

PiCPro™ for Windows®

Software Manual

Version 13.0

NOTE

Progress is an on-going commitment at Giddings & Lewis. We continually strive to offer the most advanced products in the industry; therefore, information in this document is subject to change without notice. The illustrations and specifications are not binding in detail. Giddings & Lewis shall not be liable for any technical or editorial omissions occurring in this document, nor for any consequential or incidental damages resulting from the use of this document.

DO NOT ATTEMPT to use any Giddings & Lewis product until the use of such product is completely understood. It is the responsibility of the user to make certain proper operation practices are understood. Giddings & Lewis products should be used only by qualified personnel and for the express purpose for which said products were designed.

Should information not covered in this document be required, contact the Customer Service Department, Giddings & Lewis, 660 South Military Road, P.O. Box 1658, Fond du Lac, WI 54936-1658. Giddings & Lewis can be reached by telephone at (920) 921-7100.

Release 2002

© 1995-2002 Giddings & Lewis, Controls, Measurement, and Sensing, A Company of Thyssen Krupp Technologies

IBM is a registered trademark of International Business Machines Corporation.
Windows 95, 98, NT, Microsoft, and MS-DOS are registered trademarks of Microsoft Corporation.
Pentium and PentiumPro are trademarks of Intel Corporation.
ARCNET is a registered trademark of Datapoint.
PiC900, PiCPro, PiCMicroTerm, MMC are trademarks of Giddings & Lewis.

Table of Contents

PiCPro for Windows Software Manual

CHAPTER 1-Getting Started with PiCPro for Windows	1-1
Introduction to PiCPro for Windows	1-1
PiCPro for Windows Professional Edition	1-2
PiCPro for Windows Standalone MMC Edition	1-2
PiCPro for Windows Monitor Edition	1-2
Customer Services	1-3
Principal Technical Support Services	1-3
Hands-on Training Schools	1-3
Before Calling Technical Support	1-3
World Wide Web Site	1-3
The Computer Workstation	1-4
Hardware Connections	1-4
Infrared Communications Drivers	1-4
Installing PiCPro for Windows	1-5
The Work Area in PiCPro for Windows	1-6
The Main Ladder Area	1-6
The Information Window	1-7
Status Bar	1-8
Window Elements	1-10
The Title Bar	1-10
The Menu Bar	1-11
Dialog Boxes	1-11
Help Files	1-11
Customizing PiCPro	1-12
User Preferences	1-12
Color	1-14
To change a color.....	1-14
Customizing Toolbars	1-15
Moving Toolbars	1-15
Displaying/Hiding Toolbars	1-16
Organizing your PiCPro Files	1-17
PiCPro Libraries	1-18
Library Contents	1-18
Using On-line Help	1-20
Using What's This? Help	1-20
Retrieving On-line Help	1-20
Printing On-line Help	1-20
Using Function/Function Block Help	1-21
Creating Function/Function Block Help	1-21
Starting PiCPro	1-22

CHAPTER 2-Working with Projects	2-1
Overview: Using Projects	2-1
Menu Commands that Manage Projects	2-1
The Project Tree and Its Components	2-3
Project Tree Categories	2-4
Category: PROJECT NAME	2-4
Category: MAIN.LDO	2-6
Item: MAIN.LDO	2-7
Category: COMPRESSED FILE	2-7
Item: COMPRESSED FILE	2-9
Category: PiCPro VERSION	2-9
Item: PiCPro VERSION	2-11
Category: PiCPro LIBRARY PATHS	2-12
Item: PiCPro LIBRARY PATHS	2-13
Source and Library File Categories	2-13
Category: Source and Library Files	2-14
Item: Source and Library Files	2-15
Category: OTHER FILES	2-16
Item: OTHER FILES	2-17
Creating a New Project	2-18
Opening an Existing Project	2-20
Opening a Project File (.PRJ)	2-21
Opening a Compressed Project File (.G&L)	2-22
Opening a DOS Project File(.PRJ)	2-24
Opening a DOS Compressed Project File(.G&L)	2-25
Compressing a Project	2-26
Launching a Project	2-28
Printing a Project	2-29
Copying a Project (Save As)	2-29
CHAPTER 3-Working with Ladders	3-1
A PiCPro Session	3-1
Creating New Files	3-1
Opening a New Ladder File	3-2
Opening an Existing Ladder File	3-2
Opening a Recently Opened File	3-4
PiCPro Tools	3-4
Using the Pointer Tool	3-4
Zooming In and Out	3-4
Using Zoom Command from View Menu	3-5
Using Zoom Command from Navigator Toolbar	3-5
Selecting and Deselecting Items	3-6
Focus	3-6
Selection	3-7

