. Place a Compare block in the flow chart.
2. Edit the compare expression in the “Compare Block... Expression” tab, specifying the string data items to compare. One of the strings can be a string constant.
3. Select the comparison type you want to make from the options: = (Equal to), > Greater than), < (Less than), or Instr (is contained in).

A typical use for string compares is for alphanumeric sorting.



Consider using two arrays for string sorting — one for the original list and one for the sorted list. Set up a flow chart with a Do Loop. Inside the Do Loop, increment numbers that serve to index the arrays.

The Instr function returns TRUE if the first string occurs anywhere in the second string.



Using the Instr function can save a lot of programming time. You can avoid using the Mid function and a Do Loop searching flow chart if it does not matter where in the string the substring occurs.

All string comparison operations may be case-sensitive. If you select the “Case Sensitive” check box, the comparison can only be true if the upper/lower case of each pair of characters match.

7.10 SQL Database Operations

The SQL block provides a dialog box that helps construct SQL queries for database access. After placing an SQL block in a flow chart, double-click the block to display the “SQL Block... General” tab.

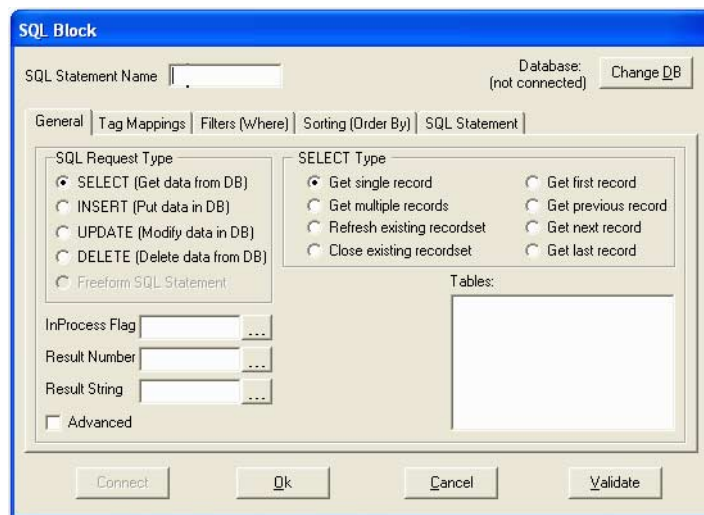


Figure 7-13 The “SQL Block” dialog box helps construct SQL statements

7.10.1 SQL Block Overview

The “SQL Block” dialog box has the following tabs:

- General: Configures the request type, record type, and query result data items
- Tag Mappings: Map record fields to project tagnames
- Filters (Where): Specify operators and conditions for field names
- Sorting (Order By): Specifies field names and the sorting order
- SQL Statement: Displays the resulting SQL statement (cannot be directly edited — use the tabs to change the SQL statement)

There are fields and buttons that appear on all tabs. They are:

SQL Statement Name

This is a general purpose comment field that (currently) plays no specific purpose at this time. You can use it as a simple identifier/description.

Change DB

The “Change DB” button selects an OLE DB Provider to be used for the SQL request. The OLE DB provider can be any one of many provided with Windows including (but not limited to) Microsoft SQL Server, Microsoft Access, Oracle, and ODBC.



To use Microsoft Access databases, select the “Microsoft Jet 4.0 OLE DB Provider.”

Additionally, there are four buttons at the bottom of the dialog box which are global to the dialog and pertinent to the SQL block operation.

Connect

Click the “Connect” button to connect to the database or data source identified by the “Change DB” button. Connecting to the database reads the list of tables and displays them in the “Tables” list on the “General” tab.

This button is NOT available when in “Online Monitor” or “Online Monitor and Edit” modes.



You must reconnect whenever you close and re-open the SQL block.

OK

The “OK” button accepts and validates the existing SQL request choices, commits the data to the block configuration, and closes the “SQL Block” dialog box.

This button is NOT available when in “Online Monitor” or “Online Monitor and Edit” modes.



Click this button after fully defining the SQL statement using all the tabs in this dialog box.

Cancel

The “Cancel” button cancels any changes made during the current configuration session. If the block had an existing configuration when the dialog was displayed, all previous configuration data is preserved.

This button is NOT available when in “Online Monitor” or “Online Monitor and Edit” modes.

Validate

The “Validate” button updates all existing choices in the “SQL Block” dialog box and validates selections and entries. It does not save the data to the block configuration and does not close the “SQL Block” dialog box.

This choice is NOT available when in “Online Monitor” or “Online Monitor and Edit” modes.

The “SQL Block” dialog box has five tabs. Use one or more of these to build a valid SQL request. Each tab and its contents are described in the next several sections.

7.10.2 General Tab

The “SQL Block... General” tab provides options to select the database, name the query, specify the query type and select type, tables to query, and status flags.

The General tab defines the top-level SQL request type and the number of records accessed.

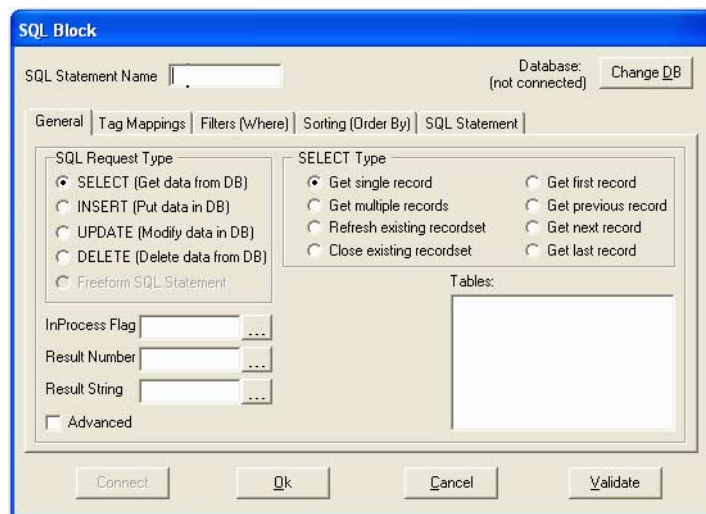


Figure 7-14 The “SQL Block... General” tab

SQL Request Type

These radio buttons represent the top-level SQL request types available. The types and purposes are:

- “SELECT (Get data from DB)” — retrieves existing record(s) from the database or data source and writes the fields’ data to tags as defined in the “Tag Mappings” tab. When selected, this radio button enables the “SELECT Type” radio buttons.
- “INSERT (Put data in DB)” — adds new records to a database or data source using the fields and respective tag data as defined in the “Tag Mappings” tab.
- “UPDATE (Update data in DB)” — changes existing records in the database or data source with the data in the tags as defined in the “Tag Mappings” tab.

- DELETE (Delete data in DB) — removes records from the database or data source and does not require tag mapping.



The Delete function removes records from the database. Be sure to have a backup image of the data in the event of a programming error.

SELECT Type

These radio buttons are available after choosing the “SELECT (Get data from DB)” radio button. It determines the type of SELECT request executed. The types available are:

- “Get single record” — generates an SQL request that retrieves all records matching the SQL request conditions, copies the first record’s field values to the corresponding tags as defined in the “Tag Mappings” tab, and then closes the recordset.
- “Get multiple records” — This option requires closing unless the “Tag Mappings” tab defines arrays with the Fill Arrays option selected (“Tag Mapping Tab” on page 7-42). It generates an SQL request that retrieves all records matching the SQL request conditions and copies the first record’s field values to the corresponding tags as defined in the “Tag Mappings” tab. The recordset is automatically closed when “Fill Arrays” is selected, otherwise it is not closed. When “Fill Arrays” is not used, the recordset is left open for subsequent SQL blocks that may perform operations on the same recordset. A complementary “Close existing recordset” request is required for each “Get multiple records” request.
- “Refresh existing recordset” — retrieves the current recordset from the database. This may be necessary because selection of multiple records caches selected records in memory. Get first, previous, next and last commands operate on the cached recordset not on the actual database records. If an update modifies the database after the “Get multiple records” is selected, the updated records are not reflected in the cached records unless the “Refresh” command is executed.
- “Close existing recordset” — generates a request to close an existing recordset after its use. This does not generate any data transfer between the database or data source and tag values.
- “Get first record” — generates a request to move to the first record in an open recordset (independent of the current record position), and copy that record’s field values to the corresponding tags as defined in the “Tag Mappings” tab.
- “Get previous record” — generates a request to move to the previous record in an open recordset (given the current recordset’s order), and copy that record’s field values to the corresponding tags as defined in the “Tag Mappings” tab.
- “Get next record” — generates a request to move to the next record in an open recordset (given the current recordset’s order), and copy that record’s field values to the corresponding tags as defined in the “Tag Mappings” tab.
- “Get last record” — generates a request to move to the last record in an open recordset (independent of the current record position), and copy that record’s field values to the corresponding tags as defined in the “Tag Mappings” tab.

There are three required tag mappings that must exist for the SQL block to operate:

InProcess Flag

This required item must be mapped to a tag (Data Type: Flag), which gets set to ON (TRUE) when the SQL block executes. Since the actual SQL request gets processed asynchronously with the Runtime, a decision block must be inserted immediately following the SQL block, and then the flag item must go OFF (FALSE) before proceeding to other logic in the same flow chart (other flow charts will continue to be processed during this time).

To select a tag, one can double-click the box to receive the tag or single-click the button with ellipses to the right of the edit box. Each of these actions opens a dialog box from which to choose an appropriate type of tag for this item. Manual entry is not allowed in this cell.

Check this Flag after each SQL block. It is also a prudent choice (though not necessary) to use a different InProcess Flag tag for each SQL block.

Result Number

This required item must be mapped to a tag (Data Type: Number), which will be set to a code representing the result of the request. The value of this code can have one of two meanings. If the resulting code is negative, an error occurred. The negative value indicates the error that occurred. If the value is non-negative (zero or greater than zero), the requested operation was successful. This code holds the quantity of records affected by the operation.

For a SELECT request, the tag holds the number of records retrieved by the SQL request. For an INSERT request, it holds the number of records inserted into the database or data source. For an UPDATE request, it holds the number of records updated by the request. For a DELETE request, it holds the number of records deleted from the database or data source.

To select a tag, double-click the box to receive the tag or single-click the button with ellipses to the right of the edit box. Each of these actions opens a dialog box from which to choose the appropriate type of tag for this item. Manual entry is not allowed in this cell.

Note that a value of 0 (zero) is a perfectly valid result code. A Close request returns 0 if the recordset is open (it returns an error if already closed). Requests that are valid, but whose filter conditions do not return or affect any records return 0.

This Number should be checked after each InProcess flag check. It is also prudent (but not necessary) to use a different Result Number tag for each SQL block.

Result String

This required item must be mapped to a tag (Data Type: String), which receives one of two types of values. For successful requests, the tag receives "Request complete" after all mapped tags have been written. For unsuccessful requests, this tag holds a descriptive string indicating the cause of the failed request.

To select a tag, double-click the box to receive the tag or single-click the button with ellipses to the right of the edit box. Each of these actions opens a dialog box from which to choose the appropriate type of tag for this item. Manual entry is not allowed in this cell.

This String need not be checked after each SQL block, as it is intended to provide diagnostic information as opposed to error information. However, it is still a prudent choice (though not necessary) to use a different Result String tag for each SQL block.

Other items on the General tab that are not in logical groups are:

Advanced

Select the "Advanced" check box to display a database's or data source's *schema*. A *schema* is a logical collection of a database's or data source's objects, which includes *tables* and *views* or *queries*. Not all OLE database providers support or provide this information (Microsoft Access, for instance, does not). If this item is checked and the provider supports it, the schema objects appear in the Schema list.

Schema List

The Schema list is exposed by the “Advanced” check box item once the database or data source is connected to via the “Connect” button. If the “Advanced” button is not checked, this list is not shown. If it is checked and the Provider does not support schemas, this list is empty. A check mark in the box next to a schema object adds the selected schema’s tables and views to the Tables list.

Tables List

This list is filled with tables and views or queries once the database or data source has been connected via the “Connect” button. If the database or data source supports schemas, this list changes with any change of the schema list selection. A check mark next to a table, view, or query adds that object’s fields to the list available for tag mapping.

Use this list to select one or more tables to be used in the SQL request. The “Tables” list is only available from the “General” tab and only after selecting a database or data source.

7.10.3 Tag Mapping Tab

The “SQL Block... Tag Mappings” tab (Figure 7-15) maps database fields to Think & Do tags.

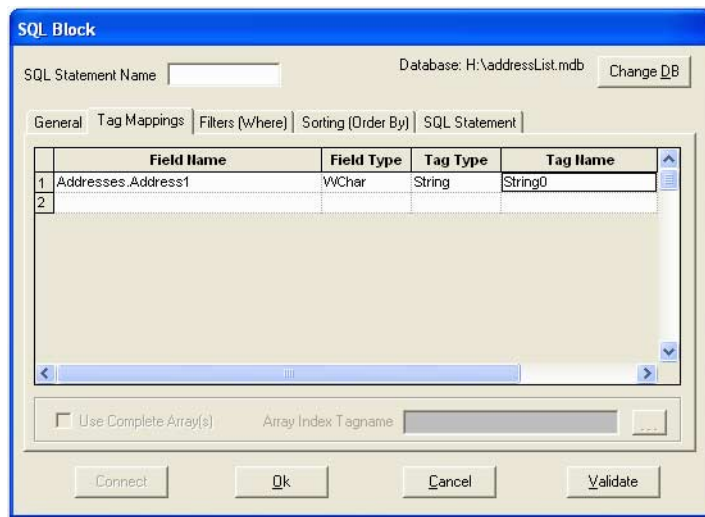


Figure 7-15 The “SQL Block... Tag Mappings” tab

This mapping occurs between the “Field Name” column and the “Tagname” column.

Field Name

The “Field Name” column shows the fields that are available in the tables, views, and queries selected on the “General” tab. Click in the column and select a field from the drop-down list that appears. Manual entry is not allowed in this field.

Field Type

The “Field Type” column displays the native database or data source type for the field selected in the “Field Name” column. This cell is read-only and is provided solely for informational purposes.

Tag Type

The “Tag Type” column displays the Think & Do data type for the tag selected in the “Tagname” column. This cell is read-only and is provided solely for informational purposes.

Tagname

The “Tagname” column permits selection of a Think & Do tag (via a pop-up dialog window) that receives transfers with the database or data source field name specified in the “Field Name” column. Double-clicking pops up the tag choice dialog box for tag selection. The dialog box shows the tag data type that matches the field type in a drop-down list field. Use the drop-down list to select “Array” if choosing an array of the appropriate type. Manual entry is not allowed in this field.

Use Complete Array(s)

The “Use Complete Array(s)” check box is available when selecting “Get multiple records” on the “General” tab. With this check box selected, Think & Do retrieves all records that satisfy the query and fills the field arrays with entries from the recordset until the smallest array is full or all records have been transferred. With this option selected, it is not necessary to explicitly close the recordset.



The “Use Complete Array(s)” operation does not clear unused array elements. If you want unused elements cleared, you must perform this operation prior to executing the SQL block.

When used with “INSERT (Put data in DB)” function, the SQL block inserts records into the database until all elements of the smallest array are exhausted.

Array Index Tagname

Use the “Array Index Tagname” field to enter a Number tagname with an array index. Double-click the field, or click the browse button, to display the tag chooser. Enter an array index tagname for either single or multiple select, or insert or update operations. Entering a tagname in this field with “Get multiple records” selected, transfers a single record from the recordset and leaves the recordset open. Manual entry is not allowed in this field.

7.10.4 Filters (Where) Tab

The “SQL Block... Filters (Where)” tab (Figure 7-16) lets you apply conditions to the query so it returns fewer records.

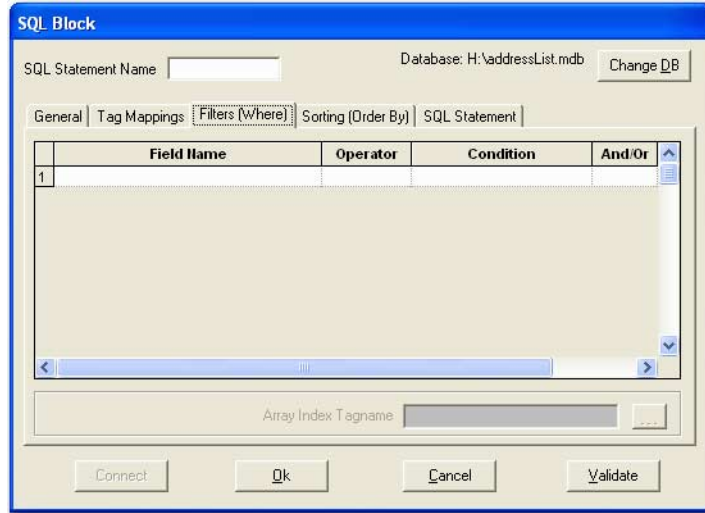


Figure 7-16 The “SQL Block... Filters” tab

Field Name

The “Field Name” column selects fields available in the tables, views, and queries on the “General” tab. Click in the column and select a field from the drop-down list that appears. Manual entry is not allowed in this field.

Operator

The “Operator” column exposes a list of available relational operators:

- =
- >
- >=
- <
- <=
- <>
- Like

One of these operators is required for each row in the Filter table. Although most of these operators are self-explanatory, the “Like” operator provides a wild card-style lookup. The format needed for the Like operator usage is database or data source dependent.

Condition

The “Condition” column allows selection of a Think & Do tag, array, or another database or data source field via a pop-up dialog window. Its value is compared to the value of the field specified in the “Field Name” column. By specifying another database or data source field in this column, one can accomplish JOINS. Manual entry is not allowed in this field.



The condition is crucially important when more than one table is selected on the “General” tab. Without the proper Field Name equates, a Cartesian Join results, returning every possible combination of the two tables!

AND/OR

If specifying more than one filter, they must connect with logical operators: AND and OR. Be careful how the filters are ordered. Think & Do resolves ANDs first (top to bottom) followed by ORs (also top to bottom).

Array Index Tagname

Use the “Array Index Tagname” field to enter a Number tagname with an array index. Double-click the field, or click the browse button, to display the tag chooser. Enter an array index tagname for either single or multiple select or insert, update, or delete operations. Manual entry is not allowed in this field.

7.10.5 Sorting (Order By) Tab

The “SQL Block... Sorting” tab (Figure 7-17) controls the order in which it returns records.

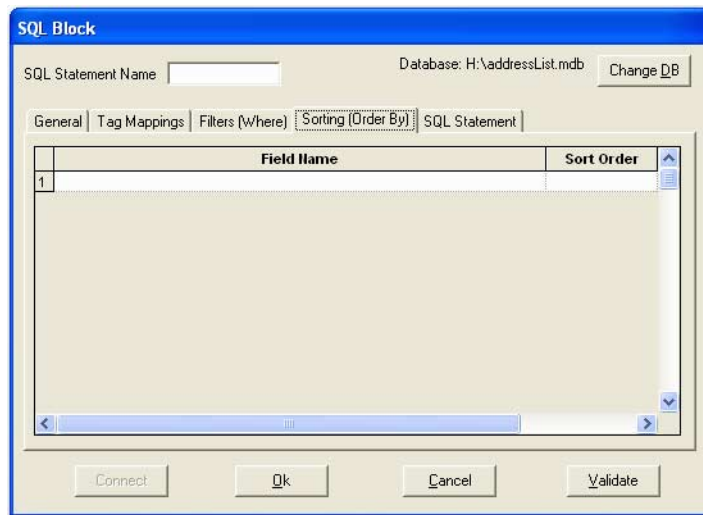


Figure 7-17 The “SQL Block... Sorting” tab

7.10.5.1 Field Name

The “Field Name” column exposes a list of the table (or view or query) fields available based upon the tables (or views or queries) selected on the “General” tab. Click in the column and select a field from the drop-down list that appears. Manual entry is not allowed in this cell.

7.10.5.2 Sort Order

The “Sort Order” column specifies how to sort each field: in ascending (ASC) or descending (DESC) order.

7.10.6 SQL Statement Tab

The “SQL Statement” tab (Figure 7-18) displays the resulting SQL statement. It is read-only.

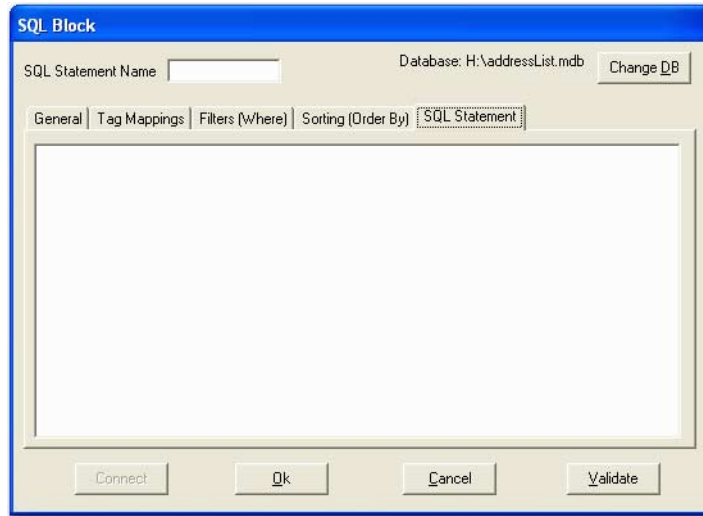


Figure 7-18 The resulting SQL in the “SQL Statement” tab

7.11 PID Block for Process Control

Think & Do includes PID (Proportional-Integral-Derivative) process control with a sophisticated set of features to address many application needs. The main features are:

- Up to 64 loops
- Loops have independent sample rates
- Manual / Automatic / Cascaded looping
- Bumpless Transfer mode
- Position or Velocity PID algorithms

The process control loop capability is accessible in flow charts using the PID Block. You just add one PID Block for each loop in your process. All sensor and actuator wiring connects to the appropriate modules in the I/O sub-system. The process variables, gain values, and other loop data are stored in the common tagname database. Most of the scalar values are floating point numbers (type Float). Loop configuration is made easy with tabbed dialog boxes that configure each PID Block.

