Twido
TwidoSoft Operation Guide
Online Help

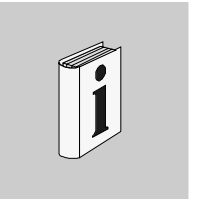
eng Version 3.2

a brand of Schneider Electric

Telemecanique

Table of Contents

About the Book



At a Glance

Document Scope	This is Online Help for TwidoSoft.
Validity Note	The information in this manual is applicable **only** for Twido programmable controllers.
Product Related Warnings	Schneider Electric assumes no responsibility for any errors that appear in this document. No part of this document may be reproduced in any form or means, including electronic, without prior written permission of Schneider Electric.
User Comments	We welcome your comments about this document. You can reach us by e-mail at techpub@schneider-electric.com

Introduction to TwidoSoft

1

At a Glance

Overview

This chapter provides an introduction to TwidoSoft - the software used to program and configure Twido programmable controllers. Instructions are also included for updating the operating system for your controller.

Note: For communication between TwidoSoft and the controller to function correctly, the TCP/IP protocol must be installed.

What's in this Chapter?

This chapter contains the following topics:

What is TwidoSoft?

Introduction

TwidoSoft is a graphical development environment for creating, configuring, and maintaining applications for Twido programmable controllers. TwidoSoft is a 32-bit Windows-based program for a personal computer (PC) running Microsoft Windows 98 Second Edition or Microsoft Windows 2000 Professional operating systems. Versions of TwidoSoft of v. 1.13 or later also run on Microsoft Windows XP.

Software Features

Main software features of TwidoSoft:
- Standard Windows Interface
- Application browser and multiple window views
- Programming and configuration support
- Communication with controller

Standard Windows Interface

Key standard Windows features:
- Easy use of keyboard or mouse
- Dockable windows and toolbars
- Standard menu organization
- ToolTips, status bar, and shortcut menus
- Online help including context-sensitive help

Programming and Configuration

Major programming and configuration features:
- Reversible ladder and list programming
- Two-step, point and click ladder programming
- Offline and online programming
- Program and/or data animation
- Easy configuration by using the Application Browser
- Editors for main programming and configuration features
- Cut, copy, and paste program editing
- Symbolic programming
- Cross references
- Printouts of programs and configuration

Controller Communication and Control

Main TwidoSoft features for controller support:
- Connecting and disconnecting a controller
- Operating the controller
- Monitoring application use of memory by the Resource Monitor
- Downloading and uploading controller programs
- Backing up controller programs to optional EEPROM

Additional Information

Refer to the following for additional information:

- For information on standard Windows interface features, refer to Microsoft Windows documentation and help files.
- For information on Twido controller hardware and programming using the List and Ladder languages, see the TwidoSoft User Reference Manual.
- For context sensitive help, click on the **Help** buttons in dialog boxes.

Conventions

The following typographic conventions are used in this operation guide.

Format	Represents
bold	For user input, enter words or phrases shown in **bold** exactly as they appear. Menu names and options, commands and toolbar names, and dialog box names and options are also shown in **bold** type.
UPPER CASE	Keyboard names, combinations, and sequences are shown in all uppercase letters. For example, the keyboard shortcut for creating a new application is CTRL+N. To perform this shortcut, press and hold the CTRL key and then press the N key.
File → **Open**	The arrow indicates a menu selection. In this instance, go to the **File** menu to select the **Open** command.

Connecting a PC to a Twido controller

Introduction

A communications cable is required to connect a PC to a Twido controller. Connections can be made in any of three ways. Either:

- using special multi-function cables for connecting the COM or USB serial port on the PC to Port 1 on the controller:
  - The TSX PCX 1031 (2.5m) cable converts signals between RS-485 and RS-232 devices,
  - The TSX PCX 3030 (2.5m) cable converts signals between RS-485 and USB devices.
- using a telephone connection linking a receiving modem connected to port 1 on the controller and a modem connected to the COM serial port on the PC (or internal modem to the PC).
- or, if your Twido controller is Ethernet-capable (such as the TWDLCAE40DRF Compact controller), using a SFTP Cat5 RJ45 Ethernet cable to connect both PC and controller RJ-45 ports to an Ethernet hub/switch.

Port 1 for all Twido controllers is a built-in RS-485 port, which must be used to communicate to the TwidoSoft programming software. When the cable is connected to Port 1, protocol is automatically set for TwidoSoft communications (see *Using the Communications Cable, p. 159*).

Note: TSX PCX 1031 and TSX PCX 3030 communication cables are equipped with a four-position rotary switch allowing the cable to be used for different modes. Set the switch to the position labeled **2** for TwidoSoft communications.

⚠ CAUTION

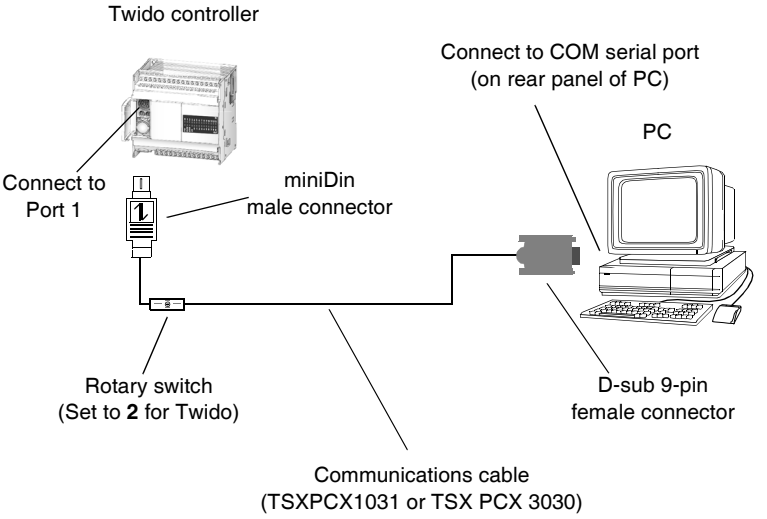
POSSIBLE ELECTRICAL DAMAGE TO CONTROLLER COMPONENTS.

Do not connect the communications cable to the controller before connecting it to the PC. Always connect the cable to the PC first.

Failure to follow this instruction can result in injury or equipment damage.

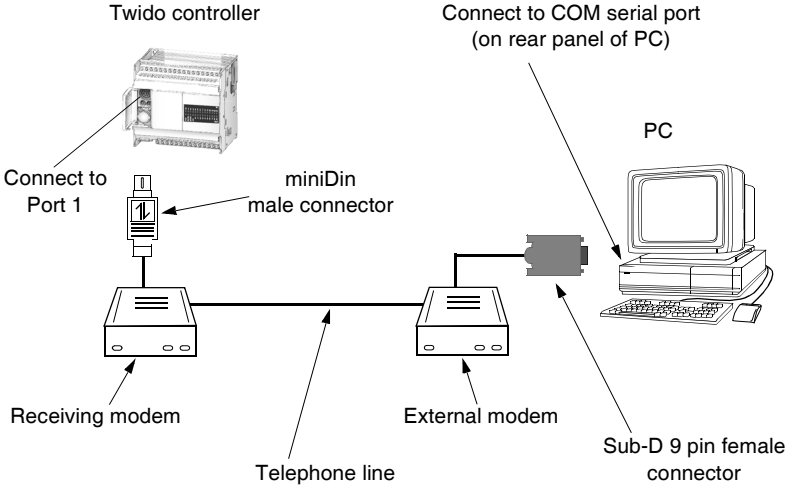
Connecting the TSX PCX 1031 or TSX PCX 3030 Communication Cable

The following diagram shows an example of the communications cable connecting a PC to a Twido controller. The location of Port 1 depends on the model of the Twido controller. Consult your PC documentation for details on using COM serial ports.



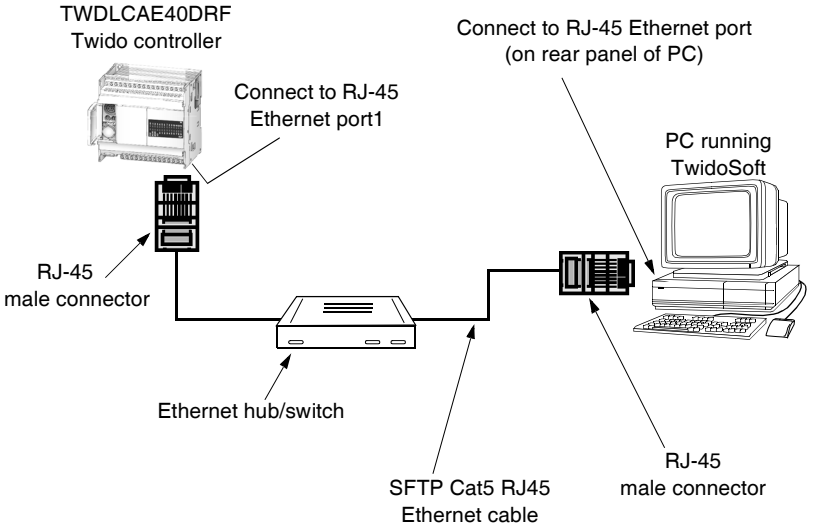
Telephone Line Connection

The following diagram shows an example of a telephone connection between the modems of a PC and a Twido controller. The location of Port 1 depends on the model of the Twido controller. Consult your PC documentation for details on using modems.



Connection via an Ethernet Network

The following diagram shows an example of a onnection between the PC running the TwidoSoft application program and a Twido controller via an Ethernet network hub/switch. This type of connection is feasible only with an Ethernet-capable Twido controller, such as the TWDLCAE40DRF Compact controller.



Updating the Operating System

Overview

The **Twido Windows Executive Loader Wizard** is a Windows-based program that guides you through the steps needed to update the **Firmware Executive** in your Schneider Electric - Telemecanique Twido programmable controller. The Firmware Executive is the operating system that runs your applications and manages controller operation.
The wizard provides an intuitive point-and-click interface that progressively steps you through the updating procedure to quickly download the latest version of the executive for your controller.
See the Executive Loader Wizard User Guide for detailed instructions. This guide can be accessed using the Exec Loader utility.

Opening the Exec Loader Wizard

You can open the Executive Loader Wizard from TwidoSoft by selecting **Tools** → **Update Controller Firmware** from the main menu, or by double-clicking on the program to open it directly. The wizard program, TWEL.EXE, is included in the TwidoSoft installation and can be found in the \bin sub-directory of the default installation directory.

Basic Steps

There are four steps required to upgrade your executive. The wizard provides a screen for each step and online help is available for each screen. The following table summarizes each step:

Step	Screen	Function
1	Welcome	Introduces the Executive Loader Wizard.
2	Parameters	Selects the correct executive file to transfer to your controller, and selects the PC serial port to communicate with the controller.
3	File and Device Properties	Compares the hardware IDs and the executive version information for the executive file and the controller.
4	Transfer Progress	Monitors the transfer of the executive file to the controller.

Supported Twido Controllers

The following table lists all Twido devices that are supported by the Executive Loader Wizard program:

Controller Module	Description
TWDLCAA10DRF	Compact base unit, 230V AC(1), with 6 In (24V DC), 4 Out (2A Relays). Screw terminal blocks, non-removable.
TWDLCDA10DRF	Compact base unit, 24V DC, with 6 In (24V DC), 4 Out (2A Relays). Screw terminal blocks, non-removable.
TWDLCAA16DRF	Compact base unit, 230V AC(1), with 9 In (24V DC), 7 Out (2A Relays). Screw terminal blocks, non-removable.
TWDLCDA16DRF	Compact base unit, 24V DC, with 9 In (24V DC), 7 Out (2A Relays). Screw terminal blocks, non-removable.
TWDLCAA24DRF	Compact base unit, 230V AC(1), with 14 In (24V DC), 10 Out (2A Relays). Screw terminal blocks, non-removable.
TWDLCDA24DRF	Compact base unit, 24V DC, with 14 In (24V DC), 10 Out (2A Relays). Screw terminal blocks, non-removable.
TWDLCAA40DRF	Compact base Unit, 230V AC(1), with 24In (24VDC), 2 Transistor(24V DC), 14 Relays (2A) Out, Embedded RTC, Screw terminal block, non-removable.
TWDLCAE40DRF	Compact base Unit, 230V AC(1), with 24In (24VDC), 2 Transistor(24V DC), 14 Relays (2A) Out, Embedded RTC, 100Base Tx Ethernet, Screw terminal block, non-removable.
TWDLMDA20DUK	Modular base unit with 12 In (24V DC), 8 Out (0.3A Transistors, Sink). MIL connectors(2), removable.
TWDLMDA20DTK	Modular base unit with 12 In (24V DC), 8 Out (0.3A Transistors, Source). MIL connectors(2), removable.
TWDLMDA20DRT	Modular base unit with 12 In (24V DC), 8 Out (2 Out 0.3A Transistors, Source and 6 Out 2A Relays). Screw terminal blocks, removable.
TWDLMDA40DUK	Modular base controller 24 inputs (24V DC), 16 sink transistor outputs (0.3 A)). Removable MIL connectors(2).
TWDLMDA40DTK	Modular base unit with 24 In (24V DC), 16 Out (0.3A Transistors, Source). MIL connectors(2), removable.
Note 1: Products described as 230 VAC power supply are 120/230 VAC compatible. **Note 2: MIL connectors are also known as HE10 connectors.**	

Software Features

2

At a Glance

Overview

This chapter provides details on using the software features of TwidoSoft.

What's in this Chapter?

This chapter contains the following sections:

2.1 User Interface

At a Glance

Overview

This section provides an introduction to the main components of the TwidoSoft user interface.

What's in this Section?

This section contains the following topics:

Key Concepts

Introduction

TwidoSoft provides an intuitive Windows-based user interface consisting of standard Windows features including ToolTips and online help.

Key TwidoSoft Concepts

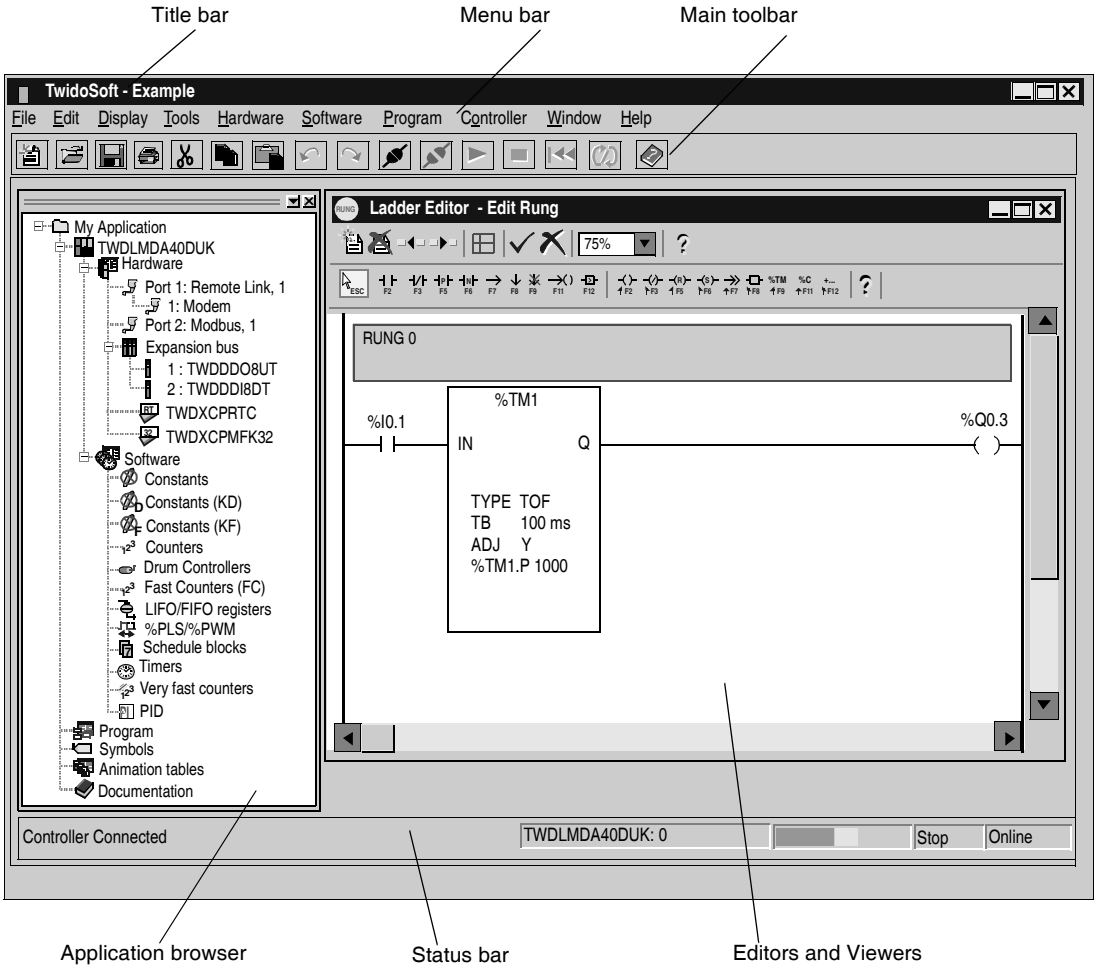
The following are key concepts for using TwidoSoft.

- Application browser
  Use the Application Browser to view, configure, program, and maintain an application. Conveniently perform the same commands that are available from menus. Configure hardware using a graphical representation of controllers, expansion I/O, and options.
  See *Application Browser, p. 27*.
- Memory allocation
  View the status bar memory usage indicator (visible in the memory editor) to determine the percentage of total memory used by a program. A warning is provided when available memory is getting low. See *Status Bar, p. 30*.
- Online and offline operations
  In offline operation, TwidoSoft is not connected to the controller and changes can only be made to the application in the PC memory. Use offline operation to develop and configure an application.
  In online operation, TwidoSoft is connected directly to the controller and changes can be made to the application in the controller memory. Use online operation to run the controller and debug and modify an application.
  See *Offline and Online Operation, p. 58*.
- Application protection
  An application can be protected to prevent unauthorized viewing, changing, or duplication. See *Security Features, p. 66*.

Main Window

Introduction The TwidoSoft **Main Window** provides easy access to menus and commands, windows and toolbars, and views of an application.

Main Window The following illustration shows the TwidoSoft Main Window.



Description of Main Window

The following table describes the above components of the Main Window.

Component	Description
Title bar	Displays TwidoSoft application icon and title, application path and file name, and the current editor or viewer if maximized in display area.
Menu bar	This is the main menu displayed in the form of a horizontal bar near the top of the main window containing the names of TwidoSoft menus.
Main toolbar	Panel beneath menu bar containing buttons for frequently used menu commands.
Application browser	Provides a convenient, tree-like view of the structure of an application.
Editors and Viewers	Editors and viewers are TwidoSoft windows that organize programming and configuration controls for efficient development of applications.
Status bar	Displays information about the application, the controller, and TwidoSoft.

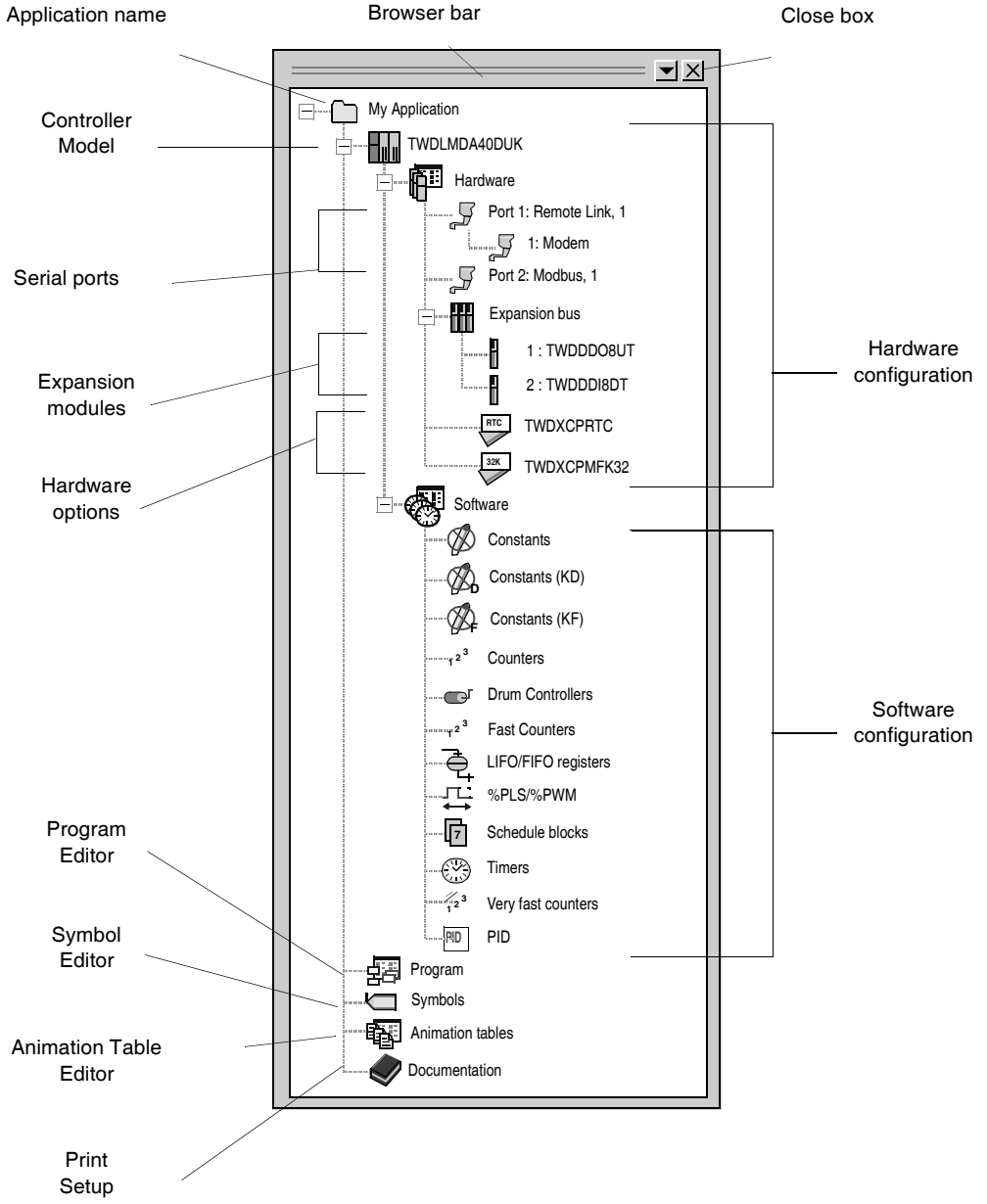
Application Browser

Introduction

The **Application Browser** is a dockable window that provides a tree view of an application. Windows and toolbars that are dockable can be moved around and attached to the borders of a parent window. Elements of an application appear in a logical hierarchy based on their relationship in the application. The elements are organized as an indented outline that can be expanded or collapsed. The Application Browser conveniently organizes items in an application so that you do not have to search for these items individually.

Application Browser Window

The following illustration is an example of the Application Browser. Selections will vary for each application depending on configuration choices.



Description of Application Browser

The following table describes the components of the Application Browser.

Component	Description
Application name	This is the top most item in the tree structure. Displays the name of the open application.
Controller Model	The model number of the base controller.
Hardware configuration	Displays the hardware configuration for the base controller: communications setup, expansion modules, and hardware options. Right-clicking on hardware items displays shortcut menus for commands on the **Hardware** menu.
Software configuration	Displays configuration of software resources such as timers and counters. Right-clicking on software items displays shortcut menus for some commands on the **Software** menu.
Program	Opens the default program editor (see *Preferences, p. 31*). Right-clicking displays a shortcut menu for some commands on the **Program** menu.
Symbol	Opens the Symbol Editor.
Animation	Opens the Animation Tables Editor.
Documentation	Opens the Print Setup dialog box to configure printing options.
Browser bar	Click and hold the left mouse button to move the Application Browser in the display area. Double-click to reduce the browser to a small, floating window.
Close box	Click to close the Application Browser.

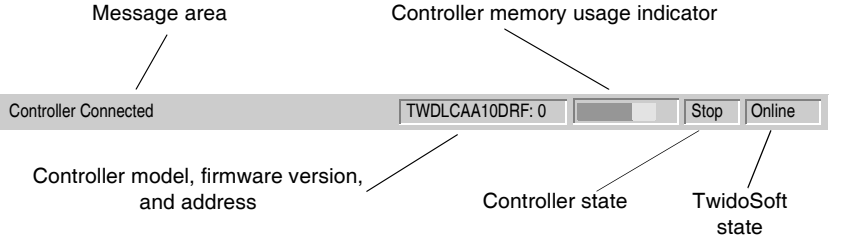
Status Bar

Introduction

The **Status Bar** is a panel at the bottom of the main window that displays information about the application, the controller, and TwidoSoft. The Status Bar can be turned on or off by selecting **Display** → **Status Bar** from the main menu bar.

Status bar

The following illustration shows the TwidoSoft Status Bar.



Description of Status Bar

The following table describes the components of the Status Bar.

Component	Description
Message area	Displays status messages and prompts.
Controller model, firmware version, and address	If TwidoSoft is online: displays controller model number, version number of the firmware (version and patch numbers), and the controller address.
Controller memory usage indicator. (double click to access the memory viewer)	Displays a graphical indication of controller memory usage. The indicator bar moves from left to right to indicate the amount of controller memory that is being used by an application. The color of the bar changes to warn of scarce memory resources. When the bar fills more than 50% of the available space, the color changes from green to yellow. When the bar fills more than 75% of the available space, the color changes to red.
Controller state	Displays controller state: run, stop, or error. TwidoSoft must be online.
TwidoSoft state	Displays TwidoSoft state: initial, offline, online, or monitor.

Preferences

Introduction

Use the **Preferences** dialog box to select user preferences for using Ladder and List editors and viewing toolbars.

Displaying Preferences

With an open application, click **File** → **Preferences** from the main menu bar to display the **Preferences** dialog box.

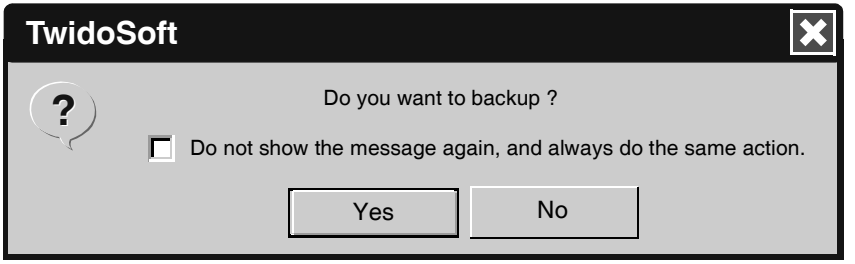
Selecting Preferences

The following table describes the options available for user preferences.

Option	Description
Default Program Editor	Select the List or Ladder editor as the program editor to display when creating a new application.
Ladder Information	Select a format for ladder entries.
List/Ladder Animation	Select Hexadecimal or Decimal format for display of data when animating a program.
Display Attributes	Select the default attribute, symbols or addresses, for display of variables in List and Ladder editors.
Automatic save	Select it to have a backup done automatically every time you disconnect. **Note:** An automatic backup is always done at connection.
Save message	After you have done some online modifications and you want to do a request, to disconnect or to close TwidoSoft, a saving message window pops up if you select **Save message**.
Close Ladder viewer on Edit Rung	Select it to close Ladder viewer when editing a rung.
Display Toolbars	Select it to display toolbars for editors.
Auto Line Validate	Select to fix errors before leaving a List instruction. With Auto Line Validate selected, any errors in an instruction will stop the List editor from moving to another instruction until the errors are fixed. If Auto Line Validate is not selected, you can leave a line with errors to create or edit other lines. A question mark (?) is inserted at the front of a line indicating that it contains a program error.
Automatic validation of configuration editor	Select it to validate automatically the operations done in the Configuration Editor, whenever necessary.
Connections management	Select it to open a window and create, delete or modify connection parameters.
Connection	Select from one of the connections previously configured on the serial port to be used to connect the PC to controller.

Automatic Saving

Combining **Automatic save** and **Save message** options affects the following saving message window:



When you have done online modification(s):

If...	**Then ...**
Automatic save and **Save message** are both checked	The saving message window pops up and default button associated to "**Do you want to backup?**" question is **Yes**. If you check "**Do not show the message again, and always do the same action**", the saving window will no longer pop up. Automatic backup is done at each disconnection.
Only **Automatic save** is checked	The saving message window does not pop up. An automatic backup will be done before disconnection.
Only **Save message** is checked	The saving message window pops up and default button associated to "**Do you want to backup?**" question is **No**. If you check "**Do not show the message again, and always do the same action**", the saving window will no longer pop up. No automatic backup is done at disconnection.
Automatic save and **Save message** are both unchecked	The saving message window does not pop up. No automatic backup is done at disconnection.

2.2 Editors and Viewers

At a Glance

Overview

This section is an introduction to the editors and viewers used to create, modify, and debug TwidoSoft applications.

What's in this Section?

This section contains the following topics:

Editors and Viewers

Introduction

TwidoSoft provides specialized windows called editors and viewers for performing important tasks in developing an application. A TwidoSoft **application** consists of a program, configuration data, symbols, and documentation. These components can be used in any order when creating an application. For example, an application can be developed in the following order:

- First, define configuration data,
- Next, define symbols,
- Then, write program instructions,
- Finally, add comments and print documentation.

This is just one method to organize these tasks; these could just as easily been done in reverse order. Developing each part of an application using separate editors makes the application development process more systematic, resulting in more clearly defined applications.

List and Ladder Editors

Use the *List Editor, p. 43* and the *Ladder Editor, p. 38* to develop the main program which is the core of an application. The development of the main program includes creating the order in which the program is executed.

Selecting List or Ladder

You can write a controller program in either List or Ladder format. Selecting one method over the other is a matter of preference and does not affect the application.

- Ladder language consists of diagrams that use both graphics and text.
- List language is a text-based instruction set.

In either list or ladder language, you write the program in the logical order required to control a machine or process. You can automatically convert or **reverse** ladder instructions to list instructions. You can also reverse list to ladder instructions if the program structure and instructions comply with reversibility rules. See *Reversing a Program, p. 84*.

In either language, document your program by using comments in both the List and Ladder editors. Comments consist of text that you insert into instructions to document the meaning and purpose of the program.

Animation Tables Editor

Use the *Animation Tables Editor, p. 50* to create and save animation tables. A animation table lists all or part of the data variables used in the main program, and consists of addresses of variables, current and retained values, and associated symbols. This information is useful for adjusting and debugging an application. See *Animating a Table, p. 349*.

Configuration editor	Use the *Configuration Editor, p. 44* to assign specific values to the hardware and software resources of the controller to control these resources, such as timers, counters, latched inputs, external run/stop switches, and so on. These assigned values are called configuration data. See *Hardware and Software Configuration , p. 127*.
Symbol Editor	Use the *Symbol Editor, p. 45* to document a program by assigning tag words (symbols) to the data variables used in the program. Symbols are easier to remember than the memory addresses of variables. Use the Symbol Editor to define and edit symbols. See *Defining Symbols, p. 81*.
Cross References Viewer	The *Cross References Viewer, p. 49* provides a convenient list of application elements: operands, symbols, rung or line numbers, and operators. Cross references identify program elements used in an application, and where they are used. During troubleshooting or debugging an application, these items can be easily located and cross-referenced to other elements in the program without searching through the entire program. See *Cross Referencing an Application, p. 113*.
Program Error Viewer	The *Program Error Viewer, p. 48* displays the results of the program analysis of an application. The following information is provided for each error: type of error message (error or warning), number of rung or line, and description of error. See *Analyzing a Program, p. 79*.

Ladder Viewer

Introduction

The **Ladder Viewer** window provides a graphical view of a ladder program and its component rungs. Use it to scroll through and view a ladder program. To edit or modify rungs, use the Ladder Editor.
See *Programming in Ladder Language , p. 241*.

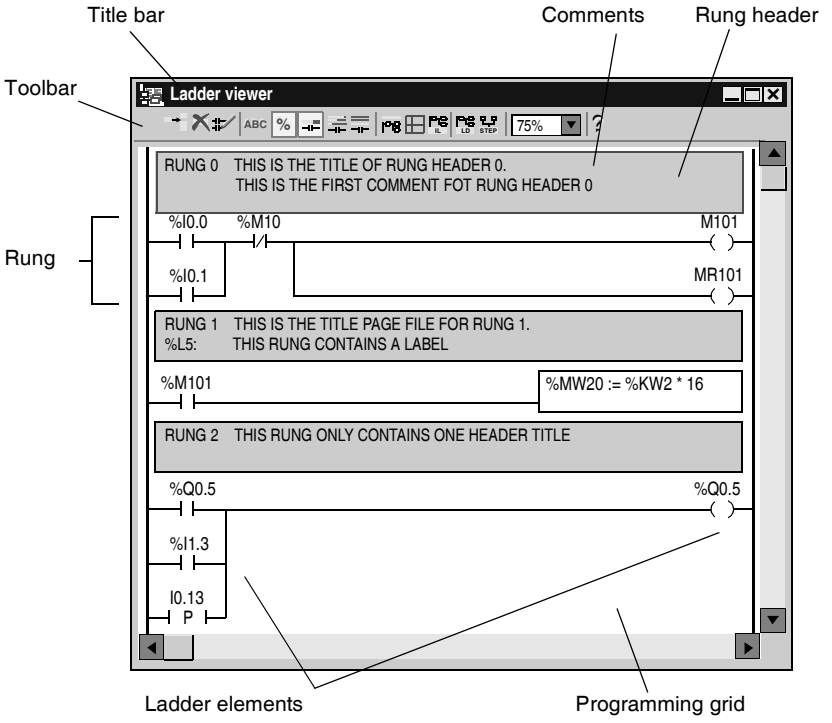
Available Commands

Functions available in the Ladder Viewer:

- View a ladder program.
- Customize the view of ladder diagrams.
- Open the Ladder Editor to create and edit rungs.
- Open the List Rung Editor to edit a non-reversible list instruction.
- Insert or delete a rung.

Ladder Viewer Window

The following is an example of the Ladder Viewer window.



Parts of the Ladder Viewer

The following table describes the parts of the Ladder Viewer:

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer.
Toolbar	A bar below the title bar that displays frequently used commands for the Ladder Viewer.
Rung	Panel that contains graphic elements and connections for a single ladder rung.
Rung header	A panel located immediately above a rung which identifies the rung and can include user comments.
Programming grid	Each rung consists of a matrix of 7 rows by 11 columns of cells. Each cell can contain a ladder element. Displayed area is from the first row to the last row in which there are ladder elements. Double-click on an empty cell to open the Ladder Editor. Double-click on a ladder element to edit the properties for that element.
Ladder elements	Symbols for functions in ladder programs such as, coils, contacts, and function blocks. When selected in the programming grid, elements are surrounded by a red rectangle.

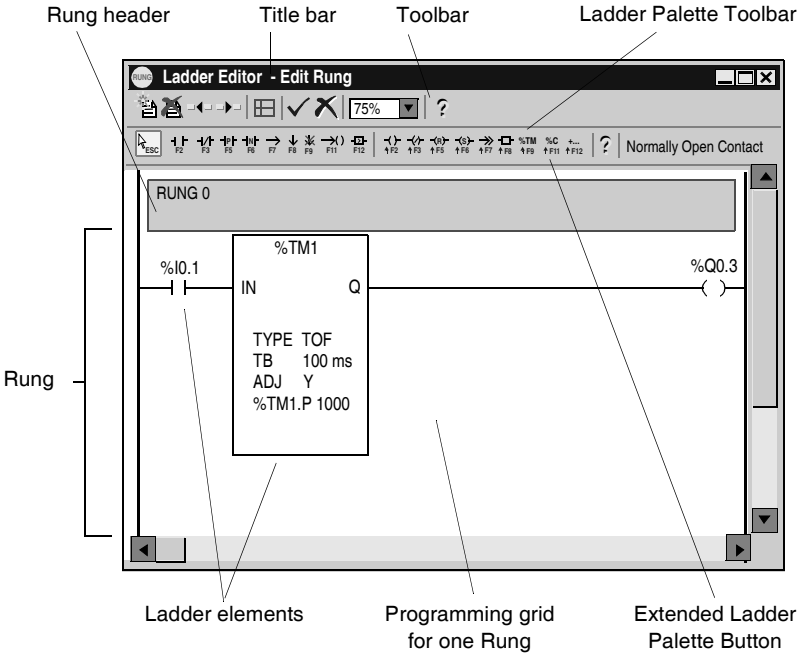
Ladder Editor

Introduction

The **Ladder Editor** is a graphics-based, program editor used to create and edit ladder diagrams.
See *Programming in Ladder Language , p. 241*.

Ladder Editor Window

The following is an example of the Ladder Editor window.



Parts of the Ladder Editor

The following table describes parts of the Ladder Editor.

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer. Indicates if editing or inserting a rung.
Toolbar	A bar below the title bar that displays buttons for frequently used commands.
Ladder Palette Toolbar	A bar beneath the toolbar that displays buttons for most commonly used ladder elements. A single left click on a symbol selects the element, then right-click in a cell to insert the element.
Extended Ladder Palette Button	A special selection on the instruction bar that opens a dialog box with additional options for special contacts, function blocks, and special coils. See *Extended Ladder Palette, p. 40*.
Rung	Panel that contains graphic elements and connections for a single ladder rung. The Ladder Editor displays the programming grid for only one rung at a time.
Rung header	A panel located immediately above a rung which identifies the rung and can include user comments.
Programming grid	Each rung consists of a matrix of 7 rows by 11 columns of cells. Each cell can contain a ladder element. Double-click on a ladder element to edit the properties for that element.
Ladder elements	Symbols for functions in ladder programs such as coils, contacts, and function blocks. When selected in the programming grid, elements are surrounded by a red rectangle.

Extended Ladder Palette

Introduction

The **Extended Ladder Palette** dialog box provides additional ladder elements that are not available from the Ladder Palette Bar. This dialog box is selected by clicking the button just next to the help button on the Ladder Palette Bar (Maj+F12). The ladder elements are organized into three groups in the dialog box:

- Special contacts
- Function blocks
- Special coils

See *Using the Extended Ladder Palette, p. 259*.

Making Selections

To make selections from the Extended Ladder Palette, left-click on an item and then left-click again in a cell in the programming grid to insert the element.

Rung Header

Introduction

Use the **Rung Header** dialog box to edit a Rung Header. A Rung Header appears directly above a rung and documents the purpose of the rung in a ladder diagram. See *Using Rung Headers, p. 255* and *Programming in Ladder Language , p. 241*.

Opening the Rung Header Dialog Box

To open a Rung Header dialog box, double-click on the Rung Header in the Ladder Viewer or double-click or right-click on the Rung Header in the Ladder Editor. If Rung Headers are not visible when using the Ladder Viewer, select **Tools** → **Toggle Rung Headers** from the main menu. Rung Headers are always visible when using the Ladder Editor.

Display Options

The following options can be selected to display in the rung header. Enter selectable numbers in the **Label/Subroutine/Step Number** box:

- **Standard**
  Select to display the number of the rung in the ladder program (for example, Rung 1). The number is automatically determined by the numeric order of the rung in the program.
- **Label %Li:**
  Select to display a label number in the rung header (for example, %L1:). This number is selectable.
- **Subroutine SRi:**
  Select to display a subroutine number in the rung header (for example, SR10:). This number is selectable.
- **Initial Step =*=i**
  Select to display the Initial Step of a Grafcet program in the rung header (for example, =*=10). This number is selectable.
- **Begin Step -*-i**
  Select to display the Begin Step of a Grafcet program in the rung header (for example, -*-7). This number is selectable.
- **=*= POST**
  Select to display the Post-Processing Step of a Grafcet program in the rung header.

Label/ Subroutine/Step Number

Enter a number for the selected display option. Rung numbering is automatic when selecting **Standard**.

Title

Enter one line of text for the name or description of the rung. Can consist of 1 to 122 characters.

Comment

Enter up to four lines of text for comments about the purpose of the rung. Each line can consist of 0 to 122 characters.

List Rung Editor

Introduction

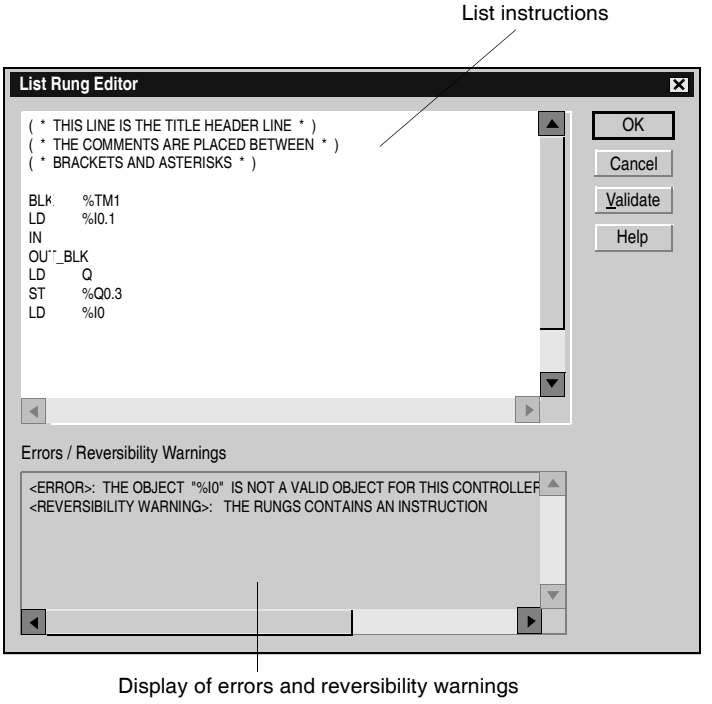
The **List Rung Editor** is a secondary program editor opened from the Ladder Viewer to edit rungs that could not be reversed to Ladder language.
This dialog box consists of two sections:

- The upper area contains the list program segment that could not be reversed to ladder.
- The lower area contains program errors and reversibility warnings.

See *Programming in Ladder Language , p. 241*.

List Rung Editor Window

The following is an example of the List Rung Editor window.



Analyzing Modifications

After making changes, press **Analyze** to check the list program segment and update the **Errors/Reversibility Warnings** section. Press **OK** to accept modifications, close the dialog box, and return to the Ladder Viewer.

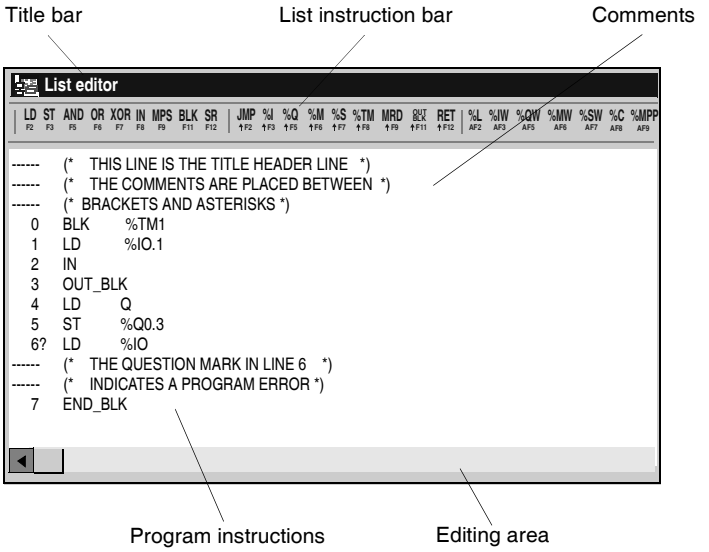
List Editor

Introduction

The **List Editor** is a simple line editor for creating and editing instruction list programs in List language.
See *Programming in List Language , p. 289*.

List Editor Window

The following is an example of the List Editor window.



Parts of the List Editor

The following table describes parts of the List Editor.

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor.
List instruction bar	A bar beneath the title bar that displays symbols and associated keyboard shortcuts for instructions. A single left click on a symbol inserts the selected instruction at the location of the cursor in the editing area.
Editing area	Contains instructions and comments. Instructions are entered and modified here.
Program instructions	Line of instruction includes a line number, instruction code, and operand.
Comments	Text you enter to document the program. Comments must be inserted within parenthesis and asterisks such as: (* comments go here *).

Configuration Editor

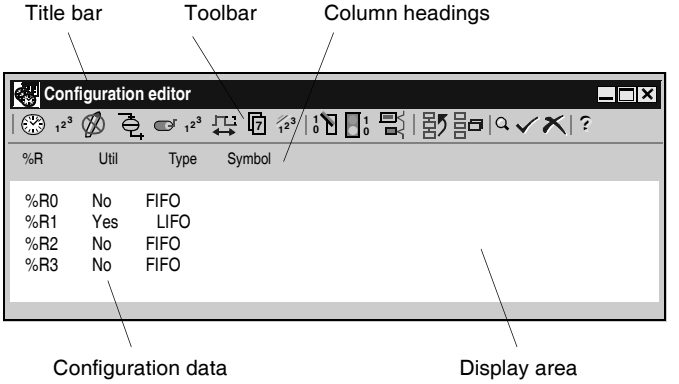
Introduction

Use the **Configuration Editor** to view and assign configuration values for software resources and some hardware resources (see the Configuration Editor toolbar to identify which resources can be configured.)

- For configuring hardware not available from the toolbar, see *Configuring Hardware, p. 136*.
- See *Using the Configuration Editor, p. 132*.

Configuration Editor Window

The following is an example of the Configuration Editor window.



Parts of the Configuration Editor

The following table describes parts of the Configuration Editor.

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer.
Toolbar	A bar beneath the main menu bar that displays buttons for selecting software and hardware resources. Allows switching between configuring resources without closing the Configuration Editor.
Configuration data	Displays current configuration for all available resource entries. Data is derived from the configuration dialog box for each resource.
Display area	Configuration data is displayed here.
Column heading	A panel above the configuration data with headings identifying the associated column of data. Some headings are displayed as part of the configuration data. The **Used?**column identifies which resources are used in the current application.

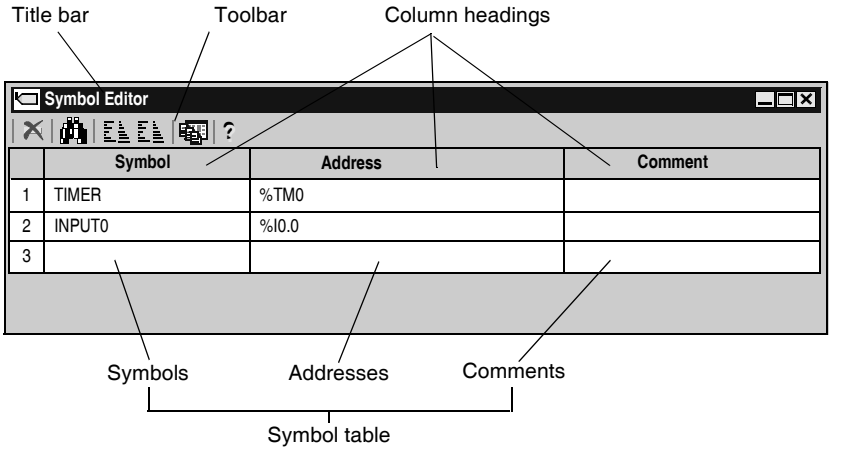
Symbol Editor

Introduction

Use the **Symbol Editor** to create and manage symbols used in a program. The Symbol Editor can be viewed in both online and offline modes, but it is not available in the monitor state. The Symbol Table consists of columns for Symbols, Comments, and Addresses, organized like a spreadsheet with rows for defining individual symbols.
See *Defining Symbols, p. 81*.

Symbol Editor Window

The following is an example of the Symbol Editor window.



Parts of the Symbol Editor

The following table describes the parts of the Symbol Editor.

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer.
Toolbar	Displays symbols for frequently used commands.
Symbol table	Consists of Symbol, Comment, and Address columns. Displays information for currently assigned symbols.
Column headings	A panel above the symbol data with headings identifying the associated column of data.
Address	Column containing addresses of variables for assigned symbols.
Symbol	Column containing the alphanumeric characters that describe a variable.
Comment	Displays text you enter to describe a symbol.

Find Symbol

Introduction

Use the **Find** dialog box to locate a symbol or address in the symbol table of the Symbol Editor. This dialog box is opened from the *Symbol Editor, p. 45*.

Finding an Address or Symbol

To find an address or symbol:

Step	Action
1	Enter the symbol or address in the **Find** box.
2	Select **OK** to close the box.
3	If found, the entry will be highlighted in the symbol table.
4	If you have searched for an address and it is not found, a warning box is displayed asking if you would like to create the address. Click on **OK** to open the *Object Browser, p. 47* which you can use to create the address.
5	If searching for a symbol and it is not found, a warning box is displayed asking if you would like to create the symbol. Select **OK** to open the Object Browser. See *Defining Symbols, p. 81*.

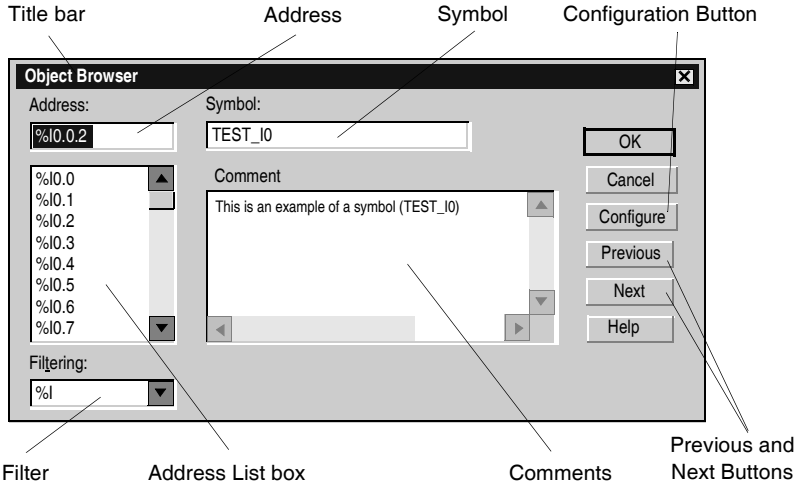
Object Browser

Introduction

The **Object Browser** dialog box is used to insert or edit symbols when using the Symbol Editor.
See *Defining Symbols, p. 81*.

Object Browser Dialog Box

The following screen is an example of the Object Browser dialog box.



Parts of the Object Browser

The following table describes parts of the Object Browser.