Deselecting.....	3-7
Entering Ladder Networks	3-8
Inserting a Network	3-8
Right Mouse Click Method	3-8
Menu Pull-Down Method	3-8
Hotkey Method	3-8
Toolbar Method	3-8
Inserting a ST network with focus in a ST element.....	3-9
Inserting a LD network with focus in a ST element	3-10
Inserting a ST network with focus in a LD element	3-10
Overview - Cut, Copy, Delete and Paste	3-11
Delete	3-11
Deleting LD or ST Networks	3-11
Deleting LD or ST Network Elements	3-11
Deleting LD Elements or Structured Text	3-12
To use the Delete command	3-12
Cut, Copy, and Paste	3-12
Cutting Network Elements	3-12
Cutting LD Elements or Structured Text	3-12
Copying LD Elements or Structured Text	3-13
Copy Rules.....	3-13
Pasting LD Elements or Structured Text	3-13
To use the Paste command.....	3-13
Paste Rules.....	3-13
Paste Insert Rules.....	3-15
Drag and Drop	3-15
Drag and Drop Copy	3-15
Drag and Drop Copy Rules.....	3-16
Drag and Drop Move	3-16
Drag and Drop Move Rules	3-16
Drop Rules	3-17
Other Notes - Drag and Drop	3-17
Finding and Replacing	3-18
Finding and Replacing - Procedures	3-20
To search for an element in the ladder.....	3-20
To search and replace an element in the ladder	3-21
Filtering Find and Replace	3-21
Finding Duplicates	3-22
To search for duplicated elements	3-22
Saving, Closing, and Exiting	3-23
Saving Files	3-23
To save a new file	3-23
To save an existing file	3-23
To save all files open within PiCPro	3-23
Saving Files with a Different Name	3-24
To save a file under a different name	3-24

Closing Files	3-24
To close a file.....	3-24
Exiting PiCPro	3-24
To exit	3-24
Working with a Split Screen	3-25
To split the screen	3-25
To unsplit the screen	3-25
Hardware Declarations	3-26
Hardware Declaration Restrictions	3-26
Entering Hardware Declarations	3-27
To bring up the Hardware Declarations table.....	3-27
To enter hardware declarations - master rack (all CPU types).....	3-29
Viewing Hardware Declarations	3-29
Hardware Declarations Table - PiC CPU	3-30
Editing Hardware Declarations	3-36
To insert additional remote I/O expansion racks or block I/O	3-36
To delete a remote I/O expansion rack or a block I/O.....	3-36
Cutting and Copying Hardware Declarations.....	3-37
To copy an object in the hardware declarations table:.....	3-37
To cut an object in the hardware declarations table:	3-38
Pasting and Paste Insert in Hardware Declarations	3-38
Using the Paste command.....	3-38
Using the Paste Insert After command	3-38
Things to note about pasting objects in the hardware declarations table	3-39
Pasting to Software Declarations	3-39
Printing Hardware Declarations	3-39
To print your hardware declarations.....	3-39
Closing and Saving Hardware Declarations	3-39
Closing and saving the hardware declarations table.....	3-39
Software Declarations	3-40
Entering Software Declarations	3-40
Names and Long Names of Variables	3-41
Names	3-41
Long Names.....	3-41
Reserved Keywords.....	3-42
I/O Points	3-43
Standalone MMC I/O Points.....	3-43
Fast Inputs.....	3-44
MMC for PC I/O Points.....	3-45
ASIU I/O.....	3-45
MMC for PC Fast Inputs	3-46
Numbering	3-46
PiC CPU.....	3-46
Master Rack, PiC CPU	3-46
Expansion Rack I/O.....	3-47
Master Rack Standalone MMC CPU.....	3-47