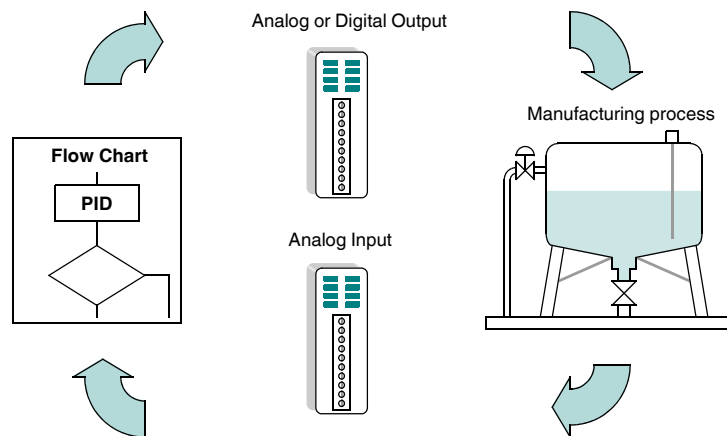


Figure 7-19 PID Control

At Runtime, the flow chart must execute a loop's PID Block at a rate suitable for stable loop control. The I/O associated with each loop is read and written during each scan. The PID Block's loop calculations use the data only when it does the loop PID calculation. The result of the PID calculation is the control output command.

For a complete description of how the loops operate and what you must do to use them, see the "PID for Process Control User Manual."



The tabbed dialog boxes for the PID Block provide an easy-to-use loop-configuring tool. After completing the setup, the Trend object in HMI screen objects will make a good loop trending tool to help tune the loop(s).

The following table provides a feature-by-feature description of the PID loops.

Table 7-10 PID Loops

Feature	Specifications
Number of loops	Up to 64 PID Blocks can be used in one project
PID algorithm	Position or Velocity form of the PID equation
Control Output polarity	Output range is selectable, can be unipolar or bipolar
Error term curves	Selectable as linear and error squared
Loop update rate (time between PID calculation)	The flow chart executing the PID Block determines the loop update rate. The fastest time a PID Block can be executed is once per 10 ms.
Loop modes	Automatic, Manual (operator control), or Cascade control
Setpoint Range	Specify Lower Span and Upper Span
Process Variable Range	Utilizes the same span endpoints as the Setpoint
Proportional Gain	Specify gains in full floating point number range (unit-less)
Integrator (Reset)	Specify reset time in full floating point number range (in minutes)
Derivative (Rate)	Specify derivative time in full floating point number range (in minutes)
Rate Limits	Specify derivative smoothing factor (a low-pass filter constant)
Bumpless Transfer	In Manual Mode, the Setpoint can track the PV, or hold its current value
Error Bias	Provides instant bias offset adjustment to implement feed forward control
Anti-windup, Freeze bias	Inhibits further increases or decreases of the output by implementing either anti-windup or freeze bias to help recovery time from loop saturation
Error Deadband	Specify a tolerance (plus and minus) for the error term (SP-PV), so that no change in control output value is made
Output Limits	Specify lower and upper limits for the control output data item
Wild Flow	Use a second PV input for doing flow control with the ratio PV1/PV2

This section informs you about:

- Serial port communications
- Using OPC Server
- Using DDE Server
- Setting up remote communications

Communications	8-3
8.1 Serial Port Communications	8-3
8.1.1 Adding the Serial Driver	8-3
8.1.2 Adding a Serial Port	8-4
8.1.3 Serial Port Settings	8-6
8.1.4 Creating Status Data Items	8-7
8.1.5 Mapping the Serial Port	8-7
8.1.6 Testing Communications	8-8
8.2 Using OPC Server	8-10
8.2.1 OPC Client Applications	8-11
8.2.2 OPC Explorer.....	8-12
8.2.3 Configuring OPC Server	8-13
8.2.4 Configuring a Project as an OPC Client	8-14
8.3 Using DDE Server	8-16
8.3.1 Introduction to DDE Servers	8-17
8.3.2 Getting DDE Communications Started.....	8-18
8.3.3 DDE Addresses	8-18
8.3.4 Launching the DDE Server	8-19
8.3.5 Configuring DDE Server	8-20
8.3.6 Reading and Writing Tagname Values (Peek and Poke)	8-20
8.3.7 Reading and Writing Tagname Values with External Programs	8-21
8.4 Setting Up Remote Communications.....	8-23
8.4.1 Runtime System Setup	8-24
8.4.2 Development System Setup	8-25
8.4.3 Synchronize PC Time-of-Day Clocks.....	8-25
8.4.4 Connect to the Remote System	8-25

Think & Do

8 Communications

Think & Do projects can communicate with a number of external devices and with other programs. A project can read the status of and control other devices through the Runtime target's serial port and expansion slots. It can also read from and send information to other programs using one of several communication standards.

8.1 Serial Port Communications

The serial communications (COM) port on the Runtime target provides a convenient way for a Think & Do project to communicate with various external devices. These include bar code readers, weigh scales, message displays and printers. While most I/O systems use dedicated high-speed PC interface cards, Think & Do can also utilize serial communications for a variety of I/O systems. In this section we will focus on communications with a typical peripheral such as a bar code reader (Figure 8-1).

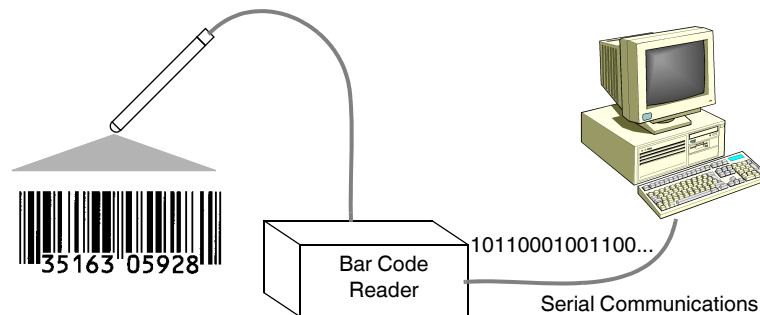


Figure 8-1 Think & Do can communicate with many different devices

To configure serial communications (details provided in subsequent sections):

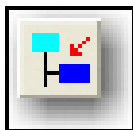
1. Add and configure the serial driver in IOView.
2. Create a data item for the COM port.
3. Create Byte array data items or String data items to store data that is being sent or received.
4. Use a Communications Block in a flow chart to communicate through the driver and comm port with the external serial device.

8.1.1 Adding the Serial Driver

Project flow charts require configuring the Think & Do serial driver to communicate through a serial port to external devices.

To add the serial driver to a project:

1. Launch IOView.
2. Click the "Add Driver" button on the "Configuration" toolbar to open the "Add I/O Driver" dialog box.
3. Select Serial Driver from the list.



Add Driver
Button

4. Click the "OK" button to close the dialog box and complete the addition.

A graphical representation of the driver appears in the Board View (upper left) pane of the IOView window (Figure 8-2). The graphic identifies the driver and indicates that it has been successfully added. A table appears on the Board Info tab (lower left) that shows the driver's name, which cannot be edited.

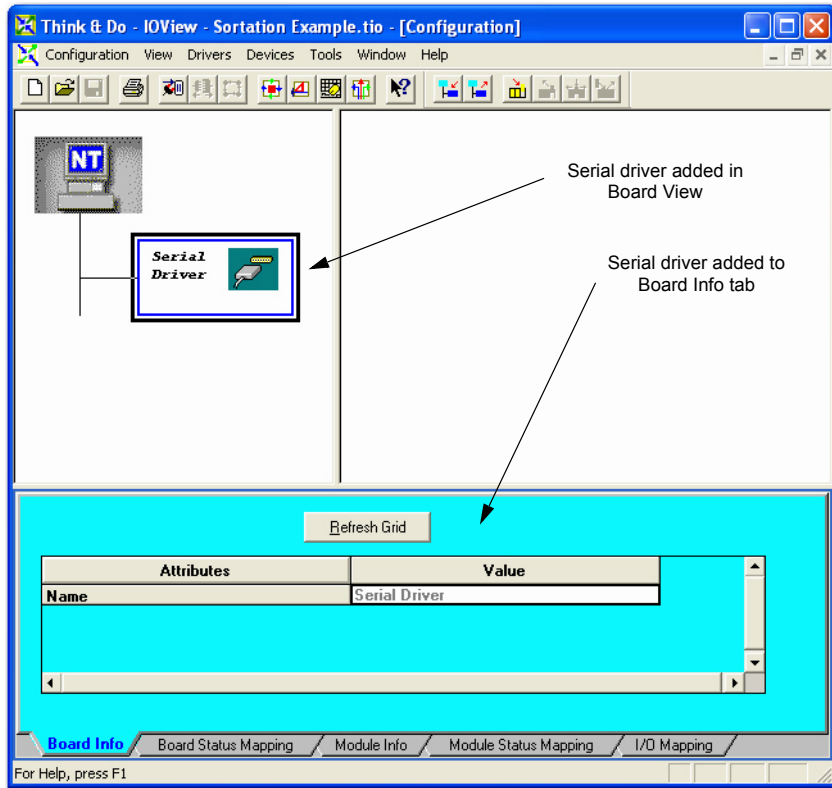


Figure 8-2 IOView window

8.1.2 Adding a Serial Port

In IOView, physical COM ports or serial ports are classified with all devices.

To add a particular serial port:

1. Select the Serial Driver graphic in the Board View pane, highlighting the driver outline.
2. Click the "Add Device" button on the Configuration toolbar, or select the "Devices... Add..." menu. The "Add Serial Device" dialog box appears as shown in Figure 8-3.



Add Devices Button

- Choose an available serial port from the list box. Be sure the mouse or other device is not already attached to the port you choose.

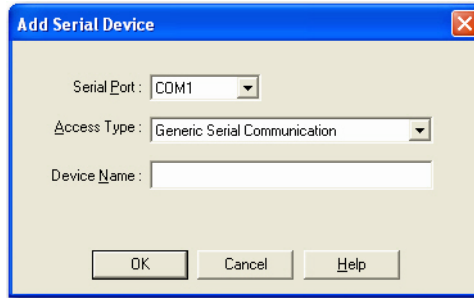


Figure 8-3 The “Add Serial Device” dialog box

- Type a name for the device in the “Device Name” field. Use the hardware port name or the name of the external device being connected.
- Click the “OK” button to close the dialog box.



After adding the port, the name (such as COM1) automatically disappears from the drop-down list. However, COM ports in use by other drivers in the system will still be on the list. If configuring these for use in IOView, a communications conflict will occur. Therefore, check for COM port usage before assigning a COM port to a Think & Do device.

After adding a serial port device, click the “Module Info” tab for the view shown below.

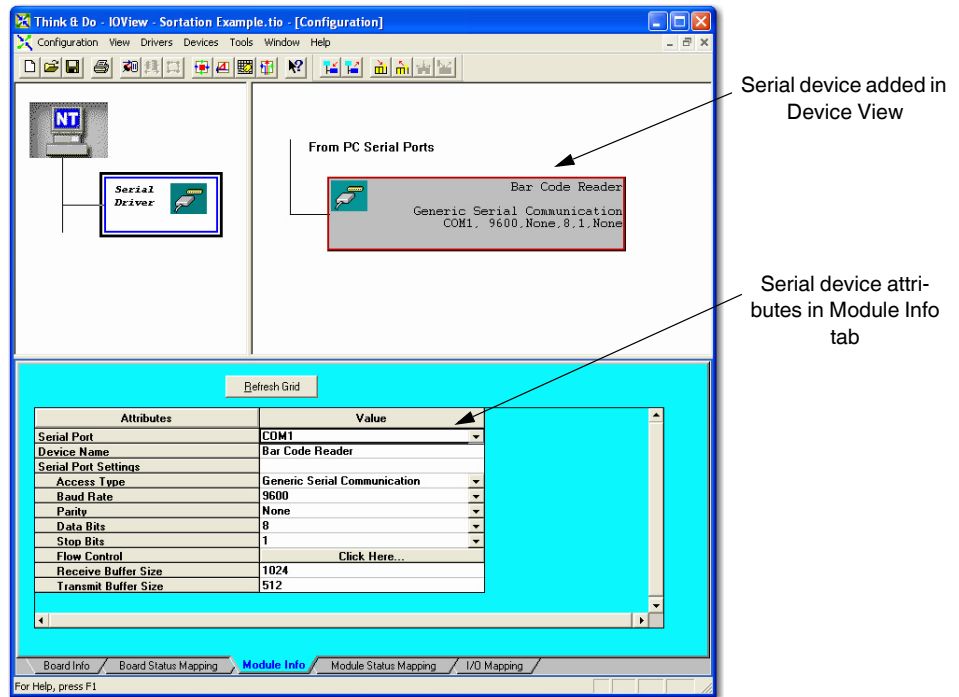


Figure 8-4 IOView “Module Info” tab

The serial port device is now represented in the Module View (upper right pane). The graphic identifies the port device and shows it has been successfully added. The port and default information are listed in the “Module Info” tab. All of the attributes of the port may be edited in the fields of the “Module Info” tab.



The attributes in the “Module Info” tab are used only by the serial port driver. Changes made do not reconfigure the device attached to the serial port. If communications problems are encountered, please consult the documentation for the attached device. In most cases, a change to one or more the driver settings will correct the problem.

8.1.3 Serial Port Settings

The serial port settings appear in the “Module Info” tab (Figure 8-5). These configure the driver for the proper speed of transmission and reception, and how to structure and interpret the data, etc. Usually, the documentation accompanying the device attached to the serial port will specify the communications settings.

Attributes	Value
Serial Port	COM1
Device Name	Bar Code Reader
Serial Port Settings	
Access Type	Generic Serial Communication
Baud Rate	9600
Parity	None
Data Bits	8
Stop Bits	1
Flow Control	Click Here...
Receive Buffer Size	1024
Transmit Buffer Size	512

Figure 8-5 Serial Port Settings (“Module Info” tab)

Baud Rate

The transmission speed, in bits per second (bps), or baud. The default speed is 9600 baud.

Parity

The error checking bit. Each 8-bit character may optionally have a 9th parity bit. The parity setting specifies the polarity and other conventions about the parity bit, if used. The setting options are: none, odd, even, mark, and space.

Data bits

The number of bits used for a character. Most modern devices use eight (8) bits per ASCII character transmitted.

Stop bits

The number of bits used to indicate the end of a character. The default setting is 1 bit.

Receive buffer size

Specifies the size of the memory space allocated for the storage of incoming data on your Runtime hardware. The default size is 1024 bytes. Valid options are from 128 to 32767 bytes.

Transmit buffer size

Specifies the size of the memory space allocated for the storage of data awaiting transmission. The default is 512 bytes. Valid options are from 128 bytes to 32767 bytes.



Some Windows CE targets, such as the WinPLC, may not provide all the serial port settings listed above. See the “Module Info” tab in IOView when configured for your particular Runtime device for applicable settings.

8.1.4 Creating Status Data Items

When properly configured, serial communications provide simple, inexpensive and reliable communications. Occasionally, you will need some status information. For troubleshooting, Think & Do provides extensive diagnostic information. With the serial device graphic selected in IOView, click the “Module Status Mapping” tab to access the status information.

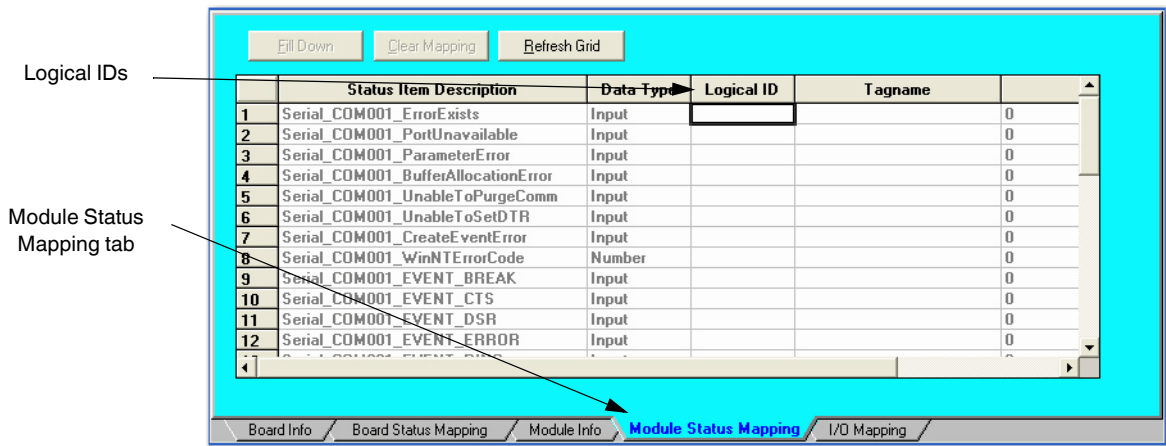


Figure 8-6 Serial port “Module Status Mapping” tab

A status item’s value equals 1 when an error exists or to indicate a busy condition. State information is useful in flow charts and screens (you only need to add names to status items that you intend to use in flow charts or screens). First, if a data item already exists, just enter its ID number in the field provided. The data type is already shown. If a number, such as “3,” is typed in the top field, Input I-3 is assigned to “Serial-COM001_ErrorExists”. The tagname for I-3 could be “COMerr” for example, as specified in the project’s Data Item grid.



Some status data items are not available on certain Windows CE Runtime devices. For example, the WinPLC does not provide CTR and DTR items.

8.1.5 Mapping the Serial Port

Each serial port device maps to a data item of the Comm data type.

To map a serial port to a data item:

1. In IOView, select the serial port device in the “Module View” window.
2. Click the right-most tab to access the “I/O Mapping” tab.
3. In the “Logical ID” column, do one of the following:
 - Enter the logical ID number of an existing Comm data item.
 - Bring up the Data Item grid and select or define a Comm data item — just double-click the tagname field. Enter the Comm data item tagname and return to IOView.

In our example, the logical ID is “COM-1”, but it is more convenient to refer to it by the name of the attached device (“BarCodeData”). This is stored with the project, along with all other data items.

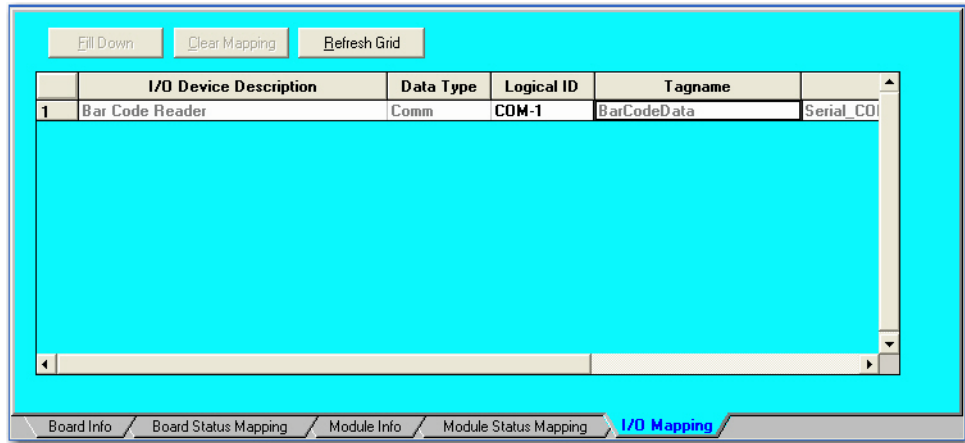


Figure 8-7 Serial port “I/O Mapping” tab

8.1.6 Testing Communications

After configuring the communications port, we recommend doing some simple tests. Think & Do’s IOView can perform single transactions by command.

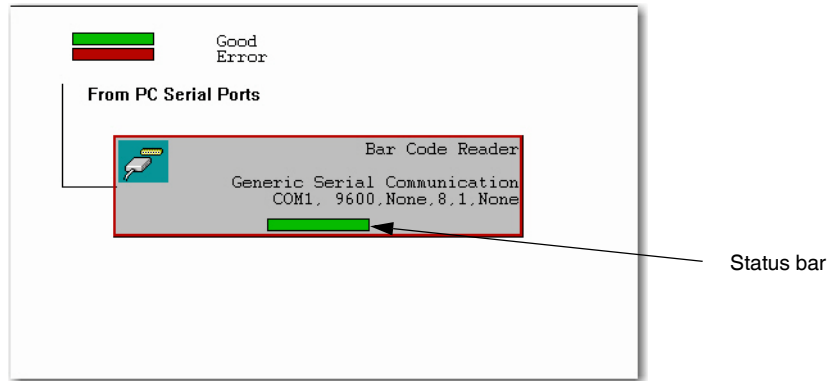
To conduct a basic serial port check:

1. Be sure the communications cable (RS-232, RS-422, etc.) is connected to the external device and that the device is powered and ready.
2. Click the “Connect” button on the toolbar in IOView. This only connects Think & Do to the serial port in the Runtime target (not the external device).