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer.
Address	Unique identifier of a controller variable. All addresses start with a percent (%) character.
Symbol	An alphanumeric name associated with the corresponding address. A symbol can use a maximum of 32 characters and can only consist of the following characters: 0 to 9, A to Z, and underscores (_).
Filter	Select from this list of all variables to display a specific type of variable.
Address List box	Displays all instances of variable type selected in the **Filtering** box.
Comments	Optional text to provide a more detailed description of the symbol.
Configuration Button	Click to open the configuration dialog box for the selected variable.
Previous/Next Buttons	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box.

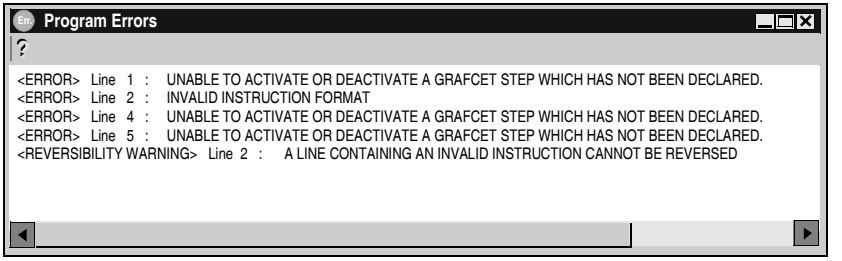
Program Error Viewer

Introduction

The **Program Errors Viewer** window displays program errors and warnings when analyzing a program by selecting **Program** → **Analyze Program** from the main menu.
See *Analyzing a Program, p. 79*.

Program Error Viewer Window

The following is an example of the Program Errors Viewer window.



Types of Errors

There two categories of errors reported in the Program Error Viewer window:

Part	Description
Errors	Indicates a problem in the application that prevents creation of an executable program.
Reversibility warning	Reports on instructions that are not reversible or may cause operational uncertainties.

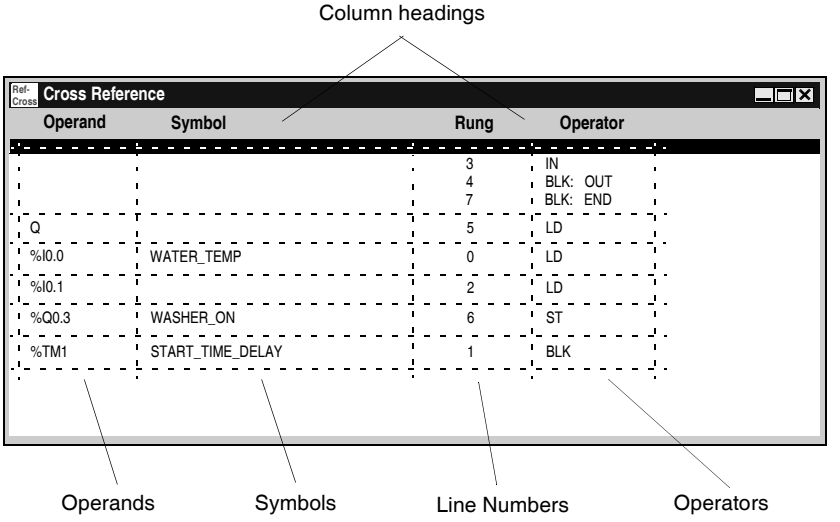
Cross References Viewer

Introduction

The **Cross References Viewer** displays a list of operands, symbols, line or rung numbers, and operators. This simplifies locating these items in the application for troubleshooting and debugging.
See *Cross Referencing an Application, p. 113*.

Cross References Viewer

The following is an example of the Cross References Viewer window.



Parts of the Cross Reference Viewer

The following table describes the parts of the Cross References Viewer.

Part	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer.
Column headings	A panel above the display area with headings identifying the associated column of data.
Operands	Addresses of variables.
Symbols	Alphanumeric characters that describe a variable.
Lines	Line or rung numbers of the item in a program.
Operators	Indicate the operation to be performed.

Animation Tables Editor

Introduction

Use the **Animation Tables Editor** to view values of variables when a PC is connected to a controller which is either running or stopped. Animating variables is useful for *Debugging and Adjusting a Program Online , p. 333*. You can view changes in variables as a program runs to compare the actual values against expected values.
See *Using the Animation Tables Editor, p. 338*.

Using Animation Tables

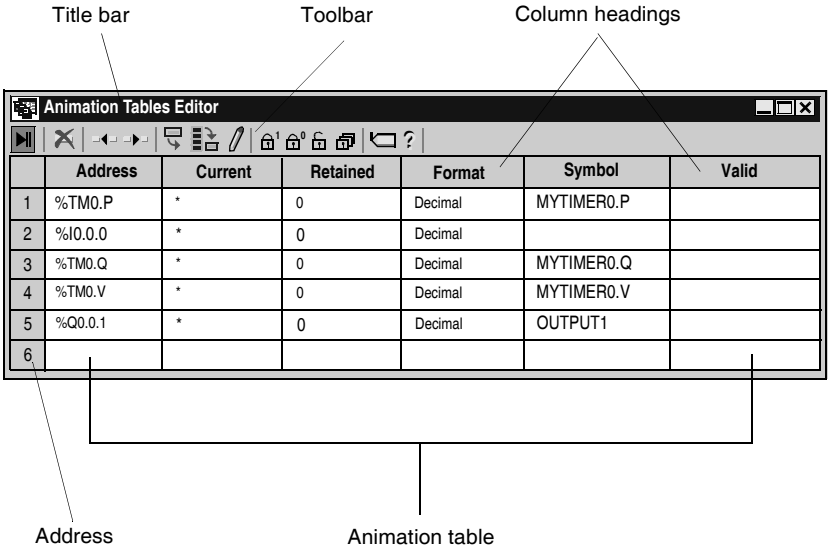
Use the Animation Tables Editor to:

- Monitor the current value of any variable in the controller.
- Force inputs and outputs.
- Change the value of variables independent of calculated values, until they are updated again.
- Backup the current values of all variables in an animation table.
- Restore the retained values of all variables in an animation table to the controller.

See *Defining Symbols, p. 81*.

Animation Tables Editor Window

The following is an example of the Animation Tables Editor window.



Parts of the Animation Table Editor

The following table describes the parts of the Animation Tables Editor.

Item	Description
Title bar	A bar at the top of the window that displays the name of the editor or viewer.
Toolbar	Displays buttons for frequently used commands.
Animation table	Consists of data in the editor: Address, Current, Retained, Format, Symbol, and Valid.
Column headings	A panel above the table with headings identifying the associated column of data.
Address	Unique identifier of a variable always preceded with a percent symbol (%).
Current	During animation, displays the Current value of the corresponding variable.
Retained	Retained value to be written to the controller when a Write retained values action is performed.
Format	Identifies the number format of the data variable. Options are Decimal, Hexadecimal, Binary, Floating Point or ASCII.
Symbol	Alphanumeric characters associated with the address, if any.
Valid	Indicates if the variable or object has been validated. If not validated, the variable has not been allocated memory space in the controller and cannot be animated. A green check mark indicates a valid variable or object that is part of the currently opened application, while a red X identifies a variable or object as not valid and is not part of the currently opened application.

Memory editor

Introduction

The memory editor enables you to view and optimize the memory resources of the PLC.
It is composed of two tabs:

- The **View** (See *Memory editor: Display Tab, p. 53*) tab, which indicates memory usage concerning:
  - the PLC's memory usage (data, program, configuration and system),
  - the memory distribution of the application.
- an **Edit** (See *Memory editor: Edit Tab, p. 55*) tab, which enables you to modify the assignment of the PLC objects (function blocks, registers, memory objects).

Access to the memory editor

The table below describes the procedure for accessing the memory editor.

Step	Action
1	Open the required application.
2	Double-click the **Memory indicator** zone located in the status bar. or select the **Controller** → **Memory usage** command.

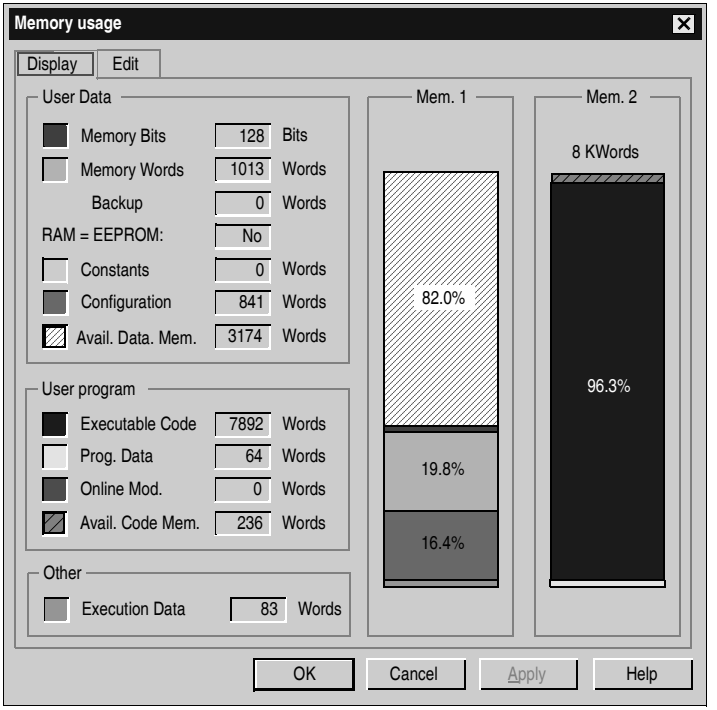
Memory editor: Display Tab

Introduction

The **View** tab includes all information relating to the PLC's memory usage. The information is updated on opening the editor.

Illustration

The illustration below shows the View tab of the memory editor.



Note: When the application or data are using too large for the memory available a single red block is displayed ≥ 100%.

Description

The following table describes the various components of the Display tab.

Field	Item	Description
User data	Memory bits	Number of internal %Mi bits used by the application.
	Memory words	Number of internal %MWi words used by the application.
	Backup	Size of backup application. **Note:** the sign **???**appears if an application backup has not been performed.
	RAM = EEPROM	Indicates if the RAM memory content is identical to the EEPROM memory. **Note:** the sign **???**appears if an application backup has not been performed.
	Constants	Number of %KWi configuration constants used by the application.
	Configuration	Number of words used by the application (other than those indicated above).
	Available memory data	Size of available memory (excluding program).
User program	Executable code	Size of user program.
	Program data	Number of words used by the application data.
	Online Mod.	Size of memory allocated for modifying the application program in online mode. On switching to online mode, or when the application is transferred in the PLC, this field is reset to 0. On switching to offline mode, the current value is saved to the memory.
	Available memory code	Size of available program memory.
Others	Executable data	Size of memory reserved for execution of the program in the PLC.
Mem. 1	-	Graphic representation of the memory usage rate for the internal RAM (available on all Twido controllers).
Mem. 2	-	Graphic representation of the memory usage rate for the external RAM (not available on TWDLCAA10DRF controllers).

Memory editor: Edit Tab

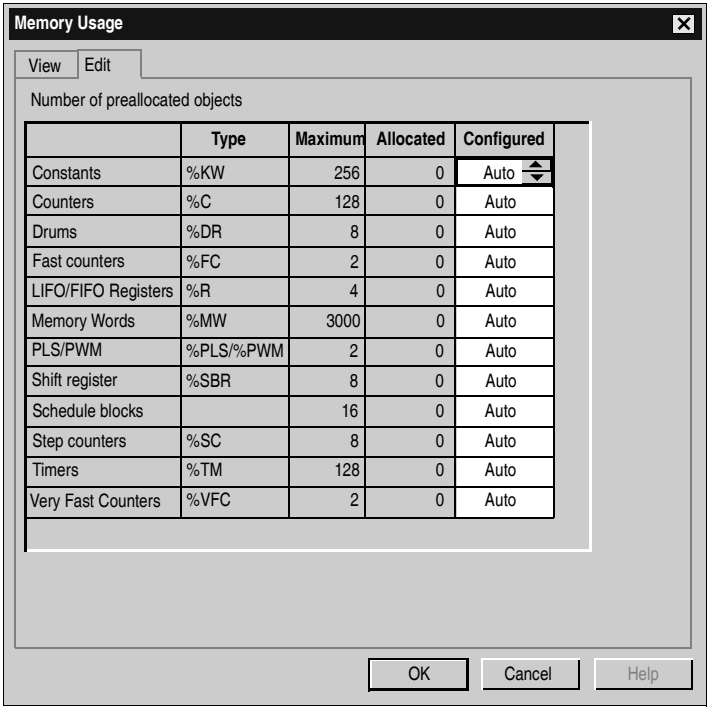
Introduction

The Edit tab presents the list of data objects that are used and usable by the application.
All data objects configured or used occupy memory space. It is therefore possible to optimize the PLC's memory requirements by only configuring the number of data objects actually needed by the application.

Note: Optimization of memory space can only be performed in offline mode.

Illustration

The illustration below shows the Edit tab of the memory editor.



	Type	Maximum	Allocated	Configured
Constants	%KW	256	0	Auto
Counters	%C	128	0	Auto
Drums	%DR	8	0	Auto
Fast counters	%FC	2	0	Auto
LIFO/FIFO Registers	%R	4	0	Auto
Memory Words	%MW	3000	0	Auto
PLS/PWM	%PLS/%PWM	2	0	Auto
Shift register	%SBR	8	0	Auto
Schedule blocks		16	0	Auto
Step counters	%SC	8	0	Auto
Timers	%TM	128	0	Auto
Very Fast Counters	%VFC	2	0	Auto

Description

The following table describes the various components of the Edit tab.

Column	Description
"Data Object"	List of data objects predefined by the application.
Type	Address associated with the corresponding data object type.
Maximum	Maximum number of data objects authorized for the controller.
Allocated	Number of data objects used by the application (1).
Configured	Number of data objects configured. The possible values are Auto, then 1 and the maximum value of the data object in question. By default, the value displayed is Auto. This means that the number of data objects configured is equal to the number of data objects used (1). To modify a value, enter the value directly or use the drop-down menu. Carry out the same procedure for each data object type and then confirm by clicking **Ok**.
(1)	**It is better first to use data objects with the lowest index, then continue in ascending order (e.g.: %TM0, %TM1). In fact, an application that uses for example the data object %TM3 automatically configures the timers %TM0 to %TM3 and so needlessly takes up memory space.**

Note: In the event that the value entered is less than the number of data objects actually used, an error appears during the application transfer. However, we recommend you carry out a program analysis after the optimization operation. The error objects appear in the error window of the program.

2.3 Operating Modes

At a Glance

Overview

This chapter describes the operating modes and operating states of TwidoSoft.

What's in this Section?

This section contains the following topics:

Offline and Online Operation

Introduction

TwidoSoft operations are either **Online** or **Offline** depending on the connection between the PC and the controller.

- In Online operation, the PC is connected to the controller.
- In Offline operation, the PC and controller are disconnected.

See *Connecting a PC to the Controller, p. 97*.

Offline

Use Offline operation to develop an application.
The PC is not connected to the controller. Changes are made only to the application in PC memory. The application must be transferred from the PC memory to the controller memory to run the application on the controller (see *Transferring an Application, p. 87*).

Online

Use Online operation to debug and adjust an application.
The PC is connected to the controller. The application in the PC memory is the same as the application in the controller memory. Changes can be made directly to the application in the controller memory (see *Debugging and Adjusting a Program Online , p. 333*).

Operating States

Introduction

Operating States indicate the offline or online status of TwidoSoft, and specify the operations that are allowable. There are four operating states:

- Initial
- Offline
- Online
- Monitor

The current operating state is displayed at the rightmost of the status bar.
See *Offline and Online Operation, p. 58*.

Initial State

The **Initial state** is the startup state when TwidoSoft is first started or when an application is closed. Available options:

- Open an existing application.
- Create a new application.

Only one application can be open at the same time.

Offline State

When you open a new or existing application, the operating state changes to the **Offline state**. Available options:

- Create and edit program instructions and configuration data.
- Create and edit data variables, symbols, and comments.

To connect the PC to the controller, select **Controller** → **Connect** from the main menu.

Online State

An application in the **Online state** is directly connected to the controller memory. You have unrestricted access to an application, and changes to the application program and data variables are directly written to controller memory. Program documentation such as symbols and comments remain in the PC memory. Available options:

- Operate the controller.
- View animation of program and data.
- Limited access to program for debugging.

Both of the following conditions are required for the online state:

- The controller application must **not** be protected.
- The application in the PC must match the application in the controller.

There are two methods for matching applications:

- Download the application from the PC to the controller.
- Upload the application from the controller to the PC.

To return to the Offline state, select **Controller** → **Disconnect** from the main menu bar. You can also select the **Close** option from the **File** menu to return to the Initial state without passing through the Offline state. If you select **Close**, you will be prompted to save your changes.

Monitor State

In the **Monitor state**, you can change operating states and make adjustments to the controller. Available options:

- Start or stop the controller.
- View, modify, or transfer data using the Animation Tables Editor.

You can not use the List or Ladder editors, Symbol Editor, or the Configuration Editor in the Monitor state. If the controller is protected, the Monitor state is the only online state available.

If the controller is in the Offline state, and you want to monitor an unprotected application in the controller that is different than the application in PC memory, enter the Monitor state.

To return to the Offline state, select **Controller** → **Disconnect** from the main menu bar.

Managing Applications

3

At a Glance

Overview

This chapter provides details for creating and managing applications for Twido programmable controllers.

What's in this Chapter?

This chapter contains the following sections:

3.1 Accessing Applications

At a Glance

Overview

This section provides procedures for accessing TwidoSoft applications.

What's in this Section?

This section contains the following topics:

Starting and Exiting TwidoSoft

Starting TwidoSoft

The installation procedure for TwidoSoft created one or more of the following options for starting the software, depending on selections made during installation.
Select one of the following options to start TwidoSoft:

- Double-click on the TwidoSoft icon on the Windows desktop.
- Select the TwidoSoft program from the Windows Programs menu.
- Select **Run** from the Windows Start menu and browse for the TwidoSoft program.

TwidoSoft starts and the main window is displayed (see *Main Window, p. 25*).

Exiting TwidoSoft

To exit TwidoSoft:

Step	Action
1	Select **File** → **Exit** from the main menu. If there no changes, the application closes and TwidoSoft exits to the Windows desktop. If there are changes, an **Exit Warning** dialog box is displayed prompting you to save changes.
2	Select **Yes** to save changes and exit TwidoSoft. Select **No** to discard changes and exit TwidoSoft, Select **Cancel** to return to TwidoSoft with changes intact,
3	If this is the first time the application has been saved, see *Save As, p. 78*.

Opening and Closing Applications

Introduction

Only one application can be open at the same time. If you attempt to open a second application, a dialog box is displayed prompting you to close the current application before opening another.

Opening an Application

From the main window, perform the following steps to open an application.

Step	Action
1	Select **File** → **Open** from the main menu.
2	Type a file name of the application file or browse to locate the file (default extension is .TWD).
3	When the application file is located, press **Open**. The file opens with the Application Browser displayed and the List Editor or the Ladder Editor displayed depending on the status of the settings saved in user preferences (see *Preferences, p. 31*). On the status bar, the state has changed from initial to offline. The file name and path of the application file appears in the title bar of the main software window.

Closing an Application

From the main window, perform the following steps to close an application.

Step	Action
1	Select **File** → **Close** from the main menu. If there are changes to the open application, the **Exit Warning** dialog box will be displayed.
2	In the **Exit Warning** dialog box, select **Yes** to save changes, select **No** to discard changes, or select **Cancel** to return to the application file without exiting the software. If there are no changes, TwidoSoft closes and returns to the Windows desktop.

Security Features

Introduction

TwidoSoft provides two options for protecting an application:

- **Controller application protection**: The controller application cannot be viewed, changed, or transferred.
- **Password protection**: Passwords are used to control access to an application.

Both protections work independently. They can be used either individually or at the same time.

Controller Application Protection

Controller application protection, accessed via **PLC** → **Protect application** menu, can be checked or unchecked. If checked, it prevents access to the controller application, but not to the PC application. This option prohibits unauthorized transfers of an application and is selectable when transferring an application to the controller (see *Transferring an Application, p. 87*).

A controller application that is protected:

- Cannot be modified.
- Cannot be transferred to PC RAM memory.

If **controller application protection is checked and used only** (password protection is not used), you can / cannot do what follows:

You can ...	**You cannot ...**
Access the application on PC.	
Upload, provided you have an opened application on TwidoSoft that is identical to the controller one.	Upload, in all other cases.
If you own the same application on your PC and the controller is connected and in Run mode, you can observe the application.	
If you own a different application on your PC, you can download the application, you can also observe it.	

If controller application protection is unchecked and used only (password protection is not used), you can upload an application from the controller to a PC.

Password Protection

Password protection controls access to both PC and controller application after an application has been transferred from a PC to the controller. A controller application that is password protected:

- Cannot be modified unless the password is entered.
- Cannot be transferred to PC RAM memory unless the password is entered.
- Cannot be overwritten by the application in PC RAM memory unless the password is entered.
- Can be erased.

If **password protection is used only** (controller application protection is not used) and you know the password, you can use TwidoSoft normally: modify the application, save the application, transfer a new application to the controller, ...
If **password protection is used only** (controller application protection is not used) and you don't know the password, you can / cannot do what follows:.

You can ...	**You cannot ...**
Read the application.	Modify the application.
Print the application.	Save the application to the PC hard disk.
	Read the password in ".twd" application file.
	Modify the Symbol Editor window.
	Modify the Memory Report window.

Note: It is very important to remember the password. Without it, you will not have access to the application!

See *Set Security Level, p. 69*.

Protections used simultaneously Though independant, **Password protection** and **Controller application protection** can be applied simultaneously.

Action	**Application protection is checked and password is used (but not known)**		**Application protection is checked and password is used (and known)**	
	You can ...	**You cannot ...**	**You can ...**	**You cannot ...**
At a PC level	Cancel the application protection	Modify the %KW constant values	Cancel the application protection	
	See the application		See the application	
	Save the application		Save the application	
			Modify the application	
			Modify the %KW constant values	
At upload	See the application	Modify the %KW constant values	See the application	Save the application
		Save the application	Modify the %KW constant values	Cancel the application protection
		Cancel the application protection		Modify the password
		Modify the password		
At download		Download the application		

Set Security Level

Introduction

Use the **Set Security Level** dialog box to apply password protection to an application. Password protection controls access to an application to prevent unauthorized changes.
See *Change Password, p. 71*.

Password Protection

Defining a password restricts access by creating two Security Levels:

- Operator Level: Does not allow changes to the application program and configuration data. Allows changes only to symbols and animation tables.
- Supervisor Level: Allows changes to any part of an application.

A password must consist of from one to eight characters.

> **Note:** At the Operator Level, you must enter the password to transfer a password-protected application from the PC to the controller.

Applying Password Protection

To apply password protection to an application:

Step	Action
1	Open an application requiring password protection.
2	Select **Program** → **Change Password** from the main menu.
3	Type a password in **Enter New Password**.
4	Select **OK** to close the box. The application must be saved to retain the password.

Removing Password Protection

To remove password protection from an application:

Step	Action
1	Open an application with password protection.
2	Select **Program** → **Change Password** from the main menu.
3	Type the current password in **Enter Old Password**. Do not enter a new password, tab through the other boxes.
4	Select **OK** to close the box. The application must be saved to remove password protection.

Opening an Application with Password Protection

Security level can be selected when opening an application. A password is required to use the Supervisor Level. To set the security level to Operator or Supervisor:

Step	Action
1	Open an application that has password protection. An information box dialog box is displayed allowing you to change security levels.
2	To open application at Operator Level, select **Cancel**. The application opens at the Operator Level.
3	To open the application at the Supervisor Level, select **OK**. The **Security** dialog box is displayed.
4	Type in the password and select **OK** to open the application at the Supervisor Level.

Changing from Operator to Supervisor

In an open application, you can change the security level in order to modify the application. The Supervisor Level is required to make changes to an application. To change from Operator to Supervisor level:

Step	Action
1	Open an application at the Operator Level.
2	Select **Program** → **Set Security Level** from the main menu.
3	Type the password in the **Security** dialog box.
4	Select **OK** to close the box. The application must be saved to retain the Supervisor Level.

Changing from Supervisor to Operator

The security level must be reset to Operator Level after making changes to ensure unauthorized changes are not made by anyone else using the application. The **Set Security** dialog is not available at the Supervisor Level, so you will need to save your changes, close the application, and then reopen it at the Operator Level. To change from Supervisor to Operator level:

Step	Action
1	Open an application at the Supervisor Level.
2	Save any changes and close the application.
3	Reopen the application. An information box dialog box is displayed allowing you to change security levels.
4	Select **Cancel** to open the application at the Operator Level. You will need the password to change the security level to Supervisor in order to make changes to the application.

Change Password

Introduction

Use the Change Password dialog box to change an existing password.
See *Security Features, p. 66*.

Changing a Password

To change a password:

Step	Action
1	Open an application with password protection.
2	Select **Program** → **Change Password** from the main menu.
3	Enter the old password.
4	Enter the new password. A password must consist of one to eight characters.
5	Confirm the new password by entering it again.
6	Select **OK** to close the box. The application must be saved to retain the new password.

3.2 Creating and Developing Applications

At a Glance

Overview

This section provides procedures for creating and developing TwidoSoft applications.

What's in this Section?

This section contains the following topics:

Creating an Application

Introduction

A TwidoSoft application consists of a program, configuration data, symbols, and documentation. These components can be used in any order when creating an application. For example, an application can be developed in the following order:

1. Define configuration data.
2. Define symbols,
3. Write program instructions,
4. Add comments and print documentation.

This is just one method to organize these tasks; these could just as easily been done in reverse order. The TwidoSoft editors and viewers allow separate development of each part of an application, as a result, the application development process is more systematic with more clearly defined applications.

Creating a New Application

From the main software window, perform the following steps to create a new application.

Step	Action
1	Select **File** → **New** from the main menu. The Application Browser is displayed and either the List Editor or the Ladder Editor is opened depending on the default editor selection (see *Preferences, p. 31*). On the status bar, the state has changed from Initial to Offline Mode.
2	A **Functional levels management** window pops up on top of the Editor. See *Functional Level Management, p. 139* for guidelines in choosing a functional level.
3	Select **File** → **Save** or **File** → **Save As** from the main menu. The **File Selection** dialog box is displayed.
4	Enter a file name for the application file. This is the file name external to the application. The application can also have an internal name which is different from the file name (see *Naming an Application, p. 77*). The file name is created with the default extension of .TWD. Unless another directory is selected, the file will be saved in the \applications sub-directory of the TwidoSoft installation directory. The file name and path are displayed in the title bar of the main software window.
5	See *Developing an Application, p. 74* for guidelines in developing an application.

Developing an Application

Introduction

The following flow charts identify recommended steps and associated sections in this manual for creating applications using TwidoSoft.

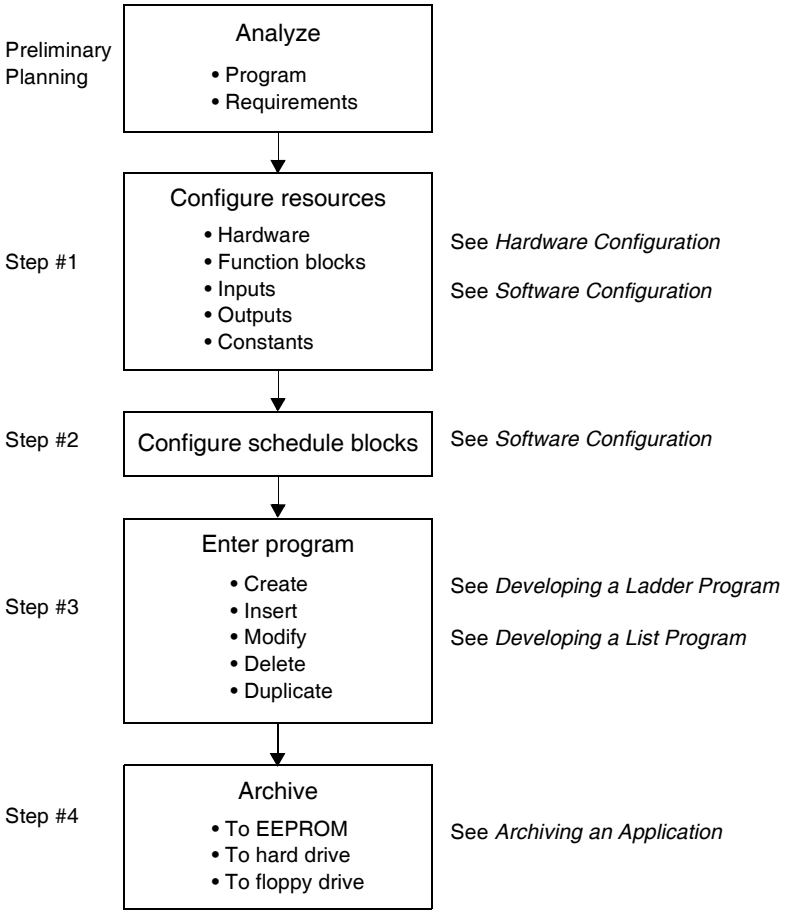
Development Phases

In general, developing an application requires two phases:

- Design Phase
  This is the first stage of development and requires a careful analysis of requirements for your application. Once a plan is created, the application is created requiring configuration, program creation, and archiving.
- Debug and Adjust Phase
  The final stage begins once an application is created. The application is transferred to the controller, started, and checked for errors or any required adjustments or modifications. Further backups and transfers may be needed.

Design Phase

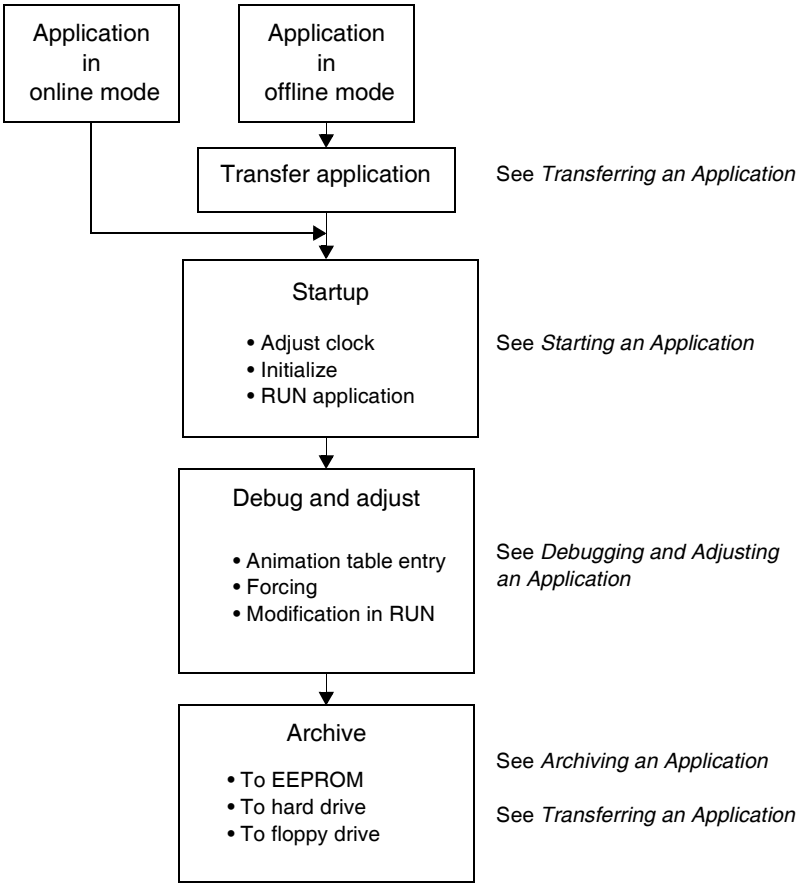
The following flow chart illustrates the design phase for an application.



Note: An optional approach is to perform Steps #1 and #2 in parallel with Step #3. The application can be configured at the same time the program is created.

Debug and Adjust Phase

Perform the debugging and adjusting phase in the online or monitor state. The following flow chart illustrates debugging and adjusting an application.



Naming an Application

Introduction

An application can have two names:

- File Name
  The external name for a TwidoSoft application with a 3-letter extension indicating that it is an executable program. Used by the Windows operating system for file management operations and must follow Windows guidelines for file names. For example, NEW_APP.TWD.
- Application name
  The internal name for a TwidoSoft application. An application name can be modified without any effect on the external file name. Provides more flexibility for application names. For example, car wash #2.

File Name

A file name can contain up to 255 characters, including spaces. But, it cannot contain any of the following characters: \ / : * ? " < > . Use **File** → **Save** or **File** → **Save As** to assign a file name for an application.
See *Saving an Application, p. 78*.

Application name

A new application has a default name of untitled. Use the *Application Name, p. 239* dialog box to change an application name. To open this dialog box, select **Program** → **Rename** from the main menu, or right-click on the application name at the top of the *Application Browser, p. 27*.

Saving an Application

Introduction

An application can be archived by selecting **Save** or **Save As** commands from the File menu.

Save As

Select **File** → **Save As** from the main menu for the initial save of an application, or to save the application to a different file name (see *Naming an Application, p. 77*). From the main window, perform the following steps.

Step	Action
1	Select **File** → **Save As** from the main menu.
2	Select a directory. The default current directory is the installed directory for the TwidoSoft program.
3	Enter a file name for the application. File names can be up to 255 characters. **Note:** Do not use any of the following characters: \:*?<>|".
4	Click the **Save** button. The application file name is saved with the default extension of TWD. The file name and path are displayed in the title bar of the main window.

Save

Select **File** → **Save** from the main menu to save changes to an open application after the initial save or rename of the application. Changes will be made to the open application file in the current directory. Selecting **Save** for an application that has not yet been saved will open the **Save As** dialog box.

Analyzing a Program

Introduction

Programs must be free from errors to be transferred to the controller. Select **Program** → **Analyze Program** from the main menu to compile a program and check for errors.

Analyze Program

In addition to compiling a program, Analyze Program performs the following:

- Checks that the application will fit in available PLC memory.
- Checks for correct syntax of each program line or rung.
- Checks that each symbol used in a program has a corresponding address.
- Checks that the structure of the program is correct.
- Checks that the resources used by the program are compatible with the hardware configuration (I/O addresses are valid for the configured PLC, expansion modules, and so).
- Displays messages in the Program Errors window.

Analyzing a Program

Use the following steps to analyze a program in the offline or online states.

Step	Action
1	Offline state: Select **Program** → **Analyze Program** from the main menu to check and compile a program at any time and from any editor.
2	Online state: In the online state, each program line is automatically validated as it is entered and before it is sent to the PLC. Programs entered in the online state do not need to be validated. (You can still validate a program as a precaution.)
3	If a program does not contain any errors, an Information dialog box will be displayed stating the following: • The program contains no errors or warnings. • The program is executable and CAN be transferred to the PLC.
4	If a program contains any errors, an Information dialog box will be displayed stating the following: • The number of program errors and warnings. • The program is not executable and CANNOT be transferred to the PLC. • See *Program Error Viewer, p. 48*.

Viewing Program Errors

Select **Program** → **View Program Errors** from the main menu to display the **Program Errors** window.
The format for each message:

- The type of message - error or warning.
- The number of the line or rung containing the error or warning.
- An explanation of the problem.

Types of Error Messages

There are two types of error messages:

- Error
  An error message indicates a problem in the application that prevents the creation of an executable program.
- Warning
  A warning message reports on instructions that are not reversible or may cause erratic operation.

Note: Double-clicking on the error returns you to the location of the error program (list or ladder) window.

Defining Symbols

Introduction

Use the *Symbol Editor, p. 45* to assign easy-to-recognize alphanumeric names called symbols to data variables in a program. Using symbols allows for quick examination and analysis of program logic, and greatly simplifies the development and testing of an application.
For example, WASH_END is a symbol that could be used to identify a timer function block that represents the end of a wash cycle. Recalling the purpose of this name should be easier then trying to remember the role of a program address such as %TM3.

Guidelines for Using Symbols

When to define symbols:

- You can define symbols using the Symbol Editor before, during, or after writing a program.
- A symbol without a variable address is an unresolved symbol. You can write a program with unresolved symbols while you are designing program logic. You can complete the symbol table once you complete the program.

Rules for Defining Symbols

Defining a symbol requires:

- A maximum of 32 characters.
- Letters (A-Z), numbers (0 -9), or underscores (_).
- First character must be an alphabetical or accented character.
- No spaces or special characters.
- Not case-sensitive. For example, Pump1 and PUMP1 are the same symbol and can only be used once in an application.

Creating a Symbol

To create a new symbol:

Step	Action
1	Open the Symbol Editor.
2	Left-click on an empty **Symbol** cell and enter a variable or object.
3	Press ENTER. The *Object Browser, p. 47* opens.
4	If you know the address of the variable, enter in the **Address** list box, or use the **Filter** box to narrow your choices to one type of variable.
5	Enter an alphanumeric name for the variable in the **Symbol** box. A symbol can be up to 32 characters and consist of the following: 0 to 9, A to Z, or underscores.
6	Optionally, enter text in the **Comment** box describing the symbol usage in more detail (up to 128 characters.)
7	A variable can be configured directly from the Symbol Editor by clicking on the **Configure** button.
8	Use the **Previous** and **Next** buttons to select other instances of the variable.
9	Click on **OK** to save changes and close the dialog box and return to the Symbol Editor.

Editing a Symbol

To edit an existing symbol:

Step	Action
1	Open the Symbol Editor.
2	Double-click on a symbol entry in a cell to open the *Object Browser, p. 47*.
3	Make changes to the symbol using the Object Browser.
4	Click on **OK** to save changes and close the **Object Browser** dialog box.
5	Optionally, enter text in the **Comment** box describing the symbol usage in more detail (up to 128 characters.)
6	A variable can be configured directly from the Symbol Editor by clicking on the **Configure** button.
7	Click on **OK** to save changes and close the dialog box and return to the Symbol Editor.

Deleting a Symbol

To delete a symbol:

Step	Action
1	Open the Symbol Editor.
2	Click on the symbol cell of the entry you want to delete.
3	Select **Tools** → **Delete Row**, or click on the toolbar shortcut, or click on the DELETE button. The symbol row is deleted.

Sorting the Symbol Table

Entries in a symbol table can be sorted by address or symbols:

- To sort by address, select **Tools** → **Sort by Address** from the main menu or click on the toolbar shortcut.
- To sort by symbol, select **Tools** → **Sort by Symbol** from the main menu or click on the toolbar shortcut.

Finding a Symbol

To search for a symbol in the symbol table:

Step	Action
1	Open the Symbol Editor.
2	Select **Tools** → **Find Symbol** from the main menu, or click on the toolbar shortcut. The **Find Symbol** dialog box is opened.
3	See *Find Symbol, p. 46* for instructions.

Reversing a Program

Introduction

An application written in ladder language can be converted or reversed to list language. A program written in list language can be reversed to ladder language if the instructions follow the **Reversibility Guidelines** which are located in the Software Reference portion of the Twido User Reference Manual.

Note: Any portions of a List program that do not meet the rules for reversibility will not be displayed as Ladder rungs in the Ladder Editor but as List instructions in the *List Rung Editor, p. 42*.

Reversing a Program

Reversing a program is simply alternating the selection of the list and ladder editors from the **Program** menu. Use the following steps to reverse a program in the offline or online states.

Step	Action
1	With an open application, select **Program** → **Ladder Editor** from the main menu. The Ladder Viewer is displayed, and the program is displayed as ladder rungs.
2	To reverse the program to list language, select **Program** → **List Editor** from the main menu. The List Editor is displayed, and the program is displayed as list instructions.

Importing and Exporting ASCII

Introduction

Program instructions that have been exported in ASCII text format from a TwidoSoft application can be imported into another TwidoSoft application program. Only program instructions can be imported or exported. These ASCII files are saved with a file extension of TXT.
If you try to import ASCII text created in a program other than TwidoSoft (e.g. PL707), you must carefully check that the user application continues to function correctly. You should note that the following PL707 instructions have been deleted or re-defined in TwidoSoft: %FC, MCR, MCS, EXCH, %MSG, %PLS, %PWM.

Exporting ASCII Text

To export an ASCII text file from an open application:

Step	Action
1	Select **Program** → **Export ASCII Program** from the main menu. The **Save As** dialog box opens.
2	Enter a name for the text file and press **Save**. The program instructions from the open application are saved in ASCII format.

Importing ASCII Text

To import an ASCII text file into an open application or a new application:

Step	Action
1	Select **Program** → **Import ASCII Program** from the main menu. The **Open** dialog box opens.
2	Select an ASCII text file with the file extension .txt, and press **Open**. (This file must have been exported from a TwidoSoft application.) The program instructions are inserted as new instructions for a new application, or are added to the end of a program in a open application.

3.3 Transferring Applications

At a Glance

Overview

This section provides procedures for transferring TwidoSoft applications.

What's in this Section?

This section contains the following topics:

Transferring an Application

Introduction

An application can be transferred to or from the following hardware storage areas:
- PC RAM
- Controller RAM
- Controller EEPROM
- Backup cartridge

Types of Transfers

The following table describes the types of application transfers that are available from the **Controller** menu. Applications can be transferred in the Offline and Online states, but the controller must be stopped to transfer applications in the Online state (see *Operating States, p. 59*).

Selection	Description
Transfer PC => Controller	Transfers a copy of an application from PC RAM memory to controller RAM memory. See *Transferring an Application: PC => Controller, p. 88*.
Backup	Transfers a copy of an application from controller RAM memory to the internal EEPROM and to the optional external backup cartridge (if installed). See *Backup, p. 89*.
Restore	Transfers a copy of an application from the internal EEPROM to controller RAM memory. See *Restore, p. 90*.
Erase	Although not a transfer procedure, the **Erase** command selectively erases the contents of the above storage areas. Erase has two options: • Erase all • Erase backup cartridge only See *Erase, p. 93*.

Note:
If there is no application in TwidoSoft at the time of connection, the application is transferred from the controller automatically.
If there is an application in TwidoSoft, a window is displayed asking you to choose the direction of transfer. You can either overwrite the application in the controller with that from the PC (PC=>Controller transfer direction), or you can update the application in TwidoSoft (Controller =>PC transfer direction).
This type of transfer is only performed at the time of connection.

Transferring an Application: PC => Controller

Introduction

To run an application, you must first connect the PC to the controller then transfer the application to the controller.

Transfer PC => Controller

To transfer an application from the PC RAM to the controller RAM:

Step	Action
1	Select **Controller** → **Transfer PC => Controller** from the main menu. If the controller application is password protected, you are prompted for a password.
2	Enter the password and select **OK** to continue the transfer. If the controller is running, a warning box is displayed stating that to continue the transfer, the controller must be stopped.
3	If the controller is stopped, the transfer begins. If the controller is running, select **OK** to stop the controller and continue the transfer. Select **Cancel** to close the box and cancel the transfer. A warning dialog box is displayed indicating that the contents of the controller application will be overwritten.
4	Select **OK** to continue the transfer. A controller menu option can be used to protect the application (see *Security Features, p. 66*).
5	To run the application, select **Controller** → **Run** (RUN) from the main menu. See *Run, Stop, and Initialize, p. 104*.

Symbols and program comments are not transferred to the controller.

Backup

Introduction

The Backup command copies an application (program and constants) from the controller RAM memory into the internal EEPROM memory and the optional Backup Cartridge (if installed). It is strongly recommended that an application be backed up to EEPROM as soon as it is debugged. Backup can be performed in both online and offline states.
See *Memory Backup Cartridges, p. 91*.

Note: While a backup is in progress, the controller cannot be running, and no other operations can be performed by TwidoSoft. While the backup is being performed the controller is in STOP mode.

Backing Up an Application

To back up an application from controller RAM to internal EEPROM memory and a backup cartridge (if installed):

Step	Action
1	Select **PLC** → **Backup** from the main menu. The controller must be stopped to perform a backup. If the controller is running, a dialog box is displayed allowing you to stop the controller.
2	If the controller is stopped, the transfer begins. If the controller is running, select **OK** to stop the controller and continue the transfer. Select **Cancel** to close the box and cancel the transfer. During the transfer, **Backing up application** is displayed in the message area of the *Status Bar, p. 30*. When the transfer is complete, **Backup complete** is displayed in the message area. The application is now in EEPROM memory and the backup cartridge (if installed).

Restore

Introduction

The **Restore** command copies an application from the internal EEPROM to controller RAM memory. Restore must be performed in the Offline state.

Note: While a restore is in progress, the controller cannot be running, and no other operations can be performed by TwidoSoft. While the restore is being performed, the controller is in STOP mode.

Restoring an Application from Internal EEPROM

To Restore an application from the internal EEPROM to controller RAM:

Step	Action
1	Confirm that a Backup Cartridge is not installed, and select **Restore** from the **Controller** menu. The controller must be stopped to perform a restore. If the controller is running, a dialog box is displayed allowing you to stop the controller.
2	If the controller is stopped, the transfer begins. If the controller is running, select **OK** to stop the controller and continue the transfer. Select **Cancel** to close the box and cancel the transfer. During the transfer, the message **Restoring application** is displayed in the message area of the *Status Bar, p. 30*. When the transfer is complete, **Restore complete** is displayed in the message area. The application is now in controller RAM memory.

Memory Backup Cartridges

Introduction

Twido provides optional **Memory Backup Cartridges** for the following functions:
- Backing Up an Application
- Expansion Memory

> **Note:** The memory cartridge overwrites the RAM and internal EEPROM at power up. Any application in RAM and EEPROM will be lost. Be sure to remove the memory cartridge when this functionality is not desirable.
> If the internal battery is no longer charged, when you restart Twido after a power interruption, you will restore from either the external cartridge or the internal EEPROM. The external cartridge takes priority.

Available External Backup Cartridges

There are two options available:
- 32 KB Memory Backup Cartridge
- 64 KB Memory Backup Cartridge

The 32K cartridge can be used by all controller models, while the 64K cartridge is available for the following models:
- TWDLCAA40DRF
- TWDLCAE40DRF
- TWDLMDA20DRT
- TWDLMDA40DUK
- TWDLMDA40DTK

Using Backup Cartridges

An application can be transferred between a backup cartridge and controller RAM by the following methods:
- Manually by using *Backup, p. 89*.
- Automatically at powerup if the optional backup cartridge is installed. At powerup, the application in the controller memory is compared to the application in the backup cartridge. If they do not match, the application in the backup cartridge is automatically downloaded to the controller memory and the EEPROM.

Restoring an Application from a Backup Cartridge

To Restore an application from an optional external Backup Cartridge to controller RAM:

Step	Action
1	Insert a Backup Cartridge into the controller, if not already installed.
2	Powerup the controller. If the application in the cartridge and the application in the controller memory do not match, the application in the cartridge is automatically transferred to both the controller memory and the internal EEPROM.

Expansion Memory

The 64K cartridge provides memory expansion as well as backup. With the 64K cartridge installed, you can create an application that is up to 64K in size. The 64K cartridge must remain installed for running and backing up the application.

Programs Larger than 32K

The 64K Expansion Cartridge is required to run applications larger than 32K. The 64K Expansion Cartridge must be "permanently" installed to provide this function. That is, if you are planning to create an application larger than 32K, install a 64K memory cartridge and leave it installed throughout development and configuration of the application.

Erase

Introduction

This is a command that can be used in two ways:

- To delete the contents of the controller RAM, the controller internal EEPROM, and the installed optional backup cartridge, all at the same time.
- To delete the contents of the installed optional backup cartridge only.

Options

There are two options available in this dialog box:

- Erase all
  This is the default selection. Selecting this option erases the contents of the controller RAM, the controller internal EEPROM, and the external backup cartridge (if installed).
- Erase backup cartridge only
  Selecting this option erases the external backup cartridge only (if installed).

Note: Use the **Erase backup cartridge only** option when you do not want to lose your application when you reinsert the backup cartridge. If the applications do not match, the controller memory will be overwritten with the contents of the external cartridge.

Note: While an erase is in progress, the controller cannot be running, and no other operations can be performed by TwidoSoft. While an erase is being performed, the controller is in STOP mode.

Using Erase

To Erase an application:

Step	Action
1	Select **Erase** from the **Controller** menu. The **Erase** dialog box is displayed.
2	Select **Erase All** or **Erase backup cartridge only** in the dialog box, and select **OK** to close the dialog box and start the command. The controller must be stopped to perform an erase. If the controller is running, a dialog box is displayed allowing you to stop the controller.
3	If the controller is stopped, the erase begins. If the controller is running, select **OK** to stop the controller and continue the erase. Select **Cancel** to close the box and cancel the erase. During the erase, **Erasing application** is displayed in the message area of the *Status Bar, p. 30*. When the erase is complete, **Erase complete** is displayed in the message area.

3.4 Running Applications

At a Glance

Overview

This section provides procedures for running TwidoSoft applications.

What's in this Section?

This section contains the following topics:

Controller Operations

Introduction

TwidoSoft allows use of the PC to control the operation of the programmable controller. The dialog box is accessed by selecting "Controller" then "Operate Controller...". This switches you into Online mode if you were in Offline mode before. You can:

- Connect the PC to the controller.
- Disconnect the PC from the controller.
- Use the PC to Run, Stop, and Initialize the controller.
- Use the PC to operate the controller.
- Use the PC to view live statistical data pertaining to the current Ethernet connection of the Twido controller.

Connecting a PC to the Controller

Introduction

The following conditions are required to connect a PC to the controller:

- The hardware configurations and the application on the PC must be compatible with those on the controller.
- The controller is not protected (see Controller Application Protection in *Security Features, p. 66*).

If the PC fails to connect, see *Connect, p. 99*.

Overview of Connect Process

When connecting a PC to the controller, the following occurs:

- The application is analyzed for program errors.
- TwidoSoft attempts to connect to the controller.
- The application in the PC is compared to the application in the controller. If they do not match, the Connect (See *Connect, p. 99*) dialog box is opened. If they are the same, the PC is connected to the controller.

Connecting a PC to the Controller

To initiate communication between a PC and a controller:

Step	Action
1	In an open application, select **Connect** from the **Controller** menu, or right-click on the Application Name in the Application Browser and select **Connect**. "Connecting to the controller" is displayed on the Status bar.
2	The Connect (See *Connect, p. 99*) dialog box is displayed when connecting to the controller if: ● Hardware configurations do not match for the PC and the controller. ● Applications do not match for the PC and the controller. ● The application in the controller is protected.
3	Select **Cancel** to exit the **Connect** dialog box and return to the Offline state.
4	Select **Monitor** if you only want to access Animation Tables and do not want to modify the program, configuration, and symbols for the application in the controller. The operating state changes from Offline to Monitor.
5	If the application in the controller is protected, the only available option is to click on the **Monitor** button to connect to the controller in Monitor state.
6	If the applications do not match, select one of the following: ● **PC => Controller**: transfer the open application in the PC to the controller. ● **Controller => PC**: transfer the application in the controller to the PC.
7	If you selected **PC => Controller**, the download of the application from the PC to the controller is started. An information dialog box opens to confirm the following: ● Controller is running ● Overwrite the controller
8	Select **OK** to continue the transfer, or select **Cancel** to abort the transfer. If you have selected **OK** the transfer is completed and the PC connects to the controller. The main window is displayed, the status bar displays Controller Connected, and the operating state changes from Offline to Online.
9	If you selected **Controller => PC**, the application in the controller is transferred to the PC. The operating state changes from Offline to Online. The main window is displayed, the status bar displays Controller Connected, and the operating state changes from Offline to Online.