ASIU I/O for MMC for PC CPU	3-47
Block Expansion Rack I/O	3-48
Blown Fuse Status	3-48
Short Circuit Detection	3-49
Fast Inputs	3-49
PiC CPU.....	3-49
Standalone MMC CPU	3-49
MMC For PC CPU	3-49
Initial Values	3-50
Prefixes for initial values	3-50
Working with Data Types in Software Declarations	3-50
Bitwise	3-51
Numeric	3-52
String	3-53
Time	3-54
Time of Day	3-54
Time duration.....	3-54
Function/Function Block	3-55
Groups of data - Structures and Arrays	3-55
Arrays.....	3-56
To declare an array in the software declarations table	3-56
To resize an array in the software declarations table	3-56
To remove an array from the software declarations table	3-57
Structures	3-57
Attributes	3-57
Retentive Attribute.....	3-57
Global Attribute	3-58
Both Global and Retentive Attribute	3-58
External Attribute	3-58
In and Out Variable Attributes for UDFBs.....	3-58
Editing Software Declarations	3-59
Inserting/Deleting Software Declarations	3-59
To insert software declarations	3-59
To delete software declarations	3-59
Cutting, Copying, and Pasting Software Declarations	3-60
To cut software declarations	3-60
To copy software declarations	3-61
To paste software declarations in another table.....	3-61
Reserved Keywords when pasting.....	3-61
Duplicate Symbols when Pasting	3-62
Searching in the Software Declarations Table	3-64
Find by Name Only.....	3-64
Find by Type.....	3-65
Find by I/O Point	3-65
Using the Find Next Command	3-65
Purging Unused Variables from the Software Declarations Table	3-65

To purge unused variables	3-65
Working with Networks and Network Elements	3-66
Network Size	3-66
LD Network Size	3-66
ST Network Size	3-67
Network Labels in Networks	3-68
To assign or edit a label on a network	3-68
Comments in Network Elements	3-68
Add comments to your ladder	3-68
Choose the number of comment lines to display	3-68
Constants and Variables in Network Elements	3-69
Constants	3-69
Variables	3-69
Variable Names in Networks	3-70
LD Elements	3-71
Contacts	3-71
Normally Open Contact	3-72
Normally Closed Contact	3-72
Normally Open Positive Transition Contact	3-72
Normally Closed Positive Transition Contact	3-72
Normally Open Negative Transition Contact	3-72
Normally Closed Negative Transition Contact	3-72
Coils	3-73
Energize Coil	3-73
Deenergize Coil	3-73
Set (latch) coil	3-73
Reset (unlatch) coil	3-73
Wires	3-74
Functions/Function Blocks in LD Network	3-75
Data	3-75
Enable Function/Function Block Connections	3-75
Jump in LD Network	3-76
To insert a jump in your ladder	3-77
ST Elements and Structured Text	3-77
Expressions	3-77
Expressions - Operators and Precedence Order	3-78
Comments in ST Elements	3-80
Assignment Statements	3-81
Conditional Statements	3-82
IF-THEN	3-82
IF-THEN-ELSE	3-83
ELSIF-THEN-ELSE	3-84
CASE	3-85
Iteration Statements	3-86
WHILE-DO	3-87
FOR-DO	3-88

REPEAT-UNTIL	3-91
Other Statements in ST	3-92
EXIT	3-92
RETURN	3-93
Inserting Conditional and Iteration Statements in ST	3-93
Rules for a Structured Text Template	3-94
Functions in ST	3-95
Function Blocks in ST	3-97
ST Network Considerations	3-98
Variables in ST Networks	3-99
Adding Variables to a ST Network	3-99
Adding Undefined Variables to Software Declarations - ST Network	3-99
Checking Syntax in an ST Element	3-102
Using the Right Click Menu in an ST Element	3-102
Compiling and Downloading	3-103
Compiling	3-103
Downloading	3-103
Compiling a Bin File	3-104
Compile and Download	3-104
Compile Only	3-105
Compiling a Hex File	3-106
To Compile a Hex File	3-106
Compiling a Task	3-107
To Compile a Task	3-107
Compiling a UDFB	3-108
To Compile a UDFB	3-108
Settings	3-109
Ignore direct I/O	3-109
Force soft bit memory	3-109
Generate Map File (Main Ladder Only)	3-109
Operator interface	3-110
Extended Data Memory Feature	3-110
Tips on Working with Extended Data Memory	3-111
Animating the Ladder	3-112
To Turn Animation On	3-112
To Turn Animation Off	3-113
Forcing & Viewing Variables	3-114
Entering Variables in the Force List	3-115
Copy/Paste	3-115
Drag-n-Drop	3-115
Right Click Menu in LDO	3-116
Manual Entry	3-116
Turn Forcing/Grouping On/Off	3-116
Turn Forcing/Grouping On	3-116
Turning On Forcing/Grouping from the On-Line Menu	3-116
Turning On Forcing/Grouping from the Tool Bar Buttons	3-117