Connect Button

- Verify the status bar in the serial device graphic is green (status okay). If a red status bar indicates an error, check to see if that serial port actually exists, or if there are interrupt or address conflicts, etc.). Correct any conflict before continuing the test.



- Click the "Scan" button on the toolbar. This enables the Runtime target's serial port to communicate externally. The "Serial Port Scanning" dialog box (Figure 8-8) appears, which allows manually transmitting and receiving characters to/from the device.

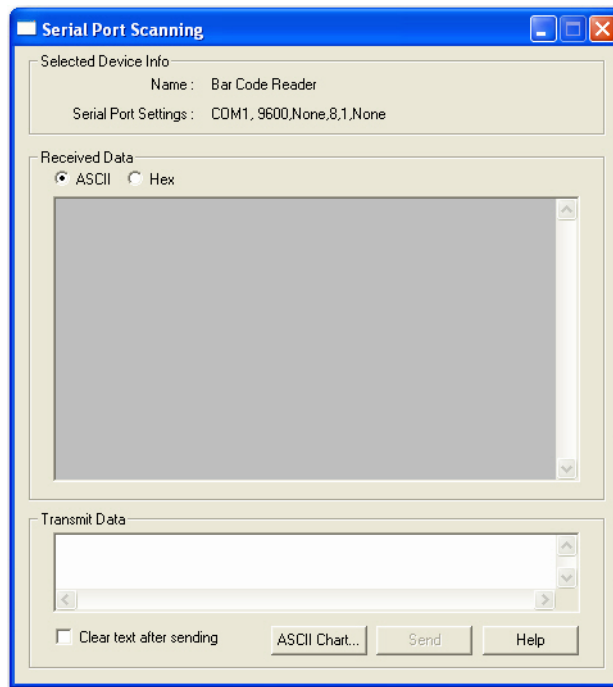


Figure 8-8 The "Serial Port Scanning" dialog box for manual communication

Type a command (or status request) to the external device in the "Transmit Data" field. You may have to refer to the documentation for the external device at this point. Note that for many devices, carriage return and line feed [CRLF] characters are required to complete a command. Press <Enter> on the PC keyboard and Think & Do supplies the [CRLF]. If only a [CR] is required, enter 0x0D (the ASCII value for [CR]).

Characters received from the external device appear in the “Received Data” field in the dialog box.

While using the “Serial Port Scanning” dialog box, you may need to refer to an ASCII chart. Click the “ASCII Chart...” button to open the chart. Both decimal and hex values are listed along with all 255 characters (PC ASCII symbols). If a character is selected from the ASCII chart, it is appended to the send string.

Notice that the “Serial Port Scanning” dialog box displays the “Receive Data” field in either ASCII characters or as hex values. The proper setting depends on the format and content of message from the external device. Sometimes just trying both settings will reveal precisely the information being sought.



Important! Be sure to check that the read or write operation is done and that the system is checked for timeout and operation errors. Finally, you will need flow chart logic to handle error conditions.

For information on using the Communications Block see “Communication Block” on page 5-9. After a project has a serial port configured in IOView and an external device is attached, flow charts can be written and screens created that communicate with an external device using the tagname assigned to the Comm data item.

8.2 Using OPC Server

The Object Linking and Embedding (OLE) for Process Control (OPC®) Server is similar to the DDE server, but uses newer technology and is more suited for process control.

Think & Do includes two OPC Servers:

- TNDOPC.EXE — Controls Think & Do projects running on a Certified PC.

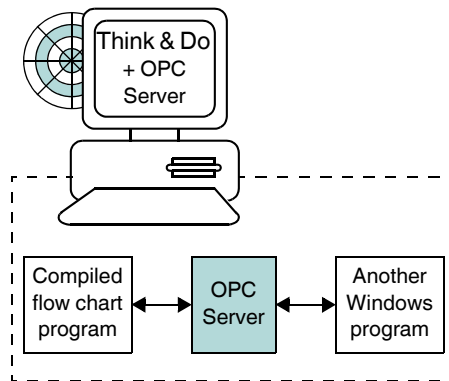


Figure 8-9 The OPC Server controls Think & Do projects on a Certified PC

- TNDCEOPC.EXE — Controls Think & Do projects running under Windows CE.

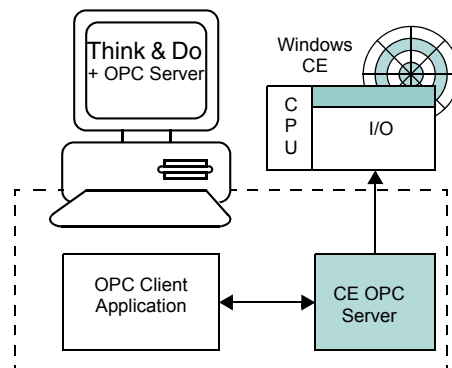


Figure 8-10 The OPC CE Server controls Think & Do projects on Windows CE

OPC is a standard for communications between software applications. The focus is on control applications, field devices, and office automation systems. See www.opcfoundation.org for more information.

8.2.1 OPC Client Applications

For most applications, OPC is a faster and more robust communications link than DDE. The OPC Server can handle requests from multiple clients to one Think & Do project. Many process control applications (for example, Wonderware®, Intellution®, GE Cimplicity®, and Honeywell PlantScape®) are already OPC-compliant.

For an application to work with an OPC Server, it must specifically include OPC client functionality. To use such an application with Think & Do, you will need to map local Think & Do tagnames to tags in the OPC client. See the documentation that shipped with the application for details on mapping. Most applications require that you create a property sheet for each tagname whose value is to be retrieved by the server.

Some terms used in OPC server conventions are:

- ProgID — OPC server program ID (ProgID):
ThinkNDo.OPCDA.1 (for a Certified PC Runtime),
ThinkNDoCE.OPCDA.1 (for CE Runtime)
- DataType.tagname — This is the item naming convention used by the Think & Do OPC Server. For example, a counter with the tagname PartCount is designated in the OPC client as “Counter.PartCount.”
- <CEstationname>.DataType.tagname — Windows CE systems use the tag mapping format of <CE station name>.DataType.Tagname. For example, a WinPLC might be accessed by WinPLC1.Output.Relay1. This format easily allows an OPC client running on a Certified PC to access the Think & Do Windows CE OPC Server (ThinkNDoCE.OPCDA). That server can then get tagname values from many Windows CE units (for example, ten WinPLCs on a network).

OPC tagnames permit alphanumeric characters as do Think & Do tagnames. Some applications are case-sensitive or have case requirements for tagnames. Check the documentation for the client application you are using.



OPC clients may need to access data from a Think & Do project on the local system (same PC as the client) or on a remote Runtime system (such as Windows CE Runtime systems or another remote PC). If your Runtime system has not been configured for remote communications, you may need to set up remote communications before using the OPC Server (see “Setting Up Remote Communications” on page 8-23).

8.2.2 OPC Explorer

The Canadian company Matrikon has created the software application named OPC Explorer. This tool provides a good introduction to using OPC for accessing data in other Windows-based applications such as Think & Do. It allows selection of either Think & Do OPC or Think & Do CE OPC Server. Then tagnames can be mapped to tags and data written. OPC Explorer is available for download on the Matrikon website at www.matrikon.com. Refer to Matrikon for any application support needed on the OPC Explorer.

When an OPC client connects to the Think & Do server, it opens the “Think & Do OPC Server” dialog box (Figure 8-11). The logo is animated and the log display lists messages pertaining to the OPC connection.

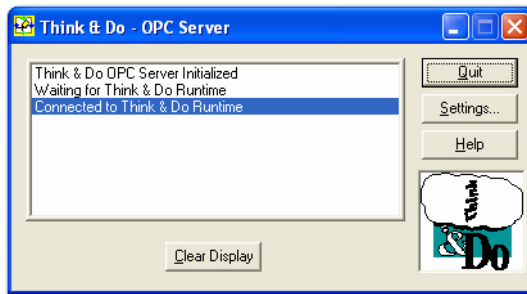


Figure 8-11 The “Think & Do OPC Server” dialog box



Think & Do software also features COM/DCOM for interfacing to other applications and network communications. More information on COM/DCOM is in an application note on the software CD, and on the Web at www.phoenixcontact.com.

8.2.3 Configuring OPC Server

There are a few configurable options available for OPC Server. To access them, click the “Settings...” button in the “Think & Do - OPC Server” dialog box. This displays the dialog box shown in Figure 8-12.

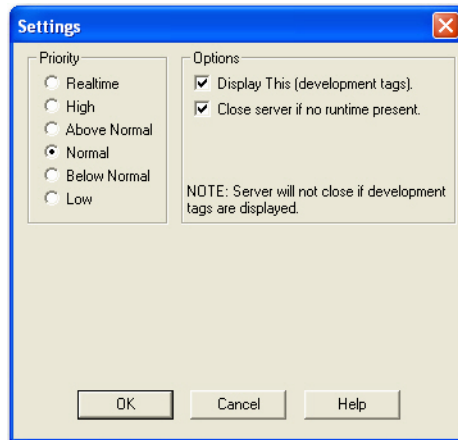


Figure 8-12 The OPC Server “Settings” dialog box

The available options are:

- “Priority” — Select a priority for OPC Server operation relative to other Windows tasks. The default setting is “Normal”.
- “Display Development tags” — Selecting this check box (the default) creates a structure of tags (under a “This” folder) that are available during design and Runtime. This is a valuable technique that lets you access tags during development and have them available at Runtime. “This” refers to “this development project” if the Runtime isn’t executing, or “this Runtime project” if the Runtime is executing. When the project is not executing, OPC Server returns a “Bad” status to any connected OPC client that requests a data item in the “This” structure. With this check box cleared, tags are only available to the OPC Server when the Runtime is executing.
- “Close Server if no Runtime present” — Selecting this check box causes the OPC Server to close if Think & Do Runtime isn’t executing on this PC.

The Windows CE OPC Server has a somewhat different layout and capability. The Windows CE version of the OPC Server has a “Settings” dialog box (Figure 8-13) that has the priority settings plus a list of available CE Runtimes.

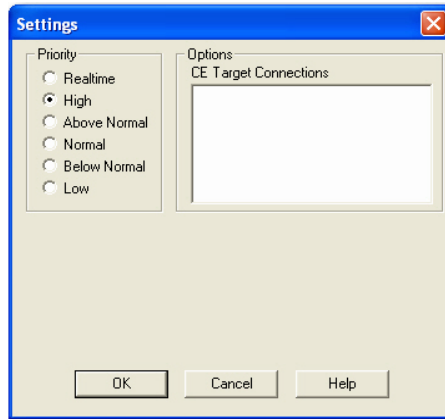


Figure 8-13 The Windows CE OPC Server “Settings” dialog box

- “Options” — This list shows available CE Runtimes. By default, it has all available CE Runtimes selected and available for OPC-related communication. You can choose to select only those you want to avoid unnecessary connections. Press the <Ctrl> key and click on entries in the list to toggle their selection.

8.2.4 Configuring a Project as an OPC Client

A Think & Do project can easily function as an OPC client. As an OPC client, the project can initiate read and write operations for data in another Windows-based application. That application may be on the PC where the Think & Do project is executing or it can be on another computer on the network. Note that the OPC server in this case is not the Think & Do OPC or Think & Do CE OPC Servers, which are part of Think & Do. In this case, the vendor of the other Windows-based program being accessed supplies the OPC server. Applications communicating with the Think & Do OPC Client must include OPC server functionality. Refer to vendor documentation to configure other OPC servers.



In order to configure an OPC server or client, you must have the same user account on both PCs.

The OPC Client feature of Think & Do uses the Tag Link driver in IOView as the communications mechanism.

To add OPC communications to a project:

1. Start Think & Do and load the desired project for OPC Client capability. This will also load its existing I/O configuration.
2. Launch IOView.
3. In IOView, select the “Drivers... Add” menu to get the list of I/O drivers.
4. Choose the “Tag Link Driver” from the drop-down list.

The Tag Link driver icon appears in the I/O configuration. This device can link tagnames in the Think & Do project with tagnames in other applications. However, you need to connect a device to the Tag Link driver to fully configure the communications mechanism.

1. Highlight the Tag Link Driver in the left pane (the right pane is initially clear of devices).
2. Select “Devices... Add” from the main menu.
3. In the “Add Tag Link” dialog box, choose the “T&D OPC Client Tag Link Module” as shown.

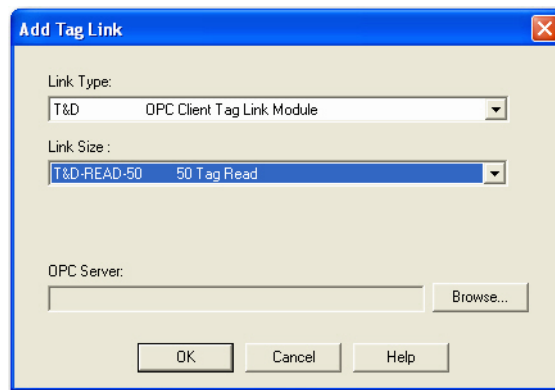


Figure 8-14 The “Add Tag Link” dialog box

The types of available tagname communications transactions include:

- T&D-READ-50 — 50 Tag Read
- T&D-READ-100 — 100 Tag Read
- T&D-WRITE-50 — 50 Tag Write
- T&D-WRITE-100 — 100 Tag Write
- T&D-READ/WRITE-50 — 50 Tag Read/Write
- T&D-READ/WRITE-100 — 100 Tag Read/Write

Thus, the “packet size” can be optimized for the amount of data typically read/written, trading off the number of tags versus system loading.

Now select the name of the OPC server that provides the connection to the other application’s data.

To select the OPC server for the Tag Link:

1. Click the “Browse” button in the “Add Tag Link” dialog box. The icons for the PC and the Network Neighborhood (if applicable) appears as shown in Figure 8-15.

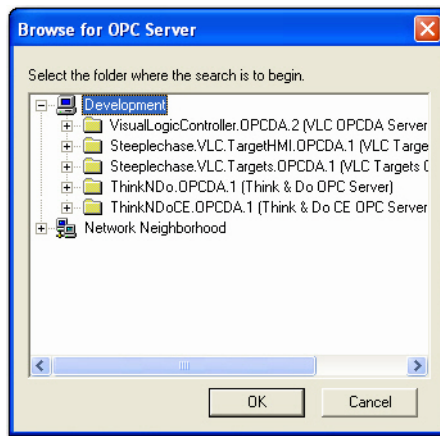


Figure 8-15 The “Browse for OPC Server” dialog box

2. Click the “+” to expand the listing for the PC (“Development” in the example) and the “Browse” function automatically locates any OPC servers on the PC.
3. Click the name of the OPC server of the other application to which the Tag Link driver will communicate. Note that Think & Do OPC Servers will be listed, but they do not apply for this OPC Client configuration.



The “Browse for OPC Server” dialog box searches other computers on the network. The is search faster if OPC Data Access Components 2.0 (or later) is on the other computer.

4. Click the “OK” button in the “Browse for OPC Server” dialog box. The name of the server selected appears in the “Add Tag Link” dialog box.
5. In the “Add Tag Link” dialog box, click the “OK” button.

To perform both Read and Write functions for a Think & Do project as an OPC client, you need to add read/write OPC devices to the Tag Link driver. Additional settings, such as “Update Rate” and error status tags, are available in the “Module Info” tab and other tabs. The “Tag Mapping” tab maps a list of 50 (or 100) data items to Remote tagnames. Double-click a “Remote Tagname” field to browse for tags on the remote system. When starting a project, note that both the other application, such as HMI or SCADA, and its OPC server must be running in order for the project to read/write data item values.

8.3 Using DDE Server

A Think & Do Runtime project often needs to exchange data with other Windows-based applications. Think & Do includes a DDE server that provides access to local/remote Certified PC systems or Windows CE Runtime systems.



DDE server is no longer the preferred solution for this requirement. It is included for backward compatibility. For new projects, we recommend that you use OPC server (see “Using OPC Server” on page 8-10).

In the example shown in Figure 8-16, the DDE server transfers data from the flow chart to another Windows-based program, which then formats the data and prints it. This is a typical Think & Do DDE Server application.

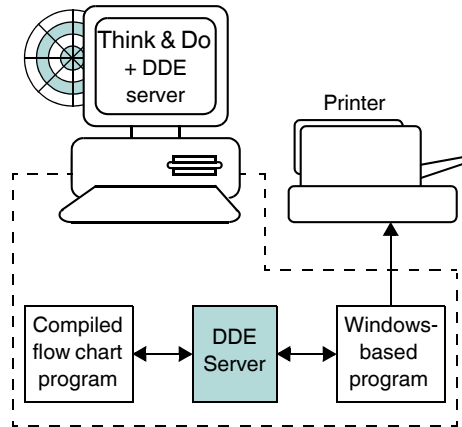


Figure 8-16 Use DDE to communicate with other Windows-based programs

The DDE server can access remote Windows CE Runtime systems. The following example shows a local system with Microsoft Excel accessing a remote Windows CE system.

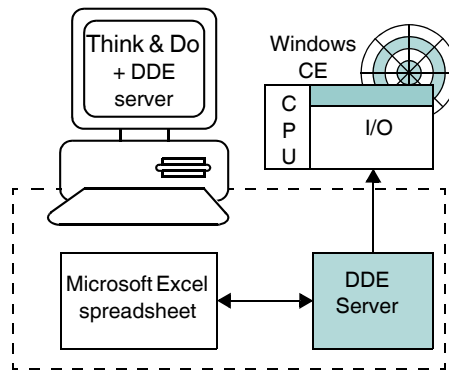


Figure 8-17 Use Excel and DDE to access a remote Windows CE system.



The DDE executable file is located in the following path:
 C:\Program Files\Phoenix Contact\ThinkNDo\BIN\TND2DDE.exe

8.3.1 Introduction to DDE Servers

The acronym DDE stands for Dynamic Data Exchange. The basis of DDE is a communications protocol that allows one Windows-based application to send data and commands to another. The DDE protocol is part of Microsoft's overall Windows environment design.



This manual uses the word "application" in this DDE server discussion to describe a Windows-based program and not the Think & Do project.

The DDE server is simply a program to let other applications read and write data from/to the database in a running Think & Do project. Think & Do software includes the "TND2DDE.exe", which provides this DDE server function. The "client" is the program requesting data and the "server" is the program providing it. However, both the client and server applications have to be DDE-compliant in order for the communications link to work.

8.3.2 Getting DDE Communications Started

To conduct communications between applications, the Client application opens a channel to the DDE server by establishing the following:

- Server application name (such as "TND2DDE").
- Topic name of interest (such as "Production Spreadsheet") — Think & Do defaults to "data" for local systems. For remote Windows CE Runtime systems, the name of the station must be entered.
- After the channel is open, the client can request data items in the server's specified topic name of interest.
- Item name (the specific cell in the spreadsheet), for example tagname: "ProductionTotal"

When the client application sets up the link to the server, it requests notification whenever data changes. This arrangement reduces the amount of communication necessary for most situations. The link can remain open, even if the client is not requesting data, until the client or server terminates the link.

8.3.3 DDE Addresses

The standard DDE server protocol identifies a specific data item with a three-part address specification based on the location of the running Think & Do project (see Table 8-1).

- When the Runtime target is the local PC, use "data" for the topic name.
- When the client needs to connect to a remote Windows CE target, use the network name of the target Windows CE Runtime station for the topic name.

Table 8-1 DDE Server Address Structure

Runtime Target	Application	Topic	Item
Local PC	TND2DDE	data	tagname
Windows CE Device	TND2DDE	<nameofstation>	tagname

The client program uses these three pieces of information when requesting data from the server application. Each of the three are progressively specific in the addressing convention. Some examples of DDE addressing follow:

Table 8-2 Client and server applications on a local Certified PC

Client	Server	Topic	Item (tagname)
Excel	TND2DDE	data	ProductionTotal

Formulas, such as the Excel type, create an address with field separators:
 =TND2DDE!Data!ProductionTotal

Table 8-3 Client and server applications for a remote Windows CE system

Client	Server	Topic	Item
Excel	TND2DDE	<nameofstation>	ProductionTotal

Formulas, such as the Excel type, create an address with field separators. In this example, the client is accessing a remote WinPLC named "WinPLC1."
 =TND2DDE!WinPLC1!ProductionTotal

8.3.4 Launching the DDE Server

You can start the Think & Do DDE server by selecting "Start... Programs... Phoenix Contact... Think & Do... DDE Server".

The main dialog box (Figure 8-18) opens when starting DDE Server. It lets you:

- Configure the server
- View the status of Think & Do data items
- Clear the message log
- Quit (exit) the program.



Figure 8-18 The "DDE Server" dialog box

The logo is animated when DDE requests from a client application are occurring. If the logo animation stops, the requests have stopped. However, the link may still be open.

8.3.5 Configuring DDE Server

Click the “Configure” button to configure DDE Server. The “Configure” dialog box (Figure 8-19) opens.

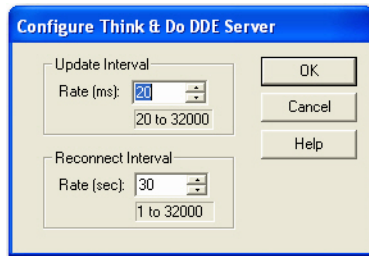


Figure 8-19 The “DDE Server Configure” dialog box

You can set the following:

Table 8-4 DDE Server Configuration Options

Control	Description	Units	Range
Update Interval	The rate at which the server sends changing data to clients	Milliseconds	20-32,000
Reconnect Interval	The amount of time the server will wait before trying to reconnect to the Runtime system	Seconds	1-32,000

8.3.6 Reading and Writing Tagname Values (Peek and Poke)

You can use a “Think & Do DDE Server” dialog box to read (“peek” at) a data item value in a project. This feature is especially useful during project development, since it does not require configuring a client application to make DDE requests.