Application protection

By default, the application in the controller is not protected.
To protect an application:

Step	Action
1	Select **PLC** → **Protect application** menu. **Result:** A checkmark indicates that the application protection is enabled.
2	To unprotect the application, select it again. **Result:** The checkmark has been removed.

Connect

Introduction

This dialog box is displayed if a PC failed to connect to the controller because the application on the PC is different from that on the controller.
See *Connecting a PC to the Controller, p. 97*.

Reasons for Failing to Connect

The following will cause a PC to fail to connect to the controller:
- Hardware configurations for PC and controller do not match.
- The application on the PC does not match that on the controller.
- Application in controller is protected (see *Security Features, p. 66*).

Additional Messages

The **Connect** dialog box also displays the following:
- Hardware configuration different (cannot transfer PC => Controller)
- Controller application protected (cannot transfer Controller => PC)

An applicable condition is indicated by a checked box.

Options

The following options are available from the **Connect** dialog box:
- PC => Controller
  Transfer the application in the PC to the controller. Not available if the hardware configurations are not compatible.
- Controller => PC
  Transfer the application in the controller to the PC. Not available if the controller is protected. See Controller Application Protection in *Security Features, p. 66*.
- Monitor
  Enter the Monitor state without a transfer for a limited access to an application running in the controller. See *Operating States, p. 59*.
- Cancel
  Exit the **Connect** dialog box.

Hardware Configurations

If the hardware configuration of the controller is not compatible with the configuration specified in the application on your PC, you cannot transfer the application to the controller. You can enter the Monitor state to perform a subset of controller operations (see *Operating States, p. 59*).

Controller Application Protected

If the application in the controller is protected, then the option to transfer from the controller to the PC is disabled. You can also enter the Monitor state in this case.

Matching Applications

If the applications do not match, there are two options:

- Transfer the application in the PC to the controller (PC => Controller).
- Transfer the application in the controller to the PC (Controller => PC).

PC => Controller

Select this button to transfer your application on the PC to the controller. If the application on the controller is protected, you can choose to overwrite it. You can also select to protect the application you are currently transferring.

Controller => PC

Select this button to transfer the application on the controller to the PC. If you have an open application, you will be prompted to save any changes before the transfer begins.

Monitor

If the application and configuration on the PC do not match exactly those on the controller or the controller is protected, you can still connect in the Monitor state. This allows access to the application data pages and to operate the controller. You will be restricted from accessing the application program, configuration, or symbols.

Disconnecting a PC from the Controller

Disconnecting PC From the Controller

To disconnect a PC from the controller:

Step	Action
1	Select **Disconnect** from the **Controller** menu, or right-click on the Application Name in the Application Browser and select **Disconnect.**
2	When completed, "Controller Disconnected" is displayed on the status bar and the operating state changes to Offline.

Controller Operations

Introduction

Use this dialog box to remotely operate and view your controller. From this dialog box, you can:

- View status and system information.
- Run, Stop, and Initialize the controller.
- Enter time and date settings for the optional RTC.
- Configure the RTC Correction Factor.
- Perform Advanced Operations.

Status Information

The **Status** check boxes indicate the following:

- One or more input or output bits are forced to either a 1 or 0.
- An executable application is stored in the controller's RAM.
- The controller's RAM protection bit was set to 1 during the transfer process.

Switches

The **Switches** boxes display the settings for all the potentiometers available on your controller. This is one 10-bit value for each of the controllers. All controllers have at least one potentiometer with the exception of the TWDLC•A24DRF and TWDLCA•40DRF which have two. For the TWDLC•A24DRF and TWDLCA•40DRF, the second potentiometer can be set between 0 and 511.

Scan Time

The Current box in **Scan Time** displays the time in milliseconds that is required to solve the user logic, process inputs/outputs, and service the system and outstanding communications. This is the actual measured time required to complete the tasks to support the user application. This is different than the requested Periodic Scan Period that you can specify as the maximum allotted time to complete a scan (see *Scan Mode, p. 236*).
The minimum (Shortest) and maximum (Longest) scan times since the last power down are also displayed in milliseconds.

Real Time Clock (RTC)

These boxes are only valid when an RTC option cartridge (TWDXCPRTC) is installed on the controller. Note that the TWDLCA•40DRF series of compact controllers have RTC onboard. The **Real Time Clock** box display the current date, time, and the RTC correction factor set. You can change the date and time by selecting the **Set Time** button.

Configure RTC

This button is only enabled when the Real Time Clock (RTC) optional cartridge (TWDXCPRTC) is installed on the controller. Note that the TWDLCA•40DRF series of compact controllers have RTC onboard. Click on this button to enter the RTC Correction Factor for the RTC.

LED Display

The LEDs on the base controller are simulated in the **Controller Operations** dialog box (for details about these LEDs, see the Twido Hardware Reference Guide). The following are states of the simulated LEDs as they appear in the **Controller Operations** dialog box. For all simulated LEDs in the dialog box, a question mark (?) indicates communication has yet not been established.
For Inputs/Outputs (**IN** and **OUT**) LEDs:

LED State	Description
Extinguished	I/O point is Off
Steady green	I/O point is On
Has a lock symbol	I/O point is forced to a 0 or 1 0 is extinguished with lock, and 1 is green with lock

For **RUN** and **ERR** LEDs:

RUN LED	ERR LED	Description
Extinguished	Blinking red	No valid configuration in controller
Blinking green	Extinguished	Controller is stopped
Steady green	Extinguished	Controller is running
Blinking green	Blinking red	Controller is halted
Extinguished	Steady red	Hardware or system error. Must cold restart

The operation of the **STAT** LED is defined by user logic.
For the **BATT** LED (TWDLCA•40DRF series controllers only):

LED State	Description
Extinguished	Indicates that either: • the external battery is functioning normally, • the BATT LED has been disabled by user by setting the %S66 system bit to 1.
Steady red	Indicates that either: • the power of the external battery is low (voltage below 2.5V), • there is no external battery installed in the battery compartment.

The **LAN ACT** and **LAN ST** LEDs (TWDLCAE40DRF controller only) show the current status of the Ethernet connection. For detailed information about the Ethernet LEDs, please refer to*Ethernet LED Indicators, p. 208*.

Run, Stop, and Initialize

Introduction

You can use the PC to issue the following commands to the controller:
- Run
- Stop
- Initialize

You must confirm the command in order for it to be performed.

Selecting These Commands

Note: The Run/Stop/Initialize are not available in offline mode.

When the TwidoSoft application is in online mode, these commands can be selected from either:
- the **Controller** menu;
- the toolbar of the main menu;
- the Application Browser, by right-clicking on the controller model number; or
- the **Controller Operations** dialog box.

Run

When the controller is running, the application program is executing. The controller's inputs are updated and data values are set according to the application instructions. This is the only state where the actual outputs are updated.

Stop

When the controller is stopped, the application is not being executed. The controller's inputs are updated and internal data is held at its last values. Outputs are not updated in this state.

Initialize (Init)

When the controller is initialized, the application is not executed. The controller's inputs are updated and data values are set to their initial state. No outputs are updated from this state.

Set Time

Introduction

Use the **Set Time** dialog box to enter time and date settings for the controller's Real Time Clock (RTC). This button is only enabled when an RTC option cartridge (TWDXCPRTC) is installed on the controller. Note that the TWDLCA•40DRF series of compact controllers have RTC onboard.
See *RTC Configuration, p. 106*.

Setting the RTC

To set the time and date for the RTC:

Step	Action
1	Select **Set Time** from the **Controller Operations** dialog box.
2	Enter the current date in **Controller Date**. A valid date is in the format of day/month/year (DD/MM/YY). For example, March 28, 2002 is entered as 28/03/02.
3	Enter the current time in **Controller Time**, A valid time value is standard "military" format HH:MM. For example, 2:15 PM is 14:15.

RTC Configuration

Introduction

Use the **RTC Configuration** dialog box to enter the RTC Correction Factor for the Real Time Clock (RTC) option. The RTC option allows the controller to have a hardware Time of Day clock.
The RTC will keep the time even when the controller is not powered for a limited amount of time. When the RTC is not powered by the controller, the time will continue to be updated for 1,000 hours at 25 °C and 300 hours at 55 °C using a fully charged battery.
The RTC option cartridge (TWDXCPRTC) must be installed on the controller. This option is also required for using Schedule Blocks. Note that the TWDLCA•40DRF series of compact controllers have RTC onboard.

RTC Correction Factor

Configuring the RTC option requires entering the RTC Correction Factor. Each RTC has a correction value in decimal located on the part label. This correction value must be entered in the **RTC Configuration** dialog box so that TwidoSoft can initialize the RTC cartridge.

RTC Display

When there is an RTC installed on the controller and it has a valid correction value, the **Controller Operations** dialog displays the following:
- Controller Date
- Controller Time
- RTC Correction
- **Set Time** button is enabled

If there is no RTC installed or is has an invalid correction value the date and time displays are blank and the **Set Time** button is disabled.

Configuring the RTC

To configure the RTC option:

Step	Action
1	Click on the **Configure RTC** button in the **Controller Operations** dialog box. The installed RTC option is required. The **RTC Configuration** dialog box is displayed.
2	Enter a decimal value between 0 and 127. See the label on the RTC part for this value. Default value is 0.
3	Select **OK** to accept the entered value and close the dialog box. Select **Cancel** to close the dialog box without making any changes. The entered RTC Correction Value is added to the configuration of the PC and the controller (if connected).
4	The **Controller Operations** dialog now displays the current date and time and the current correction value and enables the **Set Time** button. Select **Set Time** to enter the current time and date for the RTC.

Ethernet Statistics

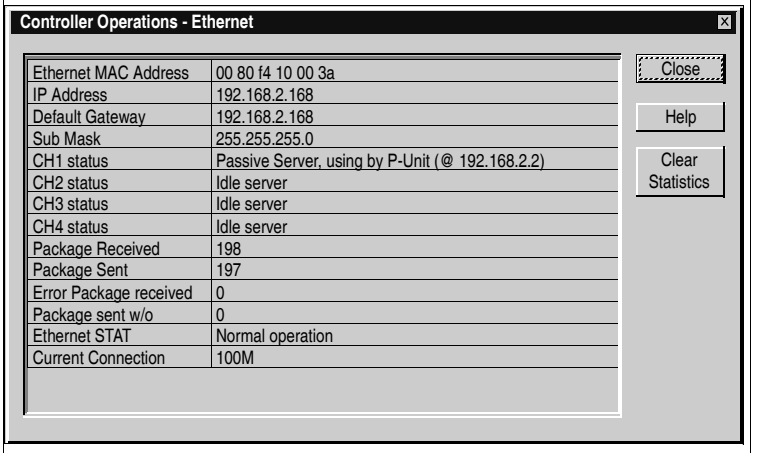
Overview

Use the **Ethernet** dialog box to view MAC address, IP configuration settings, and Ethernet network statistics for your Twido controller.

Note:
1. This dialogbox is available on TWDLCAE40DRF Ethernet-capable base controllers only.
2. To view network statistics correctly, it is assumed that your PC running the TwidoSoft application is connect to a TWDLCAE40DRF controller over the Ethernet network.

Using the Ethernet Statistics Table

To call up and use the Ethernet network statistics table, follow these directions:

Step	Action
1	Select **PLC > Check PLC** from the TwidoSoft application menu bar. **Result:** This brings up the **Controller Operations** dialog box
2	Click the **Ethernet** button located in the right portion of the current Controllers Operation dialog box. **Result:** The **Ethernet Statistics** table appears, as shown in the following figure: 
3	The above table contains information organized as two categories: ● Four lines at the top of the table display the MAC and IP information about your Twido controller. ● The remainder of the table displays live statistics about the Ethernet connection of your Twido controller over the network.
4	You may also click **Clear Statistics** to reset the current statistical values. **Note:** Displayed statistics are gathered from the time of the last Ethernet connection of the Twido controller over the network, or from the last time you cleared them using the Clear Statistics button.

Description of the Ethernet Statistics Table

The following table describes the information in the **Ethernet Statistics** table:

Field	Description	Control
Ethernet MAC Address	48-bit Universal LAN MAC address uniquely assigned to the TWDLCAE40DRF base controller.	R
IP Address	32-bit default IP address derived from the MAC address. Note that the IP address you have configured for the base controller may different from the default address.	R
Default Gateway	32-bit IP address of the network gateway device. Note that the gateway device must be located on the same subnet as the controller. Therefore, the gateway's IP must be compatible with the selected subnet mask.	R
Sub Mask	32-bit subnetwork mask compatible with the IP class of the controller's IP address. Note that the subnetwork you have configured for the base controller may different from the default subnet mask.	R
Current Connection Speed	Indicates the current network connection speed. Three possible readings: • OFF (No current Ethernet connection) • 10M (10 Mbps network connection speed) • 100M (100 Mbps network connection speed	R
Ethernet STAT	Indicates operational status of the Ethernet network connection.	R
CH1 Status	Indicates the current state of TCP Channel 1. Possible readings are as follows: • Idle server. • Passive server, Messaging active. Connected with [IP address]. • Passive server, timeout expired. Connected with [IP address]. • Marked server, Messaging active. Connected with [IP address]. • Marked server, timeout expired. Connected with [IP address]. • Passive Server, using by P-Unit . (@ [IP address]) • Passive Server, using by P-unit(@ [IP address]) but timeout expired. • Passive Server, using by P-unit(@ [IP address]) but timeout expired. • Waiting for TCP ACK from [IP address]. • Waiting for TCP ACK from [IP address] but timeout expired. • Waiting for Modbus TCP response from [IP address]. • Waiting for Modbus TCP response from [IP address] but timeout expired.	C

Field	Description	Control
CH2 Status	Indicates the current state of TCP Channel 2. Possible readings: see CH1 above	C
CH3 Status	Indicates the current state of TCP Channel 3. Possible readings: see CH1 above	C
CH4 Status	Indicates the current state of TCP Channel 4. Possible readings: see CH1 above	C
Package Received	Number frames received by the controller.	C
Package Sent	Number frames sent by the controller.	C
Error package received	Number of frame errors to date.	C
Packages sent without response	Number of frames successfully sent, but not answered.	C

Note: About user-control over Ethernet statistical data.

- R: This is a read-only, non-statistical information that cannot be cleared in this table.
- C: This statistical information is clearable by clicking on the **Clear Statistics** button.

Controller Operations - Advanced

Introduction

The **Controller Operations - Advanced** dialog box displays the controller state system information in read-only. Use this information to check the controller RAM, internal EEPROM, and backup cartridge status and to diagnose problems.

RAM

The following table describes the checked state of items in the RAM section.

Item	Description
Executable	A valid application is in controller RAM.
Protected	The application in controller RAM is protected.
Compatible with Exec	The version of the operating system in the controller is compatible with the application in controller RAM.
Different than EEPROM	The application in the controller RAM is different than the application in the internal EEPROM.
Different than Cartridge	The application in the controller RAM is different than the application in the backup cartridge. Only checked if there is a backup cartridge installed.

EEPROM

The following table describes the checked state of items in the EEPROM section.

Item	Description
Executable	A valid application is stored in the internal EEPROM.
Protected	The application in the internal EEPROM is protected.
Compatible with Exec	The version of the operating system in the controller is compatible with the application in the internal EEPROM.

PLC

The following table describes the checked state of items in the Controller section.

Item	Description
Watchdog time-out	The scan time of a program exceeded the maximum time allowed causing the controller to stop.
Real-time clock	The controller has the Real Time Clock option cartridge (TWDXCPRTC) installed.
I/O Forced	One or more base input or output bits were forced on or off.
Last Controller Stop	Displays date and time of last controller stop.

Backup cartridge

The following table describes the checked state of items in the Backup Cartridge section.

Item	Description
Executable	A valid application is stored in the backup cartridge.
Protected	The application in the backup cartridge is protected.
Compatible with Exec	The version of the operating system in the controller is compatible with the application in the backup cartridge.

3.5 Cross Referencing an Application

At a Glance

Overview

This section describes how to Cross Reference an application.

What's in this Section?

This section contains the following topics:

Cross Referencing an Application

Introduction

Cross referencing consists of selecting items in a program such as an address that you want to track in an application. Once the items are selected, then a list is generated displaying where the items are used in the application.
Cross referencing provides a useful list of the operands, symbols, lines and operators used in all or part of an application. Its primary use is during troubleshooting or debugging. It allows any item of interest to be easily located and cross-referenced to its locations in the application without having to search through the entire program.
See *Generate Cross References, p. 114* and *Cross References Viewer, p. 49*.

Tips

Some tips for using Cross References:
- Analyze the program before generating cross references.
- The Cross Reference List is not saved when the application is closed. You will have to regenerate it when the application is opened again.

Basic Steps

There are five basic steps for cross referencing an application (see comment relating to *Generate Cross References, p. 114*):

Step	Action
1	Select either list numbers or rung numbers for use as line references.
2	Select the scope of the display by specifying a range of lines.
3	Select the sort order of the list to be generated.
4	Select the specific program items to included in the display.
5	Generate the cross reference list.

Generate Cross References

Introduction

Use this dialog box to create a list of cross references. You can:
- Select references using either line or rung numbers.
- Select the range of lines to include when generating the cross reference list.
- Select a sort order to display selected items.

See *Cross Referencing an Application, p. 113* and *Cross References Viewer, p. 49*.

Opening the Generate Cross References Dialog Box

You can open this dialog box by selecting **Program** → **Cross References** from the main menu. However, once a cross reference list is created for an open application, you will have to select **Tools** → **Generate Cross Reference** from the main menu to regenerate the list.

When to Regenerate

In general, an existing cross reference list will need to be regenerated when any of the basic elements of an application are modified or the scope of the cross reference changed. Examples of these changes include:
- Lines or rungs have been added or deleted.
- Changes to the content or symbols for an application.
- The scope of the lines to include in the cross reference list has changed.
- The list of elements to include in the cross reference list has changed.

Reference By

This box determines whether you want the line references to be by List line number or by Ladder rung number. By default, Ladder **Rung Number** is selected.

Range

You can specify a range of lines by clicking on **Start** and entering the starting line number of a range in the **Start** box and an ending line number in the **End** box. The default selection is **All**: the cross reference list will use all lines of the application.

Sort By

This selection determines the sort order to use when the cross reference list is displayed. These values correspond to the options in the **Tools** menu. By default, **Operand** is used to determine the sort order.
When the list is sorted by operand or symbol, a separation line is used to frame the information to make it easier to read.

Elements

These boxes allow you to narrow the scope of the program items in you application to be displayed. For example, you can choose to list program lines or rungs that have timer function blocks by only selecting the **Timers** check box.

Generating a Cross Reference List

To generate a cross reference list:

Step	Action
1	In **Reference By**, select either rung or line numbers to use as line references.
2	Select a **Range** to use for generating the cross references. **Options**: Select **All Elements** to include all lines in the program. Select **Start** to set a range by entering a **Start** line number and an **End** line number.
3	In **Sort By**, select the item to use for the sorting order. **Options**: Operand, Symbol, Operator, or Line/Rung Number.
4	In **Elements**, select the items in the application to be displayed in the cross-reference list. Select **All Elements** to check all selections.
5	Select **OK** to generate the cross-reference list and close the dialog box. Select **Cancel** to close the dialog box without changes.
6	If there are no occurrences for the selected elements, an information box is displayed with the following message: Cross Reference Empty. Select **OK** to close the information box.
7	If there are occurrences for the selected elements, the *Cross References Viewer, p. 49* is opened displaying the cross-reference list.

3.6 Printing Applications

At a Glance

Overview

This section provides procedures for printing TwidoSoft applications.

What's in this Section?

This section contains the following topics:

Printing an Application

Overview

TwidoSoft allows you to print an entire application or only a specific part of an application.

Note: Print settings are associated and saved along with the Twido application project file "*filename*.twd".

Basic Steps

There are 8 basic steps for printing an application:

Step	Action
1	Define the appropriate print margins, headers and footers (See *The "Documentation" Dialog Box, "Folder" Option, p. 118*).
2	Specify the content for the cover page, add comments (See *The "Documentation" Dialog Box, "Cover Page" Option, p. 119*).
3	Record the application history: date, author, version and comments (See *The "Documentation" Dialog Box, "History" Option, p. 120*).
4	View the content of the file to print (See *The "Documentation" Dialog Box, "Contents" Option, p. 121*).
5	Define the configuration of your file (See *The "Documentation" Dialog Box, "Configuration" Option, p. 122*) and select the hardware configuration, memory objects, memory report and application.
6	Select the type of language used (See *The "Documentation" Dialog Box, "Program" Option, p. 124*).
7	Define the print order for the symbols (See *The "Documentation" Dialog Box, "Symbols" Option, p. 125*).
8	Define the print parameters for the cross references (See *The "Documentation" Dialog Box, "Cross reference" Option, p. 126*).

The "Documentation" Dialog Box, "Folder" Option

Introduction

This option is the most important for printing a Twido application. If this option is:
- unchecked, printing will not take place,
- checked, the user is able to select the header, footer and margins (layout).

Note: If no option is checked, the "Print" button is grayed, making it impossible to print an application.

Description

The table below provides a description of the **Folder** screen:

Zone	Fields	Description
Header	...	all drop-down menus can display the following information: ● Print date, ● Creation date, ● Modification date, ● Project name, ● Free, ● Nothing, ● Page i/n ● Page i, ● TwidoSoft version.
Footer	...	see description of the **Header** zone.
Margins	Left, Right, Top, Bottom	use **margins** settings to select page layout for each page.
	Backup	used to store the current settings for future printing operations. The settings are saved to the twido.ini file
	Restore	used to update the print parameters.

Note: The **margin** parameter unit depends on your computer's regional settings:
- in French, the unit is cm,
- in English, the unit is inches,

The "Documentation" Dialog Box, "Cover Page" Option

Introduction

Use this option to set up the cover page of the folder for the printing application.

Description

The table below provides a description of the **Cover Page** screen:

Zone	Fields	Description
...	Author	the name of the project author,
	Print date	check the box if you want the print date to appear on the cover page,
	Service	name of the service that created the project,
	Issue	current version of the project,
	Destination	PLC type used,
	Industrial Property	name of the project's owner company,
	Comments	add your comments.

The "Documentation" Dialog Box, "History" Option

Introduction

If you have a Twido application that you need to keep a record of when it changes, use the History option and fill out the Date, Author, Version and Comments fields when you want to print the report.

Note: Any changes made to history fields are saved along with the Twido application project file under "*filename*.twd".

Description

The table below provides a description of the **History** dialog box:

Zone	Fields	Action
...	Date	Enter the Twido project's date.
	Author	Enter the author's name for this version of the Twido project.
	Version	Enter the Twido project's version.
	Comments	Add your comments.

The "Documentation" Dialog Box, "Contents" Option

Introduction

Use this option to display the contents in the folder.
When you choose this option, the contents appears in the right-hand part of the dialog box. The contents includes all the options of the **Documentation** dialog box. If this option is checked, the page number appears, otherwise it is marked **not requested**.

Note: The size of the contents displayed on screen is non contractual. In the contents, the header and footer only appear when printed, and cannot be seen on screen.

The "Documentation" Dialog Box, "Configuration" Option

Introduction

Use this dialog box to select the configuration data you want to print.

Hardware configuration

The table below provides a description of the **Hardware configuration** screen:

Zone	Fields	Description
Base	Reference number	gives the reference number of the Twido PLC used (e.g.: TWDLCAA24DRF),
	Description	gives the description of the PLC used (e.g.: 14-input compact base controller, etc.),
	Inputs/Outputs	shows whether inputs, outputs or both can be printed,
	All/used	shows whether ALL I/Os can be printed, or just those that are used,
Expansion bus modules	Reference number	see description of the **Base** zone,
	Description	
	Configuration	• if an AS-Interface module is used: a list of all the AS-Interface slaves configured on the bus is printed, • if an I/O module is used: a list of the I/Os of the expansion module is printed,
	Inputs/Outputs	see description of the **Base** zone,
	All/used	
Communication	Serial ports	used to print its characteristics and configuration,
	Modem	
	Ethernet	
Options	RTC	used to print the value of the clock correction factor,
	Cartridge	used to print the reference number and description of the cartridge,

Memory Objects Configuration

The table below provides a description of the **Memory objects configuration** screen:

Zone	Fields	Description
Simple objects	...	used to select simple objects to print
Function blocks objects	...	used to select function blocks objects to print
Method	All allocated objects	used to print all objects allocated in the symbols editor.
	All used objects	used to print all objects used by the program.
	Indicate the maximum number of objects able to be configured	the number indicated corresponds to the characteristics of the PLCs used (see the maximum column in the **Edit** tab of the **Memory usage**),

Memory Usage Configuration

The table below provides a description of the **Memory objects configuration** screen:

Zone	Fields	Description
Print memory usage	Print the memory usage statistics	print the data given in the **Display** tab of the **Memory Usage**
	Print the allocation used for each object	print the data given in the **Edit** tab of the **Memory Usage**

Application Configuration

The table below provides a description of the **Application configuration** screen:

Zone	Fields	Description
Application configuration	...	prints the event tasks and other application-specific information.

The "Documentation" Dialog Box, "Program" Option

Introduction

Use this dialog box to print all or part of a **Ladder** or **List** program.

Description

The table below provides a description of the **Memory objects configuration** screen:

Zone	Fields	Description
Program	Ladder	select the type of used program to print.
	List	
Ladder	Range	choose to print all or just a part of the program.
	Attributes	choose to print the addresses with associated symbols, just the addresses or just the symbols. If you check "Addresses + Symbols + Comments", the associated comments will be printed under each network between two lines.
List	Range	choose to print all or just a part of the program.
	Attributes	choose to print the addresses or symbols. If you check "Addresses + Symbols + Comments", the associated comments will be printed under each program line.

The "Documentation" Dialog Box, "Symbols" Option

Introduction Use this dialog box to select the sorting order for printing symbols.

Description The table below provides a description of the **Symbols configuration** screen:

Zone	Fields	Description
Sort By	Symbol name	this option is selected by default to sort operand names by alphabetical order.
	Resolved address	This option sorts operands per variable.

The "Documentation" Dialog Box, "Cross reference" Option

Introduction

Use this dialog box to print all or part of the elements in an application. By default, all elements are selected.

Description

The table below provides a description of the **Cross References** dialog box:

Zone	Fields	Description
Program elements	All	choose this option to select all the elements of the program.
	Selection	choose this option to print only the elements selected in the **Elements** zone.
Elements	...	this option can only be used where you have chosen **Selection** in the **Program elements** zone. Choose the elements you wish to appear on the print-out.
Referenced by	Rung number	check this box if your program is in **Ladder**. Here, the use of an element will be reference by the rung number.
	Line number	check this box if your program is in **List**. Here, the use of an element will be reference by the line number.
Range	All	check this box to print the elements of the entire program.
	Start/End	check this box to print the elements of part of the program (by rung or line number).
Sort by	Drop-down list: - Address - Line / Rung Number - Operator - Symbols	choose a sort parameter from the drop-down list according to how you want to see the selected elements sorted.

Hardware and Software Configuration

4

At a Glance

Overview

This chapter provides details on how to configure hardware and software for Twido programmable controllers.

What's in this Chapter?

This chapter contains the following sections:

4.1 Configuration Overview

At a Glance

Overview

This section provides an overview of configuring hardware and software for Twido programmable controllers.

What's in this Section?

This section contains the following topics:

Configuring Hardware and Software

Introduction

Configuring Twido programmable controllers consists of selecting options for the hardware and software resources of the controller. These resources can be configured at any time while creating a program.

Hardware resources

Hardware resources are: the controller itself, hardware that connects to the controller, and the connections to the hardware.
Types of hardware resources:
- Base and Remote controllers
- Expansion I/O
- AS-Interface V2 bus interface module and its slave devices
- CANopen fieldbus master module and its slave devices
- Options

See *Configuring Hardware, p. 136*.

Software resources

Software resources consist of configurable and non-configurable functions:
- Configurable software functions
  Function blocks (also called variables) are blocks of addresses that are created in memory to perform special functions that can be used by a program. For example, when a counter function block is configured, memory addresses in the controller are assigned to hold values that represent the associated characteristics of a hardware counter: current count, preset count, and so on.
- Non-Configurable software functions
  These are internal memory blocks: system bits and words, memory bits and words, and network exchange words.

Configurable Function Blocks

The following is a list of configurable function blocks:
- Timers
- Counters
- Fast counters
- Very fast counters
- Drum controllers
- LIFO/FIFO registers
- PLS/PWM pulse generators
- Schedule blocks

Constants are also configurable, but they are not function blocks.
See *Configuring Software, p. 214*.

Methods for Configuring Controller Resources

Introduction

Controller resources can be configured using the following:

1. **Hardware** or **Software** menus from the main menu
2. *Application Browser, p. 27*
3. *Ladder Editor, p. 38* or the *Ladder Viewer, p. 36*
4. *Object Browser, p. 47* in the *Symbol Editor, p. 45*
5. *Configuration Editor, p. 44*

Direct Configuration

The Configuration Editor provides a comprehensive look at all configurations in one window. The other methods allow a quick and easy method to directly configure one resource at a time while creating an application.

Configuring From the Main Menu

To configure controller resources individually and directly from the main menu:

- Select the **Hardware** or **Software** menu.
- Select a specific hardware or software resource to configure.

Configuring From the Application Browser

To configure controller resources from the Application Browser:

- Right-click on the base controller model number to change the base controller or configure base I/O.
- Right-click on **Hardware** to add options.
- Right-click on **Expansion** bus to add expansion modules.
- Right-click on the Serial Port entries to edit the controller communications setup or add a Remote controller.

Configuring From the Ladder Editor/Viewer

To configure software resources from the Ladder Editor or the Ladder Viewer, double-click on a function block to display the associated configuration dialog box.

Configuring From the Symbol Editor

To configure software resources from the Symbol Editor:

- Insert or edit a symbol to display the **Object Browser** dialog box.
- Click on the **Configure** button to configure a function block (the Configuration Editor must not be open).

Using the Configuration Editor

Introduction

The following resources can be configured directly from the Configuration Editor:

- Software resources
  All software resources: timers, counters, constants, drum controllers, fast counters, LIFO/FIFO registers, PLS/PWM, schedule blocks, very fast counters and event reports.
- Hardware resources
  Some hardware resources: digital inputs and outputs, expansion modules (analog module, AS-Interface master module and CANopen master module) and controller communications setup.
- Program settings
  Scan time and application name.

Note: When canceling changes made in the Configuration Editor, all configuration changes made since the Configuration Editor was opened are discarded.

Configuring Resources From Configuration Editor

How to configure a resource using the Configuration Editor:

Step	Action
1	With the Configuration Editor open, select a resource from the toolbar or from the **Hardware** or **Software** menus from the main menu.
2	Select a specific entry from the list of the selected resources in the display. If necessary, click on the Configure button  and use the **Previous** and **Next** buttons to locate a specific selection in the dialog box.
3	Open the configuration dialog box (if not already open) for the selected resource by doing one of the following: double-click on the resource in the Configuration Editor, click on the configure button  on the toolbar, or select **Tools** → **Edit** from the main menu. **Note:** Double-clicking on the resource in the Configuration Editor is only applicable for timers, counters, constants, and LIFO/FIFO registers.
4	Make selections in the resource configuration dialog box. Save changes when complete.

Saving Changes

How to save changes to a configuration from the Configuration Editor:

Step	Action
1	When changes are complete, either select **Accept Changes** from the **Tools** menu, or click on the accept button in the tool bar. An **Exit Warning** dialog box is displayed.
2	Press **OK** to accept changes and close the Configuration Editor.
3	Press **Cancel** to cancel acceptance and return to the Configuration Editor.

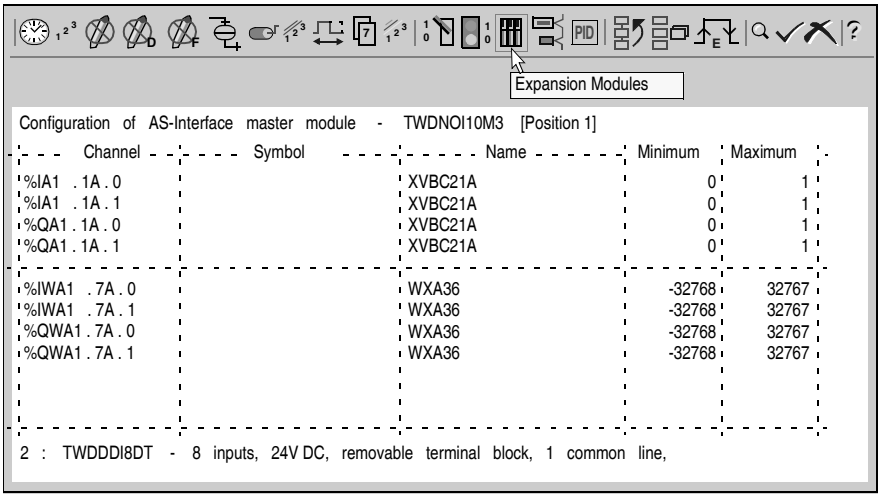
Saving Changes when Closing the Configuration Editor

To save changes to a configuration when closing the Configuration Editor:

Step	Action
1	Close the Configuration Editor by clicking on the Close button. An **Exit Warning** dialog box is displayed.
2	Press **Yes** to accept changes and close the Configuration Editor.
3	Press **No** to discard changes and close the Configuration Editor.

Display example for AS-Interface expansion modules

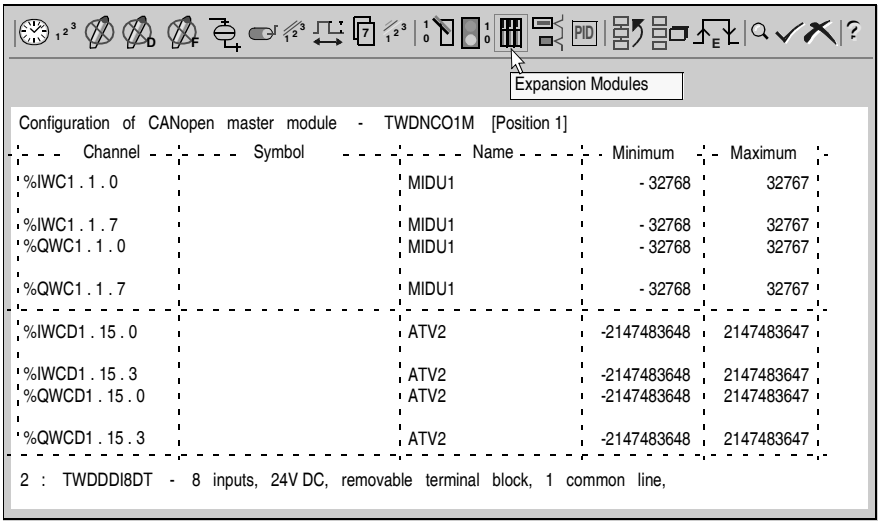
Click "Expansion modules" from the toolbar to display the configuration of **all** modules in the expansion bus.
Example:



For an AS-Interface master module, the configuration editor displays a list of the slaves planned for the project with their addresses, symbols, names and limit values.

Display example for CANopen expansion module

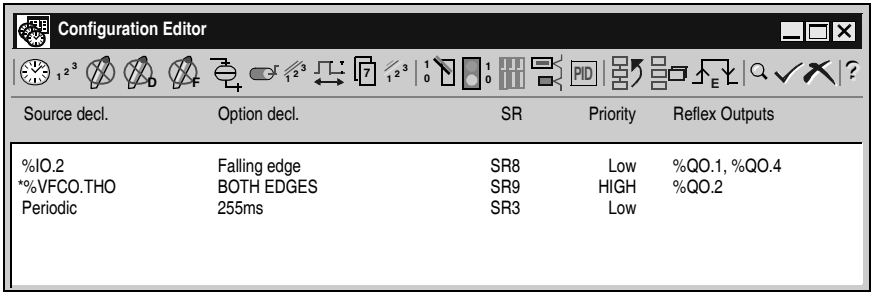
Click "Expansion modules" from the toolbar to display the configuration of **all** modules in the expansion bus.
Example:



For a CANopen master module, the configuration editor displays a list of the slaves planned for the project with their addresses, symbols, names and limit values.

Event report

Click the  icon to display the following event report:



Over its five columns, the event report shows: the source of the events, the triggering option (on rising/falling edge), the executed subroutine number, the priority and the updated outputs.
The report is updated when an event is added, deleted or modified. The line corresponding to the highest-priority event is printed in capital letters and preceded by an asterisk.

Validating a Configuration

Introduction

TwidoSoft automatically validates configurations by checking for differences between the application configuration data and the hardware selected. As you select configuration options, TwidoSoft adjusts options available for configuration selections and displays alerts to warn of incorrect or invalid configuration entries. Configurations do not require a separate validating command since validation is an automatic, built-in TwidoSoft feature.

Validating Examples

Examples of TwidoSoft validating a configuration:

- When changing the base controller, any hardware not supported by the new choice is deleted from the Application.
- Alerts you to incorrect or invalid configuration entries. For example, only available options are displayed when adding an option to a configuration.
- Won't allow you to configure more than one communications port as Remote Link.
- Checks for sufficient memory for the application.
- Displays errors for use of function blocks or I/O points that do not exist.

4.2 Configuring Hardware

At a Glance

Overview

This section contains detailed procedures for using TwidoSoft to configure Twido hardware in an application.

What's in this Section?

This section contains the following topics:

Device Properties

Introduction

The **Device Properties** dialog box displays information about Twido hardware. Information is available for the following Twido hardware:

- Controllers
- Expansion I/O modules
- Hardware options (such as RTC, communications, operator display, and backup memory cartridges)

Displaying the Properties Information

To view the Device Properties dialog box, right-click on one of the following hardware items in the *Application Browser, p. 27* and select the Properties menu:

- Controller Model
- Expansion I/O module
- Hardware option

The **Device Properties** dialog box is displayed with the Twido part number and a brief description of the hardware item.

Change Base Controller

Introduction

Use the **Change Base Controller** dialog box to select another base controller model for an application.The base controller can be changed but not deleted from the application structure.

Changing Base Controller

To select another base controller model:

Step	Action
1	Select **Hardware** → **Change Base Controller** from the main menu, or right-click on the base controller model in the Application Browser and select **Change Base Controller** from the shortcut menu.
2	Select a different model in the list window of the **Change Base Controller** dialog box, and press **Change** (or double-click on the item in the list window) to change the base controller to the new model.
3	Press **Done** to close the dialog box without changing the base controller.
4	If the new base controller model does not support the hardware configuration of the previous model, a dialog box is displayed allowing you to cancel the change. Press **Yes** to change the base controller. Those hardware configuration settings not supported by the new base model will be removed from the hardware configuration in the Application Browser.
5	Press **No** to cancel the change and return to the dialog box and make another selection or keep the current model.

Functional Level Management

Introduction

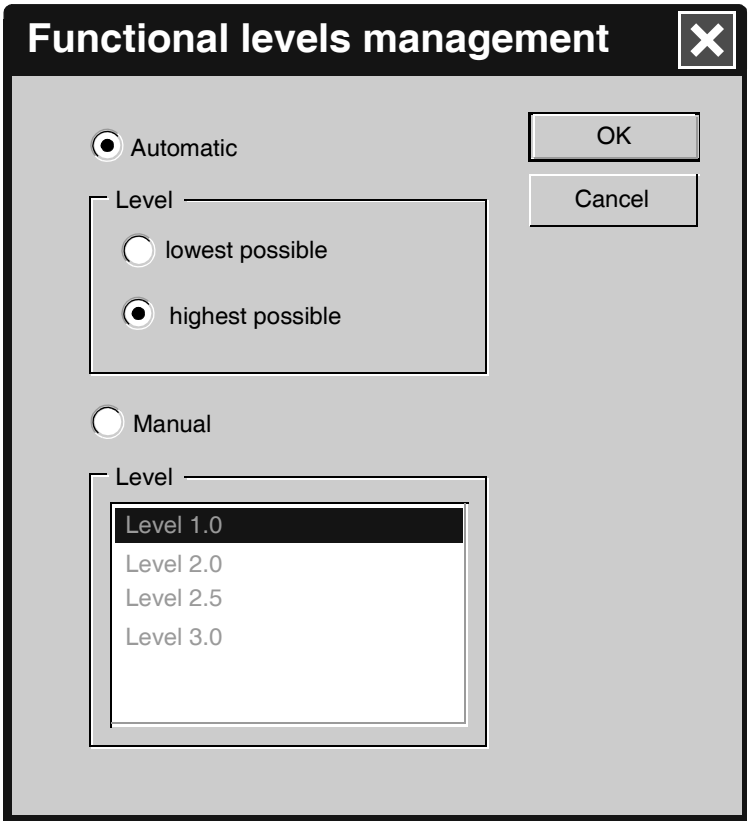
Your installation may combine different Twido controllers with different firmware versions, thus having different capability levels.
Functional level management, as proposed in TwidoSoft, allows you to determine the functional level you wish for your application.
In case Twido functional level would be lower than expected and the level required could not be reached, a message would invite you, at connection, to adjust your wish - either to update Twido firmware, or to lower the functional level and do the necessary changes to your application.

Access to Functional Levels

There are three different ways to access functional levels:
- Automatically, at creation of an application.
- Manually, via the application browser or via the main menu.

Automatically	In TwidoSoft, every time you create an application, the **Functional levels management** window pops up.
Manually	Right-click a base controller in the Application browser and a menu opens with **Functional levels management...** item. Select it.
	Select **Hardware** → **Functional levels management...** from the main menu. **Note:** If no application has been selected, this menu item is grayed and cannot be accessed.

In any case, you access the following window:



Functional Levels Description

Functional levels management window is composed of:

Element	Description
Automatic button	Choosing **Automatic** allows you to select between two levels, lowest and highest, and you do not need to know about the history. **Note:** When **Automatic** is selected, Manual level listbox is grayed and cannot be accessed.
Level box: **Lowest possible** button **Highest possible** button	You define the minimum configuration that matches your needs. You get the most of your controller possibilities.
Manual button	Choosing **Manual** allows you to select between four different levels, from 1.0 to 3.0, when you know precisely what you want.
Level box: **Level 1.0 - 3.0** listbox	Levels 1.0 to 3.0 correspond to Twido firmware possible versions. **Note:** When **Manual** is selected, Automatic level listbox is grayed and cannot be accessed.

Note: When a new application is created, or when an existing application is loaded, default values are: **Automatic** and **Highest possible**.

Input Configuration

Introduction

Use the **Input Configuration** dialog box to configure base controller digital inputs.

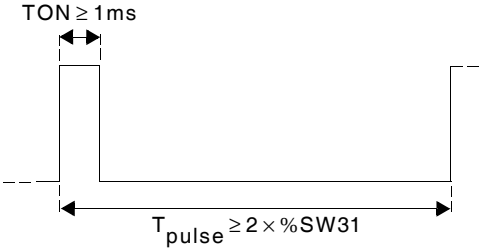
Input Configuration Dialog Box

The dialog box is organized similar to a table or spreadsheet and consists of the following columns which are described starting at the left-most column.

Column	Description	Function
Inputs (no heading)	Address of each input digital input.	For display only. Number of inputs determined by the type of base controller.
Used By	Lists any function blocks using this input, or indicates whether the input is used to trigger an event.	For display only. If more than one function block is using an input, text is displayed in red. Displays: • where input is used, function blocks or program logic, if any. • **Event** indicates whether the input is used to trigger an event
Symbol	Displays any symbols assigned for inputs.	For display only. Symbols are created by using the *Symbol Editor, p. 45*.
Filtering	Configuring Input Filtering	Input filters reduce the effect of noise on a controller input.
Retained?	Input latching setting.	Latching allows incoming pulses with amplitude widths shorter than the controller scan time to be captured and recorded.(1)
Run/Stop?	Run/Stop input setting.	The Run/Stop input can be used to run or stop a program in the controller.
Deactivation	Configuration of an event's trigger.	Only used to configure whether or not an event is triggered on an input.
High Priority	Configuration of an event's priority.	The High Priority input can be used to make an event high priority (triggered).
SR Number	Configuration of the section number.	The SR number is used to assign a subroutine number to an event (triggered).

1. Note that to allow proper detection of a pulse signal when the latching input option is selected, the pulse width (T_{ON}) and the cyclic period (T_{pulse}) must meet the following two requirements:
   - $T_{ON} \geq 1$ ms
   - The input signal cyclic period must follow the Nyquist-Shannon sampling rule stating that the cyclic period (T_{pulse}) of the input signal must be at least twice the maximum program scan time (%SW31):
     $T_{pulse} \geq 2$ x %SW31.
     Note: If this condition is not fulfilled, some pulses may be missed.

The following figure shows the input signal requirements when latching input is



used:

Configuration Guidelines

Guidelines for configuring inputs:

- The Filtering, Retained, Run/Stop, Deactivation, High Priority and SR Number inputs are the only elements from this dialog box that can be configured.
- For latching inputs, %I0.2, %I0.3, %I0.4 and %I0.5 are the only configurable inputs.
- Only inputs %I0.2, %I0.3, %I0.4 and %I0.5 can be associated with an event.
- Only one input from %I0.2, %I0.3, %I0.4 and %I0.5 can be defined with High Priority.
- All inputs can be defined for filtering or as Run/Stop.
- Only one input can be configured as Run/Stop. This Run/Stop bit can be used in your application logic.
- Inputs configured as filtered can not be used as latched inputs.
- The inputs associated with an event cannot be latched.

Configuring Input Filtering

To configure filtering for an input:

Step	Action
1	Click on the **Filtering** edit box of an input.
2	Click on the pull-down menu and select an option: **No Filtering**, **3 ms**, or **12 ms**.
3	If there are no other changes, click on **OK** to close the dialog box and apply the new configuration.

Configuring Input Latching

To configure latching for an input:

Step	Action
1	Click on a check box in the **Latch?** column for an input.
2	Setting a check mark indicates that the input is a latched.
3	If there are no other changes, click on **OK** to close the dialog box and apply the new configuration.

Configuring a RUN/STOP Input

To configure an input to be used for the Run/Stop function:

Step	Action
1	Click on a check box in the **Run/Stop?** column for an input.
2	Setting a check mark indicates that the input is used as the Run/Stop input.
3	If there are no other changes, press **OK** to close the dialog box.
4	If there are no other changes, click on **OK** to close the dialog box and apply the new configuration.

Configuration of an event's trigger.

To configure whether or not an event is triggered on an input:

Step	Action
1	Click on a **Deactivation** zone for an input.
2	Open the drop-down menu and select an option: • **Not used** for there to be no event attached to this input, • **Rising edge** for an event to be triggered when the value of the input switches to 1, • **Falling edge** for an event to be triggered when the value of the input switches to 0, • **Both edges** for an event to be triggered when the value of the input varies.
3	Configure the SR Number to be associated with the input if a trigger is used (see Configuration of the section number).
4	If there are no other changes, click on **OK** to close the dialog box and apply the new configuration.

Configuration of an event's priority.

The priority of an event can be configured for an input only where its **Deactivation** option is other than **Not used**. To configure an event as high priority:

Step	Action
1	Click on a check box in the **High Priority** column for an input.
2	Setting a check mark indicates that the input triggers a high-priority event. Note: If another input's box is checked, this will be automatically unchecked (as only one event can be high priority).
3	If there are no other changes, click on **OK** to close the dialog box and apply the new configuration.

Configuration of the section number.

The section number can be configured for an input only where its **Deactivation** option is other than **Not used**. To configure an input's section number:

Step	Action
1	Click on an **SR Number** zone for an input.
2	Open the drop-down menu, and select the number of the subroutine to be executed when an event is triggered on this input: • **0 to 15** (maximum 16 subroutines) for the Twido 16 controller, • **0 to 63** (maximum 64 subroutines) for Twido 20, 24 and 40 series controllers. Note: If the selected section number does not correspond to any implemented subroutine, or if the section number is already used by another event, the numbers are nevertheless available from the drop-down menu. Warning messages alert the user to these situations.
3	If there are no other changes, click on **OK** to close the dialog box and apply the new configuration.

Output Configuration

Introduction

Use the **Output Configuration** dialog box to configure base controller digital outputs.

Output Configuration Dialog Box

The dialog box is organized similarly to a table or spreadsheet and consists of the following columns which are described starting with the left-most column.

Column	Description	Function
Outputs (no heading)	Address of each digital output.	For display only. Number of outputs determined by the type of base controller.
Symbol	Displays any symbols assigned for outputs.	For display only. Symbols are created by using the *Symbol Editor, p. 45*.
Controller Status?	Controller Status output.	Indicates controller state. If the controller is in RUN, output is set to one. If the controller is in STOP or in an error, output set to zero. Can be used for safety circuits external to the controller for power supplies.
Used By	Lists any function blocks using this output.	For display only. If more than one function block is using an output, text is displayed in red. Displays where output is used, function blocks or program logic, if any.

Configuration Guidelines

Guidelines for configuring outputs:

- Controller Status is the only configurable item in this dialog box.
- The only configurable outputs are %Q0.1, %Q0.2, and %Q0.3.
- Only one output can be configured as Controller Status.

Configuring Controller Status

To configure an output to be used as Controller Status:

Step	Action
1	Click on a check box in the **Controller Status** column.
2	A check mark indicates that the output will be used for the Controller Status output.
3	Click on **OK** to close the dialog box and apply the new configuration.

Add Module

Introduction

Use the **Add Module** dialog box to add an Expansion I/O Module (including one AS-Interface interface module) to the configuration of the open application. Up to seven expansion modules can be added to a base controller (depending on the controller model, maximum 2 AS-Interface expansion modules and 1 CANopen master module).
There are two slightly different procedures for using this dialog box, whether it is opened from the Application Browser or the **Hardware** menu.