Turning Off Forcing/Grouping from the On-Line Menu.....	3-117
Updating the control after Force List is edited	3-117
Viewing Enabled Variables	3-118
Copy/Paste	3-118
Drag-n-Drop.....	3-118
Right Click Menu in LDO	3-119
Manual Entry	3-119
Turning Animation On in the View List	3-119
Controlling the Scan	3-120
Stopping the Scan	3-121
To Stop the Scan	3-121
Running One Scan	3-121
To Run One Scan.....	3-121
Doing a Hot Restart	3-121
To do a Hot Restart	3-121
Doing a Warm Restart	3-122
To do a Warm Restart.....	3-122
Doing a Cold Restart	3-122
To do a Cold Restart	3-122
On-line Editing	3-123
To perform an on-line edit or patch	3-124
To abort a patch	3-125
Printing	3-126
Printing the Ladder	3-126
To print the ladder file	3-126
Selections in the Ladder Network Section of the Print Dialog	3-126
Include in Listing	3-127
Cross Reference	3-127
Full Comments.....	3-128
Selections in the Software Declarations Section of the Print Dialog	3-128
Include in Listing	3-128
Selections in the Hardware Declarations Section of the Print Dialog	3-128
Include in Listing	3-128
Printing View List, Force List, or Information Window	3-129
To print the view list, force list, or information window.....	3-129
Printout Symbols	3-129
Troubleshooting the Print Option	3-130
Building a Dependency List	3-130
To build a dependency list	3-130
Comment Import / Export	3-131
Importing Comments	3-131
Exporting Comments	3-132
Communications	3-132
Communications Settings	3-132
Communication Status	3-134
Time Read / Set	3-134

Downloading a Hex File	3-135
To Download a Hex file for a PiC or Standalone MMC	3-135
To Download a Hex file to an MMC for PC	3-136
Configure Application / RAMDISK Size	3-137
Update CPU's Firmware	3-137
Update TCP/IP Module Firmware	3-138
Update SERCOS Module Firmware	3-138
Download Ladder Hex File	3-138
Clear Flash Memory	3-138
Clear Application Memory	3-138
To clear application memory with any MMC CPU.....	3-139
Backing Up and Restoring User Programs	3-140
Backing Up User Programs	3-140
Restoring User Programs	3-141
PiC Restore	3-141

CHAPTER 4-Servo Setup and Tuning **4-1**

Servo Setup Overview	4-1
Accessing Servo Setup	4-2
Create a New Servo Setup File	4-2
To Open an Existing Servo Setup File	4-3
To Open file from the setup function in your ladder	4-5
To Open file from Windows explorer	4-5
Inserting an Axis in Servo Setup	4-6
Servo Axis Limitations	4-6
To insert a Servo Axis	4-7
Input / Output Types	4-8
D/A Output Setup	4-9
Encoder Input Setup	4-10
Resolver Input Setup (Only for PiC CPU)	4-11
Analog Input Setup (Only for PiC CPU)	4-11
TTL Input Setup (Only for PiC CPU)	4-12
SERCOS Setup	4-13
Stepper Setup (PiC CPU only)	4-15
Default axes using Auto Fill (Standalone MMC)	4-16
To insert a Time Axis	4-17
Editing a Servo Setup File	4-18
Servo / Digitizing Axis Setup Data Categories	4-19
Entering Scaling Data	4-20
Entering Iterator Data for Servo Axis	4-23
Entering S-Curve Data for Servo Axis	4-27
Entering Iterator Data for Digitizing Axis	4-29
Entering Position Loop Data for Servo Axis	4-29
Time Axis Setup Data Categories	4-32

Entering Iterator Data for Time Axis	4-32
Entering S-Curve Data for Time Axis	4-33
Importing / Exporting Axis Data	4-35
Exporting Axis Data	4-35
Importing Axis Data	4-35
File Format for Servo Data File	4-36
Copying Axis Information	4-39
Saving a Servo Setup File	4-41
To save a new file	4-41
To save an existing file	4-41
To save a file in a format readable by a previous version of PiCPro	4-41
Printing Axis Information	4-42
To print axis information	4-42
Making a Servo Setup Function	4-42
To create a servo setup function	4-42
To initialize setup data in your ladder	4-43
Converting a Servo Setup File CPU Type	4-44
Axis Tuning	4-45
Axis Tuning using Forcing	4-45
To force an axis servo variable in the Servo Force list.....	4-45
Axis Tuning using Viewing	4-46
Axis Tuning Variables	4-47