To do a DDE server Peek:

1. From the “Think & Do DDE Server” dialog box, click “Peek...” The “Peek” dialog box (Figure 8-20) opens. If the Think & Do project is not running, DDE Server launches the Runtime window, from which a pre-built project must be opened.
2. Type the tagname to check in the field provided. Match the syntax of the actual tagname exactly.
3. Click the “Read Now” button. If the Think & Do Runtime project is active and the tagname is valid, its value is displayed. Note that the Peek display is a single snapshot of the value. Click the “Read Now” button to get fresh data.

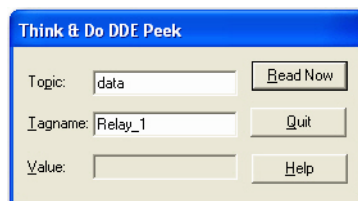


Figure 8-20 The “DDE Server Peek” dialog box

The Think & Do Peek tool is handy for resolving problems if a client application is unable to read a tagname value. Also, check the message log window as well.



Some client applications are not able to pass through tagnames that include spaces in the middle of them. Use the underline character () as an alternative character. Be sure to establish tagnames appropriately if planning on allowing a client application to read the value through DDE Server. The client may have a tagname syntax requirement. For example, Excel requires single quotes around tags with spaces.

8.3.7 Reading and Writing Tagname Values with External Programs

Use an Excel spreadsheet or another Windows-based DDE-compliant application (for example, Visual Basic) to read and write tagname values in a running Think & Do project. Please consult the documentation for those applications to learn the precise commands and syntax required to open a DDE link or channel.

Reading From Excel

To read a tagname value in Excel, type a formula with following syntax into one of the cells in your spreadsheet:

```
=<server>!<topic>!<tagname>
```

where:

<server> = The name of the Think & Do server ("TND2DDE")

<topic> depends on the operating system on which the Think & Do project is running. For a Certified PC, it is "data." For Windows CE, use the station name (for example, "WinPLC1").

<tagname> = The name of a Think & Do data tag

Do not enter any spaces in the formula.



The formula has a pipe character (|) after the server name and an exclamation point (!) before the tagname. Don't confuse them.

For example, to read the value of the tagnamed "ProductionTotal" on a Certified PC, enter the following into the cell:

```
=TND2DDE|data!ProductionTotal
```

To read the same value from a project running under a remote Windows CE Runtime, enter:

```
=TND2DDE|WinPLC1!ProductionTotal
```



Refer to the Microsoft Excel manual for details on entering remote reference formulas for cells. If using another client application, please refer to its documentation on the topics "DDE links" or "Remote Data Collection."

After entering the formula in the Excel spreadsheet cell, press <Enter> or click out of the cell. Excel begins generating client requests for information. If the Think & Do Runtime project is running and the TND2DDE program (DDE Server) is also active, the cell displays the tag-name value (see Figure 8-21). You can make other cells display values from different tag-names.

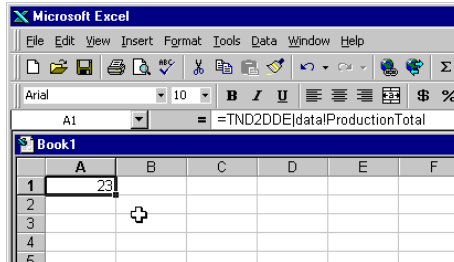


Figure 8-21 Excel can display tag values from a running Think & Do project

If saving the spreadsheet, the next time it is opened, it displays a dialog box like the one shown below, asking to update the links in the workbook.

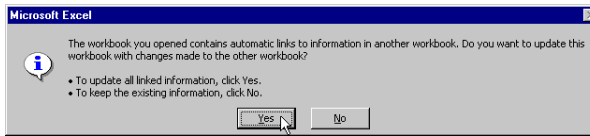


Figure 8-22 Excel may prompt to update links to Think & Do tags

If DDE server is running, click the “Yes” button to update the links and resume communication. The cells with tagname formulas display the current values.

If DDE server is not running, Excel will prompt to start the server. Click the “Yes” button to start DDE server. Data communication should begin.

Writing Tagname Values from Excel

Tagname values can be set from within an Excel spreadsheet. This is a little more advanced. You need to write a Visual Basic routine that calls the DDE “Poke” application, which writes a value to another Windows-based application (the Think & Do project). You’ll also need some way to execute the routine (for example, by clicking a button created on the spreadsheet).

For example, to place an “ActiveX Command” button on the first cell of the spreadsheet, name it “CommandButton1” and create the following subroutine:

```
Private Sub CommandButton1_Click()  
Set PokeTagname = Worksheets("Sheet1").Range("a2")  
Set PokeValue = Worksheets("Sheet1").Range("a3")  
DDE_Channel = Application.DDEInitiate("tnd2dde", "data")  
Application.DDEPoke DDE_Channel, PokeTagname, PokeValue  
Application.DDETerminate DDE_Channel  
End Sub
```


Also, put the name of the desired tag to set in cell A2 and the value to set it to in cell A3. When you click the button, the subroutine executes and sets the tagname to the value.



See Excel documentation for more information.

8.4 Setting Up Remote Communications

You can set up communications between two desktop PCs to remotely run or debug a project (see Figure 8-23). The local PC (the development system) is the one running the development tools. The remote PC (the Runtime system) is the one running the project. You can send changes from the development system to the Runtime system and use DDE or OPC functions.



Do not use this procedure to connect to a Windows CE Runtime system from a local PC.

The remote link uses Windows XP/Vista/7 DCOM capability, which lets software components communicate transparently over supported networks.

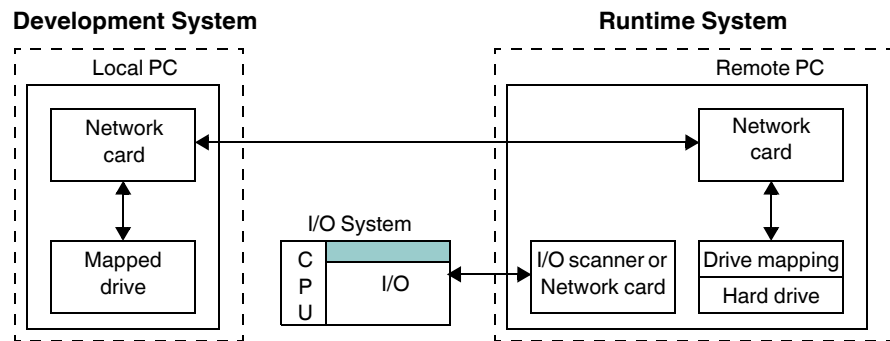


Figure 8-23 Remote communication between desktop PCs



DCOM is part of the Windows XP/Vista/7 operating systems. It is not included in Think & Do.

Verify that the development and Runtime systems meet the following requirements:

- Windows DCOM capability is enabled on both systems.
- Think & Do development tools are installed on the local system. Think & Do Runtime is installed on the remote system.
- Project files are available to both systems. Use a mapped drive with the same logical drive letter at each PC. For best performance, put the project on the drive that is physically attached to the Runtime system.
- Log onto both systems as a user with administrative privileges (but you don't need to be logged on as the "Administrator" user).

8.4.1 Runtime System Setup

On the Runtime system, set up a new user account:

- Enter the username and password (if there is one) of the user on the development system
- Complete the "Full Name" and "Description" fields
- Do not require the user to change the password at the next logon
- Add the user to the "Administrators" group

On the remote PC, enable remote connection:

- Click the "Start... Programs... Phoenix Contact... Think & Do... System... Remote Connection - Enable" icon.

Identify the computer name of the Runtime system. On the remote PC, depending on the version of Windows, do one of the following:

- Windows XP:
 1. Select the "Start... Control Panel... System" menu.



If the Control Panel is set to display Category View, click "Performance and Maintenance" and then "System."

2. Select the "Computer Name" tab and note the "Full computer name" assigned to the Runtime PC (e.g. RUNTIME).
- Windows 7/Vista:
 1. Click the "Start" button, right-click the "Computer" icon and click "Properties" from the pop-up window.
 2. From the "Computer name, domain and workgroup settings" group, note the "Computer Name" assigned to the Runtime PC (e.g. RUNTIME).

Record the computer name for later reference.

Share the drive containing the Think & Do files and the project:

1. Open Windows Explorer and highlight the disk drive or subdirectory that contains the project files.
2. Select "File... Properties" or right-click and select "Properties" from the shortcut menu.
3. Select the "Sharing" tab and click "New Share".
4. In "Share Name" enter the desired name to use. Something as simple as "H" (for drive H:) is acceptable. Click the "OK" button to close the "New Share" dialog box and "OK" again to close the "Properties" dialog box.

Map the shared drive on the Runtime system. On the remote PC, do the following:

1. In Windows Explorer, select "Tools... Map Network Drive."
2. Select a drive letter that is unused on both the Runtime and development systems.
3. Click the "Browse" button to locate the drive shared in the previous step. You may need to expand the "My Network Places... Entire Network... Microsoft Windows Network... <network name>", and station items in the directory tree view by double-clicking them.
4. Double-click the shared drive icon to complete the mapping.



The Runtime and development systems must refer to the drive containing the project using the same logical drive letter. Even though the project is on the local drive for the local Runtime system, it must be mapped to a separate drive letter to work remotely with the remote system.

8.4.2 Development System Setup

On the local PC, enable remote connection, click the “Start... Programs... Phoenix Contact... Think & Do... System... Remote Connection - Enable” menu.

Map the shared drive. On the local PC, do the following:

1. Open the “Map Network Drive” dialog box (in Windows Explorer, click “Tools...Map Network Drive”).
2. Select the same drive letter that was used when mapping the drive on the Runtime system.
3. Click the “Browse” button to locate the drive shared in the previous step. You may need to expand the “My Network Places... Entire Network... Microsoft Windows Network... <network name>”, and station items in the directory tree view by double-clicking them.
4. Double-click the shared drive to complete the mapping.

8.4.3 Synchronize PC Time-of-Day Clocks

Set the time-of-day clocks on the local and remote PCs to the same day of the week, AM/PM setting, and time zone. When AppTracker (Think & Do’s debugging tool) runs remotely, it examines the time/date stamps on Runtime files to see if the project is synchronized to the local files. If the time or date is different, it will interpret the project to be out-of-sync and will not track the project.

8.4.4 Connect to the Remote System

The remote and local systems are now ready to communicate. Use Network Neighborhood in Windows Explorer to view the remote connections. The Runtime system PC should be able to view the name assigned to the development system PC.

To connect to the remote system:

1. From the development PC, open a project on the shared drive.
2. On the ProjectCenter toolbar, enter the station name of the Runtime PC (for example, “RUNTIME”).

Build and run the project as you normally would. When running the project from ProjectCenter, the project actually runs on the remote Runtime system, using I/O devices that are connected to the remote system. You can connect to a project that is already running on the remote system.

If connection and communications are very slow the first time you connect to the Runtime system, just disconnect and reconnect to it. Performance will be normal.



All normal online editing and debugging functions work over remote connections. You can also run IOView, and make offline changes to the I/O configuration. Because IOView is actually run on the development system, you are not able to connect to the I/O in real time and create or scan a new configuration.

Section 9

This section informs you about:

- Building projects
- Running projects
- Executing on remote systems and on Windows CE targets

Building and Running Projects	9-3
9.1 Building a Project.....	9-3
9.2 Running a Project	9-4
9.3 Adjusting the Scan Interval	9-5
9.4 Stopping a Running Project	9-6
9.5 Running Remote Projects.....	9-6
9.5.1 Runtime Settings	9-7
9.5.2 Windows CE Runtime Settings	9-8
9.6 Think & Do CE Watch for Windows CE Targets	9-9
9.6.1 Security Menu (Lock).....	9-12
9.7 Instant Recall for Windows CE Target	9-12

9 Building and Running Projects

This chapter shows how to prepare a project to build and then run it. Note that this chapter discusses running a project, not just a flow chart. Running a Think & Do project means running all of its flow charts, HMI screens, and other (optional) associated programs.

For more information on HMI screens, see Section 6, “Operator Screen Techniques”. It covers data entry and many other topics on HMI screen creation.

9.1 Building a Project

Before running a project you must build an executable version of it. That’s also called compiling a project. To compile a new project:

1. Open the project in ProjectCenter. If the project is already open, save it before continuing.
2. Select the “Project... Build Project” menu or click the button on the toolbar. Build compiles only changed components (to save time in large projects). Alternately, select the “Project... Rebuild All” menu. Rebuild All recompiles the entire project.

The “Think & Do Project Build” dialog box appears. It indicates progress and any errors or warnings that might occur during the build. It also lists the number of flow charts and screens that were compiled during the build.

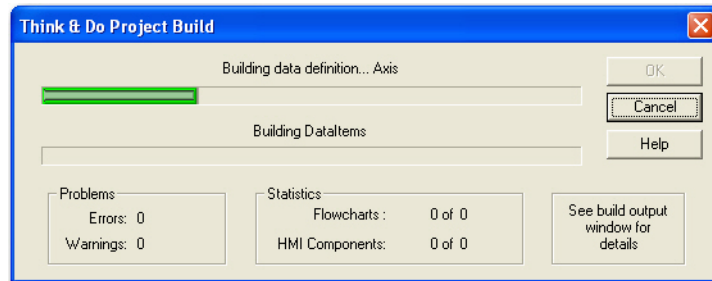
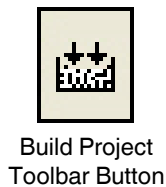


Figure 9-1 The “Think & Do Project Build” dialog box

3. When the build is complete, click the “OK” button in the dialog box.

- The Message pane in the lower part of the ProjectCenter window lists errors and warnings. If there are any, double-click each one to open the appropriate flow chart to return to the block or screen object in error. Fix all listed problems before proceeding.

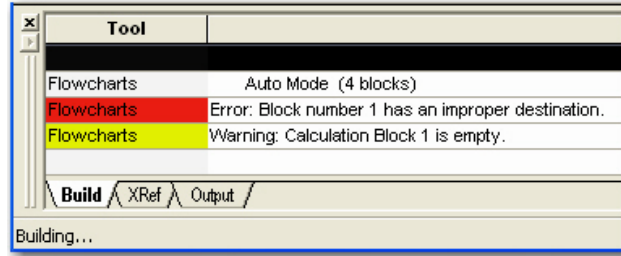


Figure 9-2 The “Output Window... Build” tab displays build messages



Run Project
Toolbar Button

9.2 Running a Project

After a successful project build and compilation, select the “Project... Run” menu or click the “Run Project” icon on the toolbar.

The Runtime window (Figure 9-3) opens. It has information about the running project, including the build date, I/O status, and scan time.

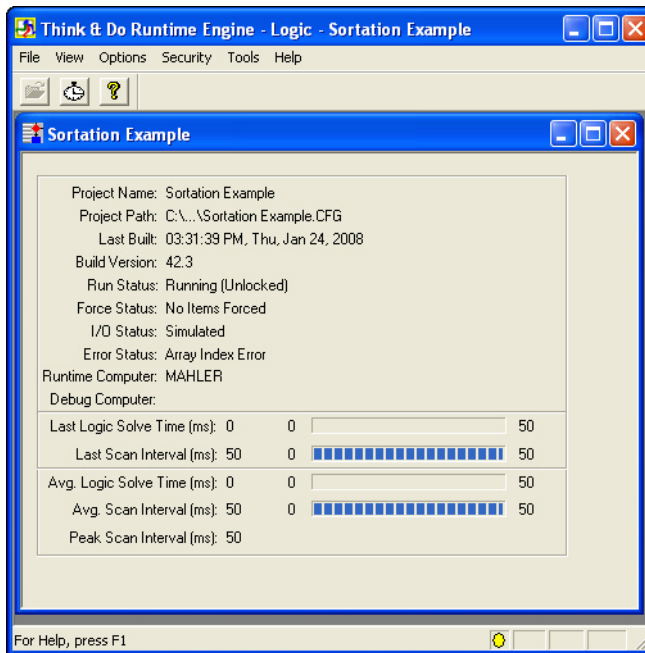


Figure 9-3 The “Think & Do Runtime Engine” window



If the Runtime window is not visible, it’s probably hidden by other windows. Click the “Think & Do Runtime Engine” button on the Windows task bar to see the window.

9.3 Adjusting the Scan Interval

The bar graphs in the Runtime window display scan performance statistics. The graph below shows the relationship between parts of the scan. You can set the scan interval, but the logic solve time depends on project size and the system processing speed.

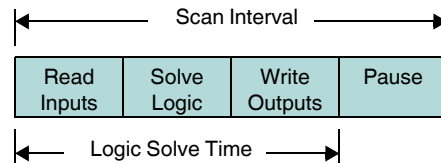


Figure 9-4 Scan Interval is the sum of Logic Solve Time and Pause time

Since Think & Do holds the scan interval at a specified rate, the Pause time is the time left for the system to handle communications and other program execution. The Runtime window's bar graph displays average and peak logic solve times. While the project runs, scan time tasks occur continuously at the scan rate. The default scan time is 50 ms. The duration of the Read Inputs and Write Outputs tasks depends on the I/O sub-system size and speed.



It is the responsibility of the programmer/application engineer to specify a scan time that allows enough Pause time for Windows to service other programs. The Think & Do Runtime enforces a Pause time equal to 20% or more of the scan interval.

The logic solve time for a very small project will register 1 to 2 ms. With a default scan interval of 50 ms, the system has about 48 ms of pause time.

To set the scan interval:

1. From the Runtime window's main menu, click "Options... Set Scan Interval" menu (or click the "Scan Interval" toolbar button) to display the "Set Scan Interval" dialog box.

To optimize performance, set the scan interval so other Windows tasks have just enough time to perform. Minimum allowable scan time settings are:

- Certified PC systems – 2 ms
- WinPLC – 3 ms



Scan Interval Button



The ProjectCenter scan time setting is stored with the project. If a Windows CE target scan time is set with Watch, it is temporary (lost after project halt).

2. Click the "OK" button to apply the scan interval. A dialog box displays a warning that scan time changes are temporary.

To set a new scan interval permanently, follow these steps:

1. Go to the ProjectCenter, and select the Project Explorer button. In the “Runtime Settings” group, select the desired “Runtime Target” and change the “Maximum Scan Interval” (see Figure 9-5). The statistics in the Runtime window should reflect the changes.

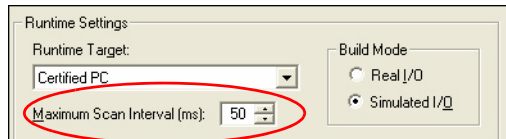


Figure 9-5 Change the scan interval permanently in ProjectCenter



If the peak scan interval ever exceeds the set scan interval, increase the set scan interval.



Clear the Runtime peak statistics after setting the Scan Interval. Click the “Options... Clear Runtime Peak Statistics” button.

9.4 Stopping a Running Project

The project will continue to run all flow charts, I/O read/writes, and HMI screens until halted. To stop the project, click “File... Close” in the Runtime window.



Any machine or process on the I/O network under Runtime control will cease operation when a project is stopped. A confirmation box prompts for confirmation before closing.

Use the Lock feature to prevent an unauthorized run/halt of a project. Click the “Security... Lock...” menu in the Runtime window and enter a password.

9.5 Running Remote Projects

The build process compiles source elements so they run the project. If the project is to run on a remote Think & Do machine, it is sent to the target as part of that process. If you have one PC for both development and Runtime activities, this is transparent. If you have a Windows CE target, there are source file transfer options that you need to understand.

Think & Do automatically downloads the compiled project to the Runtime system when running the project. By convention, a download transfers the files to the target system, and an upload transfers the project source from the target to the PC requesting the source. It is important to configure project settings in ProjectCenter early in project development.

9.5.1 Runtime Settings

To access project Runtime settings, click the “Project Explorer” bar in the left pane of the ProjectCenter window. The “Runtime Settings” group appears in the main window (Figure 9-6). For more information about Runtime screen settings, see “Creating a Project” on page 2-3.

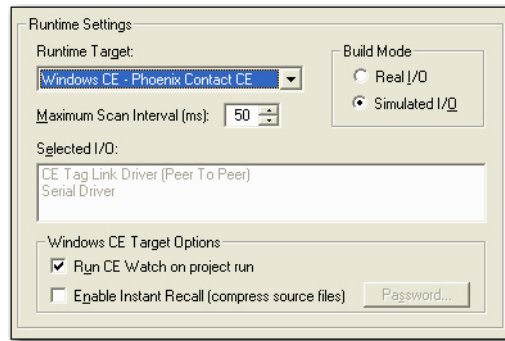


Figure 9-6 “Runtime Settings” in ProjectCenter

Runtime Target

Use this drop-down list to select the target. You can change the target type at any time, which makes it easy to switch between targets. In fact, Think & Do automatically flags items that are not compatible with the target selection and warns about the possible consequences of changing the target.

Build Mode

Select whether the project will use actual I/O data from a physical device or use flow chart logic to simulate I/O behavior when the physical I/O is unavailable or when testing. See “Simulation Flow Charts” on page 10-13.