Expansion Address

Each expansion I/O module is assigned an **Expansion Address** when added to the application configuration. The addresses are numbered in the order the modules are added, starting with 1 and ending with the highest number of allowable expansion modules.
In the **Add Module** dialog box, the Expansion Address list contains:

- All configured module addresses.
- The next available address (selected by default).

Renumbering of Addresses

If you insert a module at an already configured address, the module at that address and all other modules with higher addresses are moved up one address. The selected module is then added at the selected address. For example, adding a module at address 1 with modules already at addresses 1 and 2 results in a renumbering of the addresses:

- New module is address 1
- Module 1 is now address 2
- Module 2 is now address 3

Adding a Module

To add an Expansion I/O Module:

Step	Action
1	If you use the Application Browser: right-click the **Expansion Bus** then select **Add a Module** from the shortcut menu. If you use the **Hardware** menu: select **Hardware** → **Add a Module** from the main menu. **Result:** The **Add Module** dialog box opens.
2	Select and highlight a module in the **Module** window. **Result:** A brief description is displayed in the **Description** box.
3	Click **Add** to add the selected module to the application. **Result:** The selection is immediately displayed in the Application Browser.
4	To add more than one module without closing the dialog box, continue selecting modules and clicking **Add**.
5	When selections are completed, click **Done** to close the dialog box.

Delete Module

Introduction

Use the **Delete Module** dialog box to remove an Expansion I/O Module from the configuration of an open application. There are two slightly different procedures for using this dialog box, whether it is opened from the Application Browser or the **Hardware** menu.

Deleting a Module using the Application Browser

Using this procedure, the dialog box opens with a default selection. To delete an Expansion I/O Module from the *Application Browser, p. 27*:

Step	Action
1	Right-click the module under the **Expansion Bus**. If the module is not displayed, double-click the Expansion Bus to display the list of installed Expansion I/O modules.
2	Select **Delete** from the shortcut menu.
3	The **Delete Module** dialog box is displayed with the selected module highlighted by default in the **Module** list. Only one module can be selected at a time.
4	Select **OK** to delete the selected module and close the dialog box. Select **Cancel** to close the dialog box without changes.

Deleting a Module using the Hardware Menu

Using this procedure, the dialog box does not open with a default selection. To delete an Expansion I/O Module from the **Hardware** menu:

Step	Action
1	Select **Hardware** → **Delete a Module** from the main menu.
2	The **Delete Module** dialog box is displayed without any modules highlighted in the **Module** list window. Click a module in the list window to select it. Only one module can be selected at a time.
3	Select **OK** to delete the selected module and close the dialog box. Select **Cancel** to close the dialog box without changes.

Renumbering of Expansion Addresses

When an module is deleted, the higher addresses are renumbered so that there is no gap in the numbering sequence. For example, if the module at expansion address #4 is deleted, then the module at address #5 is renumbered to #4. If there are other modules with higher addresses, they will also be renumbered in the same way.

Configure Module - Choosing a Module

Introduction

Use the **Configure Module** dialog box to manage the parameters for configurable modules. The dialog box lists configurable modules that have been previously installed on the expansion bus.
This dialog box is only available when the PC is not connected to the controller and there are configurable expansion modules on the expansion bus.

Choosing a Module using the Application Browser

To choose a configurable module on the expansion bus:

Step	Action
1	Select a module under Expansion Bus.
2	Right-click **Configure**. **Result:** The **Configure Module - (Module ref. and position)** dialog box directly opens. (See *Configure Module - Parameters, p. 151*)

Choosing a Module using the Hardware menu

There are two dialog boxes required to configure modules: **Configure Module - Choose Module** then **Configure Module - (Module ref. and position)**. To choose a configurable module on the expansion bus:

Step	Action
1	Select **Hardware** → **Configure a Module** from the main menu. **Result:** The **Configure Module - Choose Module** dialog box is displayed.
2	Select a module from the list of configurable modules. **Result:** The **Description** box at the bottom of the dialog box lists details about the module.
3	Click **OK** to select the highlighted module and close the dialog box. **Result:** The **Configure Module - (Module ref. and position)** dialog box opens. (See *Configure Module - Parameters, p. 151*)

Configure Module - Parameters

Introduction

The **Configure Module - (Module ref. and position)** dialog box is used to configure parameters for a selected I/O module.
The following analog modules must be configured before they can be used by an application program:

- TWDALM3LT
- TWDAMM3HT
- TWDAM2HT
- TWDAMO1HT
- TWDAVO2HT
- TWDAMI4LT
- TWDAMI8HT
- TWDARI8HT

The TWDNOI10M3 AS-Interface and TWDNCO1M CANopen master modules must be configured where there are slaves on the AS-Interface bus or on the CANopen fliedbus. The configuration screens of the AS-Interface master and the CANopen master are different from the one described below (see *Description of the AS-Interface bus configuration screen* and *Description of the CANopen fieldbus configuration screen* in the manual "Twido programmable controllers: Software setup guide" – reference number **TWDUSE10AE**).

Title bar and Contents

The Title bar displays the module reference and its position on the expansion bus.
The upper part of the dialog box shows a **Description** zone.
A table shows: **Address**, **Symbol**, **Type**, **Range**, **Minimum**, **Maximum** and **Units**

- In TWDAMI4LT and TWIDAMI8HT, the table is preceded by an **Input type** list box.
- In TWDAVO2HT and TWDAMI8HT, the **Type** column is replaced by a **Used** column with check boxes.
- In TWDARI8HT, each channel (0-7) is configured individually within a tab, in which you can choose either the **Chart** or **Formula** configuration method. The table can be seen in a **Recap** tab.

Description

The **Description** zone displays a short summary of the module.

Address

Each row of the spreadsheet represents either an input or output channel of the module.
The addresses of each of these are identified in the following table, where "i" is the location of the module on the expansion bus.

Module Name	Address
TWDALM3LT	2 Inputs (%IWi.0, %IWi.1), 1 Output (%QWi.0)
TWDAMM3HT	2 Inputs (%IWi.0, %IWi.1), 1 Output (%QWi.0)
TWDAMI2HT	2 Inputs (%IWi.0, %IWi.1)
TWDAMO1HT	1 Output (%QWi.0)
TWDAVO2HT	2 Outputs (%QWi.0, %QWi.1)
TWDAMI4LT	4 Inputs (%IWi.0 to %IWi.3)
TWDAMI8HT	8 Inputs (%IWi.0 to %IWi.7)
TWDARI8HT	8 Inputs (%IWi.0 to %IWi.7)

Symbol

This is a read-only display of a symbol, if assigned, for the address.

Input type and/or Type

This identifies the mode of a channel. The choices depend on the channel and type of module.
For the TWDAMO1HT, TWDAMM3HT and TWDALM3LT, you can configure the single output channel type as:

Type
Not used
0 - 10 V
4 – 20 mA

For the TWDAMI2HT and TWDAMM3HT, you can configure the two input channel types as:

Type
Not used
0 - 10 V
4 – 20 mA

For the TWDALM3LT, you can configure the two input channel types as:

Type
Not used
Thermocouple K
Thermocouple J
Thermocouple T
PT 100

For the TWDAVO2HT, there is no type to adjust.
For the TWDAMI4LT, you can configure the four input types as:

Input type	Type
Voltage	Not used 0-10 V
Current	Not used 0-20 mA
Temperature	Not used PT 100 PT 1000 NI 100 NI 1000

For the TWDAMI8HT, you can configure the eight input types as:

Input type
0 - 10 V
0 - 20 mA

For the TWDARI8HT, you can configure each input channel (0-7) individually, from the **Operation** field in the lower part of the window. Directly choose a **Mode**, and a **Range**, if needed. You can then view a summary of all information in the Recap tab, with a **Type** column showing:

Type
Not used
NTC / CTN
PTC / CTP

⚠ CAUTION

EQUIPMENT DAMAGE

If you have wired your input for a voltage measurement, and you configure TwidoSoft for a current type of configuration, you may permanently damage the analog module. Ensure that the wiring is in agreement with the TwidoSoft configuration.

Failure to follow this instruction can result in injury or equipment damage.

Range

This identifies the range of values for a channel. The choices depend on the specific type of channel and module.
Once the **Type** is configured, you can set the corresponding **Range**. A table shows the **Minimum** and **Maximum** values accepted - either fixed or user-defined - together with the **Unit**, if needed.

Range (NTC sensors)	Minimum	Maximum	Units	I/O Analog Modules
Normal	0	4095	None	TWDALM3LT TWDAMO1HT TWDAMM3HT TWDAMI2HT TWDAMI4LT
	-2048	2047		TWDAVO2HT
	0	1023		TWDAMI8HT TWDARI8HT

Range (NTC sensors)	Minimum	Maximum	Units	I/O Analog Modules
Custom	User defined with a min. of -32768	User defined with a max. of 32767	None	All I/O Analog Modules
Celsius	-1000	5000	0.1°C	TWDALM3LT
	Dynamically updated by TwidoSoft according to user-defined parameters			TWDARI8HT
	-2000	6000		TWDAMI4LT (Pt sensor)
	-500	1500		TWDAMI4LT (Ni sensor)
Fahrenheit	-1480	9320	0.1°F	TWDALM3LT
	Dynamically updated by TwidoSoft according to user-defined parameters			TWDARI8HT
	-3280	11120		TWDAMI4LT (Pt sensor)
	-580	3020		TWDAMI4LT (Ni sensor)
Resistance	100	10000	Ohm	TWDARI8HT
	74	199		TWDAMI4LT (Ni100)
	742	1987		TWDAMI4LT (Ni1000)
	18	314		TWDAMI4LT (Pt100)
	184	3138		TWDAMI4LT (Pt1000)

Chart or Formula Method

In TWDARI8HT, each channel (0-7) is configured individually within a tab. Check the **Used** box then choose between **Chart** and **Formula** configuration methods.

- **Chart (graphical) method**
  (**R1**, **T1**) and (**R2**, **T2**) correspond to float format coordinates of two points belonging to the curve
  R1(default 8700) and **R2** (default 200) values are expressed in Ohms.
  T1 (default 233.15) and **T2** (default 398.15) values can have their unit set in the **Unit** list box: **Kelvin** (default), **Celsius** or **Farenheit**.
  Note: Changing the temperature unit after setting the T1 and T2 values will not automatically recalculate T1 and T2 values with the new unit.
- **Formula method**
  Provided you know **Rref**, **Tref** and **B** parameters, you can use this method to define sensor characteristics.
  Rref (default 330) is expressed in Ohms.
  B is default 3569 (min. 1, max. 32767).
  Tref (default 298.15) can have its unit set in the **Unit** list box: **Kelvin** (default), **Celsius** or **Farenheit**.

Here is a table of corresponding min./max. **Tref** values between units:

Unit	Min. value	Max. value
Kevin	1	650
Celsius	-272	376
Farenheit	-457	710

In both Chart and Formula windows, you can import values from another channel in the currently configured channel:

1. Select a channel number out of the **Channel No** box.
2. Press the **Import values** button.

Some error or warning messages can be associated with these windows.

Note: If you start setting values then decide to switch from Chart toFormula or from Formula to Chart, a warning message pops up, explaining that it will revert to default values and that any modified values will be lost.

Add Option

Introduction

Use the **Add Option** dialog box to add a Twido hardware option to the configuration of the open application. Available selections are determined by installed hardware and the base controller model. TwidoSoft automatically displays available options for each controller.

Adding an Option

To add an option:

Step	Action
1	Select and highlight an option in the **Hardware Option** window. A brief description of the option is displayed in the **Description** box.
2	Click on **Add** to add the selected option to the application. The selection is immediately displayed in the Application Browser.
3	To add more than one selection without closing the dialog box, continue selecting options and clicking on **Add**.
4	When selections are completed, click on **Done** to close the dialog box.

Delete Option

Introduction

Use the **Delete Option** dialog box to remove an installed Twido hardware option from the configuration of an open application. There are two slightly different procedures for using this dialog box depending on if it is opened from the Application Browser or the **Hardware** menu.

Deleting an Option using the Application Browser

Using this procedure, the dialog box opens with a default selection. To delete a hardware option from the *Application Browser, p. 27*:

Step	Action
1	Right-click on the option in the Application Browser.
2	Select **Delete** from the shortcut menu.
3	The **Delete Option** dialog box is displayed with the selected option highlighted by default in the **Hardware Option** list. Only one option can be selected at a time.
4	To delete the selected option, click on **OK** to close the box. Select **Cancel** to close the dialog box without changes.

Deleting an Option using the Hardware Menu

Using this procedure, the dialog box does not open with a default selection. To delete a hardware option from the **Hardware** menu:

Step	Action
1	Select **Hardware** → **Delete Option** from the main menu.
2	The **Delete Option** dialog box is displayed without any options highlighted in the **Hardware Option** list window. Click on an option in the list window to select it. Only one option can be selected at a time.
3	To delete the selected option, click on **OK** to close the box. Select **Cancel** to close the dialog box without changes.

Controller Communications Setup

Introduction

Use the **Controller Communications Setup** dialog box to configure serial ports. One or two serial ports may be installed:

- Port 1 (standard)
- Port 2 (optional)

Protocols

Each port can be configured for one of the following protocols:

- Remote Link (see *About Remote Links, p. 171*)
- Modbus (see *About Modbus Links, p. 177*)
- ASCII (see *About ASCII Links, p. 178*)

Only one port at a time can be configured as a Remote Link.

Using the Communications Cable

Port 1 of the controller can be configured for any protocol, but it will automatically be set to the TwidoSoft connection protocol when the PC to controller communications cable is detected. Once the cable is removed, Port 1 will revert back to the configured protocol.
See *Connecting a PC to a Twido controller, p. 14*.

Configuring a Serial Port as a Remote Link

To configure a serial port as a Remote Link (default settings are in bold):

Step	Action
1	**Note:** To setup a remote link network, connect the PC to Serial Port 1 of each controller to configure as a remote link, unless already configured.
2	Select **Port 1** or **Port 2** (if installed) tab in the dialog box.
3	Select **Remote Link** in the **Protocol Type** box.
4	Select an address number in the **Address** box. Select 0 to set the base controller as the **master** of the network of remote links. Select another number between **1** and 7 for a slave remote link. Each remote link must have a unique address.
5	Select **OK** to close the dialog box. The selected serial port in the Application Browser is updated with the Protocol Type and Address. Select **Cancel** to close the dialog box without making changes to the current settings.

Configuring a Serial Port for Modbus

To configure a serial port to use Modbus protocol (default settings are in bold):

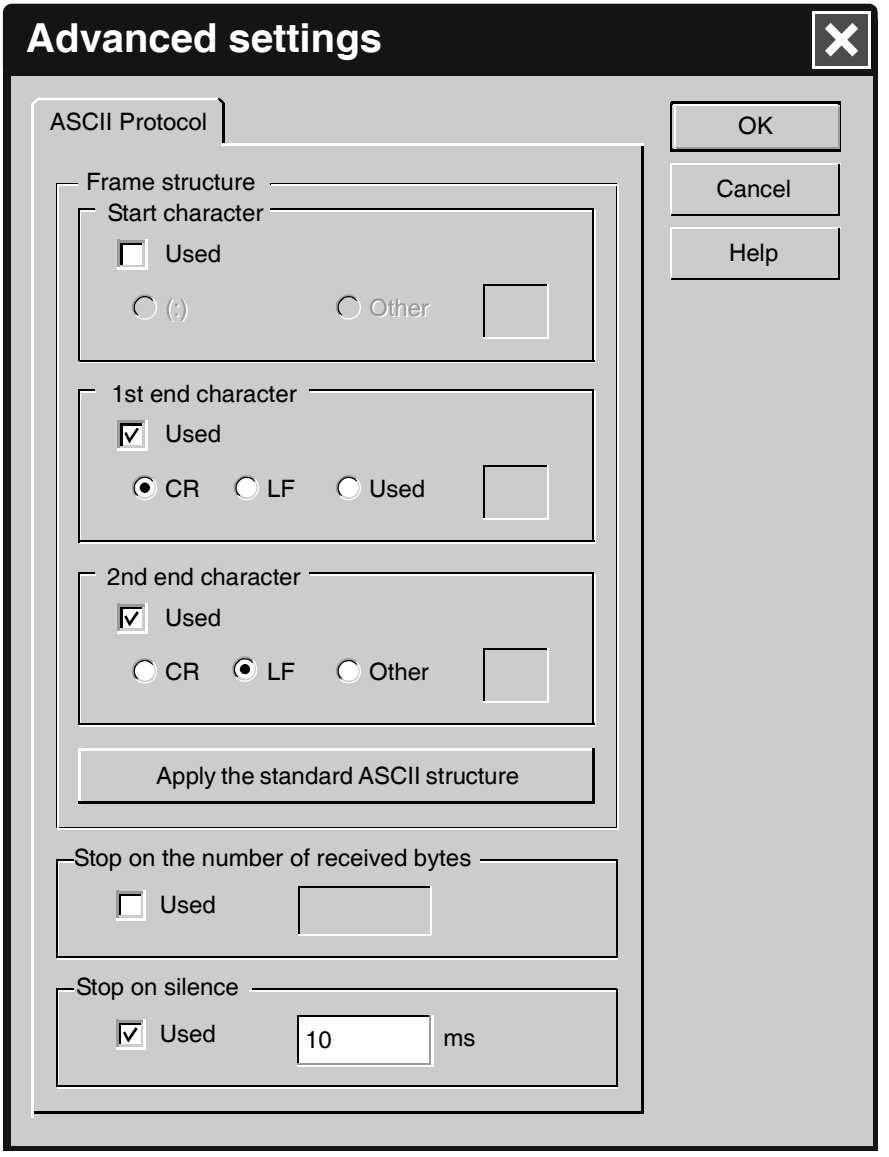
Step	Action
1	Select **Port 1** or **Port 2** (if installed) tab in the dialog box.
2	Select **Modbus** in the **Protocol Type** box.
3	Select an **Address**: **1** - 247.
4	Select a **Baud Rate**: 1200, 2400, 4800, 9600, **19200,** 38400.
5	Select **Data Bits**: 7 (ASCII), **8 (RTU)**. 7 (ASCII) indicates that Modbus ASCII protocol will be used. 8 (RTU) indicates that the Modbus RTU protocol will be used.
6	Select a **Parity**: None, **Even**, Odd.
7	Select **Stop Bits**: **1**, 2 bits.
8	Enter a value for **Response Timeout** from 0 to 255. Default is **10**, which indicates 1 second. Response Timeout defines the time (in 100 milliseconds) to wait for a completed response when using either ASCII or Modbus.
9	Enter a value for **Time between frames** from 1 to 255. Default value is **2**. It defines the time (in milliseconds) to accept data when receiving Modbus RTU. Default value is 60 milliseconds for 19200 baud.
10	Select **OK** to close the dialog box. The selected serial port in the Application Browser is updated with the Protocol Type and Address. Select **Cancel** to close the dialog box without making changes to the current settings.

Configuring a Serial Port for ASCII

To configure a serial port to use ASCII protocol (default settings are in bold):

Step	Action
1	Select **Port 1** or **Port 2** (if installed) tab in the dialog box.
2	Select **ASCII** in the **Protocol Type** box.
3	Select a **Baud Rate**: 1200, 2400, 4800, 9600, **19200,** 38400.
4	Select **Data Bits**: 7, **8**.
5	Select a **Parity**: **None**, Even, Odd.
6	Select **Stop Bits**: **1**, 2 bits.
7	Enter a value for **Response Timeout** from 1 to 255. Default is **10**, which indicates 1 second. Response Timeout defines the time (in 100 milliseconds) to wait for a completed response when using either ASCII or Modbus.
8	Click **Advanced** button. **Result:** An **Advanced settings** window opens.

Advanced settings window:



Set the Advanced parameters:

Parameter	Description
Frame structure → **Start character** box	If **Used** box is checked, **':'** radiobutton is default checked. Its value is 58, and it corresponds to the start of frame character in ASCII standard mode. You can also check **Other** radiobutton instead of **':'**, where default value is 58 but you can choose between 1-127 (if Data Bits = 7) or 1-255 (if Data Bits = 8). If you select a value out of range, when clicking OK to quit the window, a message will ask you to select a value within the corresponding range.
Frame structure → **1st end character** box	**Used** box is default checked and grayed. You can uncheck it if you use Stop on the number of received bytes or Stop on silence option. If **Used** box is checked, you can select **CR** (13), **LF** (10) or **Other** (1-127, if Data Bits = 7, or 1-255, if Data Bits = 8). Select only one value at a time. If **Other** radiobutton is checked, default value is 13.
Frame structure → **2nd end character** box	If **Other** radiobutton checked, default value is 10. If **Used** in **1st end character** box is unchecked, all radiobuttons are grayed, default value is 0.
Apply the standard ASCII structure button	Press it to obtain the parameter values.
Stop on the number of received bytes box	This parameter allows the system to conclude an end of frame at reception, when the controller received the fixed number of characters.This value, equal to the frame data, will be used like a maximum value for character count at reception. If **Used** box is checked, you can select a value between 1-255. Default value is 1. If you select a value out of range, when clicking OK to quit the window, a message will ask you to select a value within the range. If not used or disabled, the parameter default value is 0. **Note:** It cannot be used simultaneously with Stop on silence.
Stop on silence box	This parameter allows the system to conclude the end of frame at reception when the fixed period has expired, like a timer. If **Used** box is checked, you can select a value between 1-255. Default value is 10. If you select a value out of range, when clicking OK to quit the window, a message will ask you to select a value within the range. Possible cases: • You use this parameter plus the 1st end character: the received frame is validated when there is a Stop on silence and the last data received is equal to the 1st end character. • You use this parameter plus the Start character: the received frame is saved when the Start character is detected. • You use this parameter only: the received frame is validated when there is a Stop on silence. **Stop on silence** value must be smaller than Response Timeout value. Otherwise, when clicking OK to quit the window, a warning (but non blocking) message will inform you that it will not be taken into account. If not used or disabled, the parameter default value is 0. **Note:** It cannot be used simultaneously with Stop on the number of received bytes.
OK and **Cancel** buttons	Select **OK** to close the dialog box. The selected serial port in the Application Browser is updated with the Protocol Type and Address. Select **Cancel** to close the dialog box without changing the current settings.

Add and configure a modem

Introduction

Use the **Add Option** dialog box to add a modem to the configuration of the open application. TwidoSoft automatically displays available options for each controller.

Adding a Modem

To add a modem:

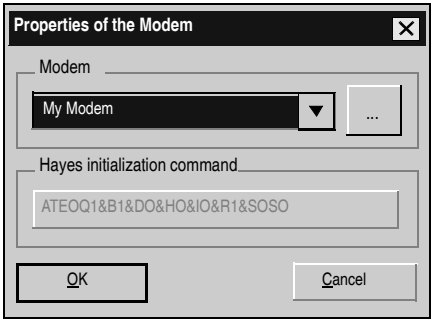
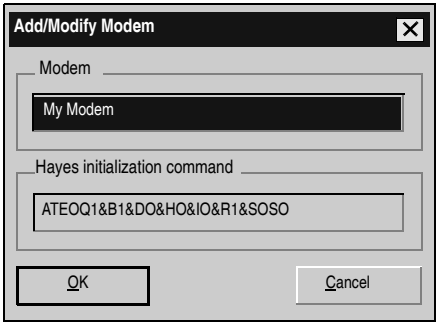
Step	Action
1	Select and highlight "MODEM" in the **Hardware Option** window. A brief description of the modem is displayed in the **Description** area: "Connection modem using TwidoSoft".
2	Click on **Add** to add the modem to the application. The selection is immediately displayed in the application browser.
3	When selections are completed, click on **Done** to close the dialog box. Result: a new "Modem" hardware device appears in the application browser.

Deleting the Modem

The procedure for deleting a modem is the same as for any other option (See *Delete Option, p. 158*).
Here, during the connection procedure, TwidoSoft sends an empty Hayes 1 command to the controller.

Configuring the Modem

To configure the modem:

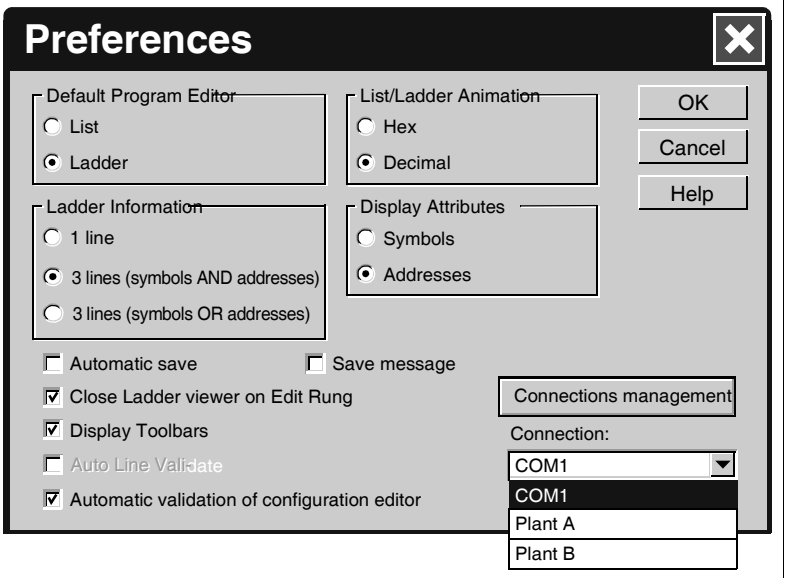
Step	Action
1	In the application browser, right-click on "Modem".
2	Select **Properties...** from the right-click menu. Result: a "Properties of the Modem" dialog box appears. Illustration of the dialog box:  The selected configuration corresponds to the one read in the controller: the Hayes initialization command, then read, is displayed in Hayes standard format.
3	From the dialog box, select the modem you wish to configure from the list (modems known to TwidoSoft, or previously-configured user modems). If no modem is selected, the modem selected by default is the first modem in the list of modems known to TwidoSoft. Note: If TwidoSoft does not manage to convert the Hayes initialization command read by the controller, a new user modem to be configured is created automatically.
4	To modify the configuration (Hayes command), click the Add/Modify Modem icon  in the "Properties of the Modem" dialog box. **Result:** A new dialog box labeled "Add/Modify Modem" appears. Illustration of the dialog box:  Known modem names are automatically replaced by "My modem i" (where i is ≥ 1). For created user modems, the name is kept.

Step	Action
5	Click **OK** to confirm the configuration. Result: the newly configured modem appears in the list of user modems.
6	Transfer the application to the controller using the serial cable for it to be configured for the use of a modem.

Note: If the modem added is already configured and the controller already has a configured Hayes 2 command, the Hayes 1 initialization command is ignored. In this case the modem icon is replaced in the browser by a warning icon.

Configuring TwidoSoft

In order to be able to dialog with the controller via the modem connection, the user must have configured the modem in the configuration of the controller (see previous section) and have connected the modem to port 1 of the controller.
After this phase, TwidoSoft must be configured to use a modem connection instead of a serial cable.
The following table describes the procedure for selecting a modem connection:

Step	Action
1	Select **File->Preferences...** from the main menu. Result: The following dialog box appears: 
2	Select the type of modem connection required for controller dialog from the list of existing connections already configured (serial, USB, modem or IP). Note: If the required connection does not appear in the list of proposed connections, follow the procedure given in the following section on creating or modifying a modem connection.
3	Click **OK** to close the "Preferences" dialog box and acknowledge the selected connection type.

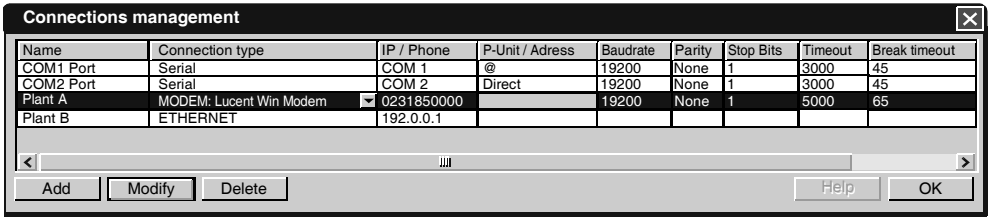
Note: The user can now connect TwidoSoft to the controller using the type of connection selected in the "Preferences" dialog box.

There is another way of selecting the type of connection, which is shown in the following table:

Step	Action
1	Select **Controller->Select a connection** from the main menu. Result: a choice of connections appears in the contextual menu.
2	Select the type of modem connection required for controller dialog from the list of connections proposed. Note: If the required connection does not appear in the list of proposed connections, follow the procedure given in the following section on creating or modifying a modem connection.

Adding or Modifying a Modem Connection in TwidoSoft Configuration

The following table describes the procedure for adding or modifying a modem connection in TwidoSoft configuration:

Step	Action
1	Select **File->Preferences...** from the main menu. Result: The Preferences dialog box appears.
2	In this dialog box, click **Connection management**. The following dialog box is then displayed: 
3	In this new dialog box, the user has three options: ● Delete the connection selected in the dialog box by clicking **Delete**. It is impossible to delete connections via COM ports, as these always exist. ● Modify the connection selected in the dialog box by clicking **Modify** or by double-clicking the selection. ● Add a new modem type connection by clicking **Add**. A new line is added.
4	Where modem connections are modified or added: ● Change or enter the required name.
5	Where modem connections are modified or added: ● Select the connection type from the list containing all the modems configured on the PC.
6	Where modem connections are modified or added: ● From the "Configuration" column, enter the telephone number to be called for this modem connection.

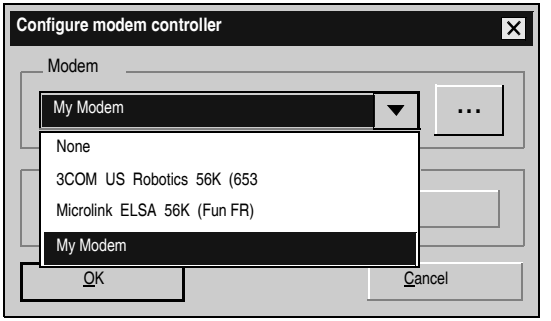
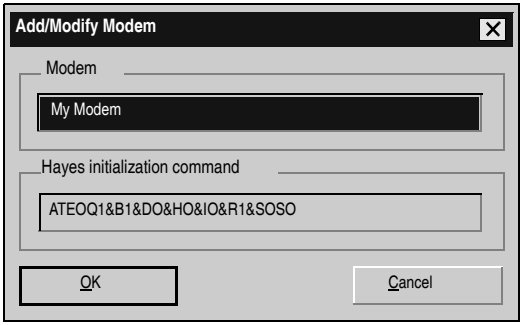
7	Where modem connections are modified or added:
	● Enter or modify the desired timeouts (values in ms):
	  ● The **Timeout** corresponds to the maximum time interval between two frames.
	  ● The **Break timeout** corresponds to the maximum time interval between two characters. When the time interval between two characters is greater than the Break timeout, the system considers this to be the beginning of the next frame. You are advised to assign a **Break Timeout** value lower than that of the **Timeout**.

8	Click **OK** to close the "Connection management" dialog box. Result: the list of connections in the "Preferences" dialog box is updated with the new connection type.

Note: If the user connects TwidoSoft to the controller without having configured the modem in the application, an empty Hayes 1 command is sent to the controller to deconfigure it.

Configuring a Hayes 2 Command

If the user has to manage several controllers using the same modem for each of them, a Hayes 2 command can be sent to dialog with a specific controller without having to modify or duplicate the program.
To configure a Hayes 2 command, follow the procedure below:

Step	Action
1	Select **Controller->Modify modem configuration...** from the main menu. Result: a dialog box appears. Illustration of the dialog box:  The selected configuration corresponds to the one read in the controller: the Hayes 2 command, then read, is displayed in a command format known to TwidoSoft. Note: If TwidoSoft does not manage to convert the Hayes 2 initialization command read by the controller, a new user modem is created automatically.
2	In the dialog box, select the modem connected to the controller. Note: The modem called "None" is used to delete the Hayes 2 command in the controller. Configuration of the None modem has an empty Hayes 2 command.
3	To modify the configuration (Hayes 2 command), click the Add/Modify Modem icon  in the "Configure modem controller" dialog box. **Result:** A new dialog box labeled "Add/Modify Modem" appears. Illustration of the dialog box: 

Step	Action
4	Click **OK** to confirm the configuration. Result: the newly configured modem appears in the list of user modems.
5	Click **OK** in the dialog box "Configure modem controller" to send the Hayes 2 command to the PLC (even if there is not yet any application or connection).

About Remote Links

Introduction

A Remote Link is a Twido controller configured to communicate with other Twido base controllers over a high-speed network. The network uses a proprietary but simple serial protocol to communicate small amounts of data between two or more Twido controllers. One controller is configured as the master and can communicate with up to seven slaves or Remote Controllers. A Remote Link can be any type of base controller model.
Only one type of data for each Remote Controller can be accessed by the Master Controller, either I/O or application data. A mixture of Remote Controller types is possible, where some can be Remote I/O and some can be Peer Controllers.
See *Setting Up a Remote Links Network, p. 173*.

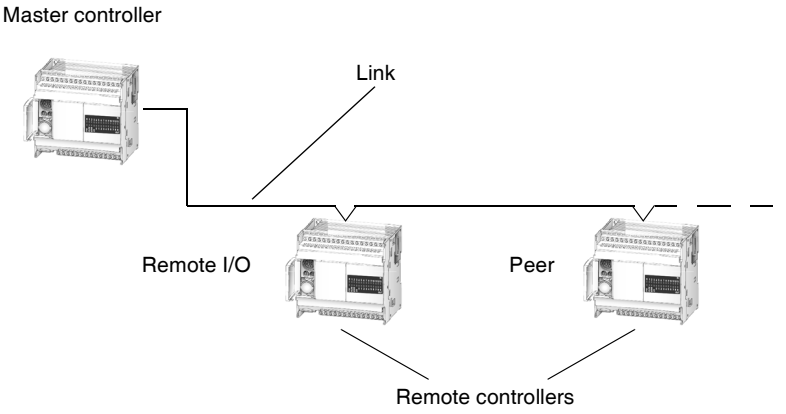
Types of Remote Links

A Remote Link has two possible configurations:

- Remote I/O
  The Remote I/O controller is not running an application program, and the Master Controller can only access I/O data. Access to remote data uses variables %I and %Q. The synchronization of remote data occurs at the end of every scan.
- Peer
  The Peer controller is running its own application program, and the Master Controller can only access application data. Access to Peer application data uses network variables %INW and %QNW. The synchronization of remote data can occur over multiple scans depending on the number of Peer controllers and the application program.

Remote Links Network

The following diagram is a simplified view of a Remote Link network.



Designating a Remote Network Master

There can only be one controller configured as the master on a Remote Link network. To designate a controller as the network master, select a Remote Link address of 0 by using the Controller Communication Setup. To designate a controller as a Remote Link slave, select a Remote Address of 1 to 7.

Guidelines for Configuring Remote Links

The following conditions must be met to configure a Remote Link:

- TwidoSoft must be alternately connected to Port 1 of each controller to configure a serial port for Protocol Type and Address.
- Configure the slave Remote Links first so that the master controller will detect the remote links after configuration.
- TwidoSoft must be offline.
- The application on the master controller must be programmed to exchange data with the remote links.
- EIA-485 is required for the port used for connecting the remote link: Port 1 or optional Port 2.
- Only one port can at a time can be configured as a Remote Link.

Setting Up a Remote Links Network

Introduction

The following three steps are required to setup a Remote Link network:

- Wiring the controllers in a network
  See the Twido User Reference Manual for details on wiring a Remote Link network.
- Configuring Remote Controllers
  Remote Controllers can be configured as either Remote I/O or Peer type, and you can mix types up to a maximum of seven Remote Controllers.
- Configuring the Master Controller
  Only one controller can be configured as the master.

See *About Remote Links, p. 171*.

Configuring Remote I/O Type

To configure a controller as a Remote I/O type of Remote Controller:

Step	Action
1	Create a new application with TwidoSoft offline.
2	Configure the application's hardware and software, and select Remote Link protocol and an address of 1 - 7. See *Controller Communications Setup, p. 159*.
3	Do not enter any programming logic since this controller will be used only for I/O data.
4	Transfer the application to the controller.

Configuring Peer Type

To configure a controller as a Peer type of Remote Controller:

Step	Action
1	Create a new application with TwidoSoft offline.
2	Configure the application's hardware and software. Select Remote Link protocol and an address of 1 - 7. See *Controller Communications Setup, p. 159*.
3	Enter programming logic since this controller will be used only for application data.
4	Transfer the application to the controller.

Configuring a Master Controller

To configure a controller as a Master Controller:

Step	Action
1	Create a new application with TwidoSoft offline.
2	Configure the application's hardware and software. Select Remote Link protocol and an address of 0. See *Controller Communications Setup, p. 159*.
3	Configure Remote Controllers on Remote Link. See *Add Remote Controller, p. 175*.
4	Enter programming logic since this controller will be used only for application data.
5	Transfer the application to the controller.

Add Remote Controller

Introduction

Use the **Add Remote Controller** dialog box to add up to seven Remote Link controllers to the hardware configuration of an open application.
See *About Remote Links, p. 171*.

Opening the Dialog Box

To open the Add Remote Controller dialog box:
- Right-click on a configured serial port under Hardware in the Application Browser.
- Select **Add Remote Controller** from the **Hardware** menu.

Controller Usage

The base controller in the hardware configuration must be configured as the network master to enable adding Remote Links. A mixture of the Remote link types is possible:
- Remote I/O
- Peer Controller

Remote Address

The address of each Remote I/O or Peer Controller is from 1 to 7. When all addresses have been used, the selection is left blank.

Adding a Remote Controller

To add an a Remote controller:

Step	Action
1	Select either Remote I/O or Peer controller for **Controller Usage**.
2	Select a Remote Address from the list of available addresses. Click on **Add** to add the selected option to the application.
3	To add more than one Remote controller without closing the dialog box, continue selecting options and clicking on **Add**.
4	When selections are completed, click on **Done** to close the dialog box.

Delete Remote Controller

Introduction

Use the **Delete Remote Controller** dialog box to remove a remote controller, Remote I/O or Peer, from the hardware configuration of an open application. This dialog box can be opened from the *Application Browser, p. 27* or the **Hardware** menu and the procedure is slightly different for each method.
See *About Remote Links, p. 171*.

Deleting a Remote Controller Using the Application Browser

Using this procedure, the dialog box opens with a default selection. To delete a remote controller from the *Application Browser, p. 27*:

Step	Action
1	Right-click on the Remote Link controller under the **Remote Link, 0**.
2	Select **Delete** from the shortcut menu.
3	The **Delete Remote Controller** dialog box is displayed with the selected controller highlighted by default. Only one controller can be selected at a time.
4	Select **OK** to delete the selected controller and close the box. Select **Cancel** to close the dialog box without changes.

Deleting a Remote Controller Using the Hardware Menu

Using this procedure, the dialog box does not open with a default selection. To delete an remote controller from the **Hardware** menu:

Step	Action
1	Select **Hardware** → **Delete Remote Controller** from the main menu.
2	The **Delete Remote Controller** dialog box is displayed without any remote controllers highlighted. Click on a controller to select it. Only one controller can be selected at a time.
3	Select **OK** to delete the selected controller and close the box. Select **Cancel** to close the dialog box without changes.

Numbering of Remote Addresses

When a remote controller is deleted, the Remote Link addresses do not automatically renumber since the numbers for the remaining remote controllers were assigned during configuration. If you change an address number for a remaining remote controller, you will have to reconnect the controller to TwidoSoft and reconfigure the controller.

About Modbus Links

Introduction

Modbus defines a master-slave communications protocol that allows one single master to request responses from slaves. The master can address individual slaves, or can initiate a broadcast message to all slaves. Slaves return a message (response) to requests from the master that are addressed to them individually. Response are not made to broadcasts from the master.

Types of Modbus Links

There are two types of Modbus devices supported by TwidoSoft:

- Master
  Initiates the transmission of a Modbus query and solicit responses from slave devices. Supported using the EXCH instruction. Both Modbus ASCII and Modbus RTU are supported in Modbus Master mode.
- Slave
  Responds to Modbus queries from a Modbus master. Supports both Modbus ASCII and Modbus RTU in Modbus Master mode.

Using Modbus

Twido controllers support the use of both Modbus ASCII and Modbus RTU modes of operation. The number of Data Bits selected in the **Controller Communications Setup** dialog box determines which mode is active.
Setting the number of Data Bits to 8 indicates that the Modbus RTU protocol will be used, while setting Data Bits equal to 7 indicates that the Modbus ASCII protocol will be used.

About ASCII Links

Introduction

ASCII protocol is a simple, half-duplex character mode protocol used to exchange a character string with a simple device. The protocol is supported using the EXCH instruction.

Types of Exchange

There are three types of communication possible using ASCII protocol:

- Transmission only
- Transmission/Reception
- Reception only

Frames and Baud Rate

The maximum size of the transmitted or received frames is 128 bytes. The maximum baud rate supported is 38,400.

4.3 Ethernet TCP/IP Communications Setup

At a Glance

Overview

This section contains Ethernet TCPI/IP connection and configuration information for the TWDLCAE40DRF Twido base controller.

What's in this Section?

This section contains the following topics:

Ethernet TCP/IP Communications Overview

Ethernet Features

The following information describes the Ethernet-capable features of the Twido TWDLCAE40DRF base controller.
The TWDLCAE40DRF base controller is an Ethernet-capable device that implements the Modbus Application Protocol (MBAP) over TCP/IP. Modbus TCP/IP provides peer-to-peer communications over the network in a client/server topology.

Frame Format

The Twido TWDLCAE40DRF compact controller supports the Ethernet II frame format only. It does not accommodate IEEE802.3 framing. Note that other PLCs available from Schneider Electric, such as the Premium and Quantum series support both Ethernet II and IEEE802.3 frame formats and are frame format selectable. Therefore, if you are planning to team up your Twido controller with Premium or Quantum PLCs, you should configure them as using Ethernet II frame format to allow for optimum compatibility.

TCP Connections

The TWDLCAE40DRF compact controller is a 4-simultaneous-channel device capable of communicating over a 100Base-TX Ethernet network. It implements 100Base-TX auto-negotiation and can work on a 10Base-T network as well. Moreover, it allows one marked IP connection, as configured in the TwidoSoft application program (see *Marked IP Tab, p. 199* for more details about Marked IP). The maximum number of server transactions supported by the Twido controller is 1 per TCP connection.

IP Address

TWDLCAE40DRF controllers implement BootP client support to obtain an IP address from a BootP server. For increased flexibility, you still have the ability to specify a static IP address through TwidoSoft programming software, along with defining the subnetwork and gateway IP addresses.
In addition, if the TWDLCAE40DRF controller fails to obtain a valid IP address from the BootP server (or if it detects a duplicate IP address when you assign a static IP address), the controller goes into fallback mode and uses the default IP address.
Each TWDLCAE40DRF controller is assigned a unique MAC physical address (IEEE Global Address) permanently stored in the compact controller. The default IP address is derived from the controller's MAC address.

Note: When using the default IP address, BootP client service is closed.

Modbus TCP Client/Server

A TWDLCAE40DRF controller can be both Modbus TCP/IP Client and Server depending on whether it is querying or answering a remote device, respectively. TCP messaging service is implemented via TCP port 502.

- The Modbus Server implements the Schneider Transparent Ready class messaging TR A15 standard.
- Modbus Client is implemented via the EXCH3 instruction and %MSG3 function. You may program several EXCH3 instructions, however one EXCH3 only can be active at a time. The TCP connection is automatically negotiated by the compact controller as soon as the
  The Modbus Client implements the Schneider Transparent Ready class messaging TR A10 standard.

Quick TCP/IP Setup Guide for PC-to-Controller Ethernet Communication

Scope

This Quick TCP/IP Setup Guide is intended to provide Ethernet connectivity information and TCP/IP configuration information to rapidly setup communication between your PC running the TwidoSoft application and the Twido Controller over a stand-alone Ethernet network.

Checking the Current IP Settings of your PC

The following procedure describes how to check the current IP settings of your PC Also, this procedure is valid for all versions of the Windows operating system:

Step	Action
1	Select **Run** from the Windows **Start** menu.
2	Type "**command**" in the **Open** textbox of the Run dialog box. **Result:** The **C:\WINDOWS\system32\command.com** prompt appears.
3	Type "**ipconfig**" at the command prompt.
4	The Windows **IP Configuration** appears, and displays the following parameters: IP Address.....: Subnet Mask.....: Default Gateway.....: **Note:** The above IP settings cannot be changed directly at the command prompt. They are available for consultation only. If you plan to change the IP configuration of your PC, please refer to the following section.

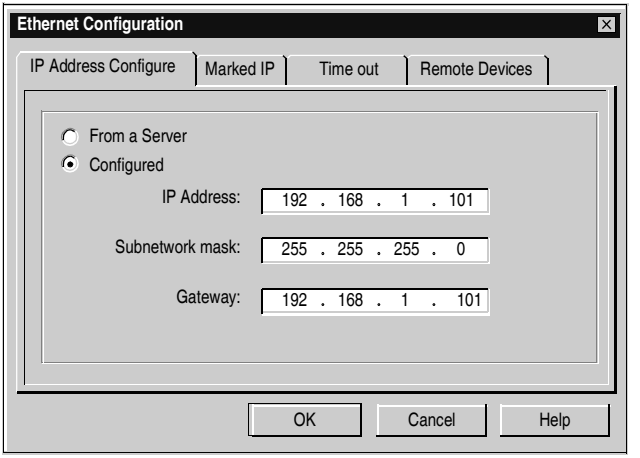
Configuring the TCP/IP Settings of your PC

The following information will help configure the TCP/IP settings of your PC running the TwidoSoft application for programming and control of the Twido controller over the network. The procedure outlined below is workable on a PC equipped with a Windows XP operating system, and is intended as an example only. (Otherwise, for other operating systems, please refer to TCP/IP setup instructions outlined in the user's guide of the particular operating system installed on your PC.)

Step	Action
Note: If your PC is already installed and the Ethernet card is configured over the existing stand-alone network, you will not need to change the IP address settings (skip steps 1-6 and continue to the following section). Follow steps 1-6 of this procedure only if you intend to change the PC's TCP/IP settings.	
1	Select **Control Panel > Network Connections** from the Windows **Start** menu.
2	Right click on the **Local Area Connection** (the stand-alone network) on which you are planning to install the Twido controller, and select **Properties**.
3	Select **TCP/IP** from the list of network components installed, and click **Properties**. **Note:** If TCP/IP protocol is not among the list of installed components, please refer to the user's manual of your operating system to find out how to install the TCP/IP network component.
4	The **TCP/IP Properties** dialog box appears and displays the current TCP/IP settings of your PC, including **IP Address** and **Subnet Mask**. **Note:** On a stand-alone network, do not use the **Obtain an IP address automatically** option. The **Specify an IP address** radio-button must be selected, and the IP Address and Subnet Mask fields must contain valid IP settings.
5	Enter a valid **static IP Address** in dotted decimal notation. Over a stand-alone network, we suggest you to specify a Class-C network IP address (see *IP Addressing, p. 189*.). For example, `192.168.1.198` is a Class-C IP address. **Note:** The IP address you specify must be compatible with the network ID of the existing network. For example, if the existing network supports `192.168.1.xxx` IP addresses (where `192.168.1` is the network ID, and `xxx = 0-255` is the host ID), than you may specify `191.168.1.198` as a valid IP address for your PC. (Make sure the host ID `198` is unique over the network).
6	Enter a valid **Subnet Mask** in dotted decimal notation. If subnetting is not used on your Class-C network, we suggest you to specify a Class-C network default subnet mask such as `255.255.255.0`.

Configuring the TCP/IP Settings of your Twido Controller

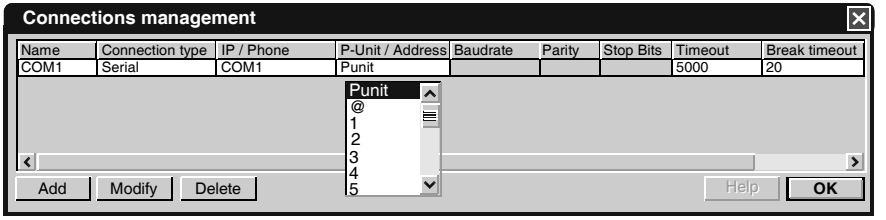
Once you have configured the TCP/IP settings of your PC hosting the TwidoSoft application, you will need to configure the TCP/P settings of the Twido controller you wish TwidoSoft to communicate with over the network, as described below:

Step	Action
1	Connect a serial cable (TSXPCX1031) from the PC running TwidoSoft to the Twido controller's RS-485 console port.
2	Launch the **TwidoSoft** application program on your PC.
3	Select a new **Hardware** from the TwisoSoft Application Brower and choose the **TWDLCAE40DRF** controller.
4	Select **PLC > Select a connection** from the TwidoSoft menu bar, and choose the **COM1** port.
5	Double-click on the **Ethernet Port** icon in the TwisoSoft Application Browser (or select **Hardware > Ethernet** from the menu bar) to call up the **Ethernet Configuration** dialog box, as shown below: 
6	From the **IP Address Configure** tab: • Select **From a Server** radio-button, to use BootP client support in order to automatically obtain a dynamic IP address from the server. *(Go directly to Step 10.)* **Note:** The TWDLCAE40DRF controller performs three, 200ms-interval retries to send BootP requests to the server. If no valid response is received, the controller uses the fallback default IP address. • Select **Configured** radio-button, and start configuring the static IP Address, Subnetwork mask and Gateway address fields as explained in steps 7-9. **Note:** At this stage, we are only dealing with the basic configuration of PC-to-controller communication over the Ethernet network. Therefore, you will not need to configure the Marked IP, Time out and Remote Devices tabs yet.

Step	Action
7	Enter a valid static **IP Address** for the Twido controller in dotted decimal notation. This IP address must be compatible with that of the PC's IP address that you have configured in the previous section. **Note:** The IP addresses of the Twido controller and the PC must share the same network ID. However, the Twido controller's host ID must be different from the PC's host ID, and unique over the network. For example, if the PC's Class-C IP address is `192.168.1.198`, then a valid address for the Twido controller is `192.168.1.xxx` (where `192.168.1` is the network ID, and `xxx = 0-197, 199-255` is the host ID).
8	Enter a valid **Subnetwork mask** in dotted decimal notation. The Twido controller and the PC running TwidoSoft must be on the same network segment. Therefore, you must enter a subnet mask that is identical to that specified for the PC. **Note:** If subnetting is not used on your Class-C network, we suggest you to specify a Class-C network default subnet mask, such as `255.255.255.0` .
9	Enter a valid **Gateway** address in dotted decimal notation. **Note:** If there is no gateway device on your stand-alone network, enter the Twido controller's own IP Address that you have just configured in step 6 in this field.
10	Click on **OK** to save the Ethernet configuration settings of your Twido controller.