CHAPTER 5-PiCPro and SERCOS **5-1**

SERCOS Setup Background Information	5-1
SERCOS Functions	5-3
Using Standard Motion Functions and Variables	5-4
Using SERCOS Functions/Blocks with TASKS	5-5
SERCOS Telegrams	5-6
IDNs	5-7
Working with Procedure Command Function IDNs	5-8
SERCOS Phases	5-9
Comparing Cyclic Data and Service Channel Transfers	5-10
Service Channel Background Information	5-11
Examples of LDO Design to Access the Service Channel	5-12
SERCOS Setup Overview	5-15
Replacing Your Analog System with SERCOS	5-15
Using SERCOS Advanced Features	5-16
Axis Positioning and Referencing in SERCOS	5-17
Opening SERCOS Setup	5-18
To create a new SERCOS setup file	5-18
To open an existing SERCOS setup file	5-18
To open an existing SERCOS setup file from the setup function in a ladder	5-19
To open an existing SERCOS setup file from Windows explorer	5-19

Inserting/Editing SERCOS Rings	5-20
Inserting/Editing SERCOS Slaves	5-21
Edit Startup IDN List	5-23
Inserting/Editing IDNs	5-24
Define Cyclic Data	5-25
Cyclic Data Structures	5-27
Define Operation Modes	5-28
Operation Mode Application Note	5-29
Copying SERCOS Data	5-30
Saving a SERCOS Setup File	5-31
Printing SERCOS Setup	5-32
Making a SERCOS Setup Function	5-32
Upload Drive Information	5-34
Changing IDN Data and Uploading IDNs Note	5-35
Converting a SERCOS setup file's CPU Type	5-36
Copying an SRS structure to the clipboard	5-38
Battery Box	5-39
Using the Battery Box	5-39
Receive IDNs	5-41
Receive Continuous IDN Data	5-42
Inserting/Editing IDNs for Receive Continuous	5-43
Send IDNs	5-44
Inserting/Editing IDNs for Send	5-45
Execute Procedure Command	5-46
Ring State	5-47
Slave Status/Control	5-49
IDN Lists	5-50
Options for IDN Display	5-50
IDN Lists	5-51
Finding a Specific IDN in an IDN List	5-52
Printing IDNs from an IDN list	5-53
Viewing Variable Length Data from an IDN List	5-54
Specifying the SRS for Viewing the Drive IDN List	5-55
Troubleshooting SERCOS	5-56
Communication Problems	5-56
Control Problems	5-56
CHAPTER 6-Working With Tasks and UDFBs	6-1
UDFBs	6-1
Creating a UDFB	6-1
Naming the UDFB	6-2
Scanning UDFB Logic	6-2
Contents of the UDFB LDO module	6-3
Size of UDFB Libraries	6-3

Data Handling in UDFBs	6-4
Compiling the UDFB	6-4
Editing a UDFB	6-4
Viewing a UDFB	6-5
UDFB Software Declarations	6-6
Naming the Variables	6-6
Order of UDFB Inputs and Outputs	6-6
Number of UDFB Inputs and Outputs	6-7
Marking UDFB Inputs and Outputs	6-7
Tasks	6-9
Comparing UDFBs and Tasks	6-9
Types of Tasks	6-9
Using the Task Template	6-10
Errs at the ERR output of the task	6-10
Using Functions from the I/O and Motion Libraries	6-11
Comparing Declaration Rules for Main and Task LDOs	6-12
Creating a Task	6-13
Naming the Task	6-14
Task Hierarchy	6-14
Interlocking Data in Tasks	6-17
Setting up Semaphore Flags	6-17
Semaphore Flag Example 1 - Data Transfer from Task LDO to Main LDO	6-17
Semaphore Flag Example 2 - Data Transfer from Main LDO to Task LDO	6-19
CHAPTER 7- DeviceNet - Ethernet - TCP/IP	7-1
G&L DeviceNet Configuration Software	7-1
Overview of setting up a DeviceNet Network	7-1
Procedure for using the G&L DeviceNet Configuration Software	7-2
Ethernet - TCP/IP Configurator	7-6
Overview of setting up an Ethernet Network	7-6
Ethernet-TCP/IP Configuration Procedure	7-7
CHAPTER 8- G&L Profibus Configuration Software	8-1
Overview of setting up a Profibus Network	8-1
Procedure for using the G&L Profibus Configuration Software	8-2
APPENDIX A - Quick Reference to In/Out Function Data	
Function Input/Output Data Type Reference	A-1