Maximum Scan Interval

The default is 50 ms. Increase or decrease this interval as required by the project. The scan interval setting minus logic solve time equals the time for data communications and other Windows-based programs (when applicable).

Selected I/O

The I/O configured in IOView is listed in this read-only field.

Windows CE Target Options

- “Run CE Watch on project run”: Select this check box (recommended) if you want to monitor and control the Runtime activity of the Windows CE target. Watch is the name of the Runtime control utility (see “Think & Do CE Watch for Windows CE Targets” on page 9-9).

- Instant Recall and Password: Select this check box to automatically store the project source files on the Windows CE target device during a download. The Instant Recall feature is for users who may want to store a copy of the project source code separately. This permits someone to retrieve (upload) an editable copy of the project from the Windows CE target at a later time. Of course, Phoenix Contact recommends always archiving the original development system copy for safe keeping.

You can password-protect the project source files that you store in the Windows CE target flash RAM. Click the “Password...” button to do so. The password is embedded in the file, not in memory in either system or on disk. You can upload the source file back to the development PC, but the password is required to open it.



Be sure to record the chosen password in a safe place. After downloading a password-protected project source to the target, you must have the password to edit the source (Phoenix Contact application support personnel cannot bypass it).

When the project is running on the target, a RunTime window (Figure 9-7) displays on the development system. The Runtime window will differ, depending on the target type.

Certified PC Runtime Window

Windows CE Runtime Viewer (Watch)

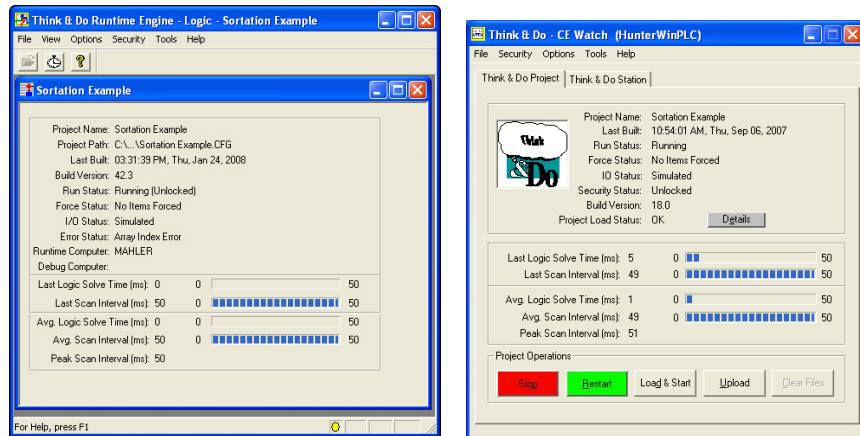


Figure 9-7 Runtime windows for Certified PC and Windows CE

9.5.2 Windows CE Runtime Settings

Windows CE targets can store two types of project files:

- Project Runtime: This file provides everything the operating system needs to run the project. This compiled code cannot be directly edited.
- Project source: Source files may be optionally stored on the Windows CE target. The files are compressed using a compressed file format. Having a copy of the source files on the Runtime target is a great convenience – no more looking for the disk or keeping track of the project version that’s running – it’s available to be recalled.

ProjectCenter's Runtime Settings group includes settings specific to Windows CE targets.

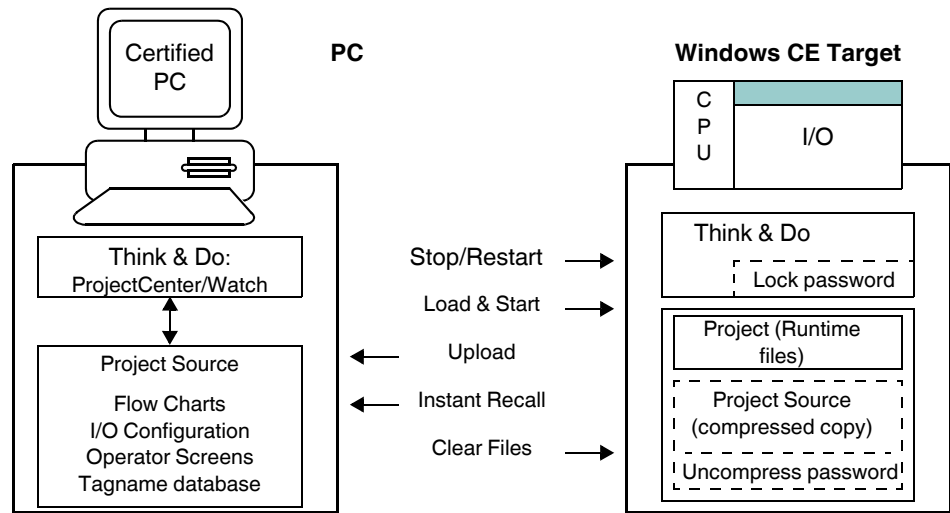


Figure 9-8 Runtime settings for Windows CE

9.6 Think & Do CE Watch for Windows CE Targets

This section describes Think & Do CE Watch in greater detail. CE Watch handles a variety of project Runtime and project source file tasks. It can manage multiple CE Runtime targets on a network. Each Runtime target on the network is a station. The "Watch" window has two tabs with project control and monitoring provisions (see Figure 9-9).

- "Think & Do Project" tab: for starting and stopping the project on the selected station and for transferring files to/from the target station.

- “Think & Do Station” tab: for viewing hardware status of the target station and selecting stations.

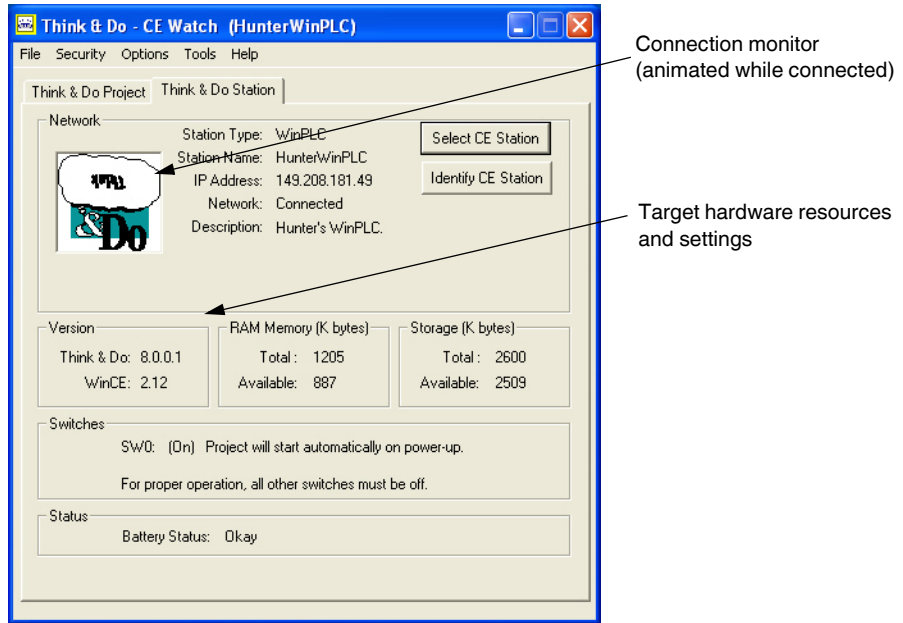


Figure 9-9 The “Think & Do - CE Watch... Think & Do Station” tab

CE Watch individually accesses each station (target) on a network. Since each target is an independent Runtime system, each can be running a different project with different scan times, or some may be stopped. CE Watch lets you control a particular CE target without disturbing the remaining targets in a system.

To configure a Think & Do CE Station:

1. Click the “Think & Do Station” tab.
2. Click the “Select CE Station” button. The “CE Watch - CE Station Selection” dialog box appears (Figure 9-10), listing all the Windows CE targets on the network.

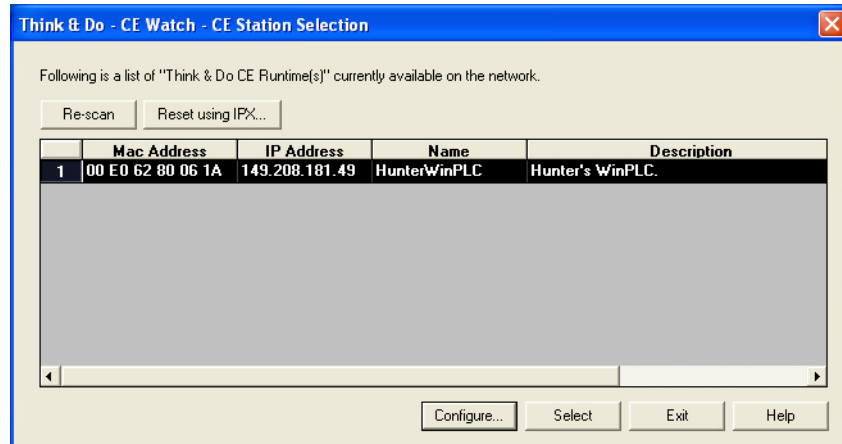


Figure 9-10 The “Think & Do - CE Watch - CE Station Selection” dialog box

3. Click the station to work with (this highlights the station name).
4. Click “Select” to make it the Watch target. Instructions on how to establish a connection to particular Windows CE hardware are in the appropriate Getting Started booklet.



If the animated Think & Do logo (connection monitor) freezes, it means that communications with the selected station has been interrupted.

After selecting a particular target station, click the “CE Watch Think & Do Project” tab to control and monitor Runtime activity on the target (see Figure 9-7 on page 9-8).

Load & Start

This command downloads the Runtime files from the development system to the selected station, overwriting the current file in the target’s memory. If you have enabled Instant Recall in the ProjectCenter’s Project explorer window settings, clicking this button also downloads the editable source file in compressed format, replacing any previous equivalent file in target memory. After downloading these files, Watch directs the target device to begin running the Runtime files, and the Watch project monitor logo becomes animated. Some notes on how Load and Start works:

- ProjectCenter opens the source files using the <projectname>.prj file name. The Runtime files are referenced using the <projectname>.cfg file name.
- The user does not have to load the project source files – only the Runtime files. This is equivalent to the “File... Open” menu on the Certified PC Runtime.
- If there is a file called <projectname>.zip in the same directory that Load and Start is using, it will download the compressed files to the WinPLC.

Restart

Clicking this button starts the project in its initial state. All flow charts begin at block 0 and initial values are loaded for data items. No files are transferred on a Restart. Watch displays a confirmation warning message before performing the restart.

Stop

Clicking this button stops the running project on the selected target station. I/O goes to the ShutDown pattern defined in IOView. For the WinPLC it's either zero, last state, or a user-defined pattern. Watch displays a confirmation warning message first. Project monitor logo animation stops.

Upload

Clicking this button transfers all files from the target station to the development system. Watch displays a "Save As..." dialog box to specify the destination directory. CE Watch transfers Runtime files and the compressed, editable source file (if it resides in target memory). However, the editable source file is not decompressed and made accessible for editing by this method. In ProjectCenter, click the "File... Launch Instant Recall..." menu to do that. If uploading the source file, you can decompress the file using a standard file compression utility.

The editable source may be password-protected. However, this does not prevent upload to the development PC. Upload does not attempt to expand (uncompress) the file, nor does it prompt for the password. In fact, you can transfer both the executable and editable source files to another target, without ever knowing the password of the protected file. To expand the editable source file, use Instant Recall in ProjectCenter, which prompts for the password if necessary.

Clear Files

This erases the executable source file in target memory. It also erases the compressed, editable source file if it is in target memory. Watch requests confirmation before erasing the file(s).

9.6.1 Security Menu (Lock)

You can prevent Watch from changing the target station's current running/stopped state. Click the "Security... Lock" menu and enter a password. Make sure to write down the password somewhere safe to unlock Watch.

The password becomes resident in the target memory and the "Stop," "Restart" and "Load & Start" buttons in Watch are disabled. A new Watch connection to the target will behave the same way.

To unlock the target, click the "Security... Unlock" menu, and enter the password. The locked buttons become active again.

9.7 Instant Recall for Windows CE Target

ProjectCenter provides a method to upload editable source files from a CE target, expanding (uncompressing) the files into a project directory. A particular Windows CE target may or may not have the editable source in memory, since that is an optional setting (see ProjectCenter's Project explorer window; click the "Enable Instant Recall" check box).

To use Instant Recall to access CE target files:

1. In ProjectCenter, select the "File... Launch Instant Recall" menu.
2. In the "Runtime Target" field, select the target type (such as "Windows CE – Phoenix Contact CE") using the drop-down list.

3. In the “Windows CE Station” field, use the drop-down list to select the particular target on the network (or you can type its name).
4. If the target station is not listed, select “Find CE Runtime” to search for targets on the network and to update the list (Figure 9-11).

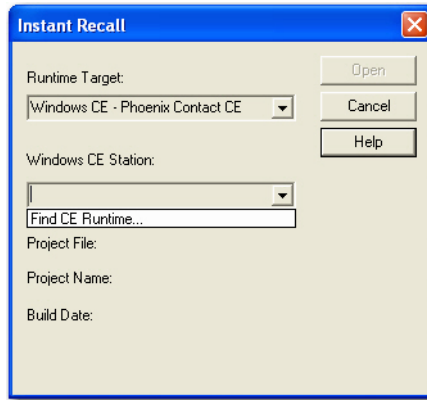


Figure 9-11 The “Instant Recall” dialog box

5. Select the Runtime station name.
6. Select the target. The “Open” button enables if the target has an editable source file in memory. The dialog box displays project information if the target is running the project. If the project is stopped, the dialog box displays the message
Station <station name> is stopped. Project information is unavailable!
However, the editable source file can still be opened if it is in target memory.
7. Click the “Open” to transfer the editable source file to the development system.
8. In the “Save As...” dialog box that appears, specify the upload directory for the project.
9. Click the “OK” button. Instant Recall expands (uncompresses) the source files, and makes them ready to edit with Think & Do tools.



Instant Recall can access a Windows CE target station’s editable source file even when the target is running. The target device transfers the file to the development system without interrupting the running project on the target.



If you have problems establishing a connection with a Windows CE target, see the Getting Started booklet or other documentation that came with the Windows CE device.

Section 10

This section informs you about:

- Using AppTracker
- Using the Watch Window
- Using FlowView in Monitor Mode
- Creating and Using Simulation Flow Charts

Debugging Projects.....	10-3
10.1 AppTracker.....	10-3
10.1.1 Launching AppTracker	10-3
10.1.2 Watch Windows.....	10-5
10.1.3 Custom Watch Windows.....	10-6
10.1.4 Using AppTracker.....	10-7
10.1.5 Using FlowView with AppTracker	10-8
10.2 Using FlowView in Monitor Mode	10-9
10.2.1 EasyTrac™	10-10
10.2.2 Flow Chart Execution Controls.....	10-11
10.3 Edit Mode	10-13
10.4 Simulation Flow Charts.....	10-13
10.4.1 Using Simulation.....	10-14
10.4.2 Disabling Simulation for Real I/O	10-15

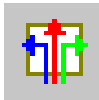
10 Debugging Projects

Think and Do provides interconnected tools that help you debug (that is, identify problems in) projects during development. This section describes how to launch and use these tools. They are:

- AppTracker
- EasyTrac®
- FlowView in Monitor Mode

10.1 AppTracker

AppTracker is a tool included with Think & Do. It is a stand-alone application that can run even if the development tools in ProjectCenter are not running.



AppTracker
Toolbar Button

10.1.1 Launching AppTracker

To start AppTracker, do the following:

1. Run the project. If the project is not running, an error message appears.
2. Click the “AppTracker” toolbar button in ProjectCenter, FlowView, etc.

The “AppTracker” window (Figure 10-1) provides a comprehensive debugging environment for project development. The example shown below is a typical view of a project.

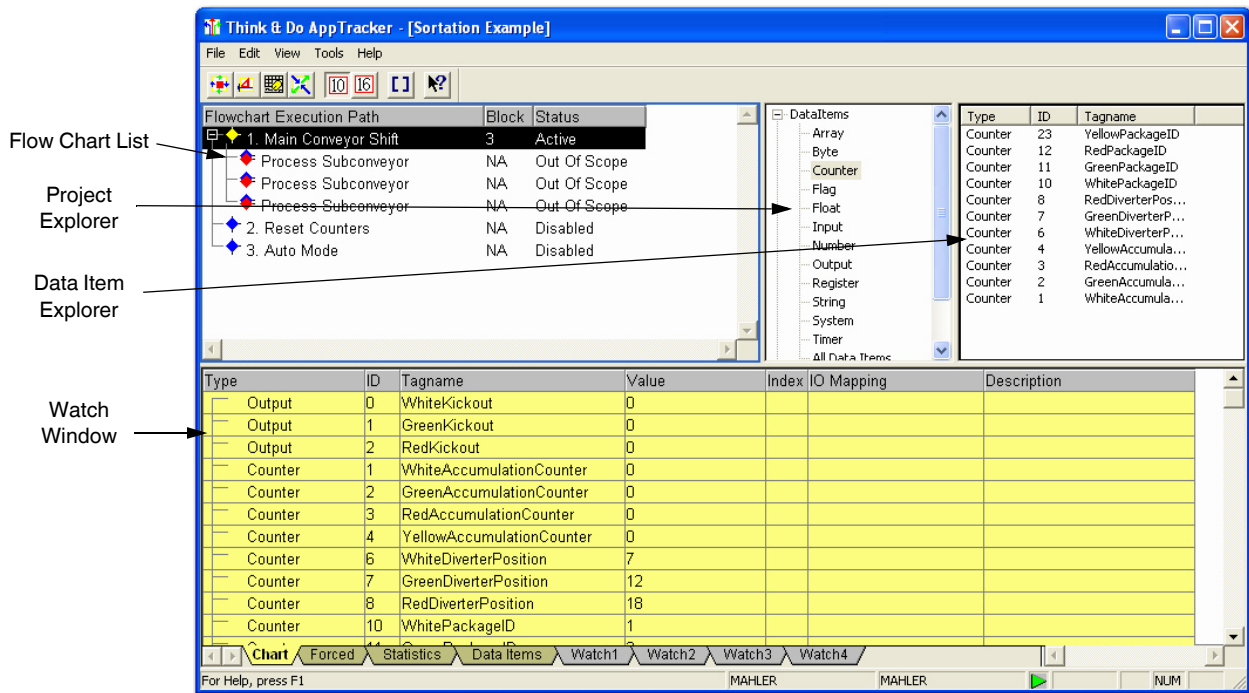


Figure 10-1 AppTracker window

The “Flow Chart Execution Path” pane (Figure 10-2) is a “treeview” diagram (like the directory list in Windows Explorer). It shows the heirarchical relationship of flow charts in the running project. Because it shows every instance of a subchart call, some subcharts appear in the list more than once.

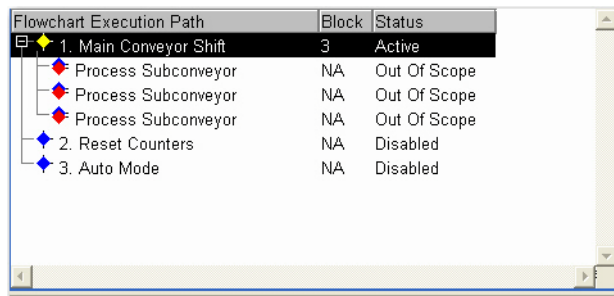


Figure 10-2 AppTracker “Flow Chart Execution Path” pane

The Status column shows the status of each flow chart in the execution path:

- Active: Flow chart is currently executing
- Enabled: Enable criteria have been met. The flow chart is ready to execute.
- Disabled: The chart is disabled and will not execute.
- Waiting: A Wait block has been encountered; flow chart is waiting as specified.
- Frozen: Freeze criteria have been met; flow chart will not execute.
- Out of scope: This only applies to subcharts. When a subchart is not active, the debugger cannot “see” inside it. The data item values remain as they were when the chart was last in scope.

The Block number column lists the number of the current block for active flow charts. “NA” is displayed for all other flow charts.

The Project Explorer pane (Figure 10-3) is also a treeview diagram. It shows the data items, flow charts, screens (if used), and I/O classes in a project.

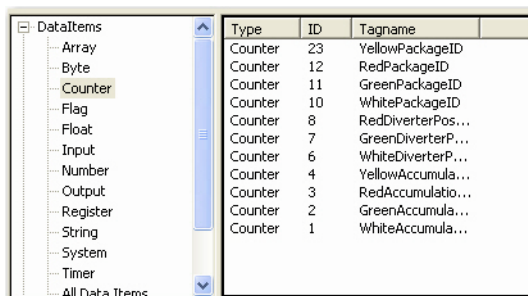


Figure 10-3 AppTracker “Project Explorer” pane

Click the “+” to the left of any project topic to open it. Data items related to the category chosen appear in the data item pane. Data item listings are not mutually exclusive among the main topics. Topics merely serve to create categories of data items for faster access.

10.1.2 Watch Windows

The bottom pane of the “AppTracker” window has watch windows that allow monitoring of selected data items in real time. The yellow tabbed windows contain pre-selected data for viewing: The remaining tabs (Watch 1 - Watch 4) allow creation of custom groups of data items.

Chart Tab

The “Chart” tab (Figure 10-4) lists data items related to the flow chart currently selected in the Flow Chart Execution Path above it. Highlight any flow chart in the Flow Chart Execution Path to view its data items.