Setting Up a New TCP/IP Connection in TwidoSoft

You will now set up a new TCP/IP connection in the TwidoSoft application. The new dedicated TCP/IP connection will allow the PC running TwidoSoft and the Twido controller to communicate over the Ethernet network.
Select **File** → **Preferences** from TwidoSoft menu bar to call up **Connexions management** dialog box:



Step	Action
1	Click the **Add** button in the **Connections Management** dialog box. **Result:** A new connection line is added. The new line displays suggested default connection settings. You will need to change these settings. **Note:** To set a new value in a field, you have two options: • Select the desired field, then click the **Modify** button. • Double click the desired field.
2	In the **Name** field, enter a descriptive name for the new connection. A valid name may contain up to 32 alphnumeric characters.
3	In the **Connection Type** field, click to unfold the dropdown list that includes: TCP/IP, Serial, Modem (if any) and USB (if any). Select **TCP/IP** as you are setting up a new Ethernet connection between your PC and an Ethernet-capable Twido controller.
4	In the **IP / Phone** field, enter a valid **IP address** which is the IP information of the Twido TWDLCAE40DRF controller you wish to connect to. **IP Address:** Enter the static IP address that you have specified for your Twido controller in a previous section.

Step	Action
5	The **Punit / Address** field can be filled in when IP / Phone has been selected. For a TCP/IP Type connection, default value is **Direct**. For a Serial Type connection, default value is **Punit**. When any of those is selected, next three fields (Baudrate, Parity and Stop Bits) are disabled. If you do not know the controller address, **@** allows you to select it later, once the program has been downloaded. *(A window pops up before the first connection to let you choose the controller to which you transfer, with a 1-247 range, and 1 as the default address value.)*
6	Use the default settings in **Timeout** and **Break timeout** fields, unless you have specific timeout needs. (For more details, please refer to *Ethernet Connections Management, p. 206*.)
7	Click the **OK** button to save the new connection settings and close the Connections management dialog box. **Result:** The names of all newly-added connections are added to the dropdown list of connections in the **File** → **Preferences** dialog box or in the **PLC** → **Select a connection** menu.

Connecting your Controller to the Network

Overview

The following information describes how to install your TDWLCAE40DRF compact controller on your Ethernet network.

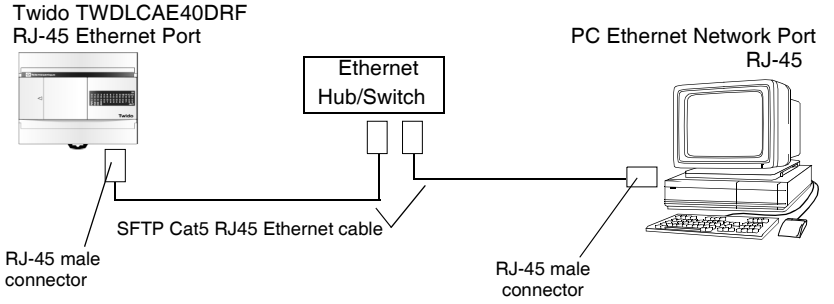
Determining the Appropriate IP Address Set

Consult your network administrator to determine if you must configure a new set of device IP, gateway and subnet mask addresses. If the administrator assigns new IP address parameters, you will need to enter this information manually in the TwidoSoft application. Follow the directions in the *TCP/IP Setup, p. 195* section hereafter.

Ethernet Network Connection

Note: Although direct cable connection (using a Ethernet crossover cable) is supported between the Twido TWDLCAE40DRF and the PC running the TwidoSoft programming software, we do not recommend it. Therefore, you should always favor a connection via a network Ethernet hub/switch.

The following figure shows a Twido network connection via an Ethernet hub/switch:



The Twido TWDLCAE40DRF features a RJ-45 connector to connect to the 100BASE-TX network Ethernet with auto negotiation. It can accommodate both 100Mbps and 10 Mbps network speeds.

Note: When connecting the Twido controller to a 100BASE-TX network, you should use at least a category 5 Ethernet cable.

IP Addressing

Overview

This section provides you with information on IP Address notation, subnet and gateway concepts as well.

IP Address

An IP address is a 32-bit quantity expressed in dotted decimal notation. It consists of four groups of numbers ranging in value from 0 to 255 and separated from one another by a dot. For example, 192.168.2.168 is an IP address in dotted decimal notation (note that this is a reserved IP address provided as an example only).
On usual networks, IP addresses fall into three categories named Class A, B, and C networks. Classes can be differentiated according to the value of their first number which ranges as described in the following table:

First decimal group	IP class
0-127	Class A
128-191	Class B
192-223	Class C

IP Subnet Mask

An IP address consists of two parts, the network ID and the host ID. The subnet mask is used to split the network portion of the IP address to artificially create subnetworks with a larger number of host IDs. Thus, subnetting is used as a means of connecting multiple physical networks to logical networks. All devices on the same subnetwork share the same network ID.
All devices on the same subnetwork share the same network ID.

Note: If you are part of a large organization, then there is a good chance that subnetting is being implemented on your company's networks. Check with your network administrator to obtain adequate subnetting information when you are installing your new Twido controller on the existing network.

Gateway Address

The Gateway is the networking device also called router that provides to your network segment access to other network segments on your company's global network, access to the Internet or to a remote Intranet.
The gateway address uses the same dotted decimal notation format as the IP address described above.

Note: Check with your network administrator to obtain adequate gateway information when you are installing your new Twido controller on the existing network.

Assigning IP Addresses

Overview

This section provides you with information on how to determine which type of IP address you can assign to the Twido TWDLCAE40DRF controller that you wish to install on your network.

Installation on a Stand-alone Network

Your Twido TWDLCAE40DRF controller is intended for installation on a stand-alone Ethernet network.

> **Note:** A network is called stand-alone when it is not linked to the Internet or a company's Intranet.

Obtaining an Address via BootP

BootP Served Address: If you choose **From a Server** in the **IP Address Configure** tab, the Twido controller will try to obtain an IP address from BootP server first.
BootP process expects a response from the BootP server. If no valid IP address is received following the BootP request transmission, Twido assumes the default IP configuration that is derived from a MAC address (see *MAC Address and Default IP Address of the Controller, p. 191* below.)

MAC Address and Default IP Address of the Controller

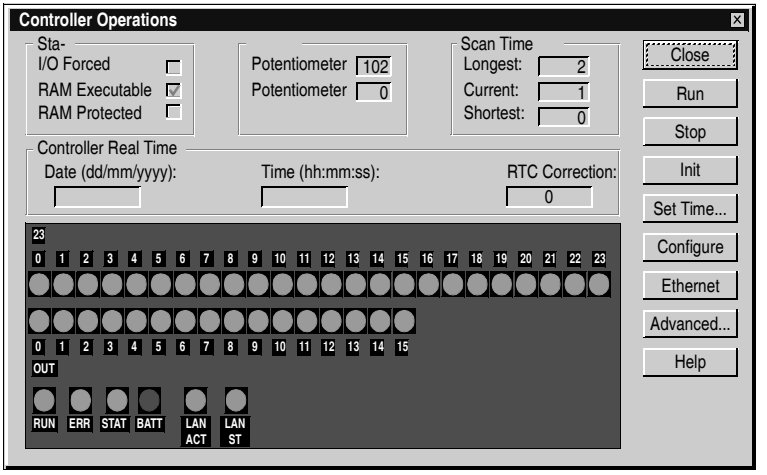
MAC Address: Each Twido TWDLCAE40DRF controller has its own factory-set MAC address that is a worldwide-unique 48-bit address assigned to each Ethernet device.
Default IP Address: The default Ethernet interface IP address of the Twido controller is derived from its unique MAC address.
The default IP address expressed in dotted decimal notation is defined as follows: `085.016.xxx.yyy`, where:

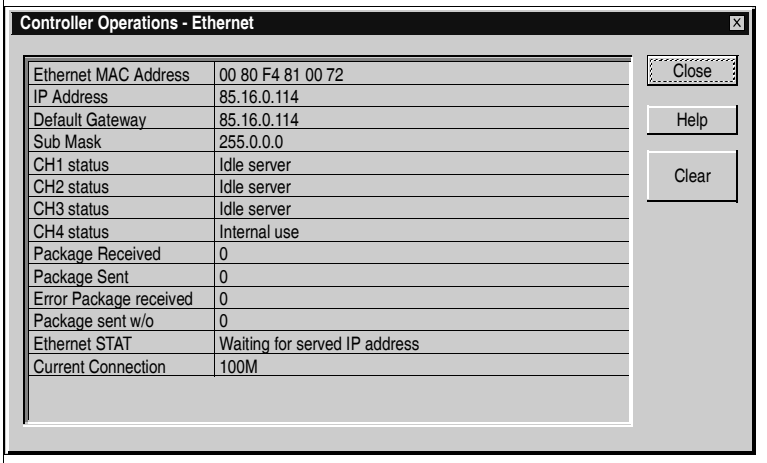
- `085.016.` is a set header shared by all IP addresses derived from MAC address,
- `xxx` and `yyy` are last two numbers of the device MAC address.

For example, the IP address derived from MAC address `00.80.F4.81.01.11` is `085.016.001.17`.

Checking the MAC Address and Current IP Address of the Controller

To check out the MAC address and the current IP address of your Twido controller, along with IP configuration settings (subnetwork mask and gateway addresses) and Ethernet connection status, follows these instructions:

Step	Action
1	In TwidoSoft application program, select **PLC** from the menu bar.
2	Select **Check PLC** from the menu items list. **Result:** The **Controller Operations** dialogbox appears, displaying the Twido LEDs on a soft front-panel, as shown in the figure below: 

Step	Action
3	Click the **Ethernet** button located in the right portion of the screen to access the connection parameters. **Result:** The **Control Operations - Ethernet** table appears, displaying MAC, current IP ,Subnet and Gateway information, as well as Ethernet connection information, as shown in the following figure:  **Controller Operations - Ethernet** Ethernet MAC Address: 00 80 F4 81 00 72 IP Address: 85.16.0.114 Default Gateway: 85.16.0.114 Sub Mask: 255.0.0.0 CH1 status: Idle server CH2 status: Idle server CH3 status: Idle server CH4 status: Internal use Package Received: 0 Package Sent: 0 Error Package received: 0 Package sent w/o: 0 Ethernet STAT: Waiting for served IP address Current Connection: 100M Close Help Clear
4	Note that the unique **MAC** address of the Twido controller is showing on the first row of the Ethernet table.
5	The IP information displayed in this table varies depending on the user-settings in the **IP Configure tab** of the **Ethernet Configuration** dialogbox (see *IP Address Configure Tab, p. 197*): ● if you selected **From a Server** in the IP Configure tab, the above table displays the default IP address (derived from MAC address) of the Twido controller, the default subnet and gateway as well. Note that the default IP address is used in fallback mode only, if no valid BootP served IP address can be obtained from the server. When one channel is used as UDP for BootP, the channel status shows **Internal use.** ● if you selected **Configured** from the IP Configure tab, the above table displays the current IP address, subnet and gateway settings that you have previouly entered in the IP Configure tab. Note: The remaining fields provide information about the current status of the Ethernet connection. To find out more information, please refer to *Ethernet Statistics, p. 107*.

Private IP Addresses

If your network is stand-alone (isolated from the Internet), you may therefore assign to your network node (Twido controller) any arbitrary IP address (as long as the IP address conforms to the IANA notation rule and it doesn't conflict with the IP address of another device already connected to the network).
Privates IP addresses meet the need for arbitrary IP addressing over a stand-alone network. Note that addresses within this private address space will only be unique within the enterprise.
The following table outlines the private IP address space:

Network	Valid range for private IP addresses
Class A	10.0.0.0 -> 10.255.255.255
Class B	172.16.0.0 -> 172.31.255.255
Class C	192.168.0.0 -> 192.168.255.255

Assigning an IP Address to your Controller

Today’s networks are rarely either totally isolated from the Internet or from the rest of the company's Ethernet network. Therefore, if you are installing and connecting your Twido base controller to an existing network, do not assign an arbitrary IP address without prior consulting with your network administrator. you should follow the directions outlined below when assigning an IP address to your controller.

Note: It is good practice to use Class-C IP addresses on stand-alone networks.

TCP/IP Setup

Overview

The following are detailed instructions on how to set up the Ethernet TCPI/IP configuration for your Twido TWDLCAE40DRF compact controller.

Note: TCP/IP setup can be performed when the TwidoSoft application program is in offline mode only

⚠ CAUTION

UNINTENDED EQUIPMENT OPERATION

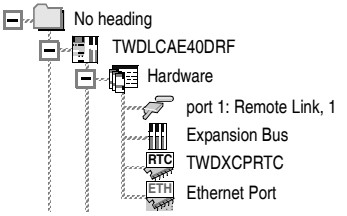
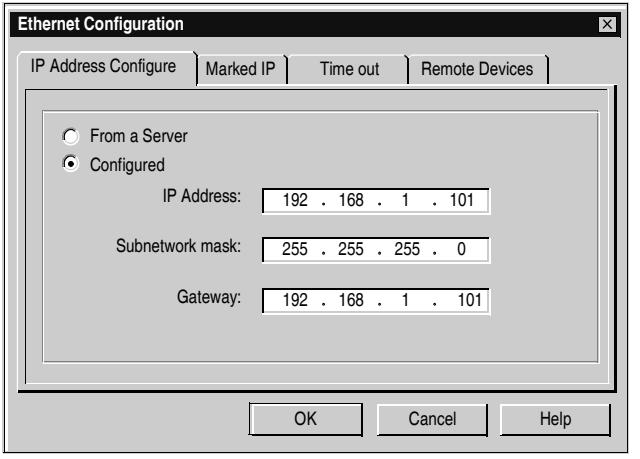
Having two devices with the same IP address can cause unpredictable operation of your network.

- Ensure that this device will receive a unique IP address.
- Always obtain your IP address from your system administrator to avoid the possibility of duplicate address.

Failure to follow this instruction can result in injury or equipment damage.

Calling up the Ethernet Configuration Dialogbox

The following steps detail how to call up the **Ethernet Configuration** dialogbox:

Step	Action
1	Open the **Application Browser**, as shown in the figure below. **Result:**  **Note:** Make sure an Ethernet-capable device such as TWDLCAE40DRF is selected as the current hardware, or otherwise the Ethernet Port hardware option will not appear.
2	Double-click on the **Ethernet Port** icon to bring up the **Ethernet Configuration** dialogbox, as shown below. **Result:**  **Note:** There are two alternate ways to call up the Ethernet Configuration screen: **1.** Right-click on the **Ethernet Port** icon and select **Edit** from the popup list. **2.** Select **Hardware > Ethernet** from the TwidoSoft menu bar.

TCP/IP Setup

The following sections detail how to configure the Twido TWDLCAE40DRF TCP/IP parameters by using the **IP Address Configure, Marked IP, Time out** and **Remote Devices** tabs.

IP Address Configure Tab

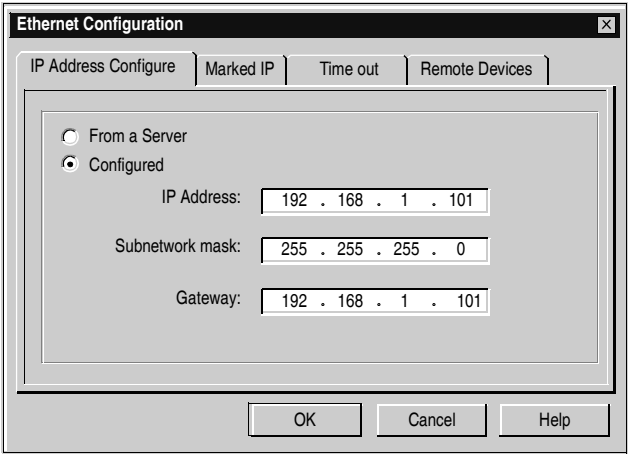
Overview

The following information describes how to configure the IP Address Configure tab of the Ethernet Configuration dialogbox.

Note: The IP address of the Twido controller can be configured when the TwidoSoft application program is in offline mode only

IP Address Configure tab

The following figure presents a sample screen of the IP Address Configure tab showing examples of IP, Subnet and Gateway addresses configured manually by the user:



Configuring the IP Address tab

The following information describes how to configure the various fields in the IP Address Configure tab:

Field	Configuring
From a Server	Check this radio button if you do not wish to set the IP address of the Twido controller manually (the IP Address, Subnetwork mask and Gateway textboxes are grayed out). The Twido controller (BootP client) will then use the IP address automatically assigned by the server. The Twido controller will choose to use the default IP address (fallback state), if it cannot obtain a valid served IP address after the three retries at 200 ms intervals. (Note that the Twido controller periodically sends requests to the server at 15s intervals until it obtains a valid IP address.) The default Ethernet interface IP address is derived from its MAC address. (Note that the default IP address will not be changed automatically when any channel (excluding the channel for internal use) of PLC is active.) **Note:** To find out more information about BootP and MAC address, please refer to *Assigning IP Addresses, p. 191*.
Configured	Check this radio button to configure the IP, subnetwork and gateway addresses manually. **Note:** Consult with your network or system administrator to obtain valid IP parameters for your network.
IP Address	Enter the static IP address of your Twido in dotted decimal notation. **Caution:** For good device communication, the IP addresses of the PC running the TwidoSoft application and the Twido controller must share the same network ID. **Note:** To allow good communication over the network, each connected device must have a unique IP address. When connected to the network, the Twido controller runs a check for duplicate IP address. If a duplicate IP address is located over the network, the LAN ST LED of the Twido controller will emit 4 flashes periodically. You must then enter a new duplicate-free IP address in this field.
Subnetwork mask	Enter the valid subnet mask assigned to your controller by your network administrator. Please note that you cannot leave this field blank; you must enter a value. As default, the TwidoSoft application automatically computes and displays a default subnet mask based on the class IP that you have provided in the IP Address field above. Default subnet mask values, according to the category of the Twido network IP address, follow this rule: Class A network -> Default subnet mask: 255.0.0.0 Class B network -> Default subnet mask: 255.255.0.0 Class C network -> Default subnet mask: 255.255.255.0 **Caution:** For good device communication, the subnet mask configured on the PC running the TwidoSoft application and the Twido controller's subnet mask must match. **Note:** Unless your Twido controller has special need for subnetting, use the default subnet mask.
Gateway	Enter the IP address of the gateway. On the LAN, the gateway must be on the same segment as your Twido controller. This information typically is provided to you by your network administrator. Please note that no default value is provided by the application, and that you must enter a valid gateway address in this field. **Note:** If there is no gateway device on your network, simply enter your Twido controller's IP address in the Gateway field.

Marked IP Tab

Overview

The following information describes how to configure the Marked IP tab of the Ethernet Configuration dialogbox.

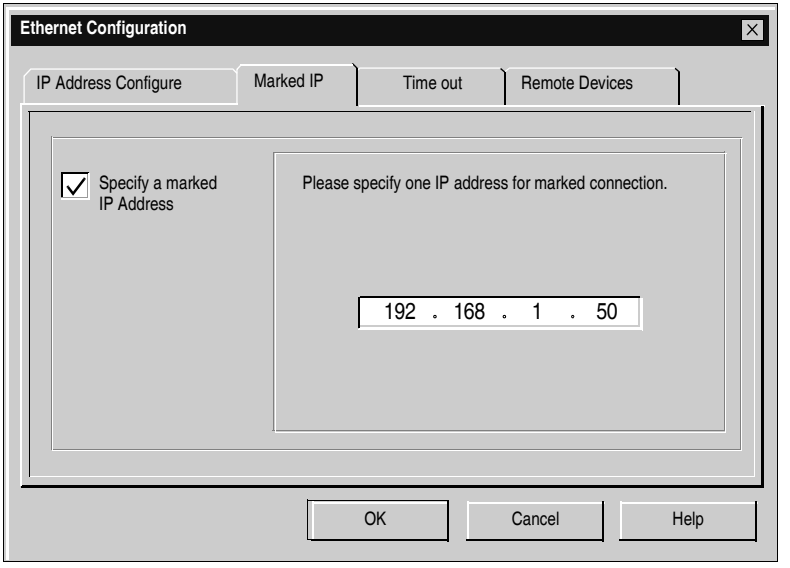
Note: The Marked IP can be configured when the TwidoSoft application program is in offline mode only.

Definition of the Marked IP Function

This function allows you to reserve one of the four Ethernet TCP connection channels supported by your Twido controller for a particular client host designated as Marked IP.
Marked IP can ensure that one TCP channel is reserved and always available for communication with the specified remote device, even if the idle timeout is disabled (idle timeout is set to "0".)

Marked IP tab

The following figure presents a sample screen of the Marked IP tab showing an example of marked IP address entered by the user:



Configuring the Marked IP tab

To configure the Marked IP tab, follow these steps:

Step	Action
1	Check the box labeled **Specify a marked IP address** to enable the Marked IP function. Note that Marked IP is disabled, as default. **Result:** The IP address box becomes active in the right portion of the frame, as shown in the previous figure.
2	Enter the IP address of the client host you wish to mark the IP in the provided IP address box. **Note:** There is no default value in this field. You must provide the IP address of the marked device, or otherwise uncheck the Specify a marked IP address box to disable this function.

Time out Tab

Overview

The following information describes how to configure the Time out tab of the Ethernet Configuration dialogbox.

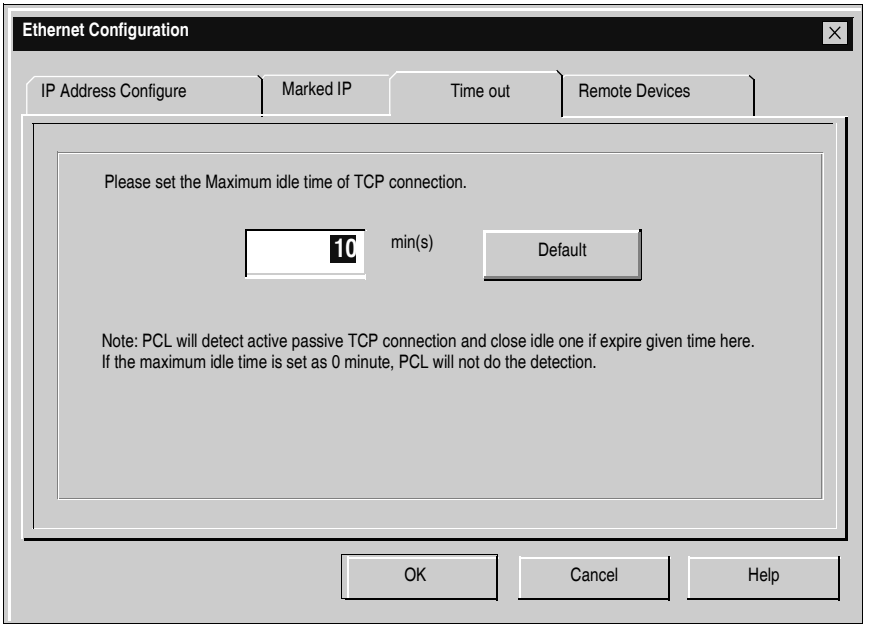
Note: The Time out of the Twido controller can be configured when the TwidoSoft application program is in offline mode only.

Definition of Time out

Time out applies an idle timeout to all current Ethernet TCP connections of the Twido controller. The idle timeout is the amount of time that any of the four Ethernet TCP connection channels may remain idle before the remote client host connection to this channel is dropped.
Note: The idle timer is reset whenever there is data traffic on the monitored connection channel.

Time out tab

The following figure presents a sample screen of the Time out tab showing the 10 min default value of the idle timer:



Configuring the Time out tab

To set the Idle timer, enter directly the elapsed time in minutes in the **min(s)** textbox, as shown in the previous figure.

Note:
1. The default elapsed time is 10 minutes. After you entering a value, to **reset** the configured elapsed time to 10 minutes, click on the **Default** button.
2. To **disable** the Time out function, set the elapsed time to **0.** The Twido controller no longer performs idle checks. As a result, the TCP connections stay up indefinitely.
3. The maximum idle time allowed to set is 255 minutes.

Remote Devices Tab

Overview

The following information describes how to configure the Remote Devices tab of the Ethernet Configuration dialogbox when you intend to use the EXCH3 instruction for the Twido controller to act as Modbus TCP/IP client.

Note: The Remote Devices tab of the Twido controller can be configured when the TwidoSoft application program is in offline mode only.

What You Should Know at First

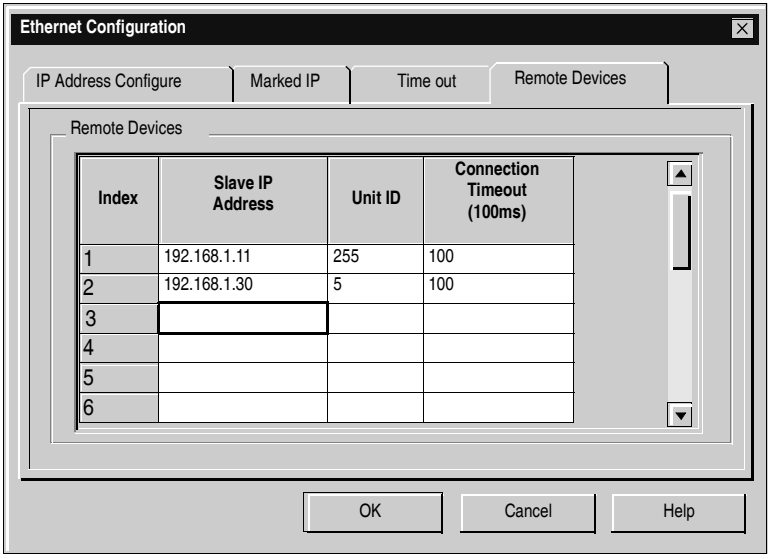
You do not need to configure the Remote Devices on any controller other than the controller that you want to use the Modbus TCP/IP client (legacy Modbus master) instruction (EXCH3).

Remote Devices Table

The Remote Devices table stores information about remote controllers (acting as Modbus TCP/IP servers) over the Ethernet network that can be queried by the Modbus TCP/IP client using the EXCH3 instruction. Therefore, you must configure the Remote Devices table properly so that the Modbus TCP/IP client controller can poll Modbus TCP/IP server controllers over the network.

Remote Devices tab

The following figure presents a sample screen of the Remote Devices tab configured on the Twido controller acting as Modbus TCP/IP client:



Configuring the Remote Devices tab

The following information describes how to configure the various fields in the Remote Devices tab:

Field	Configuring
Index	This is a read-only field that contains the MBAP Index associated with the Ethernet network IP address of the remote device (Modbus TPC/IP server specified in the Slave IP Address field). The MBAP Index is called by the EXCH3 instruction as one of the function's arguments to identify which remote controller specified in the table is being queried by the Modbus TCP/IP client. **Note:** You may specify up to 16 different remote devices indexed from 1 to 16 in this table.
Slave IP Address	Enter the IP address of the remote device (Modbus TCP/IP server) controller in this field. **Note:** You must configure the slave IP addresses starting at Index 1 and in growing index number, in a consecutive manner. For example, configuring slave IPs of index 1 than 3 is not allowed, for you must first configure the entry indexed 2 prior to index 3.
Unit ID	Enter the Modbus Unit ID (or Protocol Address) in this field. A valid Unit ID can range from 0 to 255. The default setting is 255. A Unit ID (other than 255) makes communications with a remote device across a Modbus bridge or gateway possible. If the target device is another Twido controller or a legacy Modbus device installed on another bus - serial link address via a gateway, then you may set the Unit ID of that remote device, accordingly. In the field, you should set the Slave IP as the gateway or bridge IP address, and the Unit ID as the Modbus serial link address of your target device.
Connection Timeout (100 ms)	Specify the elapsed time in units of 100 ms that the Twido controller will keep trying to establish a TCP connection with the remote device. If the connection is still not established after Timeout, the Twido controller will give up trying, until the next connection request by an EXCH3 instruction. A valid timeout setting can range from 0 to 65535 (which translates to 0 to 6553.5 s). The default setting is 100.

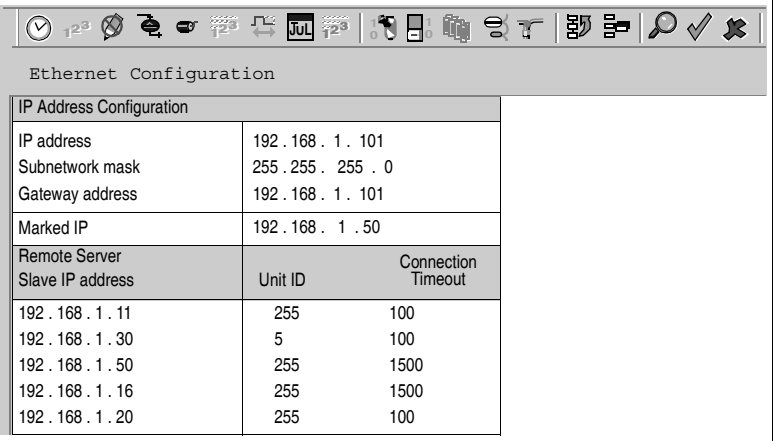
Viewing the Ethernet Configuration

Overview

You may use the TwidoSoft **Configuration Editor** to view the current Ethernet configuration of the Twido controller.

Viewing the Ethernet Configuration

To view the current Ethernet configuration settings using the Configuration Editor, follow these instructions:

Step	Action
1	Select **Program > Configuration Editor** from the TwidoSoft menu bar.
2	Click on the shortcut labeled **ETH** in the Configuration Editor taskbar or double click on the **Ethernet Port** shortcut in the Application Browser.
3	The Ethernet TCP/IP Configuration parameters appear in a table as shown in the figure below:  Ethernet Configuration IP Address Configuration IP address 192 . 168 . 1 . 101 Subnetwork mask 255 . 255 . 255 . 0 Gateway address 192 . 168 . 1 . 101 Marked IP 192 . 168 . 1 . 50 Remote Server Slave IP address / Unit ID / Connection Timeout 192 . 168 . 1 . 11 — 255 — 100 192 . 168 . 1 . 30 — 5 — 100 192 . 168 . 1 . 50 — 255 — 1500 192 . 168 . 1 . 16 — 255 — 1500 192 . 168 . 1 . 20 — 255 — 100
4	At this stage, if you have just made changes to your Twido's Ethernet TCP/IP configuration settings, you may still decide to keep the changes or to discard them and restore the previous configuration, as explained below: • Select **Tools > Accept Changes** from the TwidoSoft menu bar, to keep the changes you have made to the TCP/IP Ethernet configuration. • Select **Tools > Cancel Changes** to discard the changes and restore the previous TCP/IP Ethernet configuration settings. • Select **Tools > Edit...** to return to the Ethernet Configuration dialogbox and modify the TCP/IP configuration settings. • Select **PLC > Transfer PC=>PLC...** to download the complete PLC configuration file into the Twido controller.

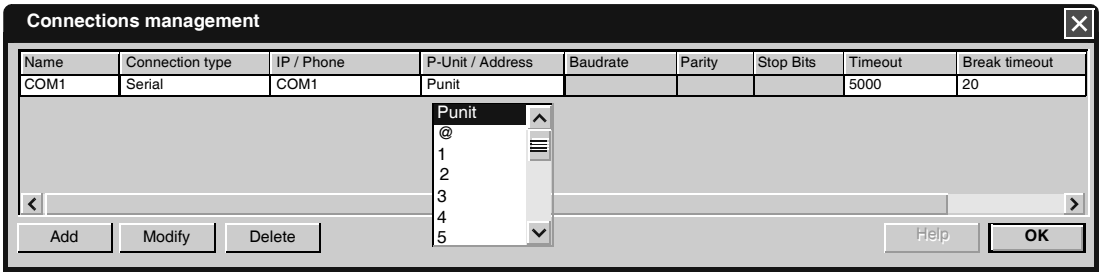
Ethernet Connections Management

Overview

The following information describes how to configure/add/delete/select a PC-to-controller Ethernet TPC/IP connection.

Setting Up a New TCP/IP Connection

To set up an Ethernet TCP/IP connection between your PC running the TwidoSoft application and a TWDLCAE40DRF controller installed on your network, follow these instructions.
Select **File** → **Preferences** from TwidoSoft menu bar to call up **Connexions management** dialog box:



Step	Action
1	Click the **Add** button in the **Connections Management** dialog box. **Result:** A new connection line is added. The new line displays suggested default connection settings. You will need to change these settings. **Note:** To set a new value in a field, you have two options: • Select the desired field, then click the **Modify** button. • Double click the desired field.
2	In the **Name** field, enter a descriptive name for the new connection. A valid name may contain up to 32 alphnumeric characters.
3	In the **Connection Type** field, click to unfold the dropdown list that includes: TCP/IP, Serial, Modem (if any) and USB (if any). Select **TCP/IP** as you are setting up a new Ethernet connection between your PC and an Ethernet-capable Twido controller.
4	In the **IP / Phone** field, enter a valid **IP address** which is the IP information of the Twido TWDLCAE40DRF controller you wish to connect to. **IP Address:** Enter the static IP address that you have specified for your Twido controller in a previous section.

Step	Action
5	The **Punit / Address** field can be filled in when IP / Phone has been selected. For a TCP/IP Type connection, default value is **Direct**. For a Serial Type connection, default value is **Punit**. When any of those is selected, next three fields (Baudrate, Parity and Stop Bits) are disabled . If you do not know the controller address, **@** allows you to select it later, once the program has been downloaded. (A window pops up before the first connection to let you choose the controller where you transfer to, with a 1-247 range, and 1 as the default address value.)
6	In the **Timeout** field, enter a timeout value in milliseconds (ms) for establishing a connection with the Twido controller. After timeout has elapsed and the PC has failed to connect to the controller, the TwidoSoft application will give up trying to establish a connection. To resume a new attempt for connection, select **PLC** → **Select a connection** from TwidoSoft menu bar. **Note:** Default Timeout value is **500** ms. Maximum Timeout value is 65535 x 100 ms (6553.5 s).
7	The **Break timeout** value is the maximum elapsed time allowed between a Modbus TCP/IP query and the reception of the response frame. If Break timeout is exceeded without receiving the requested response frame, the TwidoSoft application breaks the connection between the PC and the controller. **Note:** Default Break timeout value is **20** ms. You must set a non-zero value.
8	Click the **OK** button to save the new connection settings and close the Connections management dialog box. **Result:** The names of all newly-added connections are added to the dropdown list of connections in the **File** → **Preferences** dialog box or in the **PLC** → **Select a connection** menu.

Modifying and Deleting a TCP/IP Connection

Existing Ethernet TCP/IP connections can be deleted or have their parameters modified, as follows:

- To delete a connection from the Ethernet management dialog box, select a connection Name, then click the **Delete** button. Note that, after deletion, all connection parameters are permanently lost.
- To modify the parameters of an existing connection, select the desired field, and click the **Modify** button. Then, you can enter a new value in the selected field.

Ethernet LED Indicators

Overview

Two Ethernet communications LED indicators are located on the LED panel, at the front panel of the TWDLCAE40DRF controller and on the soft front-panel accessible via the **PLC > Check PLC** path in the TwidoSoft application as well. They are label as follows:·

- LAN ACT
- LAN ST

The Ethernet LEDs provide continuous monitoring of the Ethernet port connections status and diagnostics.

LED Status

The following table describes the status of both **LAN ACT** and **LAN ST** Ethernet LED indicators.

LED	State	Color	Description
LAN ACT	Off	-	No Ethernet signal on RJ-45 port.
	Steady	Green	10BASE-TX link beat signal to indicate a 10 Mbps connection.
	Blinking		Data packets sent or received over the 10BASE-TX connection.
	Steady	Amber	100BASE-TX link beat signal to indicate a 100 Mbps connection.
	Blinking		Data packets sent or received over the 100BASE-TX connection.
LAN ST	Steady	Green	Base controller is powered on. Ethernet port is ready to communicate over the network.
	Fast flashing		Ethernet initialization at power-up.
	2 Flashes, long off		No valid MAC address.
	3 Flashes, long off		Any of three possible causes: ● No link beat detected. ● Ethernet network cable is not plugged correctly or faulty cable. ● Network device (hub/switch) is faulty or not properly configured.
	4 Flashes, long off		Duplicate IP address detected over the network. (To remedy this situation, try assigning a different IP address to your Twido controller.)
	6 Flashes, long off		Using a valid converted default IP address; FDR safe-mode.
	9 Flashes, long off		Ethernet hardware failure.

TCP Modbus Messaging

Overview

You may use TCP Modbus messaging to allow the Modbus TCP Client (Master controller) to send and receive Ethernet messages to and from the Modbus TCP Server (Slave controller). As TCP Modbus is a peer-to-peer communications protocol, a Twido Ethernet-capable controller can be both Client and Server depending on whether it is querying or answering requests, respectively.

Message Exchange over the Ethernet Network

Ethernet messaging is handled by the EXCH3 instruction and the %MSG3 function block: Routing to an Ethernet host or via a gateway is supported by EXCH3, as well.

- **EXCH3 instruction:** to transmit/receive messages
- **%MSG3 Function Block:** to control the message exchanges.

EXCH3 Instruction

The EXCH3 instruction allows the Twido controller to send and/or receive information to/from Ethernet network nodes. The user defines a table of words (%MWi:L) containing control information and the data to be sent and/or received (up to 128 bytes in transmission and/or reception). The format for the word table is described in the following section.
A message exchange is performed using the EXCH3 instruction:

Syntax: [EXCH3 %MWi:L]
where: L = number of words in the control words, transmission and reception tables

The Twido controller must finish the exchange from the first EXCH3 instruction before a second can be launched. The %MSG3 function block must be used when sending several messages.
The processing of the EXCH3 list instruction occurs immediately, with any transmissions started under interrupt control (reception of data is also under interrupt control), which is considered background processing.

Note: Usage of the EXCH3 instruction is the same as EXCHx (where x = 1 or 2) used with legacy Modbus. Instruction syntaxes are also identical. However, there is one major difference in the information carried by Byte1 of the transmission and reception tables. While Byte1 of the legacy Modbus conveys the serial link address of the slave controller, Byte1 of the TCP Modbus carries the **Index** number of the Modbus TCP client controller. The Index number is specified and stored in the Remote Devices table of the TwidoSoft Ethernet Configuration (for more details see*Remote Devices Tab, p. 203*).

EXCH3 Word Table

The maximum size of the transmitted and/or received frames is 128 bytes (note that this limitation applies to the TCP Modbus client only, while the TCP Modbus server supports the standard Modbus PDU length of 256 bytes). Moreover, the word table associated with the EXCH3 instruction is composed of the control, transmission and reception tables, as described below:

	Most significant byte	**Least significant byte**
Control table	Command	Length (Transmission/ Reception)
	Reception Offset	Transmission Offset
Transmission table	Transmitted Byte 1 (**Index** as specified in the Remote Device Table of the TwidoSoft Ethernet Configuration dialogbox.)	Transmitted Byte 2 as Modbus serial
	...	Transmitted Byte n
	Transmitted Byte n+1	
Reception table	Received Byte 1 (**Index** as specified in the Remote Device Table of the TwidoSoft Ethernet Configuration dialogbox.)	Received Byte 2 as Modbus serial
	...	Received Byte p
	Received Byte p+1	

%MSG3 Function Block

The use of the %MSG3 function is identical to that of %MSGx used with legacy Modbus. %MSG3 is used to manage data exchanges by providing:

- Communications error checking
- Coordination of multiple messages
- Transmission of priority messages

The %MSGx function block has one input and two outputs associated with it:

Input/Output	Definition	Description
R	Reset input	Set to 1: re-initializes communication or resets block (%MSGx.E = 0 and %MSGx.D = 1).
%MSGx.D	Communication complete	0: request in progress. 1: communication done if end of transmission, end character received, error, or reset of block.
%MSGx.E	Error	0: message length OK and link OK. 1: if bad command, table incorrectly configured, incorrect character received (speed, parity, and so on.), or reception table full.

EXCH3 Error Code

When an error occurs with the EXCH3 instruction:

- bits **%MSG3.D** and **%MSG3.E** are set to **1**, and
- the Ethernet communication **error code** is recorded into system word **%SW65**.

The following table presents the EXCH3 error code:

EXCH3 Error Code (recorded into System Word %SW65)
Standard error codes common to all EXCHx (x = 1, 2, 3): 0 - operation was successful 1 – number of bytes to be transmitted is too great (> 128) 2 - transmission table too small 3 - word table too small 4 - receive table overflowed 5 - time-out elapsed (Note that eror code 5 is void with the EXCH3 instruction and replaced by the Ethernet-specific error codes 109 and 122 described below.) 6 - transmission 7 - bad command within table 8 - selected port not configured/available 9 - reception error 10 - can not use %KW if receiving 11 - transmission offset larger than transmission table 12 - reception offset larger than reception table 13 - controller stopped EXCH processing
Ethernet-specific error codes for EXCH3: 101 - no such IP address 102 - the TCP connection is broken 103 - no socket available (all connection channels are busy) 104 - network is down 105 - network cannot be reached 106 - network dropped connection on reset 107 - connection aborted by peer device 108 - connection reset by peer device 109 - connection time-out elapsed 110 - rejection on connection attempt 111 - host is down 120 - unknown index (remote device is not indexed in configuration table) 121 - fatal (MAC, Chip, Duplicated IP)122 - receiving timed-out elapsed after data was sent 123 - Ethernet initialization in progress

4.4 Configuring Software

At a Glance

Overview

This section contains detailed procedures for using TwidoSoft to configure software resources for an application.

What's in this Section?

This section contains the following topics:

Maximum Software Configuration

Introduction

The following table lists maximum software variables for each base controller model for Twido. Use this table as a guide for configuring applications. Configuration options are determined by the model.

Base Controller Models

Maximum number of configurable variables for each base controller model:

Part	Compact Base Controllers				Modular Base Controllers				
	10DRF	16DRF	24DRF	40DRF	20DUK	20DTK	20DRT	40DUK	40DTK
Timers	64	64	128	128	128	128	128	128	128
Constants	256	256	256	256	256	256	256	256	256
Counters	128	128	128	128	128	128	128	128	128
Fast counters	3	3	3	4	2	2	2	2	2
Very fast counter:	1	1	1	2	2	2	2	2	2
LIFO/FIFO registers	4	4	4	4	4	4	4	4	4
Schedule blocks	16	16	16	16	16	16	16	16	16
Drum controllers	4	4	8	8	8	8	8	8	8
PLS/PWM	0	0	0	2	2	2	2	2	2
Step counter	8	8	8	8	8	8	8	8	8
Shift bit register	8	8	8	8	8	8	8	8	8

Configuring Timers

Introduction

Use the **Timers** dialog box to configure timers.

Configuring Timers

To configure a timer using the **Timers** dialog box:

1. Enter the **Timer** number in the **%TM** box.
2. The **Symbol** box displays the alphanumeric name, if assigned, for this variable (see *Defining Symbols, p. 81* for more details.)
3. Select the **Timer Type**. Options: **TON** (Timer On Delay), **TOF** (Timer Off Delay), and **TP** (Timer Pulse). Default: TON.
4. Select the **Time Base** which sets the unit of time for the timer. Options: **1 ms** (%TM0 to %TM5 only), **10 ms**, **100 ms**, **1 sec**, and **1 min**. Default: 1 min.
5. Select a **Preset** value. Range: 0 to 9999. Default: 9999.
6. Select the **Adjustable** option to allow the **Preset** value to be changed using the Animation Tables editor.
7. Use **Previous** and **Next** to scroll through the list of timers (only available when dialog box is opened from the Configuration Editor).
8. Select **OK** to close the dialog box and accept changes.

Configuring Counters

Introduction

Use the **Counters** dialog box to configure up/down counting.

Configuring Counters

To configure a up/down counter using the **Counters** dialog box:

1. Enter the **Counter** number in the **%C** box.
2. The **Symbol** box displays the alphanumeric name, if assigned, for this variable (see *Defining Symbols, p. 81* for more details.)
3. Select a **Preset** value. Range: 0 to 9999. Default: 9999.
4. Select the **Adjustable** option to allow the **Preset** value to be changed using the Animation Tables editor.
5. Use **Previous** and **Next** to scroll through the list of counters (only available when dialog box is opened from the Configuration Editor).
6. Select **OK** to close the dialog box and accept changes.

LIFO/FIFO Registers Dialog Box

Function

Use this dialog box to configure the LIFO and FIFO registers.

Access

Use one of the following methods to access the Registers dialog box:
- Clicking on the LIFO/FIFO Registers in the Application Browser
- Clicking on the LIFO/FIFO Registers selection from the Software menu
- Selecting a specific LIFO/FIFO Register in the Configuration Editor window

Example

The following is an example of the LIFO/FIFO Register dialog box:

Dialog Box Options

The following tablle briefly identifies each selection option, describes the function of the selection, and defines selection conditions or values:

Selection	Description	Value/Action
Register %R	Enter the number of the register	Enter the number between 0 and 3
Type	Select type of register	LIFO - Last In, First Out, or FIFO - First In First Out
OK	Select to accept changes	Closes dialog box
Cancel	Select to lose changes	Closes dialog box
Previous	Selects previous timer	Only available when accessed from the Configuration Editor
Next	Selects next timer	

Entry conditions

Identify any setting conditions required in order to use this dialog box:
- Operation mode?

Drum Controllers

Introduction

Use the **Drum Controllers** dialog box to configure drum controllers function blocks.

Configuring Drum Controllers

To configure drum controllers:

Step	Action
1	Enter a **Drum** number in the **%DR** box. **Range**: see *Maximum Software Configuration, p. 215*.
2	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
3	Enter **Number of Steps**. **Range:** 1 to 8. The check boxes for steps that are not used will appear dimmed.
4	In **Outputs**, assign either a physical bit or a memory bit to each of the 16 (0 to 15) logical control bits that will be used for each defined step. All outputs %Q must be valid I/O for the hardware configuration, and cannot be expansion I/O on Remote controllers. Internal memory bit %M range is determined by the base controller model.
5	Select check boxes in each step to activate bits that have been assigned a value in **Outputs**. **For example**: When a step is run, the logical bits defined in **Output** will be activated if the corresponding check box in the step is selected
6	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
7	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

Constants

Introduction

Use the **Constants** dialog box to configure program constants. Constants are memory words that contain values or alphanumeric messages entered during configuration.

Configuring Constants

To configure constants:

Step	Action
1	Enter the **Constant %KW** number to configure. **Range**: see *Maximum Software Configuration, p. 215*.
2	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
3	Enter a **Value** for the constant. **Range**: Decimal: -32768 to 32767. Hexadecimal: 0000 to FFFF. Binary: 000000000000000 to1111111111111111 (maximum of sixteen bits). For ASCII: Any two ASCII characters.
4	Select the **Value Format**. **Options**:**Hexadecimal**, **Decimal, Binary**, or **ASCII**
5	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
6	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

constants KD

Introduction

Use the **Constants KD** dialog box to configure program constants KD. Constants KD are memory double words that contain values or alphanumeric messages entered during configuration.

Configuring Constants

To configure constants:

Step	Action
1	Enter the **Constant %KD** number to configure. **Range**: 0 to 254.
2	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
3	Enter a **Value** for the constant. **Range**: Decimal: -2147483648 to +2147483647. Hexadecimal: 80000000 to 7FFFFFFF.
4	Select the **Value Format**. **Options**:**Hexadecimal** or **Decimal**.
5	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
6	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

constants KF

Introduction

Use the **Constants KF** dialog box to configure program constants KF. Constants KF are floating constants containing the values entered on configuration.

Configuring Constants

To configure constants:

Step	Action
1	Enter the **Constant %KF** number to configure. **Range**: 0 to 254.
2	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
3	Enter a **Value** for the constant. **Range**: Floating word between: [-3.402824e+38 ; -1.175494e-38] and [1.175494e-38 ; 3.402824e+38]. Only the value 0 is accepted between -1.175494e-38 and 1.175494e-38
4	Select the **Value Format**. **Options**:**Floating word**.
5	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
6	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

%PLS/%PWM

Introduction

Use the **%PLS/%PWM** dialog box to configure the Pulse Generator function blocks.

Note: Outputs %Q0.0 and %Q0.1 are dedicated to either the %PLS or the %PWM function block. Configuring %PLS0/%PWM0 requires the dedicated output %Q0.0. If %Q0.0 is already in use, then %PLS0/%PWM0 cannot be configured. Conversely, if %PLS0/%PWM0 is configured, then %Q0.0 cannot be used elsewhere. The same is true for %PLS1/%PWM1 which requires dedicated output %Q0.1. In the **%PLS/%PWM** dialog box, the output used by each counter is displayed in the **Dedicated Output** box.

Note: You will not be able to view the value of the dedicated output bit for %PLS and %PWM on the optional Operator Display or the Animation Tables Editor. The operator display is updated at a specific interval of the controller scan cycle. This can cause confusion in interpreting the display of dedicated outputs for %PLS and %PWM pulses. At the time these outputs are sampled, their value will always be zero, and this value will be displayed. Ensure that the actual dedicated output is manipulated by the configuration of the controlling function block.

Note: The %PLS/%PWM function blocks are not available for some base controllers (see *Maximum Software Configuration, p. 215*).

Pulse Generation

The %PLS/%PWM pulse generator serves as a square wave generator with the following characteristics.

- %PLS: 50% duty cycle (on time equals the off time).
- %PWM: duty cycle can be varied by the program.

Double Word

Select **Double Word** to use 32-bit values for the pulse-generating function. Setting the pulse generator to double word mode allows to significantly increase the total number of pulses that can be produced to 4,294,967,296.

Note: Setting the pulse generator to double word mode does not impact on pulse generator's efficiency, but uses more system memory space. To check the memory usage select **PLC > Memory use** from the TwidoSoft menu bar.

Double Word Functions

The syntax of the 32-bit double-word functions used by the %PLS/%PWM function block is different from the syntax of the standard 16-bit single-word functions. The following table shows the correct syntax and valid range of the %PLS double-word functions versus the standard single-word syntax:

PLS single-word functions	PLS double-word functions		
Valid range: 0-65,535	**Valid range: 0-4,294,967,295**	**Function Definition**	**Setting**
%PLSi.N	%PLSi.ND	Number of pulses to generate	Configurable

Note: When using a double-word pulse generator (%PLSi.ND), you should no longer use single-word functions (%PLSi.N) for they contain erroneous values.