APPENDIX B - Errors

Function Error Codes	B-1
General Function Errors	B-1
I/O Function Block Error Codes	B-1
String Function Errors	B-5
Servo C-Stop, E-Stop, and Programming Error Codes	B-8
Servo C (controlled) - stop errors	B-8
Servo E (emergency) - stop errors	B-9
Servo P (programming) - errors	B-10
Servo Timing Errors	B-11
SERCOS Error Codes	B-12
Ring Errors	B-12
Slave Errors	B-14
Compile Error Messages (Ladder)	B-16
Dependency List Error Messages	B-19
Library Error Messages	B-19
Communications Error Messages	B-20
Hardware Declarations Error Messages	B-27
Ladder Error Messages	B-32
Fieldbus Error Messages	B-82
Servo Setup Error Messages	B-83
SERCOS Error Messages	B-95

APPENDIX C - PiCPro Reference Card - Errors/Variables

Reference Card	C-1
----------------------	-----

APPENDIX D - Stepper Reference Card

Reference Card	D-1
----------------------	-----

APPENDIX E - Diagnostic LED Error Codes

Error Codes	E-1
-------------------	-----

APPENDIX F - IBM ASCII Chart

ASCII Chart	F-1
-------------------	-----

APPENDIX G - Time Axes

Using a Time Axis	G-1
Referencing a Time Axis	G-1
Controlling Time Axis Velocity	G-1

Command Velocity (Variable 6)	G-1
Rollover on Position with a Time Axis	G-2
Synchronizing Slave Axes with a Time Axis	G-2

APPENDIX H - Stepper Axis Module Notes

Introduction	H-1
--------------------	-----

APPENDIX I - Toolbar Buttons

Standard Toolbar	I-1
Basic Online Operations Toolbar	I-2
Advanced Operations Toolbar	I-2
Ladder Toolbar	I-3
View Navigator Toolbar	I-4
Functions Toolbar	I-4
Compiler Toolbar	I-4
Structured Text Tools Toolbar	I-5
Insert New Network Toolbar	I-5

APPENDIX J - Module Filenames

APPENDIX K - Service Pack Installation

Service Pack Installation	K-1
Service Pack Update Installation - Manual Method	K-1
Service Pack Installation - Automatic Mode	K-5

Application Note 1

Reading and Writing STRINGS from a Structure	AN-3
--	------

INDEX	IND-1
--------------------	--------------

CHAPTER 1 Getting Started with PiCPro for Windows

Introduction to PiCPro for Windows

A PiC (Programmable Industrial Computer) and MMC (Machine and Motion Control) system each provide an integrated solution to the logic, motion, process, operator interface, and communications requirements found in today's industrial automation applications. PiCPro for Windows is the software package that allows you to program the PiC or MMC to run your application(s). It includes the following programs:

Program	Function
Ladder Diagram	Allows you to program control logic in both Ladder and Structured Text and save it as a .LDO file
Servo Setup	Allows you to program setup information for your servo application and save it as a .SRV file
SERCOS Setup	Allows you to program setup information for your SERCOS application and save it as a .SRC file
Project	Allows you to manage an application through development and production phases and then maintain the application

If PiCPro is new to you, you will soon discover how easy it is to write your application program using the interactive tools available in PiCPro. There are a tutorials on the CD you receive that may be helpful.

If you have used PiCPro in the past, you will see how the new tools and enhanced features in this Windows version enable you to work more effectively and efficiently in designing your application program(s).

Structured Text as used in PiCPro provides powerful data handling such as arrays and structures that contain different data types. It provides control structures for the following statements: IF-THEN-ELSE, CASE, FOR, REPEAT, and WHILE loops.

Included on the PiCPro CD are the Function/Function Block Reference Guide, the PiC900 Hardware Manual, the Standalone MMC Hardware Manual, and the MMC for PC Hardware Manual.