Type	ID	Tagname	Value
Output	0	WhiteKickout	0
Output	1	GreenKickout	0
Output	2	RedKickout	0
Counter	1	WhiteAccumulationCounter	0
Counter	2	GreenAccumulationCounter	0
Counter	3	RedAccumulationCounter	0
Counter	4	YellowAccumulationCounter	0
Counter	6	WhiteDiverterPosition	7
Counter	7	GreenDiverterPosition	12
Counter	8	RedDiverterPosition	18
Counter	10	WhitePackageID	1

Figure 10-4 The “Chart” tab lists data items related to the selected flow chart

The tagname and value for each data item is listed, along with other information. The “Value” column shows the current value, unless it is out of scope (subchart not executing). In that situation, the “Value” field displays diagonal bars (Figure 10-5).

Value	Index
0	

Figure 10-5 Values of tags that are out of scope appear with diagonal bars

Force Tab

The “Force” tab is similar to the “Chart” tab, except that it only displays currently forced data items (“Forcing I/O” on page 10-7). This is particularly useful in sorting out forced data items in large projects.

Statistics Tab

The “Statistics” tab provides a quick summary of commonly-used system data items and their current values (Figure 10-6). Some variables are read-only, but you may modify the value of other system variables. Use the “Edit” menu, then “Set Value”, etc. The statistics listed in the watch window are very useful in finding particular error types, or just viewing the scan-related performance of a project.

Type	ID	Tagname	Value
System	0	ScanInterval	49
System	1	PeakScanInterval	57
System	2	AvgScanInterval	49
System	3	OverScanRatio	0
System	4	LogicTime	1
System	5	PeakLogicTime	7
System	6	AvgLogicTime	1
System	11	ArrayIndexError	0
System	12	DotConnExit	0

Statistics Tab

Figure 10-6 The “Statistics” tab provides a summary of system data items

More information is available in the online Help system or see A 1, “System Data Items”.

10.1.3 Custom Watch Windows

AppTracker includes four customizable tabbed watch windows for data items in the project.

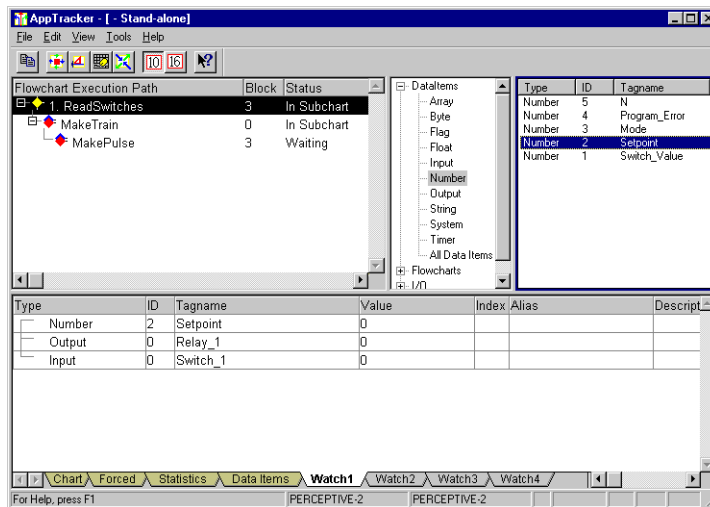


Figure 10-7 AppTracker Custom Watch window

To add a data item to a “Custom Watch” window, do the following:

1. Select the desired watch window.
2. Select the data item(s) in the data item pane. Hold down <Shift> to select a contiguous group of data items. Hold down <Ctrl> to select multiple items.

3. Drag selected data items to the watch area and release the mouse button. The mouse cursor temporarily changes to the copy cursor during the drag, indicating it has “picked up” the selected item(s).



A single data item can be moved to the “Active Watch” pane by double-clicking it.

After placing data items in a watch window, use the cut-copy-paste toolbar buttons to edit or move them to another one.

When the project uses one or more Array data items, you can set AppTracker to display array elements by clicking “View...Arrays Automatically” menu. If the project has a lot of arrays, you may want to disable this feature to improve update times.

AppTracker automatically saves custom watch windows with the project for use the next time the project is opened.

10.1.4 Using AppTracker

With AppTracker you can observe and change data values in the running project to help debug problems and verify logic.

Data Formats

Data values can be displayed in decimal or hexadecimal. From the “AppTracker” main menu, click “View... Decimal” or “View... Hex” menus. All data items are displayed in the watch window in the same format.

Forcing I/O

With a “Chart” or “Watch” tab selected, you can force an input, output or flag data item to a specified value. The term force has special meaning and usefulness in digital control systems. While debugging a project, the ability to override the current value of a data item can speed up the process.

To force a data item value, select it in the “Watch” window and then pull down the “Edit” menu. Three options are available:

Table 10-1 Force Options

Option	Result
Force ON	Turns on a data item (sets = 1) continuously. This overrides any flow charts or screen objects attempting to change the value
Force OFF	Turns off a data item (sets = 0) continuously. This overrides any flow charts or screen objects attempting to change the value
Unforce	Cancels the effect of either a Force ON or a Force OFF for the selected data item



You can also right-click the data item and use a pop-up menu to select these options.

In the “Chart” tab, the value field for forced values is highlighted in red (when a data item is not selected, green if selected). This serves as a visual reminder that the value is currently forced (Force ON or Force OFF).

Writing Data Values Once

Forcing lets you override Runtime logic indefinitely. The Set Value, Zero Value, Turn ON, and Turn OFF functions let you modify values until the project changes them. Turn ON and Turn OFF operate on single-bit data. Set Value and Zero Value are for values that are one byte or larger. For string values, simply enter the new value.

Table 10-2 Forcing Data Value Options

Option	Result
Set Value	Sets the value of the selected data item to the value you enter (can be byte-wide or larger).
Zero Value	Immediately sets the value of the selected data item to 0, all zeros, or blank, regardless of the data width.
Turn ON	Sets a bit-wide data item to 1.
Turn OFF	Sets a bit-wide data item to 0.



Turn ON and Turn OFF states are not highlighted in the Watch window.

If selecting “Set Value,” AppTracker displays a dialog box to specify the desired value. The default format is decimal. To enter the number in hexadecimal, type 0x before the number. For example, to specify 15 in hex format, enter 0x0F.



Regardless of how the value is entered, it appears in the “Watch” window in the format specified in the “View” menu.

10.1.5 Using FlowView with AppTracker

While AppTracker is connected to a running project, you can access FlowView to debug specific flow charts. In the “Flow Chart Execution Path” pane, double-click the flow chart to debug. AppTracker opens FlowView and automatically loads the flow chart.

10.2 Using FlowView in Monitor Mode

You can use FlowView to monitor Runtime operation with or without running AppTracker. With the project running, select “Debug... Monitor Mode” in ProjectCenter or FlowView. Figure 10-8 shows an example of FlowView operating in Monitor mode with the “Locals Window” open. It also shows EasyTrac highlighting execution path information.

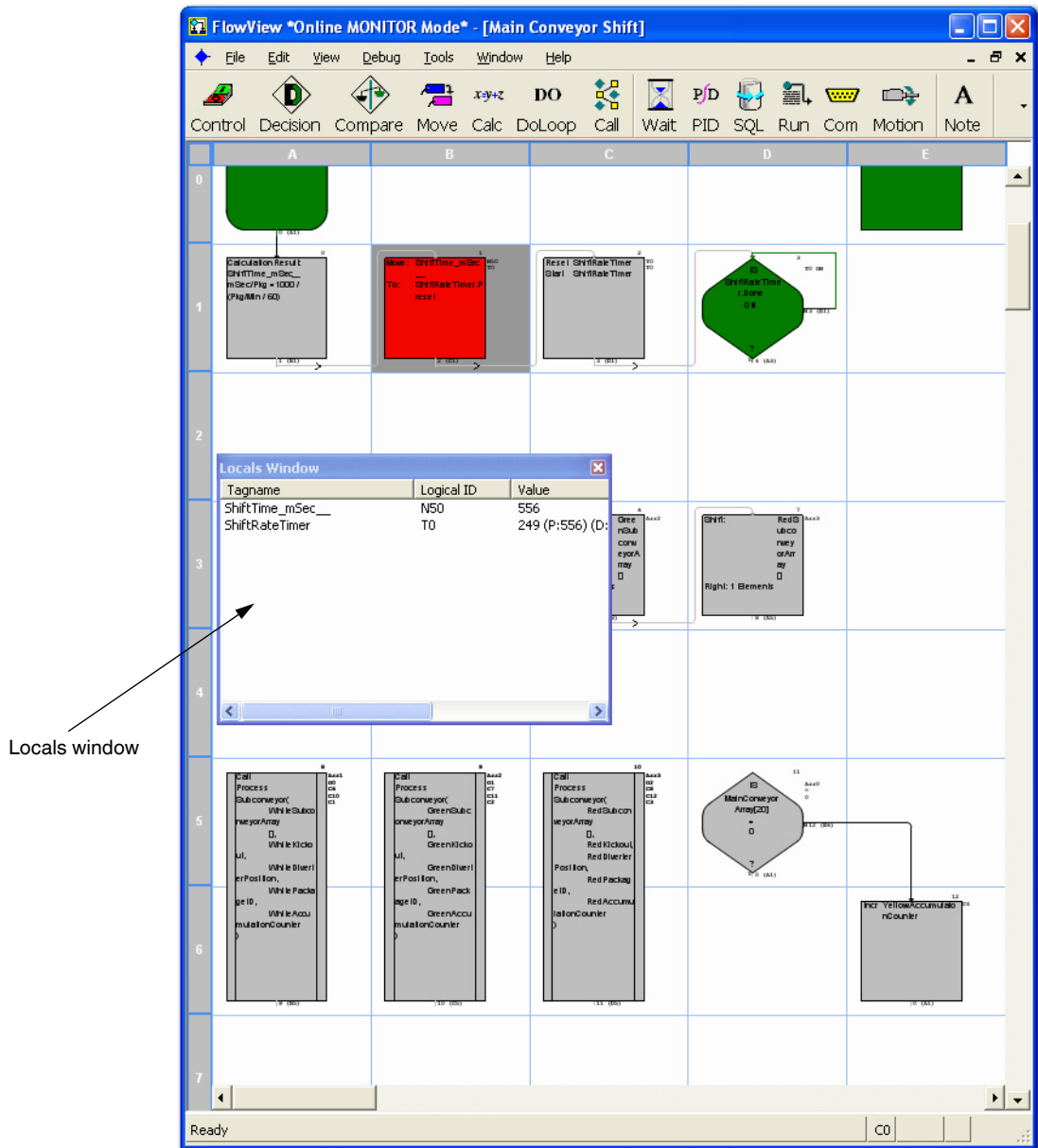


Figure 10-8 FlowView opened by AppTracker



Locals

When opened this way, FlowView’s usual flow chart editing tools are inactive. Other features (those for debugging) are active:

- “Locals Window”: a special Flow Chart Watch window for viewing data item values in blocks.
- EasyTrac: provides visual assistance and execution logging.

To access the Locals Window while in Monitor Mode, select the “Window... Locals” menu. The term locals in the context of a flow chart refers to variables (data items) that are within scope, or are local to a flow chart block. Select any block in a flow chart by clicking it. The “Locals Window” lists the state or value of all the tagnames associated with that block. To monitor values in more than one block, select multiple blocks (hold the <Ctrl> key down and click the blocks).

10.2.1 EasyTrac™

EasyTrac helps to debug by providing some visual assistance and execution logging. In FlowView, select the “Debug... EasyTrac... Active” menu.

Color Codes

The EasyTrac feature color codes flow chart blocks. The default colors are:

- Green: Always executed, on every scan
- Yellow: sometimes executed
- Gray: never executed

Change the colors by selecting the “File... Preferences” menu. Then click the “Block Colors” tab and scroll down to view the EasyTrac options (Figure 10-9).

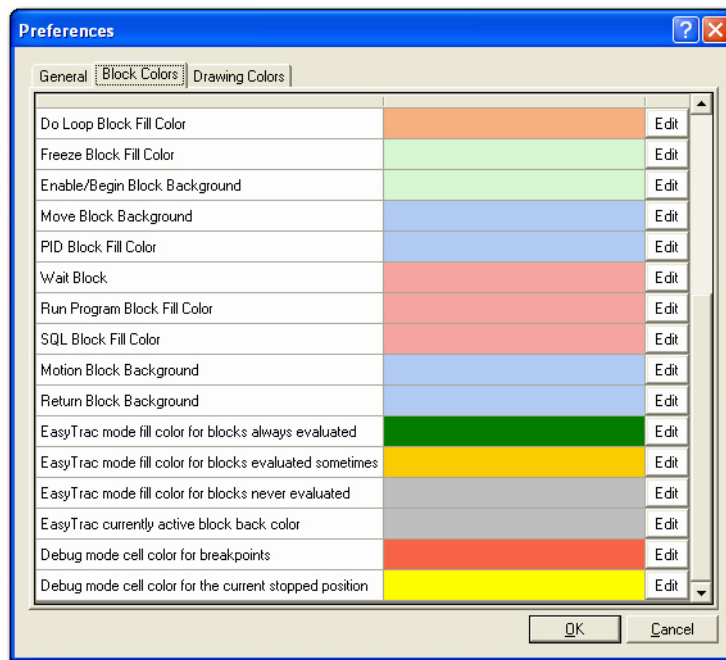


Figure 10-9 FlowView “Preferences... Block Colors” tab



Clear History Button

History Tracking

EasyTrac has a historical mode that records and accumulates logic execution of blocks on every scan. For example, if a particular block is executed just once, EasyTrac displays it as “sometimes executed.” This feature is very useful. To enable EasyTrac’s history tracking, in FlowView, select the “Debug... EasyTrac... Historical Mode” menu.

When FlowView is in Monitor mode, clear historical data by clicking the “Clear History” toolbar button.

With historical mode disabled, EasyTrac keeps very limited execution history (less than 1 second), and discards old data from the evaluation window.

10.2.2 Flow Chart Execution Controls

In addition to EasyTrac visualization, FlowView lets you control flow chart execution through menu selections.

Starting and Stopping Execution

You must halt the project or stop at a breakpoint before stepped execution is available.

To halt execution, click the “red” button, select the “Debug... Halt RTE” menu, or press <F6>. This freezes variable values and stops program execution. To continue execution, click the “green” button, select the “Debug... Go” menu, or press <F7>.



Stopping a running project can have serious effects on moving equipment. Be sure you understand the possible consequences of your action.



Toggle Breakpoint

Breakpoints

Breakpoints force execution to halt at specified locations to give you time to view and/or change (Force, see “Forcing I/O” on page 10-7) tag values. You can set breakpoints with the project running or halted. To set a breakpoint:

1. Click on a block in the flow chart at the desired execution halt point.
2. Select the “Debug... Toggle Breakpoint” menu to set a breakpoint. Note that the background color changes to indicate a set breakpoint.

To clear a breakpoint, do the following:

1. Select a program block that has a breakpoint set.
2. Select the “Debug... Toggle Breakpoint” menu.



A breakpoint halts execution of a running project can have serious effects on moving equipment. Be sure you understand the possible consequences of your action.

You can also view and remove breakpoints in the “Breakpoints” dialog box (Figure 10-10). In FlowView, select the “Debug... Breakpoints” menu.

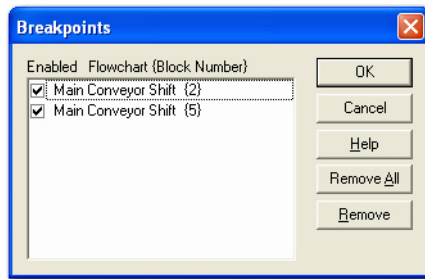


Figure 10-10 The “Breakpoints” dialog box

To disable a breakpoint, clear the checkbox. To delete a breakpoint, select it and click the “Remove” button. To delete all breakpoints, click the “Remove All” button.

Breakpoints are saved with a project.

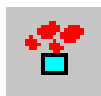
Step Control

The data sample rate of FlowView debugging is 250 ms. If you need to monitor data values that change too quickly to see in real-time, you can execute the flow charts one step at a time.



Step control can change the current action of the Runtime system. This can have dramatic effects on a running project. Be sure you understand the possible consequences of your action.

Step Block

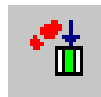


Step Block

You can incrementally execute flow charts in a project one block at a time. Execution automatically stops after each block. Program execution begins with the first flow chart in multiple-flow chart projects. Monitor the “FlowView” window while using the “Debug... Step Block” menu to see each flow chart block highlighted as each step occurs.

You can use the Step Block function in any flow chart. When the stepped execution encounters a Call block, it treats the corresponding subroutine as one block. The next step block function steps to the next standard flow chart block.

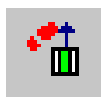
To step into a subchart, use the “Debug... Step In” menu.



Step In

Step In

To step through a flow chart normally (like Step Block) and follow a Call block into a subchart, one block at a time, use the “Step In” menu. If that subchart calls another subchart, the Step In feature follows the call down to another level. When it encounters a Return block, it automatically steps out of the subchart, back to the calling flow chart



Step Out

Step Out

To immediately execute all remaining blocks in a subchart and returns to the calling flow chart, use the “Step Out” menu. If the calling flow chart is also a subchart, a subsequent Step Out completes execution of the current subchart and goes up another level. If in a standard

flow chart, Step Out works like the Step Block. Step out in a standard flow chart executes all logic remaining in the standard chart until a loopback or Wait block occurs. Then it stops at the first block executed in the next standard flow chart



Step To Block

Step To

To execute a one-time breakpoint setting, select the “Debug... Step To” menu. When the program is halted, select the block to stop execution. Then the Step to Block function executes all flow chart blocks up to the selected block. A block must be selected before each use of the “Step to Block” toolbar button.

10.3 Edit Mode

When in Edit mode, the development system has a special connection to the Runtime project. This connection allows access to data for Watch Windows, etc., but it also permits modification of the Runtime project online. Note that new data items cannot be created in this mode. To use Edit mode, select “Debug... Edit Mode” in ProjectCenter or FlowView.

After making changes, the development and Runtime projects are different. If saving the project at this time, the edited (newest) version is active. Edit Mode permits updating either flow charts without having to stop the Runtime and do a project build! This is a very powerful feature when stopping a production line is not practical.

After completing the edit(s) for online changes, you’re ready to send them to the Runtime target. In FlowView, select the “Debug... Update RTE” menu. In the “Runtime Modification” dialog box, click the “OK” button. After this, the Runtime and development projects match.



Any changes made are offline until sent to the Runtime project.



For Windows CE targets – If you chose to enable Instant Recall, Think & Do re-compresses and downloads the source files, along with the changes, after each “Update RTE”. This keeps the compressed source files synchronized with the running project.

10.4 Simulation Flow Charts

Process simulation is useful in project development when access to an actual machine or process is restricted or not yet possible. It allows full testing and debugging of flow charts and screens in advance, provided the simulation accurately duplicates the process. Think & Do allows design of simulation flow charts to respond to real mode flow charts written for a project.

When the machine or process is not present, simulation flow charts act as a substitute. Simulation flow charts read outputs being controlled by the real mode flow charts and write input values.



When simulating I/O, you must simulate all I/O. Real and simulated I/O cannot be mixed.

10.4.1 Using Simulation

Identify which flow charts and subcharts are simulation flow charts:

1. In the ProjectCenter main window, select “Simulated I/O” in the “Build Modes” group (Figure 10-11).



Figure 10-11 Configuring simulated I/O

2. In ProjectCenter, click the “Flow Charts” Explorer bar.
3. Expand the “Flow Charts” folder and click the flow chart to configure or add a new flow chart.
4. For individual simulation flow charts, click the “Simulated” check box for the flow chart (Figure 10-12). In the “Flow Charts” explorer, the flow chart moves to the Simulation category.

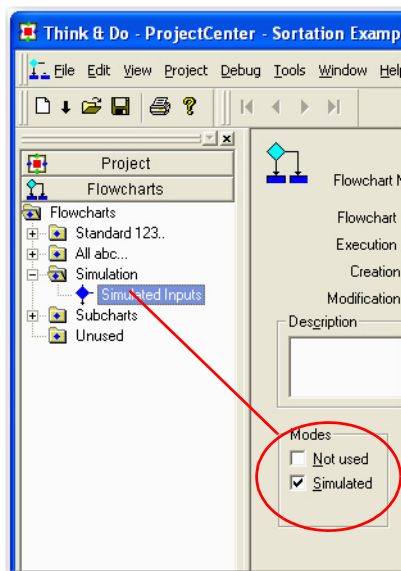


Figure 10-12 Configuring a simulation flow chart

To use simulation capability in a flow chart Control Block:

1. Place a Control Block on the drawing page.
2. Double-click the block to edit its expression.
3. Click the “Action” field and select “Turn ON Bit” from the list.
4. Click the “Type” field and choose “Input” from the list.
5. Double-click the “Tagname” field to bring up the Data Item grid. After choosing a data item, return to FlowView.

The Move Block also has added capability when its flow chart is a simulation type. The example below shows a number value being moved to sixteen discrete inputs Bit_16 to Bit_1. This method simulates analog input values.