Configuring %PLS or %PWM

To configure the %PLS or %PWM pulse generators using the **%PLS/%PWM** dialog box:

Step	Action
1	Enter the **%PLS/%PWM** number. **Range**: see *Maximum Software Configuration, p. 215*.
2	Select **%PLS** or **%PWM** in the **PLS/PWM Type** box.
3	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
4	Select a **Time Base**. **Options**: **0.142 ms**, **0.57 ms, 10 ms** and **1 s. By default**: **1 s**.
5	Select the **Double Word** option. Yes: the %PLS/%PWM function block uses 32-bit computational variables with extended value range (0 to 4,294,967,295). No: the %PLS/%PWM function block uses standard 16-bit computational variables with limited value range (0 to 65,535).
6	Enter a **Preset** (regardless of selected computational mode: standard mode or double-word mode use the same Preset value). According to selected Time Base: • for 0.142 ms and 0.57 ms time bases: 0 - 255 • for 10 ms and 1 s time bases: 0 - 32767
7	Select an **Adjustable** option. Yes: You can change the **Preset** value using the *Animation Tables Editor, p. 50*. No: You cannot change the **Preset** value.
8	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
9	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

Fast Counters

Introduction

Use the **Fast Counters** dialog box to configure fast Up/Down Counters function blocks.

Note: Inputs %I0.2, %I0.3, and %I0.4 are dedicated inputs for use by Fast Counters function blocks. Configuring %FC0 requires dedicated input %I0.2. If %I0.2 is already in use, then %FC0 cannot be configured. Conversely, if %FC0 is configured, then %I0.2 cannot be used elsewhere. The same is true for %FC1 which requires dedicated input %I0.3. In the **Fast Counters** dialog box, the input used by each counter is displayed in the **Dedicated Input** box.

Configuration Options

The Fast Counters function block is a single software resource that can be configured as one of two types of counters:

- Up Counter
- Down Counter

Compact controllers allow a maximum of three fast counters, while Modular controllers allow only two fast counters.

Double Word

Select **Double Word** to use 32-bit values for the fast counting functions. Setting the Fast Counter to double word mode allows to significantly increase the total number of events that can be counted to 4,294,967,296.

Note: Setting the Fast Counter to double word does not impact on counter efficiency, but uses more system memory space. To check the memory usage select **PLC > Memory use** from the TwidoSoft menu bar.

Double Word Functions

The syntax of the 32-bit double-word functions used by the %FC function block is different from the syntax of the standard 16-bit single-word functions.
The following table shows the correct syntax and valid range of the Fast Counter double-word functions versus the standard single-word syntax:

FC single-word functions	FC double-word functions		
Valid range: 0-65,535	**Valid range: 0-4,294,967,295**	**Function Definition**	**Setting**
%FCi.P	%FCi.PD	Preset value	Configurable
%FCi.V	%FCi.VD	Current value	Read-only

Note: When using a double-word counter (%FCi.PD and %FCi.VD), you should no longer use single-word functions (%FCi.P and %FCi.V) for they contain erroneous values.

Configuring Fast Up and Down Counters

To configure a fast Up or Down Counter:

Step	Action
1	Enter the **Fast Counter %FC** number. **Range**: see *Maximum Software Configuration, p. 215*.
2	Select **Up Counter** or **Down Counter** for the **Fast Counter Type.**
3	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
4	Select the **Double Word** option. Yes: the %FC function block uses 32-bit computational variables with extended value range (0 to 4,294,967,295). No: the %FC function block uses standard 16-bit computational variables with limited value range (0 to 65,535).
5	In **Preset**, enter the number of events to be counted. **Standar range**: 0 to 65,535. **Double-word range**: 0 to 4,294,967,295.
6	Select an **Adjustable** option. Yes: You can change the **Preset** value using the *Animation Tables Editor, p. 50*. No: You cannot change the **Preset** value.
7	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
8	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

Very Fast Counters

Introduction

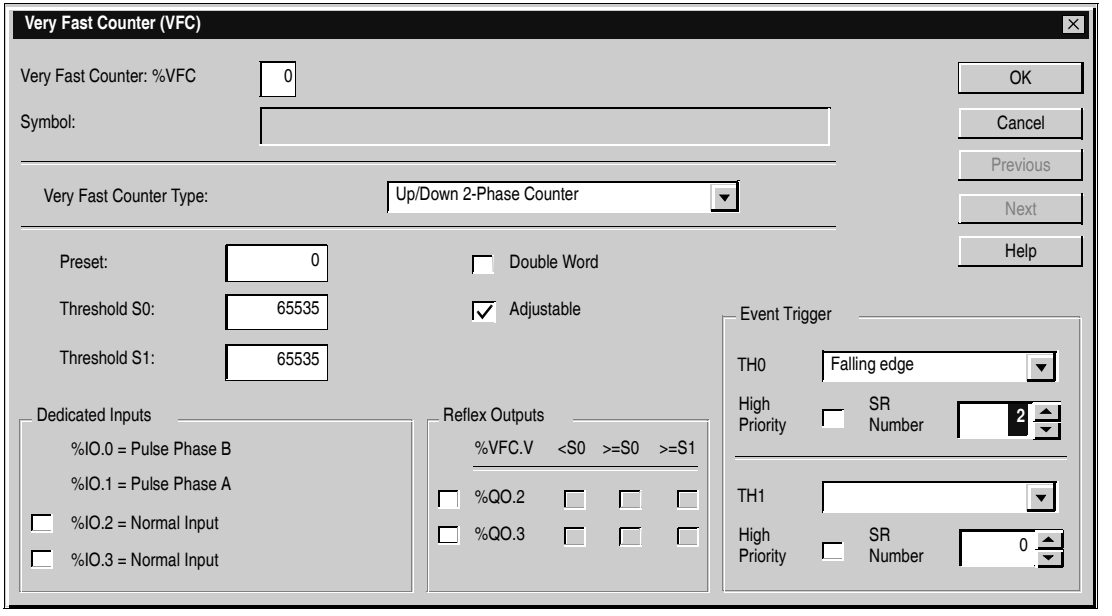
Use the **Very Fast Counters** dialog box to configure the Very Fast Timer (%VFC) function blocks.

Note: Inputs %I0.0 through %I0.7 are dedicated inputs for use by Very Fast Counters function blocks. Configuring %VFC0 requires dedicated input %I0.0 to %I0.3. If one of these inputs is already in use, it is impossible to configure %VFC0. Conversely, if %VFC0 is configured, then %I0.0 through %I0.3 cannot be used elsewhere. The same is true for %VFC1 which requires dedicated inputs %I0.1 to %I0.7. In the Very Fast Counters dialog box, the inputs used by each counter is displayed in the Dedicated Inputs box.

Note: The %VFC1 function block is not available for some base controllers (see *Maximum Software Configuration, p. 215*).

Illustration of the dialog box

The **Very Fast Counters** dialog box is displayed below:



Counter Options

The Very Fast Counter function block is a software resource that can be configured as Not Used or as one of the following types of counters:

- Up/Down Counter
- Up/Down 2-Phase Counter
- Single Up Counter
- Single Down Counter
- Frequency Meter

Not used

Setting the Very Fast Counter to Not Used releases its resources for use by other functions such as timers or schedule blocks.

Double Word

Select **Double Word** to use 32-bit values for the Very Fast Counter functions. Setting the Very Fast Counter to double word mode allows to significantly increase the total number of events that can be counted to 4,294,967,296.

Note: Setting the Very Fast Counter to double word mode does not impact on counter efficiency, but uses more system memory space. To check the memory usage select **PLC > Memory use** from the TwidoSoft menu bar.

Double Word Functions

The syntax of the 32-bit double-word functions used by the %VFC function block is different from the syntax of the standard 16-bit single-word functions.
The following table shows the correct syntax and valid range of the Very Fast Counter double-word functions versus the standard single-word syntax:

VFC single-word functions	**VFC double-word functions**		
Valid range: 0-65,535	**Valid range: 0-4,294,967,295**	**Function Definition**	**Setting**
%VFCi.P	%VFCi.PD	Preset value	Configurable
%VFCi.V	%VFCi.VD	Current value	Read-only
%VFCi.S0	%VFCi.S0D	Threshold value S0	Configurable
%VFCi.S1	%VFCi.S1D	Threshold value S1	Configurable
%VFCi.C	%VFCi.CD	Capture value	Read-only

Note: When using a double-word counter (%VFCi.PD, %VFCi.VD, %VFCi.CD, %VFCi.S0D and %VFCi.S1D), you should no longer use single-word functions (%VFCi.P, %VFCi.V, %VFCi.C, %VFCi.S0 and %VFCi.S1) for they contain erroneous values.

Preset

Set the **Preset** value to the total number of events to be counted. Enter a decimal value between 0 and 65,535 in standard mode, and between 0 and 4,294,967,295 in double-word mode.
The Preset value is loaded into the present value of the very fast counter (%VFC.V) on the rising edge of %I0.2 for %VFC0 and %I0.5 for %VFC1.

Thresholds, reflex outputs and outputs TH0 and TH1

Set **Threshold** values S0 and S1 to decimal values between 0 and 65,535 in standard mode, and between 0 and 4,294,967,296 in double-word mode.
Threshold values are used:

- in conjunction with the dedicated **Reflex** outputs %Q0.2 and %Q0.3 for VFC0, and %Q0.4 and %Q0.5 for VFC1. The counter is constantly compared to these values to set the Reflex outputs according to configuration settings. Reflex outputs are values that are available immediately.
- for potentially triggering each event according to the configuration of the **TH0 and TH1** outputs. The counter is constantly compared to these threshold values to set the TH0 and TH1 outputs. In turn, the outputs switch to 1 when the value of the counter is greater than the S0 and S1 thresholds. They switch to 0 when the value of the counter is smaller than the S0 and S1 thresholds. Events can then be triggered according to the status of the outputs (see `Configuration of an event's trigger`).

Note: For events management:
For threshold values very close to the limits (0, 65535 in standard mode or 0, 4,294,967,295 in double word mode) to high frequencies (>1KHz), the events triggered on TH0 and TH1 edges are not detected.
Example: for a threshold S1=65525 (in standard single-word mode) in up-counting mode, TH1 switches to '1' for 11 count cycles, before returning to 0. For an input frequency of 10KHz -> period = 100s; TH1 is set to 1 for 1.1ms and no event is overlooked. For an input frequency of 20KHz -> period = 50s; TH1 is set to 1 for 550s; 9 events out of 20 are overlooked.
Conclusion: the events are not scanned enough (currently every 1ms approximately) to be able to reliably detect pulses less than 1ms.

Catch Input

The current counting value does not update automatically. Use the Catch (or capture) input to force the update of %VFCi.V and copy it into %VFCi.C. %VFC0 uses the dedicated input %I0.3, and %VFC1 uses the dedicated input %I0.4. These dedicated inputs are shared with Fast Counter function blocks, the Catch Input feature can not be used if these inputs are already used.

Adjustable

Select **Adjustable** to allow an application to modify the Preset (%VFC.P) and Thresholds (%VFC.S0 and %VFC.S1). If Adjustable is selected, the Animation Tables Editor and the optional HMI can modify these values.

Frequency meter

When **Frequency Meter** is selected, only the Pulse input is needed. A frequency is measured by counting the rising edges of the input signal. In **Time Window**, select an interval of 100 ms or 1s.

Configuring Very Fast Counters

To configure a Very Fast Counter:

Step	Action
1	Enter the **Very Fast Counter %VFC** number. **Range**: see *Maximum Software Configuration, p. 215*.
2	**Symbol** is the alphanumeric name, if assigned, for this variable. See *Symbol Editor, p. 45*.
3	Select the **Fast Counter Type.** Options: Up/Down Counter, Up/Down 2-Phase Counter, Single Up Counter, Single Down Counter, or Frequency Meter.
4	Select the **Double Word** option. Yes: the %VFC function block uses 32-bit computational variables with extended value range (0 to 4,294,967,295). No: the %VFC function block uses standard 16-bit computational variables with limited value range (0 to 65,535).
5	In **Preset**, enter the number of events to be counted. **Standar range**: 0 to 65,535. **Double-word range**: 0 to 4,294,967,295.
6	Enter values for **Threshold S0** and **Threshold S1**. **Standar range**: 0 to 65,535. **Double-word range**: 0 to 4,294,967,295.
7	Select an **Adjustable** option. Yes: you can change the **Preset** value using the *Animation Tables Editor, p. 50*. No: you cannot change the **Preset** value.
8	Select an **Adjustable** option. Yes: you can change the **Preset** value using the *Animation Tables Editor, p. 50*. No: you cannot change the **Preset** value.
9	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
10	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

Configuration of an event's Deactivation edit box.

This area is divided into 2 parts: one depends on the value of the S0 threshold and the other depends on the value of the S1 threshold. Each time the counter reaches one of these 2 values , a rising or falling edge is generated on the TH0 and TH1 outputs.
The 2 parts of the event trigger zone are configured in the following independent and identical way:

Step	Action
1	Open the drop-down menu associated with the TH0 or TH1 output then select an option: ● **Not used** for there to be no event attached to this output, ● **Rising edge** for an event to be triggered when the value of the output switches to 1, ● **Falling edge** for an event to be triggered when the value of the output switches to 0, ● **Both edges** for an event to be triggered when the value of the output varies.
2	Open the drop-down menu **SR Number** only if the output option is other than **Not Used**, then select the number of the subroutine to be executed when an event is triggered on this output: ● **0** (maximum 1 subroutine) for Twido10 series controllers, ● **0 to 15** (maximum 16 subroutines) for the Twido16 controller. ● **0 to 63** (maximum 64 subroutines) for the other series of Twido controllers. Note: If the selected section number does not correspond to any implemented subroutine, or if the section number is already used by another event, the numbers are nevertheless available from the drop-down menu. Warning messages alert the user to these situations.
3	If the output option is other than **Not Used**, check the associated **High Priority** box to set the output to be used with a high-priority event trigger. Notes: ● If an output's box is checked, the other output's box can no longer be used. ● If one of the inputs from %I0.2 to %I0.5 of the controller base is already used on activation of a High Priority event, a dialog box alerts the user that the input in question is in high priority, and suggests interchanging the priority.
4	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

Schedule Blocks

Introduction

Use the **Schedule Blocks** dialog box to control Date/Clock functions, which are actions scheduled at predefined times. This function requires the Real-Time Clock option.

Configuring Schedule Blocks

To configure schedule blocks:

Step	Action
1	Enter the **Schedule Block** number to configure. **Range**: see *Maximum Software Configuration, p. 215*.
2	Click in the **Configured** check box to activate the dialog box boxes.
3	Enter an **Output Bit** address. For example, when configuring a sprinkler to turn on, enter the address of the output terminal connected to the sprinkler system. All outputs %Q must be valid I/O for the hardware configuration, and cannot be expansion I/O on Remote controllers. Internal memory bit %M range is determined by the base controller model
4	Select **Start Month** and **End Month** for starting and ending the Schedule Block. **Range**: January to December.
5	Select **Start Date** and **End Date** for the day of the month for starting and ending the Schedule Block. Range: 1 to 31 (depending on month selection).
6	Select the **Start Time** and **End Time** for setting the time of day for starting and ending the Schedule Block. Range: 00:00 to 23:59.
7	Select the **Days of the Week** to activate the schedule block output. Checked boxes indicate the days that the output device is activated.
8	Select **Previous** or **Next** to scroll to the previous or next item without closing the dialog box. (Only displayed when the Configuration Editor window is open.)
9	Select **OK** to accept the selected values, or select **Cancel** to close the dialog box without changing the configuration.

Step Counter

Introduction

Use the **Step Counter** dialog box to assign a number from 0 to 7 for a specific Step Counter function block (%SC).

Opening the Dialog Box

This dialog box is opened by double-clicking on a Step Counter function block in a ladder rung.

Configuring a Step Counter

To configure a Step Counter:

Step	Action
1	Enter a number 0 to 7 in the **Step Counter** box.
2	Select **OK** to accept the changes and close the dialog box. Select **Cancel** to close the dialog box without changes to the current settings.

Shift Bit Register

Introduction

Use the **Shift Bit Register** dialog box to select the a number from 0 to 7 for a specific Shift Bit Register function block (%SBR).

Opening the Dialog Box

This dialog box is opened by double-clicking on a Shift Bit Register function block in a ladder rung.

Configuring a Shift Bit Register

To configure a Shift Bit register:

Step	Action
1	Enter a number 0 to 7 in the **Shift Bit Register** box.
2	Select **OK** to accept the changes and close the dialog box. Select **Cancel** to close the dialog box without changes to the current settings.

4.5 Configuring Program Parameters

At a Glance

Overview

This section contains detailed procedures for using TwidoSoft to configure program or application-wide settings.

What's in this Section?

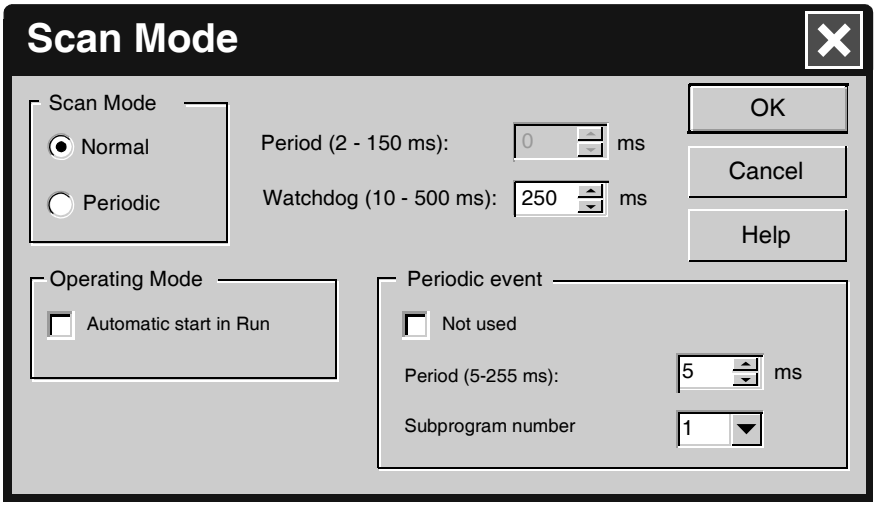
This section contains the following topics:

Scan Mode

Introduction

Use the **Scan Mode** dialog box to select how the controller will scan when running a program. Scan time is monitored by the controller's Watchdog Timer. If scan time exceeds 150 ms, a fault is generated, and the controller stops immediately.
This dialog box can also be used to configure a periodic event.

Scan mode



Scan mode can be set to normal or periodic:

- Normal (Cyclic)
  Continuous, cyclic scanning; a scan starts immediately after the current scan is completed.
- Periodic
  Controller waits until the minimum scan time has been reached before starting another scan.

Period

The duration of a periodic scan has a selectable range of 2 ms to 150 ms.
If **Periodic** Scan mode is unchecked, Period box is grayed and displays 0.

Watchdog

The execution of the application is monitored by an application watchdog timer which has a default value of 250 ms. The watchdog timer has a range of selectable values from 10 ms to a maximum of 500 ms.

Operating mode

Selecting the **Automatic Start in Run** function places the controller in automatic execution mode on start-up.

Note: If you do not check Automatic Start in Run and do not check any **Run/Stop** input in **Edit input configuration** dialog box, a warning will be displayed in the Program Error window when you transfer your application to the controller: <EXECUTION WARNING> NO AUTOMATIC START IN RUN SELECTED

⚠ WARNING

UNEXPECTED EQUIPMENT OPERATION

When using Automatic start in Run, it is essential to know in advance how automatic reactivation of the outputs will affect the process or machine being controlled.

Failure to follow this instruction can result in death, serious injury, or equipment damage.

Periodic Event

An event can be periodically triggered, with a period that can be defined between 5 and 255 ms (this is the trigger period). Every time an event is triggered, the task (subroutine) is executed, the number of which can also be defined.

Note: Restrictions:
- The periodic event cannot be high-priority,
- Only one periodic event can be configured.

Configuring Scan Mode

To configure Scan Mode:

Step	Action
1	Select **Program** → **Edit scan mode...** from the main menu.
2	Select the type of **Scan Mode**. **Options:** Periodic or Normal
3	If the type of **Scan Mode** selected is **Periodic** (this is the scan period), enter the scan time in the **Period: (2 - 150 ms)** zone.
4	Select **OK** to accept the changes and close the dialog box. Select **Cancel** to close the dialog box without changes to the current settings.

Configuring the Periodic Event

To configure the periodic event:

Step	Action
1	Select **Program** → **Edit scan mode...** from the main menu.
2	Uncheck the box **Not used** from the **Periodic event** zone.
3	Increment the period in the **Period (5 - 255 ms)** field.
4	Select the subroutine number to be periodically executed by incrementing the **Subroutine number** field. Possible selections: ● **0** (maximum 1 subroutine) for Twido10 series controllers, ● **0 to 15** (maximum 16 subroutines) for the Twido16 controller. ● **0 to 63** (maximum 64 subroutines) for the other series of Twido controllers.
5	Select **OK** to accept the changes and close the dialog box. Select **Cancel** to close the dialog box without changes to the current settings.

Application Name

Introduction

Use the **Application Name** dialog box to define a name for an application that is not restricted to the Windows operating system limits for file names and provides more flexibility for identifying applications.
The application name is not the same as the Windows file name and is only internal to the application (see *Naming an Application, p. 77*).

Naming an Application

To name an application:

Step	Action
1	Select **Program** → **Rename** from the main menu, or right-click on the current name of the application in the Application Browser.
2	Enter a new name in the **Application Name** box. An application name can be from 1 to 32 characters and consist of any characters including spaces, periods, and front and back slashes.
3	Select **OK** to accept the changes and close the dialog box. Select **Cancel** to close the dialog box without changes to the current settings.

Programming in Ladder Language

5

At a Glance

Overview

This chapter provides detailed instructions for using TwidoSoft to create Ladder programs.

What's in this Chapter?

This chapter contains the following sections:

5.1 Creating Ladder Programs

At a Glance

Overview

This section provides an overview of using TwidoSoft to create Ladder programs.

What's in this Section?

This section contains the following topics:

Overview of Ladder Programs

Introduction

A program written in Ladder language consists of networks of linked graphical elements organized into rungs that are executed sequentially by the controller. A ladder diagram is a graphical representation of a ladder program similar to a relay logic diagram.
Each rung consists of one network of linked graphical elements organized in a programming grid starting with a potential bar on the left and ending with a potential bar on the right.

Graphical Elements

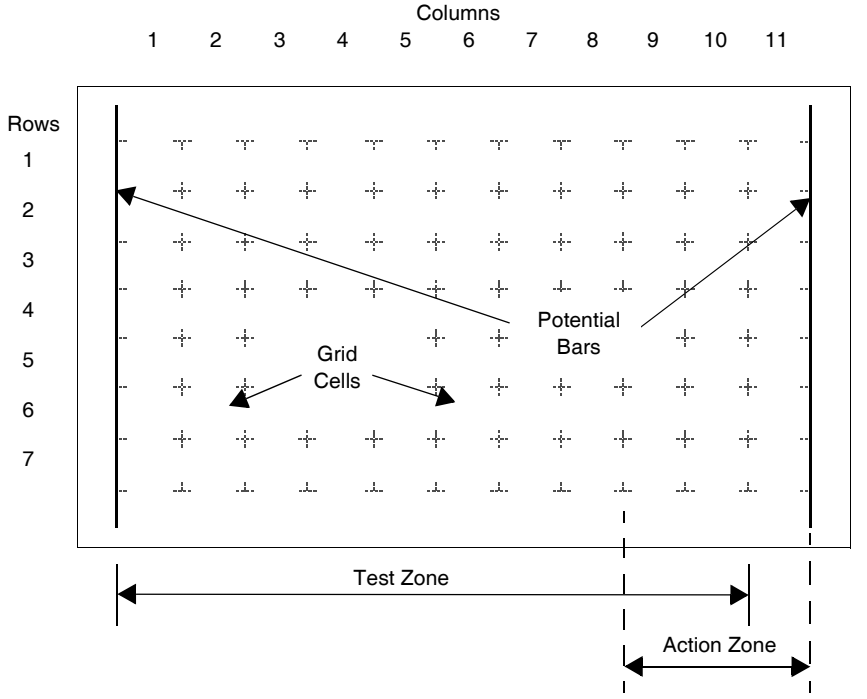
The graphical elements represent:
- Controller inputs and outputs such as sensors, push buttons, and relays.
- Arithmetic, logical, comparison, and numerical operations.
- Predefined system functions such as timers and counters.
- Controller internal variables such as bits and words.

Rungs

Each rung consists of:
- A Rung Header which documents the purpose of the rung. See *Rung Header, p. 41*.
- A programming grid of cells with a maximum of seven rows and eleven columns. Each cell is one row high by one column wide. Graphical elements are inserted into the grid and are linked with horizontal and vertical connector lines.

Programming Grid

The following is an illustration of a programming grid.



Grid Zones

The ladder diagram programming grid is divided into two zones:

- Test Zone
  Contains the conditions that are tested in order to perform actions. Consists of columns 1 - 10, and contains contacts, function blocks, and comparison blocks.
- Action Zone
  Contains the output or operation that will be performed according to the results of the tests of the conditions in the Test Zone. Consists of columns 9 - 11, and contains coils and operation blocks.

Creating a Ladder Program

Introduction

A Ladder program can be created at any time while developing an application. Once the Ladder program is completed, you can configure hardware, define symbols, and print documentation.
See *Developing an Application, p. 74*.

Basic Steps

The following are the basics steps for creating a Ladder program from an open application:

Step	Action
1	Open the *Ladder Viewer, p. 36*. See *Preferences, p. 31* to configure the view of ladder programs.
2	Insert the first rung. Select **Tools** → **Insert Rung** to open the *Ladder Editor, p. 38* and create the first rung in the ladder program.
3	Enter optional information in the Rung Header (see *Using Rung Headers, p. 255*). Double-click on the Rung Header to enter the following: ● Rung title and comments ● Declare Label (%Li) ● Declare a Subroutine (SRi) ● Declare a Grafcet step
4	Select graphic instructions from the *Ladder Palette Toolbar, p. 258* and the *Extended Ladder Palette, p. 40* and insert into the rung: ● Contacts (See *Inserting a Contact, p. 278*) ● Standard Function Blocks (See *Inserting a Timer or Counter Function Block, p. 279* and *Using the Extended Ladder Palette, p. 259*) ● Comparison Blocks (See *Inserting a Comparison Block, p. 284*) ● Operation Blocks (See *Inserting an Operate Block, p. 285*) ● Connectors (See *Inserting and Removing Connectors, p. 281*) ● Jump or Subroutine Call (See *Inserting a Coil or Jump/Subroutine Call, p. 280*) ● Coils (See *Inserting a Coil or Jump/Subroutine Call, p. 280*)
5	Select **Accept Changes** from the **Tools** menu to keep your changes, or select **Cancel Changes** from the **Tools** menu to close the rung without changes. **Note:** You can analyze your program at any time by selecting **Analyze Program** from the **Program** menu, but changes made with the Ladder Editor are not analyzed until they are "Accepted."
6	Add other rungs necessary to complete your program by repeating Step #2 though Step #5.
7	Analyze your program for errors. If there are any errors, open the Program Errors Viewer for a detailed description of errors and program reversibility problems. See *Analyzing a Program, p. 79*.
8	Save your program. Select **Save** from the **Edit** menu. See *Saving an Application, p. 78*.
9	Configure your program. See *Hardware and Software Configuration , p. 127*.
10	Run your program. See *Running Applications, p. 95*.
11	Animate your program. See *Animating a Table, p. 349*.
12	Print your program. See *Printing Applications, p. 116*.

5.2 Using the Ladder Viewer

At a Glance

Overview

This section describes using the Ladder Viewer to view Ladder programs, insert and delete ladder rungs, and open the Ladder Editor.

What's in this Section?

This section contains the following topics:

Using the Ladder Viewer

Introduction

Use the Ladder Viewer to view and edit a Ladder program. See *Using the Ladder Editor, p. 257* to insert and edit elements within an individual rung.
With the Ladder Viewer open, the following options are available:

- Opening the Ladder Editor
- Inserting, editing, and deleting rungs
- Displaying symbols and addresses
- Toggling ladder views
- Using **Edit** menu commands

Opening the Ladder Editor

Use one of the following commands to open the Ladder Editor from the Ladder Viewer to edit the instructions in an individual rung:

- Insert Rung Edit
- Edit Current Rung

Inserting, Editing, and Deleting Rungs

Introduction

With the Ladder Viewer open, the following commands are available from the **Tools** menu of the main menu:

- Insert Rung
- Insert List
- Edit Current Rung
- Delete Current Rung

Insert Rung

To insert a new rung before the currently selected rung:

Step	Action
1	Select the location where the new rung is to be inserted. The current rung, or a selected element within the current rung, is highlighted with a red rectangle.
2	Select **Insert Rung** from the **Tools** menu, or click on Insert on the toolbar.
3	The Ladder Editor opens in Insert Rung mode so that you can create a new rung.

Insert List

To insert a new rung before the currently selected rung and use the List Rung Editor to enter instructions:

Step	Action
1	Select the location where the new rung is inserted. The current rung is highlighted with a red rectangle.
2	Select **Insert List** from the **Tools** menu. The List Rung Editor opens.
3	Enter instructions in the List Rung Editor window.
4	When you are finished entering instructions, select **Validate** to check the instructions for errors and reversibility warnings.
5	Select **OK** to return to the Ladder Viewer. Press **Cancel** to return to the Ladder Viewer without adding the changes.
6	If the rung is not reversible, the new rung is displayed as List instructions.

Edit Current Rung

To open the Ladder Editor from the Ladder Viewer to edit the current rung (see *Using the Ladder Editor, p. 257*):

Step	Action
1	Select a rung to edit. The current rung is highlighted with a red rectangle.
2	Select **Edit Rung** from the **Tools** menu or Edit from the toolbar. The Ladder Editor opens in Edit Rung mode.
3	A rung can also be opened for editing by double-clicking on a unused portion of the programming grid.
4	If the rung is not reversible, the new rung is displayed as List instructions.

Delete Current Rung

To delete the current rung (removes the rung, rung header, and any labels or subroutine declarations):

Step	Action
1	Select a rung to delete by using the shift and arrow keys. The current rung must be highlighted with a blue background.
2	Select **Delete Rung** from the **Tools** men or Delete from the toolbar.
3	The rung is removed and the Ladder Viewer window is updated.

Displaying Symbols and Addresses

Introduction

With the Ladder Viewer open, you can select to view instructions in a Ladder program as symbols, address, or a combination. There are two main choices:

- View instructions as addresses or symbols.
- View the instructions in one-line or three-line format as an address, symbol, or both.

Show Symbols

To display the symbols for instructions:

Step	Action
1	Select **Show Symbols** from the **Tools** menu or from the toolbar.
2	Symbols are displayed above the instructions in either one-line or three-line format (see below).

Show Addresses

To display the addresses for instructions:

Step	Action
1	Select **Show Addresses** from the **Tools** menu or from the toolbar.
2	Addresses are displayed above the instructions in either one-line or three-line format (see below).

1 Line Address or Symbol

To display one line of either symbol or address for instructions:

Step	Action
1	Select **3 Lines Address or Symbol** from the **Tools** menu or from the toolbar.
2	Address or symbols are displayed above the instructions in a one-line format.

3 Lines Address or Symbol

To display three lines of either symbol or address for instructions:

Step	Action
1	Select **3 Lines Address or Symbol** from the **Tools** menu or from the toolbar. Only coils and contacts are displayed in three lines.
2	Address or symbols are displayed above the instructions in a three-line format.

3 Lines Address and Symbol

To display three lines of symbol and address or instructions:

Step	Action
1	Select **3 Lines Address or Symbol** from the **Tools** menu or from the toolbar. Only coils and contacts are displayed in three lines.
2	The address and symbol are displayed above the instructions in a three-line format.

Zoom function

To select a zoom scale:

Step	Action
1	Select **zoom scale** from the **View** menu. **100%** correspond to the previous version. **66%** or **33%** enable to see the most important part of the Ladder rung. **Fit** enables to adjust the display area to the window size.

Toggling Ladder Views

Introduction

With the Ladder Viewer open, you can change the view of Ladder programs by selecting the following:

- Toggle Rung Header
- Toggle Grid
- Toggle Ladder/List
- Show All As Ladder
- Grafcet Step Table

Toggle Rung Header

To switch the display of rung headers on or off:

Step	Action
1	Select **Toggle Rung Header** from the **Tools** menu or Toggle Headers from the toolbar.
2	The Ladder Viewer switches the view of rung headers: if they were previously on, they are now off.
3	Select **Toggle Rung Header** from the **Tools** menu or Toggle Headers from the toolbar again to switch the rung headers to the opposite view.

Toggle Grid

To switch the display of the programming grid on or off:

Step	Action
1	Select **Toggle Grid** from the **Tools** menu or from the toolbar.
2	The Ladder Viewer switches the view of the grid: if the grid was previously on, it is now off.
3	Select **Toggle Grid** from the **Tools** menu or from the toolbar again to switch the grid to the opposite view.

Toggle Ladder/ List

To toggle an entire program between the List and Ladder formats, select Ladder Editor or List Editor from the **Program** menu. To switch between the Ladder and List formats for a selected rung:

Step	Action
1	Select **Toggle Ladder/List** from the **Tools** menu or from the toolbar.
2	The Ladder Viewer switches the view of the selected rung. If the rung was displayed in List format, the view changes to the Ladder format. **Note:** If a rung is toggled from the List to the Ladder format, and it does not meet reversibility rules, then the rung will stay in the List format.
3	Select **Toggle Ladder/List** from the **Tools** menu or Toggle Headers from the toolbar again to switch the view back to the other format.

Grafcet Step Table

Viewing Grafcet steps is simplified by using the Grafcet Step Table which can be displayed by selecting **Tools** → **Grafcet Step Table** from the main menu while viewing either the List Editor or the Ladder Viewer.
The table displays the list of Grafcet steps defined in the program in ascending order. The initial step is represented by a square with two borders while all other steps have a single border. In the online state, the table is animated and the active step or steps are highlighted.
To select a step from the Grafcet Step Table, do one of the following:

- Single click with the mouse on a step and press the ENTER key.
- Double-click on a step.

Once selected from the table, the step will be highlighted in red for both ladder rungs and list instructions.

Using Rung Headers

Introduction

A **Rung Header** appears directly above a rung and documents the purpose of the rung in a ladder diagram.
See *Rung Header, p. 41*.

Displaying Rung Headers

If Rung Headers are not visible when using the Ladder Viewer, select **Tools** → **Toggle Rung Headers** from the main menu. Rung Headers are always visible when using the Ladder Editor.

Opening the Rung Header

To open the Rung Header in the Ladder Viewer or Ladder Editor, do one of the following:

- Double-click on the rung header.
- Use the arrow keys on the keyboard to select a rung header, press ENTER to open the highlighted rung header.

5.3 Using the Ladder Editor

At a Glance

Overview

This section describes how to use the Ladder Editor to create and edit rungs in a Ladder program.

What's in this Section?

This section contains the following topics:

Using the Ladder Editor

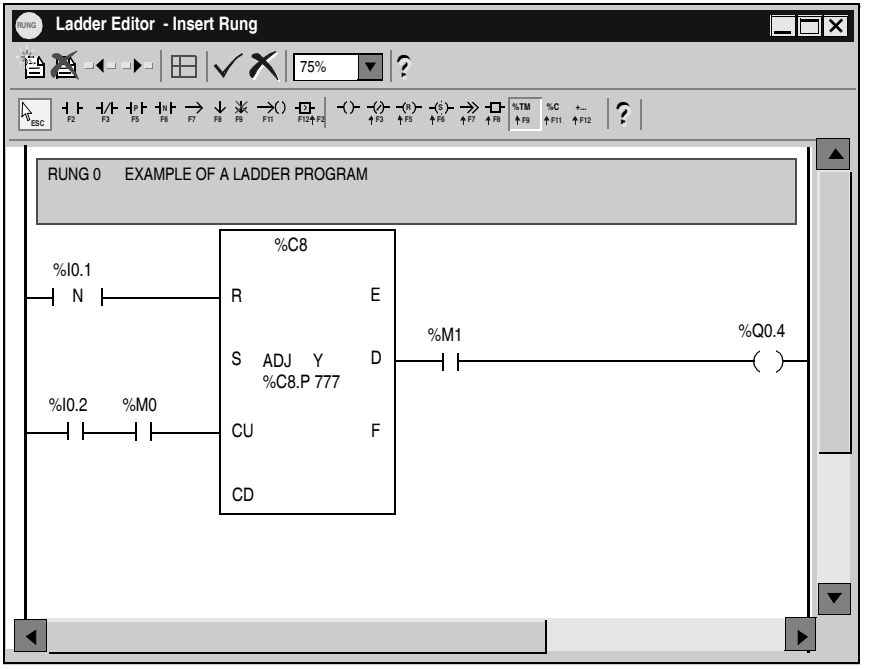
Introduction

To create a ladder program, simply select graphic elements from the Ladder Editor and insert into a ladder rung (graphic elements are the same as graphic instructions). Graphic elements are available from the following parts of the Ladder Editor:

- Ladder Palette Toolbar (See *Ladder Palette Toolbar, p. 258*)
- Extended Ladder Palette (See *Using the Extended Ladder Palette, p. 259*)

Example

The following is an example of editing a rung using the Ladder Editor.



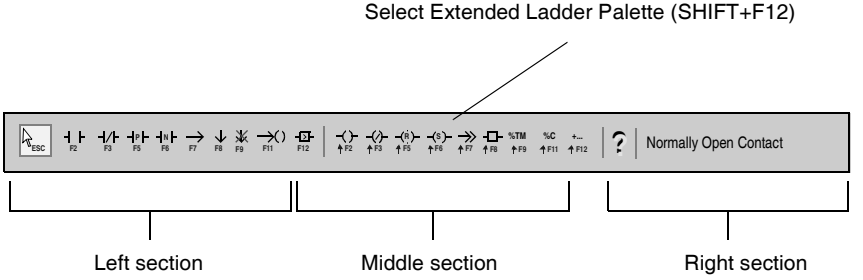
Ladder Palette Toolbar

Introduction

Use the **Ladder Palette Toolbar** to select graphic instructions to insert into a Ladder program. The Ladder Palette is part of the Ladder Editor.
Some special instructions are located on the Extended Ladder Palette which can be opened from the toolbar.

Ladder Palette Toolbar

The following is the Ladder Palette toolbar.



For easy reference, the Ladder Palette toolbar consists of three sections:

1. Left section: contains buttons for test elements such as inputs and the comparison block.
2. Middle section: contains buttons for action elements such as outputs, operate block, jump/subroutine instruction, timer and counter function blocks, and the shortcut button for the Extended Ladder Palette.
3. Right section: displays the name of the currently selected graphic element from the Ladder Palette or the Extended Ladder Palette.

Using the Extended Ladder Palette

Introduction

The **Extended Ladder Palette** contains special instructions that are not available from the Ladder Palette toolbar:

- Special contacts
- Special function blocks
- Special coils

Inserting a Special Instruction

To insert a special instruction from the Extended Ladder Palette, with the Ladder Editor open:

Step	Action
1	Select the Extended Ladder Palette by pointing and left-clicking the mouse on the last button in the Middle section of the Ladder Palette or by pressing the SHIFT + F12 function key.
2	The **Extended Ladder Palette** dialog box opens.
3	Select an instruction from the Extended Ladder Palette by pressing the TAB key repeatedly or clicking with the mouse until the instruction you want is highlighted. Press ENTER to select the highlighted instruction. The Extended Ladder Palette dialog box closes.
4	Left-click or press the space bar to insert the contact. When positioning the contact using the mouse, it is placed in the cell in which you click. When positioning the contact using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
5	To complete programming, see *Inserting an Operand or Symbol, p. 286*.

Note: To position several objects (or blocks) hold down the CTRL key.

Ladder Editor Commands

Introduction

With the Ladder Editor open, the following commands are available from the **Tools** menu:

- Accept Changes
- Cancel Changes
- New Rung
- Clear Rung
- Previous Rung
- Next Rung
- Toggle Grid
- Zoom function
- Cut function
- Copy function
- Paste function
- Cancel function
- Repeat function

Accept Changes

To accept changes made to a rung:

Step	Action
1	Select **Accept Changes** from the **Tools** menu to close an open rung with all changes when you have completed editing or creating a rung.
2	The Ladder Editor closes and the Ladder Viewer opens. (The Ladder Viewer can remain open while using the Ladder Editor, see *Preferences, p. 31*.)

Cancel Changes

To cancel changes made to a rung:

Step	Action
1	Select **Cancel Changes** from the **Tools** menu when you want to discard changes made to a rung.
2	The Ladder Editor closes and the Ladder Viewer opens. (The Ladder Viewer can remain open while using the Ladder Editor, see *Preferences, p. 31*.)

New Rung

To insert a new rung while editing another rung:

Step	Action
1	Select **New Rung** from the **Tools** menu to open a new rung while editing the current rung. **Note:** Any changes to the current rung are saved before opening a new rung.
2	The current rung is closed and a new rung opens in the Ladder Editor.

Clear Rung

To clear all instructions from the current rung:

Step	Action
1	Select **Clear Rung** from the **Tools** menu to remove all elements from the current rung.
2	The rung remains open in the Ladder Editor.

Previous Rung

To open the previous rung in a Ladder program:

Step	Action
1	Click on **Previous Rung** from the **Tools** menu.
2	The preceding rung in the Ladder program opens in the Ladder Editor.
3	Continue clicking on Previous Rung to open all rungs sequentially that precede the current rung. **Note:** If a rung is encountered that is not reversible, it can not be viewed in the Ladder Editor, and it will stop the sequence of opening preceding rungs.

Next Rung

To open the next rung in a Ladder program:

Step	Action
1	Click on **Next Rung** from the **Tools** menu.
2	The next rung in the Ladder program opens in the Ladder Editor.
3	Continue clicking on **Next Rung** to open all rungs sequentially that follow the current rung in the Ladder program. **Note:** If a rung is encountered that is not reversible, it can not be viewed in the Ladder Editor, and it will stop the sequence of opening following rungs.

Toggle Grid

To switch the display of the programming grid on or off:

Step	Action
1	Select **Toggle Grid** from the **Tools** menu or from the toolbar.
2	The Ladder Editor switches the view of the grid: if the grid was previously on, it is now off.
3	Select **Toggle Grid** from the **Tools** menu or from the toolbar again to switch the grid to the opposite view.

Zoom Function

To select a zoom scale:

Step	Action
1	Select **zoom scale** from the **View** menu. **100 %** corresponds to the previous version **66 %** or **33%** enable you to view most of the Ladder rung **Fit** enables to adjust the display area to the window size

Cut Function

Use **Cut** to remove one or more rungs and insert into another location in the same program, or in another program. To remove one or more rungs in offline or online stopped states:

Step	Action
1	See *Marking a Ladder Block, p. 268* to select a block of rungs to cut. Selections are highlighted in blue.
2	Select **Cut** from the **Edit** menu. **Note**: Press the DELETE key to remove the marked block without copying it to the Clipboard.
3	The marked block is removed from the program and the remaining rungs are renumbered.
4	Use the **Paste** command from the **Edit** menu to insert the removed marked block into another location in the same program, or insert into another program.

Copy Function

Use **Copy** to duplicate a marked block from a program and place it in the Clipboard. Copy does remove the marked block. Use **Copy** and **Paste** to duplicate rungs in a program for use in another location in the same program, or for use with another program. To copy a marked block in offline or online stopped states:

Step	Action
1	See *Marking a Ladder Block, p. 268* to select a block of rungs to copy. Selections are highlighted in blue.
2	Select **Copy** from the **Edit** menu.
3	A copy of the marked block is placed in the Windows Clipboard.
4	Use the **Paste** command from the **Edit** menu to insert a copy of the marked block into another location in the same program, or into another program.

Paste Function

Use **Paste** to insert rungs that are in the Clipboard after a Cut or Copy command. Insert rungs in another location in the same program, or in a different program. To Paste a marked block in offline or online stopped states:

Step	Action
1	Use Cut or Copy to place a marked block in the Clipboard.
2	Select the rung where you want to insert the marked block. The rung can be in the current program, or close the current program and open another program.
3	Select **Paste** from the **Edit** menu.
4	A copy of the marked block is inserted **before** the selected rung.

Cancel Function

Use **Undo** to reverse the last Cut, Paste, or Delete operation.
To use Undo:

Step	Action
1	Select **Undo** from the Edit menu.
2	The previous editing operation is reversed.

Repeat Function

Use the **Repeat** command to reverse the Undo command.
To use Repeat:

Step	Action
1	Select **Undo** from the Edit menu.
2	The last Undo operation is reversed.

5.4 Using Edit Functions in the Ladder Viewer

At a Glance

Overview

This section provides procedures for using commands from the **Edit** menu to edit Ladder programs in the Ladder Viewer,

What's in this Section?

This section contains the following topics:

Edit Menu Commands

Introduction

With the Ladder Viewer open, use the following commands from the **Edit** menu to edit a single rung, or a group of rungs:

- *Cut, Copy, and Paste, p. 266*
- *Find and Replace, p. 269*
- *Undo, p. 273*

Cut, Copy, and Paste

Introduction

Select the following commands from the **Edit** menu to edit a Ladder program:

- *Cut, p. 266*
- *Copy, p. 266*
- *Paste, p. 267*
- *Cancel, p. 267*
- *Repeat, p. 267*

See *Marking a Ladder Block, p. 268*.

Cut

Use **Cut** to remove one or more rungs and insert into another location in the same program, or in another program. To remove one or more rungs in offline or online stopped states:

Step	Action
1	See *Marking a Ladder Block, p. 268* to select a block of rungs to cut. Selections are highlighted in blue.
2	Select **Cut** from the **Edit** menu. **Note**: Press the DELETE key to remove the marked block without copying it to the Clipboard.
3	The marked block is removed from the program and the remaining rungs are renumbered.
4	Use the **Paste** command from the **Edit** menu to insert the removed marked block into another location in the same program, or insert into another program.

Copy

Use **Copy** to duplicate a marked block from a program and place it in the Clipboard. Copy does remove the marked block. Use **Copy** and **Paste** to duplicate rungs in a program for use in another location in the same program, or for use with another program. To copy a marked block in offline or online stopped states:

Step	Action
1	See *Marking a Ladder Block, p. 268* to select a block of rungs to copy. Selections are highlighted in blue.
2	Select **Copy** from the **Edit** menu.
3	A copy of the marked block is placed in the Windows Clipboard.
4	Use the **Paste** command from the **Edit** menu to insert a copy of the marked block into another location in the same program, or into another program.

Paste

Use **Paste** to insert rungs that are in the Clipboard after a Cut or Copy command. Insert rungs in another location in the same program, or in a different program. To Paste a marked block in offline or online stopped states:

Step	Action
1	Use Cut or Copy to place a marked block in the Clipboard.
2	Select the rung where you want to insert the marked block. The rung can be in the current program, or close the current program and open another program.
3	Select **Paste** from the **Edit** menu.
4	A copy of the marked block is inserted **before** the selected rung.

Cancel

Use **Undo** to reverse the last Cut, Paste, or Delete operation.
To use Undo:

Step	Action
1	Select **Undo** from the Edit menu.
2	The previous editing operation is reversed.

Repeat

Use the **Repeat** command to reverse the Undo command.
To use Repeat:

Step	Action
1	Select **Undo** from the Edit menu.
2	The last Undo operation is reversed.

Marking a Ladder Block

Introduction

Marking a block of ladder rungs for cut, copy, paste, delete, search, and other commands.

Marking a Block in a Ladder Program

To mark a rung or a number of rungs in a Ladder program:

Step	Action
1	Position the cursor in the beginning of the first rung in the block. A red rectangle highlights the selected cell.
2	Press and hold the SHIFT key. Use the up or down arrow on the keyboard to select rungs. **Note:** The first click with an arrow will select the contents of the current rung. Additional clicks select other rungs. All selected rungs are highlighted in blue.
3	Release the SHIFT key when selections are completed.

Find and Replace

Introduction

You can search for and replace each occurrence of an operand or comment string in Ladder programs. You can also search for rung numbers, but these cannot be replaced.
Find and Replace can be used in the offline state.

Types of Searches

You can search for the following items in a program.

- Operands
  Address or Symbol: Does not matter if addresses or symbols are displayed. For example, if **Show Addresses** is selected from the **Tools** menu, you can still search for symbols.
- Rung
  The rung number of a specified rung when viewing a Ladder program. If the rung number specified is greater than the highest available rung number, then the last rung is highlighted.
- Comment String
  A specific text string in the rung headers of a Ladder program.
  Searching for comment and text strings is not case sensitive.

Search Options

Three options are available to select a search area in a program:

- Search All: Search the entire program.
- Search From Cursor: Start searching at cursor and continue to end of program.
- Search Selected Region: Search only within a marked block in the program.
  See *Marking a Ladder Block, p. 268*.

Search Guidelines

Some general guidelines for searching a program:

- Exact matches only for operands. For example, when searching for %TM0, %TM0.V or AND %TM0.Q will not be found.
- Selecting Comment or Text strings searches for strings in comments, operands, operators, labels, and subroutines.
- Subroutines (SRn) and Labels (%Li) are treated as operands.

General Guidelines for Using Replace

Some general guidelines for using Replace:

- When replacing operands, only function blocks are replaced. For example, when replacing %TM0 with %TM2, %TM0.Q is not replaced with %TM2.Q.
- Source and target operands cannot be unresolved symbols. See *Defining Symbols, p. 81*.
- Replacing a label or subroutine will not change the declaration of the label or subroutine. A label or subroutine declaration may be replaced with another label or subroutine declaration.
- Subroutines (SRn) and Labels (%Li) are treated as operands.
- For Comment or Text strings, any strings that can be located using Find can be replaced.

Replacing Operands

For operands, you can replace:

- Bits with bits (For example, %I0.0 with %M2)
- Words with words (For example, %MW100 with %SW12)
- Function blocks with like function blocks (For example, %TM0 with %TM2 is allowed, %TM0 with %C3 is not allowed)
- Immediate values with immediate values (For example, 7 with 8)

Find

Introduction

Use the **Find** dialog box to search for each occurrence of an operand, rung/line numbers, or comment/text strings in Ladder and List programs. Once the first instance of an item is found, the dialog box changes to display only command buttons to simplify searching and replacing (see Step #7 below). This dialog box is displayed by selecting **Edit** → **Find** from the main menu while viewing a program in the Ladder Viewer or List Editor.
See *Find and Replace, p. 269* for Ladder programs, or *Find and Replace, p. 309* for List programs.