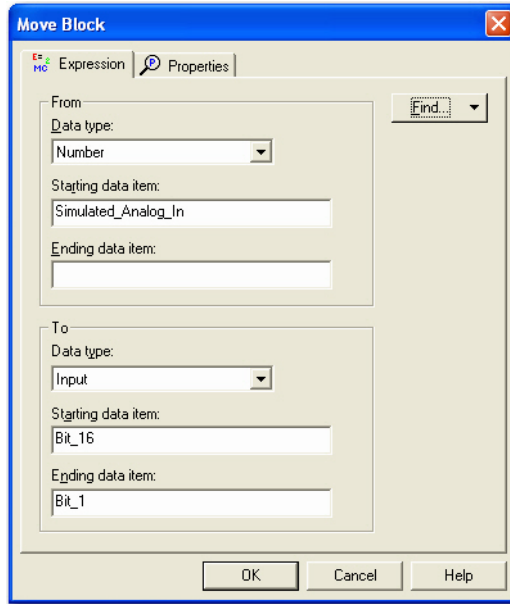


Figure 10-13 Configuring simulated analog input values



You may consider temporarily using part of an operator screen for simulation data. This provides a place to enter process values in real time, so simulation flow charts just read the data input and write them to actual input data items.

In designing simulation flow chart(s), remember to simulate the machine or process for the benefit of the real mode flow charts, which will also be running at Runtime. The better the simulation, the more confident you can be in the design of the real mode flow charts.

When finished with the simulation flow chart design, save it and return to ProjectCenter. The Flow Chart Explorer lists the flow chart in the simulation folder.



You should move any simulation flow charts to execute last in the project. Just click and drag the flow charts to the proper list position. This order ensures real mode flow charts and simulation charts “take turns” generating logical outputs without interference.

10.4.2 Disabling Simulation for Real I/O

In the Project Explorer of ProjectCenter, click the “Real I/O” radio button in the “Build Mode” group box (see Figure 10-11 on page 10-14).

Think & Do

A Appendix

A 1 System Data Items

The table in this appendix describes System Data Items that are applicable to Windows-based PCs. While they also apply to the Think & Do WinPLC Embedded CE device, not all of them apply to all Windows CE-based target devices.

*Indicates an error message. For information on interpreting errors, see “Notes on Reading Error Data” on page A-5

ID	System Data Item	Description
SYS-0	ScanInterval	Read only. The last runtime scan interval. This time is in milliseconds.
SYS-1	PeakScanInterval	The peak, or largest, scan interval encountered by the runtime. Each time the runtime updates ScanInterval, it compares it to PeakScanInterval, if ScanInterval is larger, it is written to PeakScanInterval. Typically you would set this value to zero to clear the current PeakScanInterval and observe the new value. This value is stated in milliseconds.
SYS-2	AvgScanInterval	Read only. The average scan interval of the runtime. This value is stated in milliseconds.
SYS-3	OverScanRatio	Read only. The ratio of scan intervals that are over the desired scan time for the application. This value is stated in tenths of a percent. For example, a value of 12 means that 1.2 percent of the scan intervals were over the specified scan interval.
SYS-4	LogicTime	Read only. The time taken by the runtime system to read the inputs, execute the flow charts, and write the outputs for the last scan. This value is stated in milliseconds.
SYS-5	PeakLogicTime	The peak, or largest, logic time encountered by the runtime. Each time the runtime updates LogicTime, it compares it to PeakLogicTime, if LogicTime is greater, PeakLogicTime is updated with the new value. Typically you will set this value to zero to clear the PeakLogicTime and observe the new value. This value is stated in milliseconds.
SYS-6	AvgLogicTime	Read only. The average logic time required of the runtime. This value is stated in milliseconds.
SYS-7	Reserved7	Read only. Reserved for internal use.
SYS-8	Reserved8	Read only. Reserved for internal use.
SYS-9	Reserved9	Read only. Reserved for internal use.
SYS-10	Reserved10	Read only. Reserved for internal use.

Think & Do

ID	System Data Item	Description
SYS-11	ArrayIndexError*	Contains error information on flow chart blocks that illegally index into arrays. This value is written by the runtime whenever flow charts try to use an invalid array index. The error code is set to 1 if the array index is too small and set to 2 if the array index is too large. See "Notes on Reading Error Data" on page A-5 for information on decoding this error.
SYS-12	DoLoopExit*	Contains error information on Do Loops. This value gets written to by the runtime when a flow chart exits a Do Loop because it went through the loop the maximum number of times. The error code is set to 1 to indicate the error. See "Notes on Reading Error Data" on page A-5 for information on decoding this error.
SYS-13	ScanCount	The running counter of the number of scans executed by the runtime. This is a 32-bit unsigned number, so it will eventually wrap around to zero. Do not write to this value. It may be used by a runtime component or other application.
SYS-14	Reserved14	Read only. Reserved for internal use.
SYS-15	DataLogging Status	Is used by the Think & Do Data Logging runtime to indicate the status of the data logging task. Do not write to this data item. See Data Logging in help for more details.
SYS-16	RollingMsecCounter	Read only. Counts the number of milliseconds since the project runtime was started. This is a 32-bit unsigned number, so it will eventually wrap around to zero.
SYS-17	LastShutdown	Read only. Contains an encoded time stamp of the last time the runtime shut down. It is valid only when the runtime is configured for retentive data items. Bits 0-7=hour, bits 8-15=minute, bits 16-23=second, bits 24-31=day of the week.
SYS-18	Reserved18	Read only. Reserved for internal use.
SYS-19	PowerFailShutdown	When the value is set, the logic stops executing, I/O stops scanning, and data items marked as retentive are written to disk. See online Help for more information.
SYS-20	Reserved20	Read only. Reserved for internal use.
SYS-21	Reserved21	Read only. Reserved for internal use.
SYS-22	Reserved22	Read only. Reserved for internal use.
SYS-23	Reserved23	Read only. Reserved for internal use.
SYS-24	Reserved24	Read only. Reserved for internal use.
SYS-25	Reserved25	Read only. Reserved for internal use.

ID	System Data Item	Description
SYS-26	Reserved26	Read only. Reserved for internal use.
SYS-27	Reserved27	Read only. Reserved for internal use.
SYS-28	Reserved28	Read only. Reserved for internal use.
SYS-29	OverScans	Is a 32-bit counter containing the number of times the measured scan interval exceeded the desired scan interval. This is a 32-bit unsigned number, so it will eventually wrap around to zero.
SYS-30	LastIOTime	Read only. Is the time the runtime took to read the inputs and write the outputs on the last scan.
SYS-31	PeakIOTime	Is the peak LastIOTime encountered by the runtime. Each time the runtime updates LastIOTime, it compares it to PeakIOTime, if LastIOTime is larger, it is written to PeakIOTime. Typically the user would set this value to zero to clear the current PeakIOTime to observe the new value. This value is in milliseconds.
SYS-32	ClockMin	Read only. Toggles between zero and one every minute. It is zero for 30 seconds and one for the next 30 seconds. It repeats this pattern forever.
SYS-33	ClockSec	Read only. Toggles between zero and one every second. It is zero for 0.5 seconds and one for the next 0.5 seconds. It repeats this pattern forever.
SYS-34	Clock100ms	Read only. Toggles between zero and one every 100 milliseconds. It is zero for 50 milliseconds and one for the next 50 milliseconds. It repeats this pattern forever.
SYS-35	Clock50ms	Read only. Toggles between zero and one every 50 milliseconds. It is zero for 25 milliseconds and one for the next 25 milliseconds. It repeats this pattern forever.
SYS-36	FirstScan	Read only. Is one on the very first scan of the project, from then on it is zero.
SYS-37	AltScan	Read only. Toggles between zero and one every scan. It is zero for a scan and one for the next scan. It repeats this pattern forever.
SYS-38	IOStatus	Read only. Contains information on the status of the I/O. A value of zero means the I/O is working normally. A value of one means the runtime is running in simulated I/O mode. A value of two means there is an I/O fault.
SYS-39	MathError*	The error code is set to 1 on an overflow, underflow, divide by zero, and other math errors. See "Notes on Reading Error Data" on page A-5 for information on decoding this error.
SYS-40	Reserved40	Read only. Reserved for internal use.

ID	System Data Item	Description
SYS-41	StringError*	Indicates the flow chart block, the string error code, and the flow chart number where the last error occurred. Error = 1 if string data item could not be read from data table Error = 2 if internal string buffer overflow Error = 3 if string written to data table is truncated because it is longer than the maximum length specified in the data view grid Error = 4 if illegal offset into string. See "Notes on Reading Error Data" on page A-5 for information on decoding this error.
SYS-42	ConversionError*	Indicates the flow chart block, the error code, and the flow chart number where the last error occurred. Conversion errors are generated when converting between different data types and the destination is too small for the value being written to it. See "Notes on Reading Error Data" on page A-5 for information on decoding this error.
SYS-43	Write RetentiveData	Your flow chart or screen can set this bit = 1 to cause the runtime project to save retentive data items to disk. The runtime system automatically resets this bit after saving the retentive data.
SYS-44	Reserved44	Read only. Reserved for internal use.
SYS-45	Reserved45	Read only. Reserved for internal use.
SYS-46	ExtendedFunction BlockError*	Block, error code and chart number where last error occurred. Error = 0 if timeout Error = 1 to 15 if error is returned from extended block. See "Notes on Reading Error Data" on page A-5 for information on decoding this error.
SYS-47	ScreenInput Ignored	All HMI screen data entry is ignored when this value = 1.
SYS-48	Reserved48	Read only. Reserved for internal use.
SYS-49	Reserved49	Read only. Reserved for internal use.
SYS-50	LastExtended FunctionBlockTime	Read only. Execution time (microseconds) of the last Extended Function Block call.
SYS-51	PeakExtended FunctionBlockTime	Peak execution time (microseconds) of all Extended Function Block calls.
SYS-52	AvgExtended FunctionBlockTime	Read only. Average execution time (microseconds) of all Extended Function Block calls.
SYS-53	Spare1	Reserved for internal use.
SYS-54	Spare2	Reserved for internal use.
SYS-55	Spare3	Reserved for internal use.

ID	System Data Item	Description
SYS-56	Spare4	Reserved for internal use.
SYS-57	LiveCaptureStatus	Status of the Live! capture set(s). Do not write to this data item. See Live! capture in help for more details.
SYS-58	MajorBuildVersion	Read only. Major build version number — read only
SYS-59	MinorBuildVersion	Read only. Minor build version number — read only
SYS-60	MaximumFlowchart Time	Read only. The maximum flow chart execution time (one component of the time reported in SYS-1). This value is stated in milliseconds.

A 2 Notes on Reading Error Data

Error data contains the error code and the block and flow chart where the last error occurred. The bits in the error data can be read as follows:

- Bits 0–15 contain a reference to the flow chart that has the error. If the error occurred in a standard flow chart, you can find the number in the “Standard 1,2,3...” folder of the flow chart Explorer in ProjectCenter. If the error occurred in a subchart, the number will be in the 10000 range. Subchart numbers are available in the “Chart Properties” dialog box, which you open by selecting the “File... Chart Properties” menu in FlowView. For simulation flow charts, you can calculate the flow chart number by subtracting the number of simulation flow charts found before it in the list. The runtime system writes to the system data item whenever it encounters an error, so if a project contains several errors, only the last one is captured in the data item.
- Bits 20–31 contain the block number. When an error occurs in the Enable or Freeze Blocks, the Runtime identifies the block as block 0.
- Bits 16–19 contain the error code.

Think & Do

A 3 List of Figures

Section 1

Figure 1-1:	The Think & Do “Help” menu	1-5
Figure 1-2:	PCs connect to I/O bases in racks	1-6
Figure 1-3:	A PC can connect to a network with drops to devices.....	1-7
Figure 1-4:	PC connected to multiple I/O controllers.....	1-7
Figure 1-5:	WinPLC and its one base of I/O modules	1-7
Figure 1-6:	Group development system.....	1-9
Figure 1-7:	Local PC used for development and Runtime.....	1-10
Figure 1-8:	Development and Runtime control on different systems.....	1-10

Section 2

Figure 2-1:	Project components.....	2-3
Figure 2-2:	ProjectCenter provides access to Think & Do development tools.....	2-4
Figure 2-3:	The “Rename Project” dialog box with a name filled in	2-5
Figure 2-4:	Use the “Save As” dialog box to define the project name.....	2-5
Figure 2-5:	The “Project” Explorer bar provides access to Think & Do tools	2-6
Figure 2-6:	The “Flow Charts” and “Screens” Explorer bars.....	2-7
Figure 2-7:	The “Data Items” Explorer bar.....	2-8
Figure 2-8:	The “Find...Configure CE Runtime” dialog box.....	2-9
Figure 2-9:	Flow chart showing an action and branching block.....	2-10
Figure 2-10:	“Name Flow Chart” dialog box	2-11
Figure 2-11:	“Screen Attributes” dialog box	2-13

Section 3

Figure 3-1:	ProjectCenter Data Items.....	3-4
Figure 3-2:	The Data Items grid for Counter data types	3-4
Figure 3-3:	IOView window includes a board view, module view, and tab view	3-6
Figure 3-4:	The “Add I/O Driver” dialog box and its driver listing	3-7
Figure 3-5:	IOView displays an image of the I/O scanner card or Runtime target .	3-8
Figure 3-6:	IOView showing the "Module Info" tab for selected module	3-9
Figure 3-7:	IOView showing the "I/O Mapping" tab for selected module	3-11
Figure 3-8:	Physical I/O points map to logical I/O points in any sequence	3-12
Figure 3-9:	The on-screen images in IOView reflect input and output states	3-13
Figure 3-10:	Connection-related buttons available in the toolbar	3-14

Figure 3-11: “Add” and “Remove” driver buttons 3-15
Figure 3-12: “Add,” “Remove,” “Insert” and “Replace” device buttons 3-15

Section 4

Figure 4-1: Flow chart segment showing an action and branching block 4-3
Figure 4-2: A Call block calls a subchart, which returns to the original flow chart.. 4-4
Figure 4-3: The “Project” Explorer bar showing the Flow Chart explorer 4-5
Figure 4-4: Heirarchical sublist list collapsed (left), expanded (right) 4-6
Figure 4-5: “Name Flow Chart” dialog box 4-7
Figure 4-6: Select the “Unused” or “Simulation” check box to move a flow chart .. 4-8
Figure 4-7: FlowView showing a new Flow Chart..... 4-9
Figure 4-8: Zoom tools in FlowView 4-11
Figure 4-9: Open multiple flow charts in FlowView, each with its own window 4-12
Figure 4-10: Decision, Compare, and Do Loop blocks appear as diamonds 4-15
Figure 4-11: Sample flow chart with flow lines connecting all blocks 4-17
Figure 4-12: “Compare Block... Properties” tab 4-18
Figure 4-13: Use the “Screen Attributes” dialog box to set up a new screen 4-20
Figure 4-14: ScreenView development environment 4-21
Figure 4-15: ScreenView Explorer 4-22
Figure 4-16: The “Object Properties” dialog box 4-22
Figure 4-17: The “Replace” dialog box..... 4-23
Figure 4-18: The “Colors” dialog box 4-25
Figure 4-19: The “Colors... Custom” tab 4-25
Figure 4-20: The “Fill Effects” dialog box 4-26

Section 5

Figure 5-1: Every new flow chart opens with a predefined Enable block..... 5-4
Figure 5-2: The “Enable Block... Expression” tab..... 5-4
Figure 5-3: The “Calculation Block... Expression” tab 5-5
Figure 5-4: The “Call Block... Expression” tab..... 5-7
Figure 5-5: “Note Properties” dialog box 5-8
Figure 5-6: The “Communication Block... Expression” tab 5-9
Figure 5-7: The “Compare Block... Expression” tab 5-10
Figure 5-8: The “Control Block... Expression” tab 5-13
Figure 5-9: The “Decision Block... Expression” tab 5-15
Figure 5-10: A Do Loop consists of Do Loop block linked to a Compare block 5-17
Figure 5-11: Do Loop block showing “Maximum Loop Count” 5-18

Figure 5-12:	Typical I/O subsystem and a motion control systems design.....	5-19
Figure 5-13:	The “Motion Block... Expression” tab.....	5-19
Figure 5-14:	The “Move Block... Expression” tab.....	5-21
Figure 5-15:	PID loop block diagram.....	5-23
Figure 5-16:	The “PID Block... Expression” tab.....	5-23
Figure 5-17:	The “Run Program Block... Expression” tab.....	5-24
Figure 5-18:	The “SQL Block... General” tab.....	5-25
Figure 5-19:	The “Wait Block... Expression” tab.....	5-26
Figure 5-20:	Calling flow chart passes values to subcharts via parameters.....	5-27
Figure 5-21:	The “Begin Block... Expression” tab defines parameters and locals.....	5-28
Figure 5-22:	The “Call Block... Expression” tab.....	5-29

Section 6

Figure 6-1:	The “Screens” Explorer bar in ProjectCenter.....	6-3
Figure 6-2:	Use ScreenView to create and edit HMI screens.....	6-5
Figure 6-3:	Object Properties: Check Box.....	6-7
Figure 6-4:	“Object Finder” dialog box.....	6-7
Figure 6-5:	The “Select Index” dialog box.....	6-8
Figure 6-6:	The “Member selection” dialog box.....	6-8
Figure 6-7:	The “New Tag” dialog box.....	6-9
Figure 6-8:	The “Screen Tags” dialog box.....	6-10
Figure 6-9:	The “Standard” toolbar.....	6-12
Figure 6-10:	Zoom drop-down selection.....	6-13
Figure 6-11:	The “Execution” toolbar.....	6-13
Figure 6-12:	The “Bitmap” toolbar.....	6-14
Figure 6-13:	The “Edit Image” dialog box.....	6-15
Figure 6-14:	The “Dynamic Properties” toolbar.....	6-16
Figure 6-15:	Object Properties: Command.....	6-16
Figure 6-16:	Object Properties: Hyperlink.....	6-17
Figure 6-17:	Object Properties: BarGraph.....	6-18
Figure 6-18:	Object Properties: Text I/O.....	6-19
Figure 6-19:	Object Properties: Colors.....	6-20
Figure 6-20:	The “Color Limits” dialog box.....	6-21
Figure 6-21:	Object Properties: Position.....	6-22
Figure 6-22:	Object Properties: Resize.....	6-23
Figure 6-23:	Object Properties: Rotation Property.....	6-23
Figure 6-24:	The “Mode” toolbar.....	6-24

Figure 6-25:	The “Line Selection” dialog box	6-25
Figure 6-26:	The “Static Objects” toolbar	6-26
Figure 6-27:	Object Properties: Open Polygon	6-26
Figure 6-28:	Object Properties: Closed Polygon	6-27
Figure 6-29:	Object Properties: Line	6-28
Figure 6-30:	Oval, Chord, Arc, and Ring	6-28
Figure 6-31:	Object Properties: Ellipse	6-29
Figure 6-32:	Object Properties: Rounded Rectangle	6-30
Figure 6-33:	Object Properties: Button.....	6-31
Figure 6-34:	Object Properties: Text	6-32
Figure 6-35:	Object Properties: Rectangle.....	6-33
Figure 6-36:	The “Caption” dialog box	6-33
Figure 6-37:	The “Active Objects” toolbar	6-34
Figure 6-38:	Object Properties: Alarm/Event Control	6-34
Figure 6-39:	Object Properties: Trend Control	6-35
Figure 6-40:	The “Insert ActiveX Control” dialog box	6-36
Figure 6-41:	Object Properties: ActiveX Control	6-37
Figure 6-42:	The “.NET Framework Components” dialog box.....	6-38
Figure 6-43:	Object Properties: .NET Control	6-39
Figure 6-44:	Sample Grid object	6-40
Figure 6-45:	Object Properties: Grid	6-40
Figure 6-46:	Object Properties: Combo Box	6-41
Figure 6-47:	The “Data Sources” dialog box	6-42
Figure 6-48:	Object Properties: Radio Button	6-43
Figure 6-49:	The radio button “Advanced” dialog box.....	6-44
Figure 6-50:	Object Properties: Check Box.....	6-46
Figure 6-51:	The check box “Advanced” dialog box.....	6-47
Figure 6-52:	Object Properties: List Box	6-50
Figure 6-53:	The “Messages Configuration” dialog box	6-51
Figure 6-54:	Object Properties: Smart Message	6-53
Figure 6-55:	The smart message “Configuration” dialog box	6-54
Figure 6-56:	Object Properties: Pushbutton.....	6-57
Figure 6-57:	The pushbutton “Configuration” dialog box	6-59
Figure 6-58:	Pushbutton styles	6-59
Figure 6-59:	The “Align and Distribute” toolbar	6-60
Figure 6-60:	Aligning objects left.....	6-61
Figure 6-61:	Aligning objects right	6-62