Searching a Program

With a program displayed in the Ladder Viewer or the List Editor:

Step	Action
1	Select **Edit** → **Replace** from the main menu to open the **Replace** dialog box.
2	With the **Find** dialog box open, click on the **Find** drop-down list and select the type of item for the search.
3	In the text box below **Find**, enter a specific item that you want to locate.
4	Select one of the search **Options**. For selecting a region, see *Marking a Ladder Block, p. 268* or *Marking a List Block, p. 308*.)
5	Click on **OK** to close the dialog box and start the search.
6	If the item is not found, an information box is displayed confirming "Item not found." Select **OK** to close the information box and return to the **Find** dialog box.
7	If the item is found, it is highlighted in red. The dialog box changes to display the following buttons: • **Find Next** - select to search for another instance. • **Cancel** - select to close the dialog box.
8	To continue searching, press the **Find Next** button. (Go to Step #7.)
9	To stop searching, press the **Cancel** button to close the dialog box.
10	After the last instance of the value or text is found, or if the search does not find any instance of the specified value or text, an information dialog box is displayed stating "Item not found." Press **OK** to close the box.

Replace

Introduction

Use the **Replace** dialog box to search for and replace an operand or comment/text strings in Ladder and List programs. Once the first instance of an item is found, the dialog box changes to display just command buttons to simplify searching and replacing (see Step #8). This dialog box is displayed by selecting **Edit** → **Find** from the main menu while viewing a program in the Ladder Viewer or List Editor.
See *Find and Replace, p. 269* for Ladder programs, or *Find and Replace, p. 309* for List programs.

Using Replace

To search for and replace an operand or text string:

Step	Action
1	Select **Edit** → **Replace** from the main menu to open the **Replace** dialog box.
2	Click on the **Find** box and select the type of item to be replaced.
3	In the box below the **Find** box, enter the specific item that will be replaced.
4	In the **Replace with** box, enter the specific item that will be the replacement.
5	Select one of the search **Options**.
6	Click on **OK** to close the dialog box and start the search. You can also select **Replace All** to replace all instances of the item. Found instances will not be displayed, but an information box is displayed confirming the number of occurrences (items replaced). Select **OK** to close the information box and the **Replace** dialog box.
7	If an item is not found, an information box is displayed confirming "Item not found." Select **OK** to close the box and return to the **Replace** dialog box.
8	If an item is found, it is highlighted in red. The dialog box changes to display four buttons: • **Find Next** - select to leave highlighted instance unchanged and search for another instance. • **Replace** - select to replace the highlighted instance. • **Replace All** - select to replace all instances of the item. • **Cancel** - select to close the dialog box without any further changes (previous changes are retained).
9	After the last instance of the value or text is found, or if the search does not find any instance of the specified value or text, an information dialog box is displayed stating "Item not found." Press **OK** to close the box.
10	When the **Replace** dialog box is closed, an information box is displayed confirming the number of occurrences (items replaced).

Undo

Introduction

Use **Undo** to reverse the last Cut, Paste, or Delete operation.

Using Undo

To use Undo:

Step	Action
1	Select **Undo** from the Edit menu.
2	The previous editing operation is reversed.

5.5 Inserting Graphic Instructions

At a Glance

Overview

This section provides a procedure and rules for inserting graphic instructions in ladder rungs by using either the mouse or the keyboard.

What's in this Section?

This section contains the following topics:

Inserting Graphical Elements

Introduction

General procedure for inserting graphical elements in a ladder program.
See *Ladder Palette Toolbar, p. 258*.

Inserting Instructions

To insert graphic instructions, starting at the Ladder Viewer:

Step	Action
1	Select **Tools** → **Insert Rung** to open the *Ladder Editor, p. 38*.
2	To view the programming grid, select **Tools** → **Toggle Grid**. See *Preferences, p. 31* to configure viewing ladder diagrams.
3	You can select and insert graphic instructions in a ladder rung using the mouse or the keyboard: • *Using the Mouse to Insert Graphic Instructions, p. 275*. • *Using the Keyboard to Insert Graphic Instructions, p. 276*.

Using the Mouse to Insert Graphic Instructions

To insert a graphic instruction into a ladder rung by using the mouse:

Step	Action
1	Select an instruction from the *Ladder Palette Toolbar, p. 258* by pointing at an instruction button and left-clicking the mouse.
2	For instructions that are not on the Ladder Palette, click on the last button in the Action section of the Ladder Palette to open the Extended Ladder Palette.
3	Select an instruction from the Extended Ladder Palette by pointing at the associated button and left-clicking the mouse. Once you make a selection, the Extended Ladder Palette closes.
4	The Element Name section of the Ladder Palette toolbar displays the name of the selected instruction.
5	Left-click or press the space bar to insert the instruction. When positioning the instruction using the mouse, it is placed in the cell in which you click. When positioning the instruction using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys. The inserted instruction remains active until another instruction is selected.
6	If you insert an instruction in a cell that already has an instruction, the new instruction overwrites the previous instruction.
7	Press delete to remove an instruction from a selected cell.

Using the Keyboard to Insert Graphic Instructions

To insert a graphic instruction into a ladder rung by using the keyboard:

Step	Action
1	Select an instruction from the *Ladder Palette Toolbar, p. 258* by pressing the keyboard shortcut identified on the instruction button: a function key or a combination of the SHIFT key and a function key.
2	For instructions that are not on the Ladder Palette, press SHIFT+F12 (press and hold the SHIFT key, and then press the F12 function key) to open the Extended Ladder Palette,
3	Select an instruction from the Extended Ladder Palette by pressing the TAB key repeatedly until the instruction button you want is highlighted. Press ENTER to select the highlighted instruction. The Extended Ladder Palette closes.
4	The Element Name section of the Ladder Palette toolbar displays the name of the selected instruction.
5	Left-click or press the space bar to insert the instruction. When positioning the instruction using the mouse, it is placed in the cell in which you click. When positioning the instruction using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys. **Note**: If you insert an instruction in a cell that already has an instruction, the new instruction overwrites the previous instruction.
8	Press the DELETE key to remove an instruction from a currently selected cell. to remove an instruction from a selected cell.

Note: To keep the instruction active, press Ctrl + Enter.

Rules for Inserting Graphical Elements

Introduction

Use the following list of rules as guidelines for inserting graphic instructions into a ladder rung.
See *Ladder Palette Toolbar, p. 258* and *Using the Extended Ladder Palette, p. 259*.

Rules

Rules for inserting graphical elements:

1. You can not insert the graphic instructions located on the Test section of the Ladder Palette toolbar into the last column of the grid. The **Compare Block** instruction, the last selection in this section, requires two cells and can not be inserted into the last two columns.
2. The coil, inverse coil, reset coil, set coil, and jump/routine call instructions can only be inserted in the last column of the grid.
   If you try to insert these instructions anywhere else, a horizontal connector line is automatically inserted from that point to the last column where the instruction is inserted.
3. You can only insert an **Operate Block**, which requires four cells, into the last four columns of the grid.
   If you try to insert this instruction anywhere else, a horizontal connector line is automatically inserted from that point to the last four columns where the instruction is inserted.
4. The timer and counter blocks, which require two columns by four rows each, can not be inserted in the first column or the last two columns.
5. The **Special Contacts** located on the Extended Ladder Palette can not be inserted in the first or last columns.
   The OPEN and SHORT special contacts are the exceptions to this rule and can be inserted in the first column.
6. The **Function Blocks** located on the Extended Ladder Palette occupy two columns by four rows, and can not be inserted in the first or last columns.The Very Fast Counter function block occupies two columns by five rows. Only one function block is allowed in a rung.
7. The **Special Coils**, located on the Extended Ladder Palette, can only be inserted in the last column.
   If you try to insert these instructions anywhere else, a horizontal connector line is automatically inserted from that point to the last four columns where the instruction is inserted.

Inserting a Contact

Introduction

Insert contacts in any column of the programming grid except he last column. This procedure is for inserting the following in a ladder rung:

- Normally open contact
- Normally closed contact
- Rising edge contact
- Falling edge contact

Inserting a Contact

With the Ladder Editor open:

Step	Action
1	Select a contact from the Test section of the Ladder Palette by pointing and left-clicking the mouse, or by using the assigned function key.
2	The Element Name section of the *Ladder Palette Toolbar, p. 258* displays the name of the selected contact.
3	Left-click or press the space bar to insert the contact. When positioning the contact using the mouse, it is placed in the cell in which you click. When positioning the contact using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	To complete programming, see *Inserting an Operand or Symbol, p. 286*.

Inserting a Timer or Counter Function Block

Introduction

Insert a timer or counter in any column of the programming grid except for the first column and the last two columns.

Inserting a Timer or Counter Function Block

With the Ladder Editor open:

Step	Action
1	Select a timer or counter function from the Action section of the *Ladder Palette Toolbar, p. 258* by pointing and left-clicking the mouse, or by pressing SHIFT and the assigned function key.
2	The Element Name section of the Ladder Palette toolbar displays the selected type of function block.
3	Left-click or press the space bar to insert the block. When positioning the block using the mouse, it is placed in the cell in which you click. When positioning the block using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	To complete programming timers, see *Configuring Timers, p. 216*. To complete programming counters, see *Configuring Counters, p. 217*.

Inserting a Coil or Jump/Subroutine Call

Introduction

Insert a coil or jump/subroutine call only in the last column of the programming grid. Use this procedure to insert the following into a ladder rung:

- Coil
- Inverse coil
- Reset coil
- Set coil
- Jump/Subroutine call

Inserting a Coil or Jump/ Subroutine Call

With the Ladder Editor open:

Step	Action
1	Select a contact or a jump/subroutine call from the middle section of the *Ladder Palette Toolbar, p. 258* by pointing and left-clicking the mouse, or by pressing SHIFT and the assigned function key.
2	The Element Name section of the Ladder Palette toolbar displays the name of the selected coil or jump/subroutine.
3	Left-click or press the space bar to insert the contact. When positioning the contact using the mouse, it is placed in the cell in which you click. When positioning the contact using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	To complete programming, see *Inserting an Operand or Symbol, p. 286*.

Inserting and Removing Connectors

Introduction

Use horizontal and vertical connector lines to link elements in a ladder rung.

- Horizontal Connector - a horizontal line that connects elements on the same row.
- Down Connector - a vertical line in one column that connects two rows.
- Erase Down Connector - deletes a down connector.
- Horizontal Connector Fill - inserts a unbroken horizontal connector line from the selected cell to an Action block such as an operation or output.

Inserting a Horizontal Connector

With the Ladder Editor open:

Step	Action
1	Select the Horizontal Connector from the left section of the *Ladder Palette Toolbar, p. 258* by pointing and left-clicking the mouse, or by pressing SHIFT and the assigned function key.
2	The Element Name section of the Ladder Palette toolbar displays Horizontal Connector.
3	Left-click or press the space bar to insert the connector. When positioning the connector using the mouse, it is placed in the cell in which you click. When positioning the connector using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	The horizontal connector line appears near the bottom of the cell. To insert a horizontal connector line in another location, point to a cell and left-click, or press the spacebar.

Deleting a Down Connector

With the Ladder Editor open:

Step	Action
1	Select a cell containing a horizontal connector line by left-clicking the mouse or using the arrow keys. A red rectangle highlights the selected cell.
2	Delete the horizontal connector by pressing the DELETE key.

Inserting a Down Connector

With the Ladder Editor open:

Step	Action
1	Select the Down Connector from the left section of the Ladder Palette by pointing and left-clicking the mouse, or by pressing SHIFT and the assigned function key.
2	The Element Name section of the Ladder Palette toolbar displays Down Connector.
3	Left-click or press the space bar to insert the connector. When positioning the connector using the mouse, it is placed in the cell in which you click. When positioning the connector using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	The down connector line appears and extends down to the right side of the cell immediately below and in the same column.

Deleting a Down Connector

With the Ladder Editor open:

Step	Action
1	Select a cell containing a down connector line by left-clicking the mouse or using the arrow keys. You will need to select the cell above the cell containing the down connector. A red rectangle highlights the selected cell which is located immediately above the cell containing the down connector line.
2	Select the Erase Down Connector from the left section of the Ladder Palette by pointing and left-clicking the mouse, or by pressing SHIFT and the assigned function key.
3	The Element Name section of the Ladder Palette toolbar displays Erase Down Connector.
4	Left-click, or press the spacebar to delete the down connector for the cell.

Using Horizontal Fill

With the Ladder Editor open:

Step	Action
1	Select a cell in a row that you want to insert a horizontal line from a the selected cell to the last cell while still leaving the last cell open for an output element. A red rectangle highlights the selected cell.
2	Select the Horizontal Connector Fill from the left section of the Ladder Palette by pointing and left-clicking the mouse, or by pressing SHIFT and the assigned function key.
3	The Element Name section of the Ladder Palette toolbar displays Horizontal Connector Fill.
4	Left-click, or press the spacebar to insert a horizontal connector line starting in the selected cell and ending in a block or output.

Deleting a Horizontal Connector Fill

Use the same procedure as *Deleting a Down Connector, p. 281*.

Inserting a Comparison Block

Introduction

Use a **Comparison Block** to compare two operands. A comparison block cannot be inserted in the last two columns of the programming grid.

Inserting a Comparison Block

With the Ladder Editor open:

Step	Action
1	Select the Comparison Block from the left section of the *Ladder Palette Toolbar, p. 258* by pointing and left-clicking the mouse, or by pressing the F10 function key.
2	The Element Name section of the Ladder Palette toolbar displays Compare.
3	Left-click or press the space bar to insert the block. When positioning the block using the mouse, it is placed in the cell in which you click. When positioning the block using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	To complete programming, see *Inserting an Operand or Symbol, p. 286*.

Inserting an Operate Block

Introduction

Use an **Operate Block** to perform numerical instructions. An operate block can only be inserted in the last four columns of the programming grid in a ladder rung. If you try to insert this instruction anywhere else, a horizontal connector line is automatically inserted from that point to the last four columns where the block is inserted.

Inserting an Operate Block

With the Ladder Editor open:

Step	Action
1	Select the Operate Block from the middle section of the *Ladder Palette Toolbar, p. 258* by pointing and left-clicking the mouse, or by pressing the SHIFT+F8 function key.
2	The Element Name section of the Ladder Palette toolbar displays operate.
3	Left-click or press the space bar to insert the block. When positioning the block using the mouse, it is placed in the cell in which you click. When positioning the block using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	To complete programming, see *Inserting an Operand or Symbol, p. 286*.

Inserting an Operand or Symbol

Introduction

Simple Ladder instructions, such as contacts or coils, use a single operand (for example, %I0.1). Other instructions, such as compare and operate blocks, require operation strings - multiple operands with operators or option calls.
Example of operation strings:

- **%MW50:=%MW3 + %KW5** in an operate block
- **%MW15<0** in a comparison

Enter operation strings directly from the keyboard, just as you would to specify a single operand.

Inserting an Operand or Symbol above a Contact or Coil

With the Ladder Editor open:

Step	Action
1	Select a cell containing a contact or coil by left-clicking the mouse or using the arrow keys. A red rectangle highlights the selected cell.
2	Double-click the mouse or press ENTER to open a small rectangular, text box above the instruction.
3	Enter the operand or symbol name in the text box.
4	Press ENTER to accept the entry and close the box. Press ESC to close the box without changes.

Inserting an Operation String in a Comparison or Operate Block

With the Ladder Editor open:

Step	Action
1	Select the comparison from the left section, or the operate block from the middle section of the *Ladder Palette Toolbar, p. 258* by pointing and left-clicking the mouse.
2	The Element Name section of the Ladder Palette toolbar displays Compare or Operate.
3	Left-click or press the space bar to insert the instruction. When positioning the instruction using the mouse, it is placed in the cell in which you click. When positioning the instruction using the space bar, it is placed in the cell which was previously selected. To select an cell you must call up the selection tools (arrow icon) then left-click or move to the desired cell using the arrow keys.
4	Double-click the mouse or press ENTER to open a text box above the instruction.
5	Enter an operation string into the text box. Press ENTER when complete. The operand appears inside the box.
6	Press ENTER to accept the entry and close the box. Press ESC to close the box without changes.

Programming in List Language

6

At a Glance

Overview

This chapter provides detailed instructions for using TwidoSoft to program in Instruction List language.

What's in this Chapter?

This chapter contains the following sections:

6.1 Creating List Programs

At a Glance

Overview

This section provides an overview of using TwidoSoft to create List programs.

What's in this Section?

This section contains the following topics:

Overview of List Programs

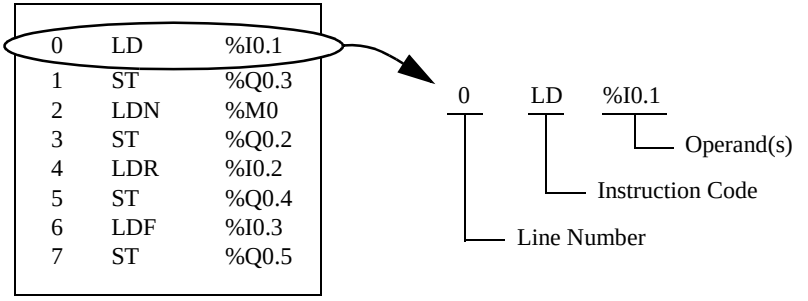
Introduction

A program written in List language consists of a series of instructions executed sequentially by the controller. Each List instruction is represented by a single program line and consists of three components:

- Line number
- Instruction code
- Operand(s)

Example of a List Program

The following is an example of a List program.



Line Number

Line numbers are generated automatically when you enter an instruction. Blank lines and Comment lines do not have line numbers.

Instruction Code

The instruction code is a symbol for an operator that identifies the operation to be performed using the operand(s). Typical operators specify Boolean and numerical operations.

For example, in the sample program above, LD is the abbreviation for the instruction code for a LOAD instruction. The LOAD instruction places (loads) the value of the operand %I0.1 into an internal register called the accumulator.

There are basically two types of instructions:

- Test instructions
  These setup or test for the necessary conditions to perform an action. For example, LOAD (LD) and AND.
- Action instructions
  These perform actions as a result of setup conditions. For example, assignment instructions such as STORE (ST) and RESET (R).

Operand

An operand is a number, address, or symbol representing a value that a program can manipulate in an instruction. For example, in the sample program above, the operand %I0.1 is an address assigned the value of an input to the controller. An instruction can have from zero to three operands depending on the type of instruction code.

Operands can represent the following:

- Controller inputs and outputs such as sensors, push buttons, and relays.
- Predefined system functions such as timers and counters.
- Arithmetic, logical, comparison, and numerical operations.
- Controller internal variables such as bits and words.

Creating a List Program

Introduction

A List program can be created at any time while developing an application. Once the program is completed, you can configure hardware, define symbols, and print documentation.
See *Developing an Application, p. 74*.

Basic Steps

The following are the basics steps for creating a List program from an open application:

Step	Action
1	Open the *List Editor, p. 43*. Select **Program** → **List Editor** from the main menu.
2	Configure the List Editor. See *Preferences, p. 31* to configure the view of List programs.
3	Enter instruction codes and operands by using one of the following methods (see *Entering List Instructions, p. 299*): ● Select from the List Instruction Bar ● Enter directly from the keyboard
4	Enter comments, See *Guidelines for Inserting List Instructions, p. 303*.
5	Analyze your program for errors. If there are any errors, open the Program Errors Viewer for a detailed description of errors and program reversibility problems. See *Analyzing a Program, p. 79*.
6	Save your program. See *Saving an Application, p. 78*.
7	Configure your program. See *Hardware and Software Configuration , p. 127*.
8	Run your program. See *Running Applications, p. 95*.
9	Animate your program. See *Animating a Program, p. 336*.
10	Print your program. See *Printing Applications, p. 116*.

6.2 Using the List Editor

At a Glance

Overview

This section describes using the List Editor to view and create List programs.

What's in this Section?

This section contains the following topics:

Using the List Editor

Introduction

Use the *List Editor, p. 43* to view and edit a List program. With the List Editor open, the following options are available:

- Inserting, modifying, and deleting instruction lines
- Entering comments
- Displaying symbols and addresses
- Using **Edit** menu commands

Opening the List Editor

To open the List Editor, select **Program** → **List Editor** from the main menu.

When You Can Use the List Editor

You can use the List Editor in the Online or Offline state, but not in the Monitor state. Typically, use the List Editor in the Offline state to develop a List program, and in the Online state to debug a program.
Some instructions require balancing or complementary instructions. For example, a BLK instruction requires an END_BLK instruction. An instruction modified by a parenthesis requires a closing parenthesis - all in the same scan.
To maintain an effective scan rate, the controller accepts a single instruction line at a time. Consequently, there is no way to write these complex instructions without severely diminishing the performance of the controller while in the Online state.

List Editor Commands

Introduction

With the List Editor open, the following commands are available from the **Tools** menu:

- Show Symbols
- Show Addresses
- Grafcet Step Table

Viewing Operands

You can view operands in a List program as symbols or as addresses. Additionally, you can quickly locate Grafcet instructions in a program by using the Grafcet Step Table.

Show Symbols

To display the symbols for operands:

Step	Action
1	Select **Show Symbols** from the **Tools** menu or from the toolbar.
2	Operands are displayed as symbols in List instructions if symbols were defined for the operands using the Symbol Editor.

Show Addresses

To display the addresses for operands:

Step	Action
1	Select **Show Addresses** from the **Tools** menu or from the toolbar.
2	Operands are displayed as addresses in List instructions.

Grafcet Step Table

Viewing Grafcet steps is simplified by using the Grafcet Step Table which can be displayed by selecting **Tools** → **Grafcet Step Table** from the main menu while viewing either the List Editor or the Ladder Viewer.
The table displays the list of Grafcet steps defined in the program in ascending order. The initial step is represented by a square with two borders while all other steps have a single border. In the online state, the table is animated and the active step or steps are highlighted.
To select a step from the Grafcet Step Table, do one of the following:

- Single click with the mouse on a step and press the ENTER key.
- Double-click on a step.

Once selected from the table, the step will be highlighted in red for both Ladder rungs and List instructions.

6.3 Inserting List Instructions

At a Glance

Overview

This section provides procedures and guidelines for inserting List instructions in a program by using either the mouse or the keyboard.

What's in this Section?

This section contains the following topics:

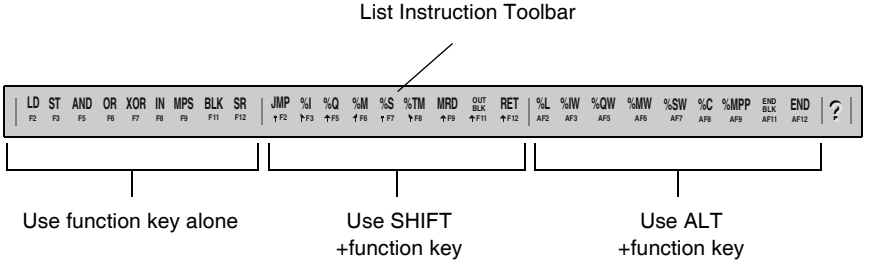
Entering List Instructions

Introduction

With the List Editor open, type List instructions directly from the keyboard or select operators and operands from the **List Instruction Toolbar** by using the keyboard or the mouse. The List Instruction Toolbar displays the most commonly used operators and operands, and this convenience enhances the speed and accuracy of creating a List program. Operators and operands that are not available from the toolbar have to be entered using the keyboard.

List Instruction Bar

The following is the List Instruction Toolbar, which is located on the List Editor.



Inserting Instructions

To insert List instructions in an existing or new List program, with an open application:

Step	Action
1	Select **Program** → **List Editor** from the main menu to open the List Editor. See *Preferences, p. 31* to configure the view of List programs.
2	The cursor is positioned at the beginning of the first program line in the editing area of the *List Editor, p. 43*.
3	You can enter List Instructions by using the mouse or the keyboard: • *Using the Mouse to Insert List Instructions, p. 300*. • *Using the Keyboard to Insert List Instructions, p. 301*.

Using the Mouse to Insert List Instructions

To insert a List instruction by using the mouse:

Step	Action
1	Left-click in the program where you want to enter an instruction. The text cursor is placed on the selected line. If you click on the last line, the text cursor is placed on the last line which is then highlighted in red.
2	Press ENTER anywhere in the program line to create a new program line that appears immediately after the current line, or press the INSERT key to insert a new blank line immediately before the current line.
3	Select an instruction from the List Instruction Toolbar by left-clicking on the toolbar button. Use the keyboard to enter instructions not available from the toolbar.
4	The selected instruction is inserted at the cursor location. If the instruction was selected from the toolbar, a space is automatically included after the instruction to allow for separation from the operand, otherwise, enter a space using the keyboard.
5	Select an operand from the List Instruction Toolbar by left-clicking on the toolbar button. Use the keyboard to enter operands not available from the toolbar.
6	The selected operand is inserted at the cursor location.
7	Enter a numerical value for the specific Instance of the operand by using the keyboard. **For example:** Click on **%TM** on the List Instruction Toolbar to insert a Timer function block into a program line. The text cursor is positioned immediately after the %TM. Enter a valid number for the instance using the keyboard. (Valid numbers for instances depend on the controller model (see *Maximum Software Configuration, p. 215*).
8	Enter optional Comments to document the program. See **Line Comments**.
9	To complete insertion of the List instruction: press ENTER to complete the line and begin a new, blank line; or, use the up and down arrows on the keyboard to move the text cursor off the current line; or, click on another line. **Note:** The program line is validated for errors, and the List Editor displays errors according to the **Auto Line Validate** setting in Preferences. (See *Program Errors When Inserting List Instructions, p. 302*.)
10	If there are no program errors for the program line, continue entering instructions or validate the entire program (see *Analyzing a Program, p. 79*).

Using the Keyboard to Insert List Instructions

To insert a List instruction by using the keyboard:

Step	Action
1	Use the Up and Down arrows to position the cursor where you want to enter an instruction.
2	Press ENTER anywhere in the program line to create a new program line that appears immediately after the current line, or press the INSERT key to insert a new blank line immediately before the current line.
3	Select an instruction from the List Instruction Toolbar by pressing the keyboard shortcut identified on the instruction button, or type in the instruction directly from the keyboard. Some operands are not available on the toolbar and must be entered using the keyboard.
4	The selected instruction is inserted at the cursor location. If the instruction was selected from the toolbar, a space is automatically included after the instruction to allow for separation from the operand, otherwise, enter a space using the keyboard.
5	Select an operand from the List Instruction Toolbar by pressing the keyboard shortcut identified on the instruction button, or type in the instruction directly from the keyboard. Some operands are not available on the toolbar and must be entered using the keyboard.
6	The selected operand is inserted at the cursor location.
7	Enter a numerical value for the specific instance of the operand by using the keyboard. **For example:** Click on **%TM** on the List Instruction Toolbar to insert a Timer function block into a program line. The text cursor is positioned immediately after the %TM. Enter a valid number for the instance using the keyboard. (Valid numbers for instances depend on the controller model (see *Maximum Software Configuration, p. 215*).
8	Enter optional Comments to document the program. See **Line Comments**.
9	To complete insertion of the List instruction: press ENTER to complete the line and begin a new, blank line; or, use the up and down arrows on the keyboard to move the text cursor off the current line. **Note:** The program line is validated for errors, and the List Editor displays errors according to the **Auto Line Validate** setting in Preferences. (See Program Errors While Entering Instructions.)
10	If there are no program errors for the program line, continue entering instructions or validate the entire program (see *Analyzing a Program, p. 79*).

Program Errors When Inserting List Instructions

When you have completed entering a List Instruction in the List Editor, the instruction is validated for program errors. The List Editor handles errors according to the state of the **Auto Line Validate** option (see *Preferences, p. 31*).

- If Auto Line Validate is selected
  The List Editor displays the errors in the message section of the Status bar, and you **can not** move the cursor to another line. You have to correct errors to the current line before the List Editor permits movement to another instruction line.
- If Auto Line Validate is not selected
  The List Editor displays the errors in the message section of the Status bar, and you **can** move the cursor to another line. A question mark (?) is displayed at the beginning of the line to mark the line as containing an error, and you may continue entering instructions. You can return to this line later to correct the error.

Line Comments

Enter optional comments to describe the purpose of a program, individual instructions, operands, and so on. Comments consist of text that must be entered within parenthesis and enclosed by asterisks as in the following example:
(*THIS IS A COMMENT.*)
Valid text entries are standard keyboard alphanumeric characters: A - Z, 0 - 9, "‘~!@#$%^&*()-_+=", and spaces. Letters that are entered in lower case are automatically converted to upper case when the line is validated.
There are two methods for inserting comments in a program:

- At the end of a program line
  1 LD START_WATER (*STARTS THE PUMP*)
- On a separate Comment Line
  1 LD START_WATER
  (*STARTS THE PUMP*)

Comment Lines do not have line numbers.

Guidelines for Inserting List Instructions

Introduction

Use the following list of guidelines for inserting List instructions.
See *Entering List Instructions, p. 299*.

Guidelines

Guidelines for inserting List instructions:

1. There are two cursors that can be used for inserting and modifying List instructions . The mouse cursor and the text cursor. The text cursor is a short, flashing vertical line that can be moved either by using the mouse or the up/down arrows on the keyboard.
2. When inserting an instruction, a space is required between an operator and an operand, and between an operand and any comments as in the following example:
   LD START_SWITCH (*START THE CONVEYOR*)
3. When entering operands using the toolbar, only the type of operand is inserted in the program, you will have to add the specific Instance for the operand.
   For example, if you inserted a timer function block %TM as an operand in an instruction, only %TM appears in the instruction. You must use the keyboard to enter a number for the specific instance of the timer to complete address for the operand (for example, %TM1).
4. You can enter a symbol such as START_SWITCH for an operand instead of the associated address such as %I0.7.
   See *Defining Symbols, p. 81*.

6.4 Using Edit Functions in the List Editor

At a Glance

Overview

This section provides procedures for using commands from the **Edit** menu to edit List programs in the List Editor.

What's in this Section?

This section contains the following topics:

Edit Menu Commands

Introduction

With the List Editor open, use the following commands from the **Edit** menu to edit a single instruction, or a group of instructions:

- *Cut, Copy, and Paste, p. 306*
- *Find and Replace, p. 309*
- *Undo, p. 313*

Cut, Copy, and Paste

Introduction

With the List Editor open, select the following commands from the **Edit** menu to edit a List program:

- *Cut, p. 306*
- *Copy, p. 306*
- *Paste, p. 307*

See *Marking a List Block, p. 308*.

Cut

Use **Cut** to remove one or more program lines and insert into another location in the same program, or in another program. To remove one or program lines in offline or online stopped states:

Step	Action
1	See *Marking a List Block, p. 308* to select a block of program lines to cut. Selections are highlighted in red.
2	Select **Cut** from the **Edit** menu. **Note**: Press the DELETE key to remove the marked block without copying it to the Clipboard.
3	The marked block is removed from the program and the remaining lines are renumbered.
4	Use the **Paste** command from the **Edit** menu to insert the removed marked block into another location in the same program, or insert into another program.

Copy

Use **Copy** to duplicate a marked block from a program and place it in the Clipboard. Use **Copy** and **Paste** to duplicate lines in a program for use in another location in the same program. To copy a marked block in offline or online stopped states:

Step	Action
1	See *Marking a List Block, p. 308* to select a block of program lines to copy. Selections are highlighted in red.
2	Select **Copy** from the **Edit** menu.
3	A copy of the marked block is placed in the Windows Clipboard.
4	Use the **Paste** command from the **Edit** menu to insert a copy of the marked block into another location in the same program, or into another program.

Paste

Use **Paste** to insert program lines that have been copied into the Clipboard after a Cut or Copy command. Insert lines in another location in the same program. To Paste a marked block in offline or online stopped states:

Step	Action
1	Use Cut or Copy to place a marked block in the Clipboard.
2	Select the line where you want to insert the marked block. The line can be in the current program, or close the current program and open another program.
3	Select **Paste** from the **Edit** menu.
4	A copy of the marked block is inserted **before** the selected line.

Marking a List Block

Introduction

Marking a block or region of a List program for cut, copy, paste, delete, search, and other commands.

Marking a List Block

To mark a block of List instructions:

Step	Action
1	Select the first instruction in the block by clicking the mouse anywhere within the instruction line, or by using the arrow keys on the keyboard to move the text cursor to the instruction line.
2	Select additional lines by clicking and dragging the mouse cursor up or down, or by pressing the SHIFT and the up/down arrows on the keyboard.
3	The marked block is the highlighted text in red.

Find and Replace

Introduction

You can search for and replace each occurrence of an operand or text string in List programs. You can also search for rung or line numbers, but these cannot be replaced.

Types of Searches

You can search for the following items in a program.

- Operands
  Address or Symbol: Does not matter if addresses or symbols are displayed. For example, if **Show Addresses** is selected from the **Tools** menu, you can still search for symbols.
- Line
  The line number of an instruction when viewing a List program. If the line number specified is greater than the highest available line number, then the cursor is positioned after the last instruction and at the beginning of a blank instruction line.
- Text String
  A specific text string in the instructions of a List program
  Searching for comment and text strings is not case sensitive.

Search Options

Three options are available to select a search area in a program:

- Search All: Search the entire program.
- Search From Cursor: Start searching at cursor and continue to end of program.
- Search Selected Region: Search only within a marked block in the program.
  See *Marking a List Block, p. 308*.

Search Guidelines

Some general guidelines for searching a program:

- Exact matches only for operands. For example, when searching for %TM0, %TM0.V or AND %TM0.Q will not be found.
- Selecting Text Strings searches for strings in comments, operands, operators, labels, and subroutines.
- Subroutines (SRn) and Labels (%Li) are treated as operands.

General Guidelines for Using Replace

Some general guidelines for using Replace:

- When replacing operands, only function blocks are replaced. For example, when replacing %TM0 with %TM2, %TM0.Q is not replaced with %TM2.Q.
- Source and target operands cannot be unresolved symbols. See *Defining Symbols, p. 81*.
- Replacing a label or subroutine will not change the declaration of the label or subroutine. A label or subroutine declaration may be replaced with another label or subroutine declaration.
- Subroutines (SRn) and Labels (%Li) are treated as operands.
- For Comment or Text strings, any strings that can be located using Find can be replaced.

Replacing Operands

For operands, you can replace:

- Bits with bits (For example, replace %I0.0 with %M2)
- Words with words (For example, replace %MW100 with %SW12)
- Function blocks with like function blocks (For example, replace %TM0 with %TM2 is allowed, %TM0 with %C3 is not allowed)
- Immediate values with immediate values (For example, replace 7 with 8)

Find

Introduction

Use the **Find** dialog box to search for each occurrence of an operand, rung/line numbers, or comment/text strings in Ladder and List programs. Once the first instance of an item is found, the dialog box changes to display only command buttons to simplify searching and replacing (see Step #7 below). This dialog box is displayed by selecting **Edit** → **Find** from the main menu while viewing a program in the Ladder Viewer or List Editor.
See *Find and Replace, p. 269* for Ladder programs, or *Find and Replace, p. 309* for List programs.

Searching a Program

With a program displayed in the Ladder Viewer or the List Editor:

Step	Action
1	Select **Edit** → **Replace** from the main menu to open the **Replace** dialog box.
2	With the **Find** dialog box open, click on the **Find** drop-down list and select the type of item for the search.
3	In the text box below **Find**, enter a specific item that you want to locate.
4	Select one of the search **Options**. For selecting a region, see *Marking a Ladder Block, p. 268* or *Marking a List Block, p. 308*.)
5	Click on **OK** to close the dialog box and start the search.
6	If the item is not found, an information box is displayed confirming "Item not found." Select **OK** to close the information box and return to the **Find** dialog box.
7	If the item is found, it is highlighted in red. The dialog box changes to display the following buttons: • **Find Next** - select to search for another instance. • **Cancel** - select to close the dialog box.
8	To continue searching, press the **Find Next** button. (Go to Step #7.)
9	To stop searching, press the **Cancel** button to close the dialog box.
10	After the last instance of the value or text is found, or if the search does not find any instance of the specified value or text, an information dialog box is displayed stating "Item not found." Press **OK** to close the box.

Replace

Introduction

Use the **Replace** dialog box to search for and replace an operand or comment/text strings in Ladder and List programs. Once the first instance of an item is found, the dialog box changes to display just command buttons to simplify searching and replacing (see Step #8). This dialog box is displayed by selecting **Edit** → **Replace** from the main menu while viewing a program in the Ladder Viewer or List Editor. See *Find and Replace, p. 269* for Ladder programs, or *Find and Replace, p. 309* for List programs.

Using Replace

To search for and replace an operand or text string:

Step	Action
1	Select **Edit** → **Replace** from the main menu to open the **Replace** dialog box.
2	Click on the **Find** box and select the type of item to be replaced.
3	In the box below the **Find** box, enter the specific item that will be replaced.
4	In the **Replace with** box, enter the specific item that will be the replacement.
5	Select one of the search **Options**.
6	Click on **OK** to close the dialog box and start the search. You can also select **Replace All** to replace all instances of the item. Found instances will not be displayed, but an information box is displayed confirming the number of occurrences (items replaced). Select **OK** to close the information box and the **Replace** dialog box.
7	If an item is not found, an information box is displayed confirming "Item not found." Select **OK** to close the box and return to the **Replace** dialog box.
8	If an item is found, it is highlighted in red. The dialog box changes to display four buttons: • **Find Next** - select to leave highlighted instance unchanged and search for another instance. • **Replace** - select to replace the highlighted instance. • **Replace All** - select to replace all instances of the item. • **Cancel** - select to close the dialog box without any further changes (previous changes are retained).
9	After the last instance of the value or text is found, or if the search does not find any instance of the specified value or text, an information dialog box is displayed stating "Item not found." Press **OK** to close the box.
10	When the **Replace** dialog box is closed, an information box is displayed confirming the number of occurrences (items replaced).

Undo

Introduction

Use **Undo** to reverse the last Cut, Paste, or Delete operation.

Using Undo

To use Undo:

Step	Action
1	Select **Undo** from the Edit menu.
2	The previous editing operation is reversed.

Programming with Macro Definition Functions (MDF)

7

At a Glance

Overview

This chapter explains how to configure and use a Macro Definition Function (MDF).

What's in this Chapter?

This chapter contains the following topics:

Defining a Macro (MDF)

Definition

A Macro Definition Function (MDF) is a complex instruction that represents a group of real instructions in LIST language and is intended to be replaced by this group every time it appears in a program.

Macro Groups

Macro Definition Functions (or Macros) are grouped in three categories. Each macro group is described in an MDF file:

Macro groups	Macro files
COMM Macros	COMM.MDF
DRIVE Macros	DRIVE.MDF
TESYS Macros	TESYS.MDF

COMM is a group of macros that allows you to send 'standard' read/write queries of a bit or a word to a modbus slave (for example to Twido Modbus slaves).
DRIVE is a group of macros that allows you to implement drives (for example ALTIVAR) through CANopen as well as Modbus.
TESYS is a group of macros that allows you to implement motor starters through CANopen as well as Modbus.

Note: Those three Macro Definition Function groups are user-customizable.

Complementary information

Macros are a help to programming. They aim at structuring the language, making programs easy to read.
For example, if you wish to read a word from a slave over the Modbus network:

- Without macros, you need to program an exchange block but, once it has been written, a simple reading of the program is not enough to understand that this block allows you to read on a slave.
- Using macros, you directly find the corresponding macro, COMM_RD1W, in your program.

There is another advantage. If, for example, you want to change the speed of a drive:

- Using macros, you write an instruction whose name will be directly linked to the function, and you get the code to program this function.

Configuration of a Macro

There are three ways to configure a Macro:
- Using the Application Browser
- Using the main menu
- Using the Configuration editor

In any case, the same configuration dialog box appears.

Note: TwidoSoft Macro menus and buttons are enabled provided that an MDF file exists in TwidoSoft directory and no error appears during the automatic initial import (checked once at TwidoSoft start).

Via the Application Browser

In the Application Browser, Macros are located below Program, in a generic Macros item that divides into:
- Comm
- Drive
- TeSys

Step	Action
1	Select a Macro software resource and right-click. **Result:** A pop-up menu opens, with **Edit** and **Help** items.
2	Click **Edit**. **Result:** A configuration dialog box opens.

The same result can be obtained in just one step if you double-click the selected Macro software resource.

Note: A Help Macro menu can be attached to each Macro group. If you click **Help** instead of **Edit**, you get information about the selected Macro type.

Via the Main Menu

In the **Program** main menu, a **Macros** item opens a sub-menu with:

- Comm
- Drive
- TeSys

Step	Action
1	Select **Program** from the main menu.
2	Click **Macros** resource item.
3	Select a macro group: **Comm**, **Drive** or **TeSys**. **Result:** A configuration dialog box opens.

Note: A Help Macro menu can be attached to each Macro group. If you click **Help** from the main menu, then **Help with Macros**, you can access the 3 different Macro Help files and get information about the one you select.

Via the Configuration Editor

Each macro software resource can be configured directly from the Configuration editor.
In the Configuration editor's tool bar, three icon buttons allow you to select a Macro group:

Macro group	Button
Comm macros	
Drive macros	
TeSys macros	

Step	Action
1	Click a button to select a macro software resource.
2	Click the **Configure** button  or select **Tools** → **Edit** from the main menu. **Result:** A configuration dialog box opens.

Note: When canceling a change made in the Configuration editor, all configuration changes made since the Configuration editor was opened are discarded.

Use of the Configuration Dialog Box

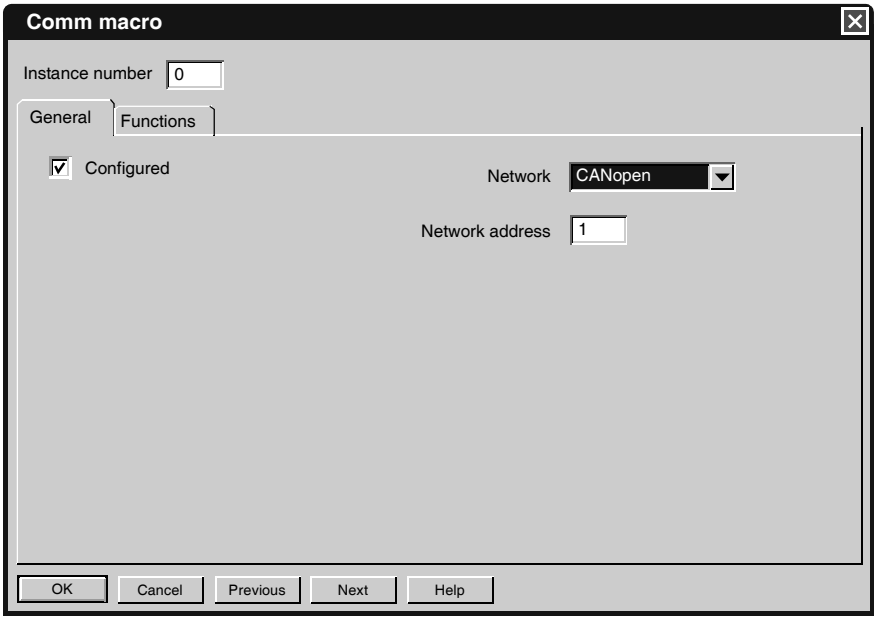
The configuration dialog box shows two tabs:
- **General** (default tab)
- **Functions**

Note: In Online mode, Macro dialog boxes are available but all controls - except the number of macros - are greyed.

In the upper part of the window, the **Macro Number** box corresponds to the macro number you wish to configure here. The value is 0 to 31 maximum per application and per group of macros.

General Tab

The following figure describes the Comm macro's General tab:



Information in the **General** tab depend on the network status:

If ...	**Then ...**
The network has not been configured yet,	All fields are disabled and a "No available and configured network" string appears in the main box, below **Used** checkbox.
The network has been configured,	**Used** chekbox is replaced by **Configured** checkbox - that is default unchecked - and **Network** and **Address on the Network** fields are enabled.

The **Network** field is a combo box in which you may choose between three items, as a maximum:

Modbus - Port 1	If the standard serial port has been configured for Modbus protocol.
Modbus - Port 2	If the optional serial port has been configured for Modbus protocol.
CANopen	If a CANopen expansion module exists in the configuration.

The **Address on the Network** field allows you to specify the network address used by the Macro.

- If a **Modbus** network, the value is between 1 and 247.
  The profile used is "MODBUS".
- If a **CANopen** network, the value is between 1 and 128.
  The network address is associated to one profile. The address gets the profile, then the profile gets the function code.

Cases of a valid or invalid profile:

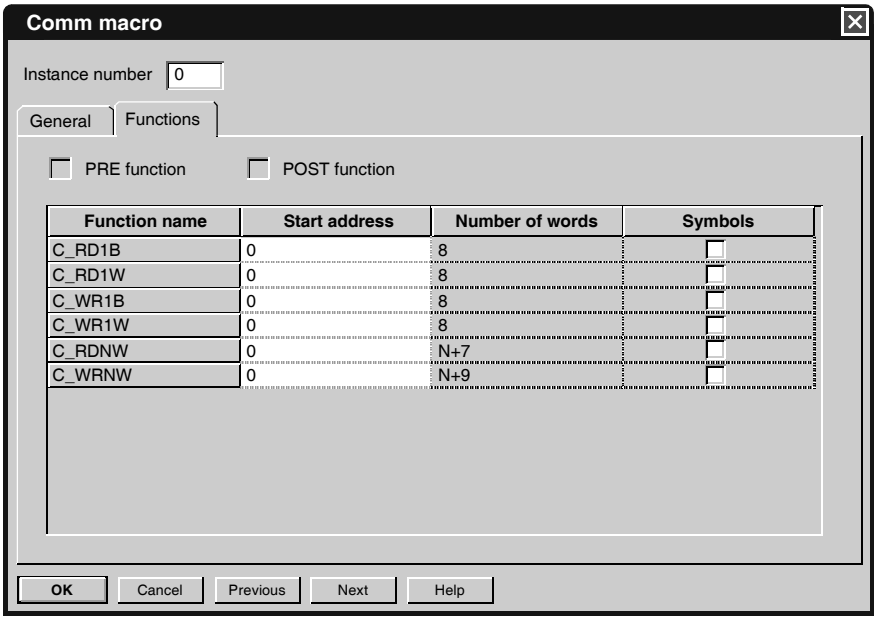
If the profile is valid	Its type is valid (COMM, DRIVE or TESYS). The MDF file has at least one function using this profile. **Result:** You can click **OK**.
If the profile is invalid	Its type is invalid or no MDF function uses this profile. **Result:** An "Invalid profile" message appears when clicking **OK**, clicking **Previous** or **Next**, changing tabs or changing the instance macro number.

To locate a specific macro, click **Previous** and/or **Next** buttons.

Note: When leaving **General** tab (changing tabs, clicking **OK** or **Previous** / **Next** buttons), TwidoSoft analyzes all configured macros to detect the presence of an already configured macro with the same kind of network and the same address. If any, it displays an error message: "Warning. Macro %s and Macro %s are configured for the same network address".

Functions Tab

In the configuration dialog box, click **Functions** tab.
The following figure describes the Comm macro's Functions tab:



The upper part of the window shows two checkboxes:

PRE function	Indicates if it must be user-added at the beginning of a LIST program or not. It is checked provided that **Configured** has been checked in **General** tab and "TYPE_PRE" function exists in the MDF file and has a description for this selected profile.
POST function	Indicates if it must be user-added at the end of a LIST program or not. It is checked provided that **Configured** has been checked in **General** tab and "TYPE_POST" function exists in the MDF file and has a description for this selected profile.

Note: Those two boxes are read-only.

The main part of the window is a table showing a list of functions with:

Function name	Functions using the current profile are listed with their **Function names**.
Start address	In the **Start address** field, you specify the MW start address used here (default address is 0). **Note:** You should not re-use already used addresses, or at your own risk.
Number of words	In the **Number of words** field, the number of MWs used by this function is provided.

Symbols	In the **Symbols** field, you can click the symbol associated to a function. The symbol will be generated by the system. **Note:** Only click the box(es) corresponding to the function(s) that you use. If you do not check any box, no symbol will be generated.

To locate a specific macro, click **Previous** and/or **Next** buttons.

Use of a Macro in a Program

A Macro can be added to a LIST or a LADDER program.

Syntax of a Macro Function

The syntax of a Macro is:
Macro_name parameter0 parameter1 parameter2 (* **comment** *)

- **parameter0** is mandatory. It corresponds to the slave instance number.
- **parameter1** and **parameter2** are optional. They depend on the function.
- **comment** is optional.

Associated Symbols

Note: If you access other symbols by an array (for example C_RDNW_VAL1[%MW0], where %MW0 is a variable index), be carefull not to use a word already used by macros.

Example: COMM_RDNW_NOACK Macro

The Macro Definition Function name is **C_RDNW**
This function allows the user to read N words over the modbus network.
The two following symbols have to be filled in before you can use the macro:

Symbol	Fill in
C_RDNW_ADDR1	This symbol has to be filled in with the first word address to be read.
C_RDNW_VAL1	This symbol is filled in by the macro with the first word value read after achievement.

C_RDNW parameter0 parameter1

- **parameter0** indicates the instance of slave on which to apply the macro.
- **parameter1** indicates the number of words to read.

```
(* This example reads 10 words from %MW5 on slave with instance 1 *)

LD 1
[ C_RDNW_ADDR1_1 := 5 ] (* Read from %MW5 on instance 1 *)

[ C_RDNW 1 10 ](* Send frame to slave with instance number 1 and parameter 10 *)
```

Inserting a Macro Instruction in a Program

To insert a Macro instruction in a LIST or a LADDER program, you must use the keyboard.
There are no buttons in the LIST instruction toolbar and no hot-keys to insert macro instructions. When you have completed entering a Macro instruction in the LIST Editor, the instruction is validated for program errors.
In a LIST as well as in a LADDER program, you must use an Operate Block to insert a Macro instruction:

Text is automatically placed between '[ ]'	in a LIST program
Text is written in a block	in a LADDER program.

A Macro instruction is valid if :
- the Macro name exists in one of the three MDF files,
- the parameter numbers are good,
- parameter 0 is in [0-31] range,
- parameters 1 and 2 are in [MDF_PARAMETER_MIN-MDF_PARAMETER_MAX] range.
- the compatibility level is not forced to less than MDF_COMPATIBILITY value.

Real replacement code is not checked.

Note: Online modification, addition or deletion is forbidden.

Automatic Symbol Addressing

Symbol addressing is automatic.
There are two different addresses:
- a constant address (starts with '%')
- a user-controlled address (starts with '@')

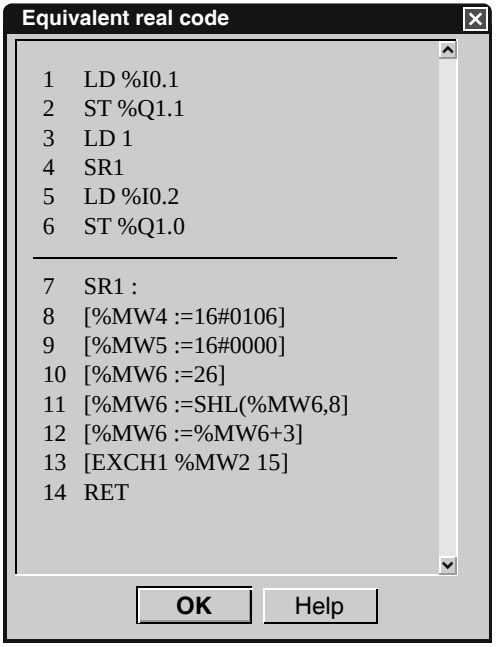
The first time a Macro is used (open the configuration dialog box, check **Configured**, then click the **OK** button to validate), TwidoSoft adds all symbols used by the slave.
When the MW start address of a function has changed (open the configuration dialog box, change the **Start address**, then click the **OK** button to validate), TwidoSoft searches for all symbols using this address, via the symbol names, then updates the addresses.

Note: If you have removed a symbol, Twido will not create it any more.

Macro Replacement and Binary Code Generation

Display of Equivalent Real Code

TwidoSoft can display equivalent real macro code. Macro Instructions are replaced using configuration and MDF file to create a Temporary equivalent LIST code.
In the LIST editor **Tools** menu, select the following item: **Display real code equivalent to macros**.
Click it to open an **Equivalent real code** dialog box, such as:



Note: LIST code displayed is read-only.

LIST blocks can be selected or not:

If ...	Then ...
No block is selected in the LIST editor	Macros are replaced in all LIST code then displayed.
Blocks are selected in the LIST editor	Macros are replaced in the selected block LIST code then displayed.

Symbols and Addresses: If the LIST editor displays symbols and/or addresses, **Equivalent real code** dialog box displays symbols and/or addresses too.

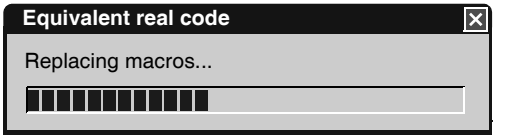
Macro not configured: If one of the Macros to display is not configured, "Macro %s is not configured and thus cannot be displayed " appears and the **Equivalent real code** dialog box is not displayed.

Source is too long: In case of more than 250 line in LIST source, the following message appears: "A big amount of code must be displayed. This operation can be time-consuming. Do you wish to continue?". You can then continue or Cancel.

Help: The Help button opens TwidoSoft **Help with Macros**.

Progress Dialog Box

Each time TwidoSoft generates equivalent real LIST code, a progress dialog box appears:



From the equivalent macro code, you can get an estimation of the memory used, via the memory editor.

Equivalent Real Code Processing

Instead of compiling source code, TwidoSoft compiles the Equivalent real code to generate binary code to be sent to Twido controller. Macros do not appear any more as macros, but as their equivalent LIST code.

Note: During this step, some errors may appear. Any macro error will appear in the Program Errors viewer.

Binary Code Generation and Sending

Whenever needed, TwidoSoft compiles the temporary code to generate binary. Binary code is then sent to Twido controller.

MDF Help File

Help Access

Help can be accessed:

From the **Help** main menu	1. Click **Help with Macros** item. 2. Select one of the 3 Macro groups: - Comm - Drive - TeSys
From the Application Browser	1. Select one of the 3 Macro groups: - Comm - Drive - TeSys 2. Right-click to open a pop-up menu. 3. Select the **Help** item.

Note: If there is no help file, TwidoSoft displays no message but all help menus and buttons are disabled.

Error Messages

Error messages can be associated to Macro Definition Functions.
Each error message has been isolated to provide you with the context in which the error comes, the error message you get and the consequence this error may have on your program:

Error context	Error message
MDF_STRUCT_VERSION tag	Error with MDF file '%s': the structure version %s of the MDF file is not supported.
MDF_COMPATIBILITY tag	Error with MDF file '%s': MDF file compatibility level %d not supported.
MDF_FILE_VERSION tag MDF_NB_MW_USED_DELTA tag	Error with MDF file '%s': contact your supplier for an update. (Incorrect value for tag %s. This tag must be numerical)
A mandatory tag is missing.	Error with MDF file '%s': contact your supplier for an update. (Missing tag)
MDF_SYMBOL_NAME tag	Error with MDF file '%s': contact your supplier for an update. (Incorrect symbol name)
MDF_SYMBOL_ADDRESS tag	Error with MDF file '%s': contact your supplier for an update. (Incorrect symbol address format)
MDF_NETWORK tag	Error with MDF file '%s': contact your supplier for an update. (Incorrect network profile for function %s)
MDF_PROFILE_LANGUAGE tag	Error with MDF file '%s': contact your supplier for an update. (Language %s not supported)
Macro configuration dialog box (General tab) that has not been configured.	No network available and configured.
If an MDF version used to create TWD is lower than the MDF version used by TwidoSoft to load TWD.	Warning: The file %s has been updated as it was created with an obsolete version of %s. **Consequence:** TWD file is loaded. All symbols needed are created as in the function configuration.
If an MDF version used to create TWD is greater than the MDF version used by TwidoSoft to load TWD.	The file %s was created with a more recent version of %s. It cannot be loaded. **Consequence:** TWD will not be loaded unless the MDF file is updated.
If you select "Display of equivalent real macro code" but a macro is not configured.	Macro %s is not configured and so cannot be displayed. **Consequence:** "Equivalent real macro" dialog box is not displayed.
If you select "Display of equivalent real macro code" but there are more than 250 lines in List source.	A big amount of code must be displayed. This operation can be time-consuming. Do you wish to continue? **Consequence:** You can decide to continue or to cancel.
If a program contains macro error(s).	<ERROR> Line %d: MACRO %s NOT CONFIGURED
If you try to open the Memory editor but there is a macro error.	Macro %s is not configured. Memory Report cannot be displayed. **Consequence:** A memory box appears to display the first error message. The Memory Report dialog box is not displayed.

Error context	Error message
If a profile selected in the configuration dialog box is not valid but you click OK, Previous/ Next, change tabs or change macro numbers.	Invalid profile.
If you change protocol types from Modbus to another protocol and press OK, but at least one configured macro uses this Modbus port.	Unable to change protocol type for port %d, as at least one macro is configured in Modbus on this communication port. **Consequence:** You cannot close the dialog box until protocol type is Modbus.
If you try to remove a CANopen module but at least one configured macro uses CANopen.	Unable to delete CANopen module, as at least one macro requires access to CANopen network.
	Macro %s not defined in any MDF file.
"parameter0" is not configured.	The instance %d of macro %s is not configured.
The number of operands is not good.	The macro %s requires %d operand(s).
"parameter1" is lower than the minimum value.	The operand %d of macro %s must be greater than or equal to %d.
"parameter1" is greater than the maximum value.	The operand %d of macro %s must be less than or equal to %d.
"parameter0" is not in the range [0-31].	The operand 1 of macro %s must be between 0 and 31.
"parameter0" has no valid profile.	The instance %d of macro %s is not correct.
In Program printing, Table of contents, page numbers are not displayed.	Click to display the page numbers. **Consequence:** Click the message to generate page numbers.
In configuration dialog box, when General tab is leaved after changing tabs, clicking OK or Previous/Next.	Warning: The %s and the %s are configured at the same network address.
If an MDF file exists but is invalid (not a correct zip format, bad password, ...)	Unable to open file %s: contact your supplier for an update.
If PRE function is needed but is not present in LIST source.	Missing macro " %s_PRE %d "
If POST function is needed but is not present in LIST source.	Missing macro " %s_POST %d "
If you chose a manual fonctional level lower than the compatibility level provided in the MDF file.	The chosen functional level does not allow you to use the instruction "%s". You must increase the level. **Consequence:** The message does not prevent to enable the menu but you will have to increase the level.
If you try to change a base controller with a macro configured for Modbus on port 2 but the final controller does not support a second port.	Unable to change the base controller for the selected type, as at least one macro is configured in Modbus on port 2 and this controller cannot have a second communication port.
If you try to change a base controller for a controller without an expansion bus and a macro is configured for CANopen.	Unable to change the base controller for the selected type, as at least one macro is configured in CANopen and this controller does not have an expansion bus.

Macro Information Printing

Information about a Macro configuration or a Macro program can be printed.

Configuration Printing

To access the printing window, select **File** main menu, then **Print Configuration**.
Macro objects configuration can be printed via the **Documentation** window.
In the left-pane tree-structure, you have to click **Folder**, **Configuration** then **Memory Objects**.

To print ...	**Click ...**
all macro objects,	**All external objects** box
a Macro group,	**Comm**, **Drive** or **TeSys** box

Program Printing

In the left-pane tree-structure, if you check **Program**, you access a new printing window.
In the main program window, if you check **Print equivalent real code**, you will generate then print out real macro code instead of edited source code.

Debugging and Adjusting a Program Online

8

At a Glance

Overview

This chapter provides details on debugging and adjusting a program online.

What's in this Chapter?

This chapter contains the following sections:

8.1 Debugging Online via the Program Animation Table

At a Glance

Overview

This section describes the way to use a Program Animation Table in order to debug a program online.

What's in this Section?

This section contains the following topics:

Animating a Program

Introduction

Animating a program provides a view of actual values of variables when a program is online, either running or stopped. This is useful for debugging because changes in values can be viewed as the program runs to compare the actual values against expected values.

Values Animated for Ladder Programs

When animating a Ladder program, the following data is displayed in the Ladder Viewer:

- Contacts, coils, and special objects with logical results of 1 are highlighted.
- The data variables of function blocks, compare blocks, and operate blocks are displayed including the current and preset values.
  Binary operands are displayed as 0 or 1, while all other values are displayed in hexadecimal or decimal, depending on the format selected in the *Preferences, p. 31* dialog box.

Animating a Ladder Program

To animate a Ladder Program, with the Ladder Viewer open, and a program online (running or stopped):

Step	Action
1	Select **PLC** → **Toggle animation** from the main menu. The Ladder Viewer displays the following: ● **Ladder Viewer - Animating** in the title bar ● Animated data.
2	Select **PLC** → **Toggle animation** from the main menu to turn off the animation.

Values Animated for List Programs

When animating a List program, the following values are animated in the List Editor:

- A column is added to the right of the Line Number column. This column contains the value of the operand for that program line.
  If an instruction contains more than one operand, the value of each operand is displayed and separated by a slash character (/).
- Binary operands are displayed as 0 or 1, while all other values are displayed in hexadecimal or decimal, depending on the format selected in the *Preferences, p. 31* dialog box.

Values Not Animated for List Programs

When animating a List program, the following values are not animated in the List Editor, but are represented by an asterisk (*):

- Labels (%Li)
- Subroutines (SRn)
- Instructions that have no operands (NOT, NOP, END)
- Immediate values
- Indexed words
- Bits extracted from words
- Tables of words
- Strings of bits (for example, %M0:5)

Display of Forced Bits

Forced bits are displayed with an F paired with the forced state, either a 0 or a 1:

- A bit operand that is forced On, is displayed as "F 1".
- A bit operand that is forced Off, is displayed as "F 0".

See *Forcing Input/Output Values, p. 345*.

Animating a List Program

To animate a List Program, with the List Editor open, and a program online (running or stopped):

Step	Action
1	Select **PLC** → **Toggle animation** from the main menu. The List Editor displays the following: • **List Editor - Animating** in the title bar • Animated values.
2	Select **PLC** → **Toggle animation** from the main menu to turn off the animation.

Using the Animation Tables Editor

Introduction

The *Animation Tables Editor, p. 50* is used to view and modify program variables to assist in debugging a program. The Animation Tables Editor can also be used to force the values of input or output bits.
The list of data variables that the Animation Tables Editor displays is known as an Animation Table.

Animation Tables

An **Animation Table** displays a list of controller variables that you want to monitor and save. Data in the table is organized similar to a spreadsheet with six columns:

- Address
- Current
- Retained
- Format
- Symbol
- Valid

Address

An **Address** is a specific location in the controller memory, always preceded by a percent sign (%). An address does not require an assigned symbol, but a symbol must have an assigned address.

Current

The **Current** value is the actual value of a variable in the controller. In the online state, this value changes as the program runs. This changing value can be monitored by animating the program.
An asterisk (*) appears in this column prior to the first time the data is animated. When the data is animated and subsequently turned off, the Current column displays the last updated value.

Retained

The **Retained** value is the initial value for variables in the controller. When the Write Retained Values command is performed, these values are written to the controller.

Format

The **Format** column identifies the number format of the data variable. Options are Decimal, Hexadecimal, Binary, Floating Point or ASCII.

Symbol

A **Symbol** is a name you assigned to the address in the *Symbol Editor, p. 45* to identify the purpose of the variable. A symbol must have an assigned address.

Valid

The **Valid** column indicates if the variable or object has been validated. A variable is Valid only when it has been configured in the currently opened application program. A green check mark indicates a valid variable or object, while a red X identifies a variable or object that is invalid or not configured in the currently configured and opened application program.

Animation Tables Editor Commands

Introduction

The animation tables' tools menu is used to construct, edit, and save an animation table. Additionally, you can modify and force values for selected variables in a program.

With the Animation Tables Editor open, the following commands are available from the **Tools** Menu:

- Add Next Instance (See *Add Next and Previous Instances, p. 344*)
- Add Previous Instance (See *Add Next and Previous Instances, p. 344*)
- Force 1 (See *Forcing Input/Output Values, p. 345*)
- Force 0 (See *Forcing Input/Output Values, p. 345*)
- Clear Force (See *Forcing Input/Output Values, p. 345*)
- Clear All Force (See *Forcing Input/Output Values, p. 345*)
- Read Retained Values (See *Reading and Writing Values, p. 342*)
- Write Retained Values (See *Reading and Writing Values, p. 342*)
- Write Data Value (See *Reading and Writing Values, p. 342*)
- Open Animation Table (See *Opening and Saving Animation Tables, p. 347*)
- Save Animation Table (See *Opening and Saving Animation Tables, p. 347*)
- Save Animation Table As (See *Opening and Saving Animation Tables, p. 347*)

Inserting, Editing, and Deleting Variables

Introduction

Use the Animation Tables Editor to insert, edit, and delete variables in an animation table. The following conditions are required:

- An application must be open.
- A PC must be connected to the controller.
- The controller can be running or stopped.

Inserting a Variable

To insert a single variable in an animation table:

Step	Action
1	Click an **Address** cell and enter a variable or object. Entry must be preceded by a percent symbol (%).
2	Press ENTER to complete the entry. The following default values are displayed automatically for the entry: ● **Current** value defaults to an asterisk (*). ● **Retained** value defaults to 0. ● **Format** defaults to Decimal. ● The **Symbol** for the variable, if previously defined in currently opened application. See *Defining Symbols, p. 81*.

Editing a Variable

To edit a single variable in an animation table:

Step	Action
1	Click an **Address** cell to edit the address for the variable or object. You can also highlight text using the mouse to edit a specific portion of the entry, or double-click the cell to highlight the entire entry. The **Current** column is read-only. It updates only when TwidoSoft is connected to the controller (online state).
2	To modify the **Retained** value, click the cell and enter a value. Only valid entries will be accepted.
3	To update the Display **Format**, click the column and select an option from a list of valid formats. Only valid options for the given data type will be available. **Options:** Decimal, Hexadecimal, Binary, Floating Point or ASCII.
4	Press ENTER to accept the display format change.
5	The symbol for the variable cannot be edited directly from the Animation Tables Editor. Click the Symbol Mode button  in the toolbar to display the Symbol Editor and create a new symbol or edit an existing symbol. See *Defining Symbols, p. 81*. **Note:** If a symbol has not been previously defined, it will not be displayed in the Animation Tables Editor. An animation table can be saved as a separate file from the application.
6	Click the Animation Editor button  in the toolbar to return to the Animation Tables Editor.

Deleting a Variable

To delete a single variable from an animation table:

Step	Action
1	Select a variable to delete by clicking the row number.
2	Select **Tools** → **Delete Row**, or click the Delete Row button  on the toolbar, or click the DELETE key. The variable row is deleted.

Reading and Writing Values

Introduction

Use read and write values commands to transfer values between an animation table and the controller. The following commands are available from the **Tools** menu when the Animation Tables Editor is open:

- Read Retained Values
  Transfer Current values on the controller to Retained values in an animation table.
- Write Retained Values
  Transfer Retained values in the animation table to the Current values in the controller.
- Write Data Value
  Momentarily send a single data value to the controller. An animation table does not need to be open.

For these commands, the program must be online, either stopped or running. An animation table must be open and animated.

Read Retained Values

To transfer Current values on the controller to Retained values in an animation table:

Step	Action
1	Select **Tools** → **Read Retained Values** from the main menu or from the Animation Tables Editor toolbar.
2	The **Retained** values column is updated with the values from the **Current** values column.

Write Retained Values

To transfer Retained values in an animation table to the Current values in the controller:

Step	Action
1	Select **Tools** → **Write Retained Values** from the main menu or from the Animation Tables Editor toolbar.
2	The **Current** values column is updated with the values from the **Retained** values column.

Write Data Value

To transfer a single data value to the controller:

Step	Action
1	Select **Tools** → **Write Data Value** from the main menu or from the Animation Tables Editor toolbar. An animation table does not need to be open. The **Write Data Value** dialog box is displayed.
2	Enter a variable in the **Data Object** box. The **Current Value** box displays the current value for the selected variable.
3	Select the format for the value from the format list. **Options:** Decimal, Hexadecimal, Binary, Floating Point or ASCII. **Default:** Decimal.
4	In the **Write Data Value** box, enter a value to write to the controller for the variable.
5	Select **OK** to write the value to the controller. Select **Cancel** to return to the Animation Tables Editor without writing the value to the controller. **Note:** Depending on the frequency of changes for the variable, the change in values may not be detectable.

Add Next and Previous Instances

Introduction

The **Add Next Instance** and the **Add Previous Instance** commands add sequential instances of a variable to the animation table. For example, if the variable %I0.3 is currently highlighted in an animation table, selecting **Add Next Instance** inserts the new variable %I0.4 immediately after %I0.3 in the animation table. The following two commands are available from the **Tools** menu with the Animation Tables Editor open:

- Add Next Instance
  Adds the next sequential instance of a variable highlighted in the table. For example, if %I0.3 is highlighted, variable %I0.4 is added to the table.
- Add Previous Instance
  Adds the previous sequential instance of a variable highlighted in the table. For example, if %I0.3 is highlighted, variable %I0.2 is added to the table.

Add Next Instance

To add the next sequential instance of a variable highlighted in the table:

Step	Action
1	Select a variable in the animation table.
2	Select **Add Next Instance** from the **Tools** menu, or from the Animation Tables Editor toolbar, or press Ctrl+Down. A new variable of the same type is added to the animation table with a sequential number incremented by one from the highlighted variable.

Add Previous Instance

To add the previous sequential instance of a variable highlighted in the table:

Step	Action
1	Select a variable in the animation table.
2	Select **Add Previous Instance** from the **Tools** menu, or from the Animation Tables Editor toolbar, or press Ctrl+Up. A new variable of the same type is added to the animation table with a sequential number decremented by one from the highlighted variable.

Forcing Input/Output Values

Introduction

⚠ WARNING

UNEXPECTED EQUIPMENT OPERATION

When an output is being forced on or off, you must have prior knowledge of how this will effect either the process or the controlled equipment.

Failure to follow this instruction can result in death, serious injury, or equipment damage.

The Animation Tables Editor can be used to force input and output bits to 0 or 1 values even when the actual values are different. The **Current** value column of the animation table displays an F next to the value if it is a forced value and the table is animated. The value remains forced even when the PC is disconnected from the controller and TwidoSoft is closed.

Force Commands

The following force commands are available in animation tables:

- **Force 1** - sets a value to 1.
- **Force 0** - sets a value to 0.
- **Clear force** - removes a forced value from a selected variable.
- **Clear All Force** - removes all forced values from variables.

The force commands are available from the **Tools** menu and the Animation Tables Editor toolbar when the Animation Tables Editor is open and the controller is online, either running or stopped. An animation table must be open and animated.

Force 1

To force the value of a variable to 1 in an animation table:

Step	Action
1	Highlight a variable in the animation table.
2	Select **Tools** → **Force 1** from the main menu. The animation table displays an F next to the current value of 1.

Force 0

To force the value of a variable to 0 in an animation table:

Step	Action
1	Highlight a variable in the animation table.
2	Select **Tools** → **Force 0** from the main menu. The animation table displays an F next to the current value of 0.

Clear Force

To remove a forced value from a variable in an animation table:

Step	Action
1	Highlight a variable in the animation table.
2	Select **Tools** → **Clear Force** from the main menu. The forced value is removed from the highlighted variable in the animated table.

Clear All Force

To remove all forced value in an animation table:

Step	Action
1	Highlight a variable in the animation table.
2	Select **Tools** → **Clear All Force** from the main menu. All forced values are removed from the animation table.

Opening and Saving Animation Tables

Introduction

Animation tables can be saved for later use. Although animation tables are usually associated with an open application, it is possible to open and use an existing animation table created by another application.

Commands

The following commands are available from the **Tools** menu with the Animation Tables Editor open:

- Open Animation Table
  Open a previously saved animation table file.
- Save Animation Table
  Save current animation table with a .tat (default) file extension. Once saved, the title of the animation table appears on the title bar of the Animation Tables Editor.
- Save Animation Table As
  Save an animation table for the first time or rename an animation table file.

Opening an Animation Table

To open a previously saved animation table:

Step	Action
1	Select **Tools** → **Open Animation Table** from the main menu. The **Open** dialog box is displayed.
2	Select a file with a .tat extension from the list, or browse to find a file.
3	Double-click to open the selected file, or click and press the **Open** button in the dialog box.
4	The Animation Tables Editor displays the selected animation table.

Initial Save of an Animation Table

To save an animation table for the first time:

Step	Action
1	Select **Tools** → **Save Animation Table As** from the main menu. The **Save As** dialog box is displayed.
2	Enter a file name for the animation table. File names can be up to 255 characters. **Note:** Do not use any of the following characters: \:*?<>|".
3	Click the **Save** button.The animation table is saved with the .tat (default) extension. The file name and path are displayed in the title bar of the Animation Tables Editor.

Saving Changes to an Animation Table

To save changes to a previously saved animation table:

Step	Action
1	Select **Tools** → **Save Animation Table** from the main menu.
2	Changes are saved to the open animation table.

Animating a Table

Introduction

Animating a table of data variables displays and updates the Current Value column of the animation table as the controller runs.
To animate a table:

- An application must be open.
- A PC must be connected to the controller.
- The controller can be running or stopped.

Animating a Table

To animate a table:

Step	Action
1	Select **Program** → **Animation table editor** from the main menu to open the Animation Tables Editor. Before the table is animated for the first time, the **Current** column contains asterisks (*). If the application is not connected to the controller already, see *Connecting a PC to the Controller, p. 97*.
2	When the PC is connected to the controller, select **PLC** → **Toggle animation** from the main menu to animate the table. The Current Values in the **Current** column are updated and "Animating" appears in the title bar of the editor.
3	Select **PLC** → **Toggle animation** from the main menu to turn off the animation. The Current Values will retain the last updated value.

8.2 Adjusting a Program Online

At a Glance

Overview

This section provides details about the way to adjust a program online.

What's in this Section?

This section contains the following topics:

Overview of Online Programming

Introduction	Online programming mainly consists in editing data when a Twido controller is connected. Editing data means that you can insert, delete or change program instructions or modify object parameters.
STOP and RUN modes	The connected Twido controller can either be in Stop or in Run mode. Some operations apply to both, but others are specific to one mode.
List and Ladder Editors	You can use the List and the Ladder Editors. Some operations can be done in both, but others are specific to one editor.

Online Ladder and List Programming in STOP Mode

Introduction

The Ladder and List Editors allow a program to be edited when the controller is connected and is in STOP Mode.

> **⚠ WARNING**
>
> **UNPREDICTABLE EQUIPMENT OPERATION**
>
> Before any modifications are performed, it is essential to **identify the consequences of modifications** made to an application while the controller is running. Perform all necessary preventative actions to ensure safe conditions while making modifications.
>
> **Failure to follow this instruction can result in death, serious injury, or equipment damage.**

Editing a Program in STOP Mode

All changes are allowed, except to unallocated objects. Their number is only limited by the available controller memory. Thus, changes are analized to determine the executable state of the user logic.

Validating a Program in STOP Mode

To edit a program,TwidoSoft requires that all modifications be validated before being transferred to the controller.

- In a Ladder program, click the **Accept** icon.
- In a List program, click outside the line.

The valid modifications take effect as soon as the current entry is confirmed.
The validation is also necessary if you want to switch between the List and Ladder programs.

Restrictions in STOP Mode

You can create changes to an application that can prevent the controller from running the application. This creates a "Stop/Not Exec" state that is displayed in the status bar. The controller will not go back into the "Stop/Exec" state until all causes for the non-executable state have been resolved. These changes can be grouped into six categories, which are described in the following table.

Function	Restrictions to running an application
Parentheses	● An opening parenthesis with no closing parenthesis. ● A closing parenthesis with no opening parenthesis (unbalanced parentheses). ● More than eight nested parentheses.
Function Blocks	● A block that has a beginning with no end (BLK is not followed by an END_BLK). ● A block that has an end/output with no start (END_BLK or OUT_BLK not preceded by BLK). ● An invalid block (block that contains invalid logic). ● BLK not followed by an LD instruction.
Subroutines	● A call to a non-existent subroutine. Two possible cases: - Referencing a non-existent subroutine. - Deleting a label that is referenced elsewhere in the application. ● A subroutine without a return. ● A return without a start for the subroutine. ● A subroutine with multiple definitions. ● A subroutine definition not followed by an LD instruction. ● Definition of subroutine or End of Subroutine (RET) in parentheses.
Labels	● A jump to a non-existent label. Two possible cases: - Referencing a non-existent label. - Deleting a label that is referenced elsewhere in the application. ● A label with multiple definitions. ● A label definition not followed by an LD instruction. ● A label definition in parentheses.
Grafcet instructions	● A transition to a non-existent step. Two possible cases: - Referencing a non-existent step. - Deleting a step that is referenced elsewhere in the application. ● A step with multiple definitions. ● A step definition not followed by an LD instruction. ● Use of Grafcet instructions in Pre or Post processing.
Stack operations (Push/Pop)	● The number of Pops (MPPs) is greater than the number of Pushes (MPSs). ● More than eight levels of nesting. Note: Special care must be taken when inserting or deleting stack instructions (MPP, MPS, MRD) in the STOP state. Analyze the program to be sure of restrictions before re-starting the controller.
Macro instructions are not allowed.	
Within a Ladder program, changes with the Network List Editor are not allowed **Note:** The Network List Editor is a window that pops up when you switch from the List to the Ladder editor and that program lines cannot be edited in Ladder language. If you try to edit such a block of List instructions, you will get the following message: "This block cannot be modified with ladder Editor, use List Editor to perform some changes into this block".	

Online Ladder and List Programming in RUN mode

Introduction

The Ladder and List Editors allow a program to be edited when the controller is connected and is in RUN Mode.

> **⚠ WARNING**
>
> **UNPREDICTABLE EQUIPMENT OPERATION**
>
> For safety reasons, it is advisable to program a controller in STOP mode. Programming a controller in RUN mode can present hazards to both equipment and personnel.
> It is possible to program a controller in RUN mode if a modification to a program does not require the controller to be stopped. However, ensuring safe operating conditions remains the responsibility of the user.
> Before any modifications are performed, it is essential to **identify the consequences of modifications** made to an application while the controller is running. Perform all necessary preventative actions to ensure safe conditions while making modifications.
>
> **Failure to follow this instruction can result in death, serious injury, or equipment damage.**

Program Version Compatibility

TwidoSoft and Twido controller firmware must have a compatibility level of 3.0 or higher. With the Ladder editor, when uploading an application from the controller, TwidoSoft can recognize the new labels used for online changes.

Editing a Program in RUN Mode

All insertions, deletions and changes of instructions are permitted, except for instructions that modify the structure of the program.

Validating a Program in RUN Mode

To edit a program,TwidoSoft requires that all modifications be validated before being transferred to the controller.

- In a Ladder program, click the **Accept** icon.
- In a List program, click outside the line.

The valid modifications take effect as soon as the current entry is confirmed.
The validation is also necessary if you want to switch between the List and Ladder programs.

Restrictions in RUN Mode

All restrictions specific to the Stop mode apply. The following function changes are not allowed:

- Parenthesis
- Grafcet Instructions
- Labels or subroutine definitions
- Jump or subroutine calls (JMP and SR)
- Block instructions (BLK, OUT_BLK and END_BLK)
- Coil routing instructions (MPS, MRD and MPP), except if the compilation of Ladder requires and generates these instructions.

Other programming restrictions concern the following instructions, that are not available:

List and Ladder restrictions	**List restriction only**	**Ladder restriction only**
Backing up an application to internal EEPROM	Find	Changes with the Network List Editor. **Note**: The Network List Editor is a window that pops up when you switch from the List to the Ladder editor and that program lines cannot be edited in Ladder language. If you try to edit such a block of List instructions, you will get the following message: "This block cannot be modified with ladder Editor, use List Editor to perform some changes into this block.
Macro instructions	Operations linked to the paperboard (cut/paste/copy/delete). *(Mentioned in the status bar)*	
	Cancel. *(Mentioned in the status bar)*	

Online Program Editing Process in Ladder Language

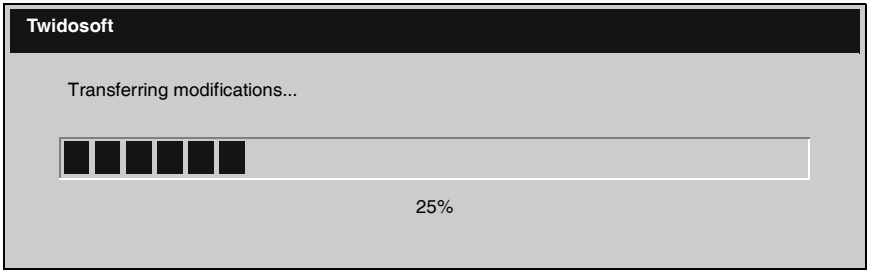
The online editing process means that you can insert, delete or change program data when the controller is connected.

Online Insert

To insert a new rung:

1. Click the **Insert** icon
   Result: The **Ladder Viewer - Insert Rung Editor** opens.
2. Edit the new rung.
3. Click the **Accept** icon to validate it.

A message box pops up, showing the transfer progress:



Note: This transfer window prevents you from performing any operation before the end of the transfer. The window cannot be resized nor closed. An hourglass can be associated with the mouse pointer.

Online Delete

To delete a rung:

1. In the **Ladder Viewer - Insert Rung Editor**, select the rung to delete.
2. Click the **Delete** icon.

A message box pops up, showing the transfer progress *(see capture above)*.

Online Change

To modify a rung:

1. Select the rung to modify.
   Result: The **Ladder Viewer - Insert Rung Editor** opens.
2. Click the **Edit** icon to open **Ladder Editor - Edit Rung**.
3. Modify the rung.
4. Click the **Accept** icon to validate it.

A message box pops up, showing the transfer progress *(see capture above)*.

Editing Data Objects Online

Overview

Editing data objects online means that you can modify parameter data. Configuration objects can have their parameters edited online. Runtime objects are not concerned.

List of Editable Data Objects

Each configuration object has one or several editable parameters:

Configuration object	Parameter
Constant (%KW, %KD and % KF)	Value
Counter (%C)	Preset
Timer (%TM)	Preset, Type and Timebase
Schedule Block	Day of week, Start and End Periods,Output Bit
Application name	Value
Input Run/Stop and PLC (%I) Output status (%Q)	Input Run/Stop
Base Input latch (%I)	Input Latch
Base Input filter (%I)	Filters
Scan mode	Normal or Periodic Mode, Watchdog
Second serial port	Protocol Type, Address and Parameters
Terminal port	Protocol Type, Address and Parameters
PLS and PWM (%PLS, %PWM)	Preset and Timebase
Fast Counter (%FC)	Preset
Very Fast Counter (%VFC)	Preset, Type,Thresholds
LIFO/FIFO Register (%R)	Type
Drum (%DR)	Number of Steps, Steps, Output
Analog modules	(Several parameters according to the type of module)

Re-initialization

All edited objects require a re-initialization.
Some require an immediate re-initialization when others do not.
An object re-initialization only affects the object that has been modified.

Note: Re-initializing an object does not affect the object family or any other program object, due to the presence of the immediate_init field in each PARAMETER_MODIF structure.

Immediate Re-initialization Is Necessary

When a parameter is only present in a configuration object, its immediate re-initialization is necessary for the modification(s) to be taken into account.
The immediate_init field of the PARAMETER_MODIF structure is equal to 1.
An immediate re-initialization is necessary for:

Configuration object	Parameter
Timer (%TM)	Type and Timebase
Second serial port	Protocol Type, Address and Parameters
Terminal port	Protocol Type, Address and Parameters
PLS and PWM (%PLS, %PWM)	Timebase
Very Fast Counter (%VFC)	Type
LIFO/FIFO Register (%R)	Type
Drum (%DR)	Number of Steps, Steps, Output

Immediate Re-initialization Is Not Necessary

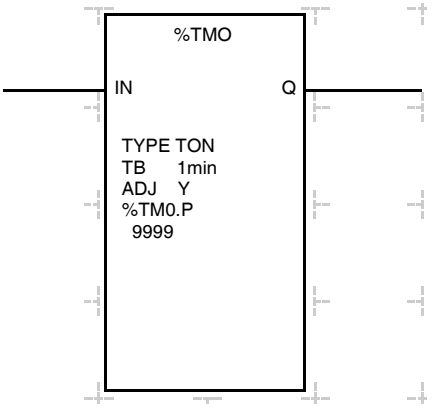
When a parameter is common to configuration and runtime objects, the object immediate re-initialization can be postponed, as the effect of the modification can be seen if you edit the runtime value.
The immediate_init field of the PARAMETER_MODIF structure is equal to 0.
No immediate re-initialization is necessary for:

Configuration object	**Parameter**
Constant (%KW, %KD and % KF)	Value
Counter (%C)	Preset
Timer (%TM)	Preset
Schedule Block	Day of Week, Start and End Periods, Output Bit
Application name	Value
Input Run/Stop and PLC (%I) Output status (%Q)	Input Run/Stop
Base Input latch (%I)	Input Latch
Base Input filter (%I)	Filters
Scan mode	Normal or Periodic Mode, Watchdog
PLS and PWM (%PLS, %PWM)	Preset
Fast Counter (%FC)	Preset
Very Fast Counter (%VFC)	Preset, Thresholds
Analog modules	(Several parameters according to the type of module)

Example of a Timer

Let us take the example of a Timer (%TM0).
This block is concerned by both re-initialization types:

- An immediate re-initialization for Type and Timebase parameters
- No immediate re-initialization for Preset parameter.



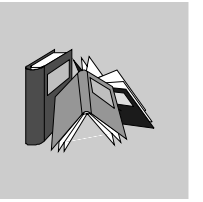
Preset parameter (%TM0.P) is also a runtime object, so you do not need to re-initialize the Timer to see the effects of a change.
TYPE and Timebase (TB) parameters are not runtime objects, so you need to reinitialize the Timer to see the effects of a change.

Note: Once the Timer has been edited and re-initialized, the configuration takes the new current values. The process is not reversible.

Case of an Analog Module Re-initialization

To take modifications to an analog module into account, all modules present on Twido internal bus need to be re-initialized. Considering the consequences of this operation, you may wish not to re-initialize immediately after the modifications are performed, but have a delay.
A dialog box opens, asking you if you want to re-initialize Twido internal bus now or later.

Glossary



!

% Prefix that identifies internal memory addresses in the controller that are used to store the value of program variables, constants, I/O, and so on.

A

Addresses Internal registers in the controller used to store values for program variables, constants, I/O, and so on. Addresses are identified with a percentage symbol (%) prefix. For example, %I0.1 specifies an address within the controller RAM memory containing the value for input channel 1.

Analog potentiometer An applied voltage that can be adjusted and converted into a digital value for use by an application.

Analyze program A command that compiles a program and checks for program errors: syntax and structure errors, symbols without corresponding addresses, resources used by the program that are not available, and if the program does not fit in available controller memory. Errors are displayed in the Program Errors Viewer.

Animation table Table created within a language editor or an operating screen. When a PC is connected to the controller, provides a view of controller variables and allows values to be forced when debugging. Can be saved as a separate file with an extension of .tat.

Animation Tables Editor A specialized window in the TwidoSoft application for viewing and creating Animation Tables.

Application A TwidoSoft application consists of a program, configuration data, symbols, and documentation.

Application browser A specialized window in the TwidoSoft that displays a graphical tree-like view of an application. Provides for convenient configuration and viewing of an application.

Application file Twido applications are stored as file type .twd.

ASCII (American Standard Code for Information Interchange) Communication protocol for representing alphanumeric characters, notably letters, figures and certain graphic and control characters.

Auto line validate When inserting or modifying List instructions, this optional setting allows for program lines to be validated as each is entered for errors and unresolved symbols. Each element must be corrected before you can exit the line. Selected using the Preferences dialog box.

Auto load A feature that is always enabled and provides for the automatic transfer of an application from a backup cartridge to the controller RAM in case of a lost or corrupted application. At power up, the controller compares the application that is presently in the controller RAM to the application in the optional backup memory cartridge (if installed). If there is a difference, then the copy in the backup cartridge is copied to the controller and the internal EEPROM. If the backup cartridge is not installed, then the application in the internal EEPROM is copied to the controller.

B

Backup A command that copies the application in controller RAM into both the controller internal EEPROM and the optional backup memory cartridge (if installed).

BootP A UDP/IP-based protocol (Bootstrap Protocol) which allows a booting host to configure itself dynamically and without user supervision. BootP provides a means to notify a host of its assigned IP address.

C

CAN **Controller Area Network**: field bus originally developed for automobile applications which is now used in many sectors, from industrial to tertiary.

CiA	**CAN in Automation**: international organization of users and manufacturers of CAN products.
Client	A computer process requesting service from other computer processes.
COB	**Communication OBject**: transport unit on CAN bus. A COB is identified by a unique identifier, which is coded on 11 bits, [0, 2047]. A COB contains a maximum of 8 data bytes. The priority of a COB transmission is shown by its identifier - the weaker the identifier, the more priority the associated COB has.
Coil	A ladder diagram element representing an output from the controller.
Cold start or restart	A start up by the controller with all data initialized to default values, and the program started from the beginning with all variables cleared. All software and hardware settings are initialized. A cold restart can be caused by loading a new application into controller RAM. Any controller without battery backup always powers up in Cold Start.
Comment lines	In List programs, comments can be entered on separate lines from instructions. Comments lines do not have line numbers, and must be inserted within parenthesis and asterisks such as: (*COMMENTS GO HERE*).
Comments	Comments are texts you enter to document the purpose of a program. For Ladder programs, enter up to three lines of text in the Rung Header to describe the purpose of the rung. Each line can consist of 1 to 64 characters. For List programs, enter text on n unnumbered program line. Comments must be inserted within parenthesis and asterisks such as: (*COMMENTS GO HERE*).
Compact controller	Type of Twido controller that provides a simple, all-in-one configuration with limited expansion. Modular is the other type of Twido controller.
Configuration editor	Specialized TwidoSoft window used to manage hardware and software configuration.
Constants	A configured value that cannot be modified by the program being executed.
Contact	A ladder diagram element representing an input to the controller.
Counter	A function block used to count events (up or down counting).
Cross references	Generation of a list of operands, symbols, line/rung numbers, and operators used in an application to simplify creating and managing applications.

Cross References Viewer	A specialized window in the TwidoSoft application for viewing cross references.

D

Data variable	See Variable.
Date/Clock functions	Allow control of events by month, day of month, and time of day. See Schedule Blocks.
Default gateway	The IP address of the network or host to which all packets addressed to an unknown network or host are sent. The default gateway is typically a router or other device.
Drum controller	A function block that operates similar to an electromechanical drum controller with step changes associated with external events.

E

EDS	**Electronic Data Sheet**: description file for each CAN device (provided by the manufacturers).
EEPROM	Electrically Erasable Programmable Read-Only Memory. Twido has an internal EEPROM and an optional external EEPROM memory cartridge.
Erase	This command deletes the application in the controller, and has two options: • To delete the contents of the controller RAM, the controller internal EEPROM, and the installed optional backup cartridge. • To delete the contents of the installed optional backup cartridge only.
Executive loader	A 32-Bit Windows application used for downloading a new Firmware Executive program to a Twido controller.
Expansion bus	Expansion I/O Modules connect to the base controller using this bus.
Expansion I/O modules	Optional Expansion I/O Modules are available to add I/O points to a Twido controller. (Not all controller models allow expansion).

F

Fast counters A function block that provides for faster up/down counting than available with the Counters function block. A Fast Counter can count up to a rate of 5 KHz.

FIFO First In, First Out. A function block used for queue operations.

Firmware executive The Firmware Executive is the operating system that executes your applications and manages controller operation.

Forcing Intentionally setting controller inputs and outputs to 0 or 1 values even if the actual values are different. Used for debugging while animating a program.

Frame A group of bits which form a discrete block of information. Frames contain network control information or data. The size and composition of a frame is determined by the network technology being used.

Framing types Two common framing types are Ethernet II and IEEE 802.3.

Function block A program unit of inputs and variables organized to calculate values for outputs based on a defined function such as a timer or a counter.

G

Gateway A device which connects networks with dissimilar network architectures and which operates at the Application Layer. This term may refer to a router.

Grafcet Grafcet is used to represent the functioning of a sequential operation in a structured and graphic form.
This is an analytical method that divides any sequential control system into a series of steps, with which actions, transitions, and conditions are associated.

H

Host A node on a network.

Hub	A device which connects a series of flexible and centralized modules to create a network.

I

Init state	The operating state of TwidoSoft that is displayed on the Status Bar when TwidoSoft is started or does not have an open application.
Initialize	A command that sets all data values to initial states. The controller must be in Stop or Error mode.
Instance	A unique object in a program that belongs to a specific type of function block. For example, in the timer format %TMi, i is a number representing the instance.
Instruction List language	A program written in instruction list language (IL) is composed of a series of instructions executed sequentially by the controller. Each instruction is composed of a line number, an instruction code, and an operand.
Internet	The global interconnection of TCP/IP based computer communication networks.
IP	Internet Protocol. A common network layer protocol. IP is most often used with TCP.
IP Address	Internet Protocol Address. A 32-bit address assigned to hosts using TCP/IP.

L

Ladder editor	Specialized TwidoSoft window used to edit a Ladder program.
Ladder language	A program written in Ladder language is composed of graphical representation of instructions of a controller program with symbols for contacts, coils, and blocks in a series of rungs executed sequentially by a controller.
Ladder list rung	Displays parts of a List program that are not reversible to Ladder language.
Latching input	Incoming pulses are captured and recorded for later examination by the application.
LIFO	Last In, First Out. A function block used for stack operations.

List editor	Simple program editor used to create and edit a List program.

M

MAC Address	Media Access Control address. The hardware address of a device. A MAC address is assigned to an Ethernet TCP/IP module in the factory.
Master controller	A Twido controller configured to be the Master on a Remote Link network.
MBAP	Modbus Application Protocol
Memory cartridge	Optional Backup Memory Cartridges that can be used to backup and restore an application (program and configuration data). There are two sizes available: 32 and 64 Kb.
Memory usage indicator	A portion of the Status Bar in the TwidoSoft main window that displays a percentage of total controller memory used by an application. Provides a warning when memory is low.
Modbus	A master-slave communications protocol that allows one single master to request responses from slaves.
Modular controller	Type of Twido controller that offers flexible configuration with expansion capabilities. Compact is the other type of Twido controller.
Monitor state	The operating state of TwidoSoft that is displayed on the Status Bar when a PC is connected to a controller in a non-write mode.

N

Network	Interconnected devices sharing a common data path and protocol for communication.
Node	An addressable device on a communications network.

O

Offline operation An operation mode of TwidoSoft when a PC is not connected to the controller and the application in PC memory is not the same as the application in controller memory. You create and develop an application in Offline operation.

Offline state The operating state of TwidoSoft that is displayed on the Status Bar when a PC is not connected to a controller.

Online operation An operation mode of TwidoSoft when a PC is connected to the controller and the application in PC memory is the same as the application in controller memory. Online operation can be used to debug an application.

Online state The operating state of TwidoSoft that is displayed on the Status Bar when a PC is connected to the controller.

Operand A number, address, or symbol representing a value that a program can manipulate in an instruction.

Operating states Indicates the TwidoSoft state. Displayed in the status bar. There are four operating states: Initial, Offline, Online, and Monitor.

Operator A symbol or code specifying the operation to be performed by an instruction.

P

Packet The unit of data sent across a network.

PC Personal Computer.

Peer controller A Twido controller configured as a slave on a Remote Link network. An application can be executed in the Peer Controller memory and the program can access both local and expansion I/O data, but I/O data can not be passed to the Master Controller. The program running in the Peer Controller passes information to the Master Controller by using network words (%INW and %QNW).

PLC Twido programmable controller. There are two types of controllers: Compact and Modular.

PLS	Pulse Generation. A function block that generates a square wave with a 50% on and 50% off duty cycle.
Preferences	A dialog box with selectable options for setting up the List and Ladder program editors.
Program errors viewer	Specialized TwidoSoft window used to view program errors and warnings.
Programmable controller	A Twido controller. There are two types of controllers: Compact and Modular.
Protection	Refers to two different types of application protection: password protection which provides access control, and controller application protection which prevents all reads and writes of the application program.
Protocol	Describes message formats and a set of rules used by two or more devices to communicate using those formats.
PWM	Pulse Width Modulation. A function block that generates a rectangular wave with a variable duty cycle that can be set by a program.

R

RAM	Random Access Memory. Twido applications are downloaded into internal volatile RAM to be executed.
Real-time clock	An option that will keep the time even when the controller is not powered for a limited amount of time.
Reflex output	In a counting mode, the very fast counter's current value (%VFC.V) is measured against its configured thresholds to determine the state of these dedicated outputs.
Registers	Special registers internal to the controller dedicated to LIFO/FIFO function blocks.
Remote controller	A Twido controller configured to communicate with a Master Controller on a Remote Link network.

Remote link High-speed master/slave bus designed to communicate a small amount of data between a Master Controller and up to seven Remote Controllers (slaves). There are two types of Remote Controllers that can be configured to transfer data to a Master Controller: a Peer Controller that can transfer application data, or a Remote I/O Controller that can transfer I/O data. A Remote link network can consist of a mixture of both types.

Resource manager A component of TwidoSoft that monitors the memory requirements of an application during programming and configuring by tracking references to software objects made by an application. An object is considered to be referenced by the application if it is used as an operand in a list instruction or ladder rung. Displays status information about the percentage of total memory used, and provides a warning if memory is getting low. See Memory Usage Indicator.

Reversible instructions A method of programming that allows instructions to be viewed alternately as List instructions or Ladder rungs.

Router A device that connects two or more sections of a network and allows information to flow between them. A router examines every packet it receives and decides whether to block the packet from the rest of the network or transmit it. The router will attempt to send the packet through the network by the most efficient path.

RTC See Real-Time Clock.

RTU Remote Terminal Unit. A protocol using eight bits that is used for communicating between a controller and a PC.

Run A command that causes the controller to run an application program.

Rung A rung is located between two potential bars in a grid and is composed of a group of graphical elements joined to each other by horizontal and vertical links. The maximum dimensions of a rung are seven rows and eleven columns.

Rung header A panel that appears directly over a Ladder rung and can be used to document the purpose of the rung.

S

Scan A controller scans a program and essentially performs three basic functions. First, it reads inputs and places these values in memory. Next, it executes the application program one instruction at a time and stores results in memory. Finally, it uses the results to update outputs.

Scan mode Specifies how the controller scans a program. There are two types of scan modes: Normal (Cyclic), the controller scans continuously, or Periodic, the controller scans for a selected duration (range of 2 - 150 msec) before starting another scan.

Schedule blocks A function block used to program Date and Time functions to control events. Requires Real-Time Clock option.

Server A computer process that provides services to clients. This term may also refer to the computer process on which the service is based.

Step A Grafcet step designates a state of sequential operation of automation.

Stop A command that causes the controller to stop running an application program.

Subnet A physical or logical network within an IP network, which shares a network address with other portions of the network.

Subnet mask A bit mask used to identify or determine which bits in an IP address correspond to the network address and which bits correspond to the subnet portions of the address. The subnet mask is the network address plus the bits reserved for identifying the subnetwork.

Switch A network device which connects two or more separate network segments and allows traffic to be passed between them. A switch determines whether a frame should be blocked or transmitted based on its destination address.

Symbol A symbol is a string of a maximum of 32 alphanumeric characters, of which the first character is alphabetic. It allows you to personalize a controller object to facilitate the maintainability of the application.

Symbol table A table of the symbols used in an application. Displayed in the Symbol Editor.

T

TCP Transmission Control Protocol.

TCP/IP A protocol suite consisting of the Transmission Control Protocol and the Internet Protocol; the suite of communications protocols on which the Internet is based.

Threshold outputs Coils that are controlled directly by the very fast counter (%VFC) according to the settings established during configuration.

Timer	A function block used to select a time duration for controlling an event.
Twido	A line of Schneider Electric controllers consisting of two types of controllers (Compact and Modular), Expansion Modules to add I/O points, and options such as Real-Time Clock, communications, operator display, and backup memory cartridges.
TwidoSoft	A 32-Bit Windows, graphical development software for configuring and programming Twido controllers.

U

UDP	A communications protocol (User Datagram Protocol) that is the part of the TCP/IP suite used by applications to transfer datagrams. UDP is also the part of TCP/IP responsible for port addresses.
Unresolved symbol	A symbol without a variable address.

V

Variable	Memory unit that can be addressed and modified by a program.
Very fast counter:	A function block that provides for faster counting than available with Counters and Fast Counters function blocks. A Very Fast Counter can count up to a rate of 20 KHz.

W

Warm restart	A power-up by the controller after a power loss without changing the application. Controller returns to the state which existed before the power loss and completes the scan which was in progress. All of the application data is preserved. This feature is only available on modular controllers.

Index

Symbols

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V