Figure 6-62:	Aligning objects top	6-62
Figure 6-63:	Aligning objects bottom	6-62
Figure 6-64:	Centering objects vertically.....	6-63
Figure 6-65:	Centering objects horizontally.....	6-63
Figure 6-66:	Distributing objects horizontally	6-63
Figure 6-67:	Distributing objects vertically	6-64
Figure 6-68:	Flipping objects horizontally.....	6-64
Figure 6-69:	Flipping objects vertically.....	6-65
Figure 6-70:	Rotating objects.....	6-65
Figure 6-71:	Moving objects to back	6-66
Figure 6-72:	Moving objects to front	6-66
Figure 6-73:	Object Properties: Alarm/Event Control	6-68
Figure 6-74:	Displaying a Grid on an Alarm/Event Control	6-68
Figure 6-75:	Displaying a Header on an Alarm/Event Control	6-69
Figure 6-76:	The Alarm/Event Control “Columns” dialog box	6-70
Figure 6-77:	The Alarm/Event Control “Filters” dialog box	6-71
Figure 6-78:	The Alarm/Event Control “Advanced” dialog box.....	6-72
Figure 6-79:	The Alarm/Event Control “Navigation Triggers” dialog box	6-73
Figure 6-80:	Object Properties: Alarm/Event Control	6-74
Figure 6-81:	Object Properties: Trend Control	6-75
Figure 6-82:	The Trend Control “Data Sources” dialog box	6-75
Figure 6-83:	The Trend Control “Points” dialog box	6-77
Figure 6-84:	The Trend Control “Pen Style” dialog box	6-78
Figure 6-85:	The Trend Control “Options” dialog box.....	6-79
Figure 6-86:	The Trend Control “Axes” dialog box	6-80
Figure 6-87:	The Trend Control “Cursor” dialog box	6-82
Figure 6-88:	The Trend Control “Position” dialog box	6-82
Figure 6-89:	The Trend Control “Legend” dialog box	6-83
Figure 6-90:	The Trend Control “Toolbar” dialog box	6-84
Figure 6-91:	The Trend Control “Advanced” dialog box	6-85
Figure 6-92:	Object Properties: ActiveX Control	6-89
Figure 6-93:	The ActiveX Control “Configuration... Properties” tab	6-89
Figure 6-94:	The ActiveX Control “Configuration... Methods” tab	6-91
Figure 6-95:	The ActiveX Control “Configuration... Events” tab	6-92
Figure 6-96:	Object Properties: .NET Control	6-93
Figure 6-97:	The .NET Control “Members... Properties” tab	6-93
Figure 6-98:	The .NET Control “Members... Methods” tab.....	6-95

Figure 6-99:	The .NET Control “Members... Events” tab.....	6-96
Figure 6-100:	Object Properties: Grid	6-97
Figure 6-101:	The “Grid Data - Text File” dialog box	6-97
Figure 6-102:	The “Grid Data - Class Tag” dialog box.....	6-98
Figure 6-103:	The Runtime “View” pop-up dialog box.....	6-98
Figure 6-104:	The “Grid Data - Class Tag” dialog box.....	6-99
Figure 6-105:	The Grid “Columns” dialog box	6-100
Figure 6-106:	The Grid “Advanced” dialog box	6-102
Figure 6-107:	Symbol Library.....	6-105
Figure 6-108:	Symbol placed on a screen	6-105
Figure 6-109:	Object Properties: Linked Symbol	6-106
Figure 6-110:	“Symbol Properties” dialog box for the linked symbol	6-106
Figure 6-111:	“Symbol Properties” dialog box with custom properties specified ...	6-107
Figure 6-112:	Creating a Master Symbol	6-107
Figure 6-113:	Symbol in Symbol Editor.....	6-108
Figure 6-114:	Object Properties: Check Box.....	6-108
Figure 6-115:	Object Properties: Check Box with a custom property specified	6-109
Figure 6-116:	Symbol Properties showing a custom property to define	6-109
Figure 6-117:	Object Properties: Check Box with a custom property specified	6-110
Figure 6-118:	A default custom property in the “Symbol Properties” dialog box....	6-110
Figure 6-119:	Object Properties: Check Box with all custom properties specified	6-110
Figure 6-120:	The “Symbol Properties” dialog box for editing tooltips.....	6-111
Figure 6-121:	The symbol “Object Properties” dialog box showing a tooltip	6-111
Figure 6-122:	The “Screen Attributes” dialog box	6-112
Figure 6-123:	“Insert Screen Group” dialog box.....	6-114
Figure 6-124:	Alarm worksheet.....	6-116
Figure 6-125:	Alarm heading	6-117
Figure 6-126:	Alarm body	6-119
Figure 6-127:	Alarm worksheet, “Advanced” settings dialog box	6-120
Figure 6-128:	Math worksheet	6-122
Figure 6-129:	Recipe worksheet.....	6-123
Figure 6-130:	Report worksheet	6-125
Figure 6-131:	Scheduler worksheet.....	6-126
Figure 6-132:	Trend worksheet.....	6-128
Figure 6-133:	Trend worksheet.....	6-130
Figure 6-134:	Global Procedure script page	6-132
Figure 6-135:	Graphics script page.....	6-133

Figure 6-136:	Startup script page	6-134
Figure 6-137:	Background script page.....	6-135

Section 7

Figure 7-1:	The Data Item grid displays all tags of a particular data type	7-3
Figure 7-2:	Use a label array with string values that correspond to data values	7-9
Figure 7-3:	The drop-down arrow displays a table with initial values.....	7-10
Figure 7-4:	Timer data items are available to different blocks	7-14
Figure 7-5:	AppTracker watch window with timer variables visible	7-15
Figure 7-6:	The “Move Block... Expression” tab	7-17
Figure 7-7:	The MSB and LSB are specified in the range fields	7-20
Figure 7-8:	To move a number to a series of output bits, specify MSB/LSB	7-21
Figure 7-9:	The “Calculation Block... Expression” tab	7-25
Figure 7-10:	IOView includes a Sign Extend option	7-30
Figure 7-11:	Map I/O in IOView.....	7-31
Figure 7-12:	Scaling equation in the “Calculation Block... Expression” tab	7-33
Figure 7-13:	The “SQL Block” dialog box helps construct SQL statements	7-37
Figure 7-14:	The “SQL Block... General” tab.....	7-39
Figure 7-15:	The “SQL Block... Tag Mappings” tab.....	7-42
Figure 7-16:	The “SQL Block... Filters” tab.....	7-44
Figure 7-17:	The “SQL Block... Sorting” tab.....	7-45
Figure 7-18:	The resulting SQL in the “SQL Statement” tab.....	7-46
Figure 7-19:	PID Control	7-47

Section 8

Figure 8-1:	Think & Do can communicate with many different devices	8-3
Figure 8-2:	IOView window	8-4
Figure 8-3:	The “Add Serial Device” dialog box	8-5
Figure 8-4:	IOView “Module Info” tab	8-5
Figure 8-5:	Serial Port Settings (“Module Info” tab).....	8-6
Figure 8-6:	Serial port “Module Status Mapping” tab	8-7
Figure 8-7:	Serial port “I/O Mapping” tab.....	8-8
Figure 8-8:	The “Serial Port Scanning” dialog box for manual communication.....	8-9
Figure 8-9:	The OPC Server controls Think & Do projects on a Certified PC	8-10
Figure 8-10:	The OPC CE Server controls Think & Do projects on Windows CE..	8-11
Figure 8-11:	The “Think & Do OPC Server” dialog box	8-12
Figure 8-12:	The OPC Server “Settings” dialog box.....	8-13

Figure 8-13:	The Windows CE OPC Server “Settings” dialog box	8-14
Figure 8-14:	The “Add Tag Link” dialog box.....	8-15
Figure 8-15:	The “Browse for OPC Server” dialog box.....	8-16
Figure 8-16:	Use DDE to communicate with other Windows-based programs.....	8-17
Figure 8-17:	Use Excel and DDE to access a remote Windows CE system.	8-17
Figure 8-18:	The “DDE Server” dialog box.....	8-19
Figure 8-19:	The “DDE Server Configure” dialog box	8-20
Figure 8-20:	The “DDE Server Peek” dialog box.....	8-20
Figure 8-21:	Excel can display tag values from a running Think & Do project	8-22
Figure 8-22:	Excel may prompt to update links to Think & Do tags	8-22
Figure 8-23:	Remote communication between desktop PCs	8-23

Section 9

Figure 9-1:	The “Think & Do Project Build” dialog box	9-3
Figure 9-2:	The “Output Window... Build” tab displays build messages	9-4
Figure 9-3:	The “Think & Do Runtime Engine” window	9-4
Figure 9-4:	Scan Interval is the sum of Logic Solve Time and Pause time	9-5
Figure 9-5:	Change the scan interval permanently in ProjectCenter	9-6
Figure 9-6:	“Runtime Settings” in ProjectCenter	9-7
Figure 9-7:	Runtime windows for Certified PC and Windows CE	9-8
Figure 9-8:	Runtime settings for Windows CE	9-9
Figure 9-9:	The “Think & Do - CE Watch... Think & Do Station” tab	9-10
Figure 9-10:	The “Think & Do - CE Watch - CE Station Selection” dialog box.....	9-11
Figure 9-11:	The “Instant Recall” dialog box	9-13

Section 10

Figure 10-1:	AppTracker window	10-3
Figure 10-2:	AppTracker “Flow Chart Execution Path” pane.....	10-4
Figure 10-3:	AppTracker “Project Explorer” pane	10-4
Figure 10-4:	The “Chart” tab lists data items related to the selected flow chart.....	10-5
Figure 10-5:	Values of tags that are out of scope appear with diagonal bars	10-5
Figure 10-6:	The “Statistics” tab provides a summary of system data items	10-6
Figure 10-7:	AppTracker Custom Watch window.....	10-6
Figure 10-8:	FlowView opened by AppTracker	10-9
Figure 10-9:	FlowView “Preferences... Block Colors” tab.....	10-10
Figure 10-10:	The “Breakpoints” dialog box.....	10-12
Figure 10-11:	Configuring simulated I/O	10-14

Figure 10-12:	Configuring a simulation flow chart	10-14
Figure 10-13:	Configuring simulated analog input values	10-15

Think & Do

A 4 List of Tables

Section 1

Table 1-1:	Manual Conventions for Keyboard Command	1-5
Table 1-2:	Help Sources	1-6

Section 4

Table 4-1:	“Object Properties” dialog box properties	4-23
Table 4-2:	Object Properties “Replace” dialog box parameters	4-24

Section 5

Table 5-1:	Valid Enable Block Expressions	5-5
Table 5-2:	Valid Compare Block Comparisons	5-12
Table 5-3:	Valid Decision Block Comparisons	5-16
Table 5-4:	Valid Moves	5-22
Table 5-5:	Begin and Call Block Parameter Relationships.....	5-30

Section 6

Table 6-1:	Available Grid “Data Source” types	6-40
Table 6-2:	Radio Button Normal Mode States	6-45
Table 6-3:	Radio Button Normal Mode States	6-45
Table 6-4:	Radio Button Tri-State Value Returns	6-46
Table 6-5:	Check Box Normal Mode States.....	6-48
Table 6-6:	Check Box Normal Mode Value Returns	6-48
Table 6-7:	check box Normal Mode States.....	6-49
Table 6-8:	Check Box Tri-State Value Returns	6-49
Table 6-9:	Smart Message CSV File Format	6-55
Table 6-10:	Data Source Settings, “Source Type” Actions.....	6-76
Table 6-11:	“Period” Properties Depend on “Data Type”.....	6-81
Table 6-12:	ScreenView Toolbar Icons.....	6-87
Table 6-13:	ScreenView Legend Icons	6-88
Table 6-14:	ActiveX Property Icons.....	6-90
Table 6-15:	ActiveX Actions	6-90
Table 6-16:	ActiveX Scan Polling Methods.....	6-91
Table 6-17:	.NET Framework Property Icons	6-94
Table 6-18:	.NET Framework Actions.....	6-94

Think & Do

Table 6-19:	Grid Column Types.....	6-100
Table 6-20:	Grid Advanced Dialog Box Export Fields.....	6-104
Table 6-21:	Alarm Worksheet Header Settings	6-117
Table 6-22:	Alarm Worksheet Body Settings	6-119
Table 6-23:	Alarm Worksheet Advanced Settings	6-120
Table 6-24:	Default Database Field Name Examples	6-129
Table 6-25:	Trend History Fields	6-130

Section 7

Table 7-1:	Think & Do Data Types.....	7-6
Table 7-2:	Clock Tagname and ID	7-12
Table 7-3:	Scan Clock Tagname and ID	7-12
Table 7-4:	Move Block summary	7-23
Table 7-5:	Runtime Math Errors.....	7-27
Table 7-6:	Math Operators.....	7-27
Table 7-7:	Bitwise Operators	7-28
Table 7-8:	Math Functions	7-29
Table 7-9:	Calculation Block Functions.....	7-35
Table 7-10:	PID Loops.....	7-48

Section 8

Table 8-1:	DDE Server Address Structure	8-18
Table 8-2:	Client and server applications on a local Certified PC.....	8-19
Table 8-3:	Client and server applications for a remote Windows CE system	8-19
Table 8-4:	DDE Server Configuration Options	8-20

Section 10

Table 10-1:	Force Options	10-7
Table 10-2:	Forcing Data Value Options.....	10-8

B Index

A

Action block	4-3
Administrator privileges	8-23
Analog value scaling.....	7-32
Analog values	7-22, 7-30
AppTracker	
Chart tab	10-5
EasyTrac(tm).....	10-10
Forcing I/O	10-7
Introduction	10-3
Monitor and Edit Mode	10-13
Monitoring a timer	7-15
Statistics tab.....	10-6
Stepped flow chart execution	10-11
Watch windows	10-6
Array data type	7-7
Defining.....	7-9
Shift and rotate.....	7-24
Understanding.....	7-8
Axis data type	7-7

B

Begin block	5-27
Begin expression.....	5-28
Interaction with Call block	5-30
Bitwise operators	7-28
Block	
Comments.....	4-18
Multiple selection.....	4-13
Selecting	4-13
Boolean operators	7-28
Branching block	4-3
Breakpoints.....	10-11
Byte data type.....	7-7

C

Calculation block.....	5-5, 7-25
Expression	7-26
Using with String data items.....	7-35
Call block	5-7
Call expression.....	5-29
Interaction with Begin block	5-30
CE Watch.....	9-8, 9-9-9-12

Certified PC	1-8, 1-9, 9-5
Clocks, fixed	7-12
Color dialog box.....	6-21
Column labels.....	4-9
Comm data type	7-7
Command button	1-4
Communication block	5-9
Communications	8-3-8-25
Compare block.....	5-10
Editing.....	5-10
Summary.....	5-12
Configuring a Studio CE station.....	9-11
Configuring I/O.....	3-5
Connecting flow chart blocks	4-16
Connections.....	4-16
Multiple selection	4-13
Selecting	4-13
ConnectivityCenter	3-5
Adding a serial port.....	8-4
Adding the serial driver	8-3
Analog value mapping	7-30
Toolbar functions	3-14
Control block.....	5-13
Editing.....	5-14
Counting	7-15
Customer support	1-11
Cut-Copy-Paste	4-13

D

Data items	
Adding.....	7-4
Data types and formats.....	7-6
Deleting, find and replace	7-4
Exploring	7-3
Data types.....	7-6
Database operations.....	5-25, 7-37-7-46
Date and Time	7-11
DCOM.....	8-23
DDE server	
Addressing	8-18
Configuring.....	8-20
Introduction	8-16
Launching	8-19

Think & Do

Writing tagnames from Excel	8-22
Debugging projects	10-3–10-15
Simulation flow charts	10-13
Decision block	4-14, 5-14
Editing	5-15
Summary	5-15
Derived functions	7-29
Development tools	1-7
Devices	3-3
Dialog	1-4
Do Loop block	5-17
Drawing area	4-9, 6-5
Drawing page size	6-5
Drivers	3-3

E

EasyTrac(tm)	10-10
Editing flow chart block expressions	4-17
Enable block	4-13, 5-4
Error	
Error code interpretation	A-5
Runtime math	7-27
Excel, writing tagnames from	8-22
Explorer Bar	2-6

F

Fill Effects dialog box	6-21
Filldown I/O mapping feature	3-12
Find (String) function	7-35
Flag data type	7-7
Float data type	7-7
Flow chart	
Calculation block	5-5, 7-25
Call block	5-7
Communication block	5-9
Compare block	5-10
Connecting	4-16
Connecting blocks	4-16
Control block	5-13
Debugging	10-3
Decision block	5-14
Do Loop block	5-17
Editing block expressions	4-17
Enable block	4-13, 5-4
Execution path	10-3

Expressions	4-17
Freeze block	4-13
Motion block	5-18
Move block	5-21, 7-16–7-25
Order of execution	4-7
PID bLock	5-23
Run Program block	5-24
Simulation	4-8, 10-13
SQL Express block	5-25
Stepped execution	10-11
Unused	4-8
Wait block	5-26

FlowView

Orientation	4-9
Forcing I/O	10-7
Freeze block	4-13

Function

asc	7-35
chr	7-35
find	7-35
left	7-35, 7-36
len	7-35, 7-36
mid	7-35, 7-36
right	7-35, 7-36
str	7-35
val	7-35

Functions

abs	7-29
cos	7-29
cosecant	7-29
cotangent	7-29
exp	7-29
log	7-29
log10	7-29
pi	7-29
round	7-29
secant	7-29
sin	7-29
sqr	7-29
trunc	7-29

H

Help	1-5, 1-6
Using help	1-5
HMI screen. See Operator Screen	

I	
I/O configuration	3-5
Adding I/O driver	3-7
Filldown feature.....	3-12
Mapping I/O points.....	3-11
Saving	3-13
Scanning and monitoring	3-13
Index	3-5, 7-18
Initial tagname values	7-5
Input data type.....	7-7
Instant Recall	9-8, 9-12
Instr.....	7-37
K	
Keyboard commands.....	1-5
Keywords	1-4
L	
Left (String) function	7-36
Len (String) function	7-36
Local systems	1-10
M	
Mapping I/O points.....	3-11
Math Block	
Basic operators	7-27
Expression display conventions	7-26
Scientific functions	7-29
Math runtime errors	7-27
Menu selections.....	1-4
Menus	4-9
Mid (String) function.....	7-35
Monitor and Edit Mode.....	10-13
Motion block.....	5-18
Move Block	
Exchanging Strings with other types.....	7-35
Moving a number to discrete outputs	7-21
Moving analog values	7-22
Moving array elements.....	7-25
Moving bits and scalar values	7-19
Moving different data types	7-19
Moving discrete inputs to a number	7-20
Moving multiple data items.....	7-18
Moving ranges of String data items.....	7-35
Moving same data types	7-17
Moving with inverter order.....	7-18
Shifting and rotating arrays	7-24
Summary.....	7-23
Using with String data items	7-34
Move block.....	4-14, 5-21, 7-16–7-25
Multiple selection	4-13
N	
Naming a project.....	2-5
Note	
Editing.....	5-8
Entering.....	5-8
Number data type	7-7
Counting.....	7-15
O	
Object Finder dialog box.....	6-7
Online changes.....	10-13
OPC Explorer.....	8-12
OPC server	
Client applications	8-11
Configuring a project as client.....	8-14
Introduction	8-10
Opening a project	2-6
Operator screen	
Creating	2-12, 4-19
Duplicate, delete, or print	6-4
Exploring in ProjectCenter	6-3
Order of flow chart execution.....	4-7
Output data type	7-7
P	
Page grid	4-9
Parameter mapping	5-27
Pass by reference.....	5-30
Pass by value	5-30
Password	
CE Watch.....	9-12
Windows CE target file.....	9-8
PID block	5-23, 7-47–7-48
PID loop	
Features	7-48
PID block.....	5-23
Placing flow chart blocks	4-13
Pointer tool.....	4-16

Think & Do

Printer paper size.....	6-5
Process control.....	5-23, 7-47–7-48
ProjectCenter.....	2-3
Pulse Value.....	7-6

R

Remote communications.....	8-23
Remote systems.....	1-10
Remote target.....	1-10, 9-6
Retentive tagname values.....	7-5
Return block.....	5-27
Right (String) function.....	7-36
Rotating arrays.....	7-24
Row labels.....	4-9
Run Program block.....	5-24
Running a project.....	9-4
On remote target platforms.....	9-6
Runtime software.....	1-9
Runtime target.....	1-7
Windows CE.....	2-8

S

Scaling analog values.....	7-32
Scan time	
Adjusting the interval.....	9-7
Components.....	9-5
Maximum.....	9-7
Scan-based clocks.....	7-12
Scientific math functions.....	7-29
Screen aspect ratio.....	6-5
Scroll bars.....	4-10
Selecting blocks and connections.....	4-13
Serial communications.....	8-3
Testing.....	8-8
Serial driver.....	8-3
Serial port	
Mapping.....	8-8
Settings.....	8-6
Status data items.....	8-7
Shifting arrays.....	7-24
Simulation flow charts.....	10-13
SQL Express Block	
Database operations.....	7-37–7-46
SQL Express block.....	5-25
Starting a new project.....	2-3
Status line.....	4-10
Stepped flow chart execution.....	10-11

Stopping a running project.....	9-6
String data type.....	7-7
Comparing and sorting.....	7-37
Using in Calculation blocks.....	7-35
String handling.....	7-34
Structured data items	
Timer.....	7-13
Structured data types	
Comm data items.....	5-9
Subchart.....	4-4
Begin block.....	5-27
Call block.....	5-7
Design summary.....	5-31
Parameter mapping.....	5-27
Return block.....	5-27
Runtime characteristics.....	5-31
using in a project.....	5-32
Subcharts.....	5-3
System data items	
Descriptions.....	A-5
Monitoring with AppTracker.....	10-6
System data type.....	7-7, 7-11

T

Tags.....	3-3
Accessing.....	6-6
Creating.....	6-9
Creating and Using.....	6-5
Target systems.....	1-8
Technical Service.....	1-11
Technical support.....	1-11
Time and date.....	7-11
Timer data items.....	7-13
Timer data type.....	7-7
Timers, fixed.....	7-12
TND.....	6-8
Toolbar.....	4-9

W

Wait block.....	4-14, 5-26
Watch windows.....	10-6
Watchdog timeout.....	3-16
Windows CE.....	1-8
As embedded controller.....	1-9
Target options.....	9-8
WinPLC.....	9-5