PiCPro for Windows Professional Edition



PiCPro for Windows Professional Edition is the full featured member of the PiCPro for Windows product line. It allows you to program PiC controls and MMC controls to run your application(s).

The title bar of the application indicates its edition.

PiCPro for Windows Standalone MMC Edition



PiCPro for Windows Standalone MMC Edition allows you to program only the Standalone MMC control to run your application(s).

The title bar of the application indicates its edition.


Note: You cannot program an MMC for PC with this edition of the software. You must use the Professional Edition to program the MMC for PC.

PiCPro for Windows Monitor Edition



PiCPro for Windows Monitor Edition is a subset of the PiCPro for Windows Professional Edition software. The Monitor Edition allows you to view, print, and monitor the execution of an application program for the PiC/MMC family of controls. It can also perform a PiC Restore.

Generally speaking, you cannot create new files or edit existing files or compile and download.

Project files are the one type of files that the Monitor edition can create and edit. You create a new project by selecting **File | New** from the main menu or by pressing the  button. A new project template is then automatically created and displayed for you. You may then make the desired edits.

The title bar of the application indicates its edition.

Customer Services

Giddings & Lewis is committed to providing customers with high-quality technical support. The following sections describe the support services available.

Principal Technical Support Services

North American customers may reach an experienced Giddings & Lewis engineer 24 hours a day, every day of the year, by dialing 1-800-558-4808. Over 90% of all problems are solved via telephone within a few hours.

Hands-on Training Schools

Giddings & Lewis offers training schools that provide training with PiC products. Classes are conducted at Giddings & Lewis or at your location. Students will receive thorough orientation and hands-on experience from our qualified instructors.

Before Calling Technical Support

Before calling Giddings & Lewis Technical Support, please have the following information available. This will assist the Technical Support representative in helping you more quickly and efficiently:

- A brief description of the problem, including the text and number of any error messages received, and the steps to recreate the problem.
- The type of computer and monitor you are using.
- The version of Microsoft Windows and PiCPro in use. Choose the About Windows command from the Help menu in Explorer to find which version of Windows you are running.

World Wide Web Site

The World Wide Web address for Giddings and Lewis on the internet is **<http://www.giddings.com>**. At this location you can search, read, print, or download information and service packs.

The Computer Workstation

The PiCPro for Windows software runs on IBM PC compatible platforms. The recommended requirements for a workstation are:

Recommended For a PiC or Standalone MMC Controller:

Computer	A 486 or Pentium processor with Windows 95, 98, NT4.0, 2000 or ME
Memory	32 MB of RAM, minimum; 64 MB of RAM, recommended
Video	800 x 600 or higher, 256 or higher color recommended
Disk drives	Typically, 60 MB of hard disk space required
Serial communications port to controller	RS232 port (COM1 or COM2)

Recommended For an MMC for PC Controller:

Computer	A 133 MHz Pentium processor with Windows NT4.0 or 2000 with PCI network card (ISA network card is not recommended)
Memory	64 MB of RAM, minimum; 128 MB of RAM, recommended
Video	800 x 600 or higher, 256 or higher color recommended
Disk drives	Typically, 60 MB of hard disk space required

Note: Power saver option in BIOS should not be set when using an MMC for PC because doing so will cause a communications error when the computer is shut down.

Hardware Connections

A cable is supplied with the PiCPro package. One end of the cable is labeled “PiC900” and the other “Computer”. Connect the “Computer” end to the serial port you specify in the **Online | Comm Settings** dialog box and connect the “PiC900” end to the PiCPro Port on the controller. The pin-out for the controller serial port is given in the appropriate Hardware Manual. The pin-out for the serial port should be in the computer’s manual.

Infrared Communications Drivers

If your system has Infrared Communications Drivers, you need to be aware that under the Windows default settings, COM1 and COM3 use the same interrupts and COM2 and COM4 use the same interrupts. The communications settings option in PiCPro for Windows, when checking for valid communications ports, may initiate execution of some Infrared Drivers. Communications problems will occur if the Infrared Driver is using one member of the above pairs and PiCPro for Windows is using the other. To avoid communication problems, do one of the following:

- Disable the Infrared Driver in the control panel or
- Change the communications port used by the Infrared Driver so that it does not conflict with the communications port used by PiCPro.

Installing PiCPro for Windows

PiCPro for Windows is available on CD.

To install from the CD, follow the instructions on the inside cover of the CD.

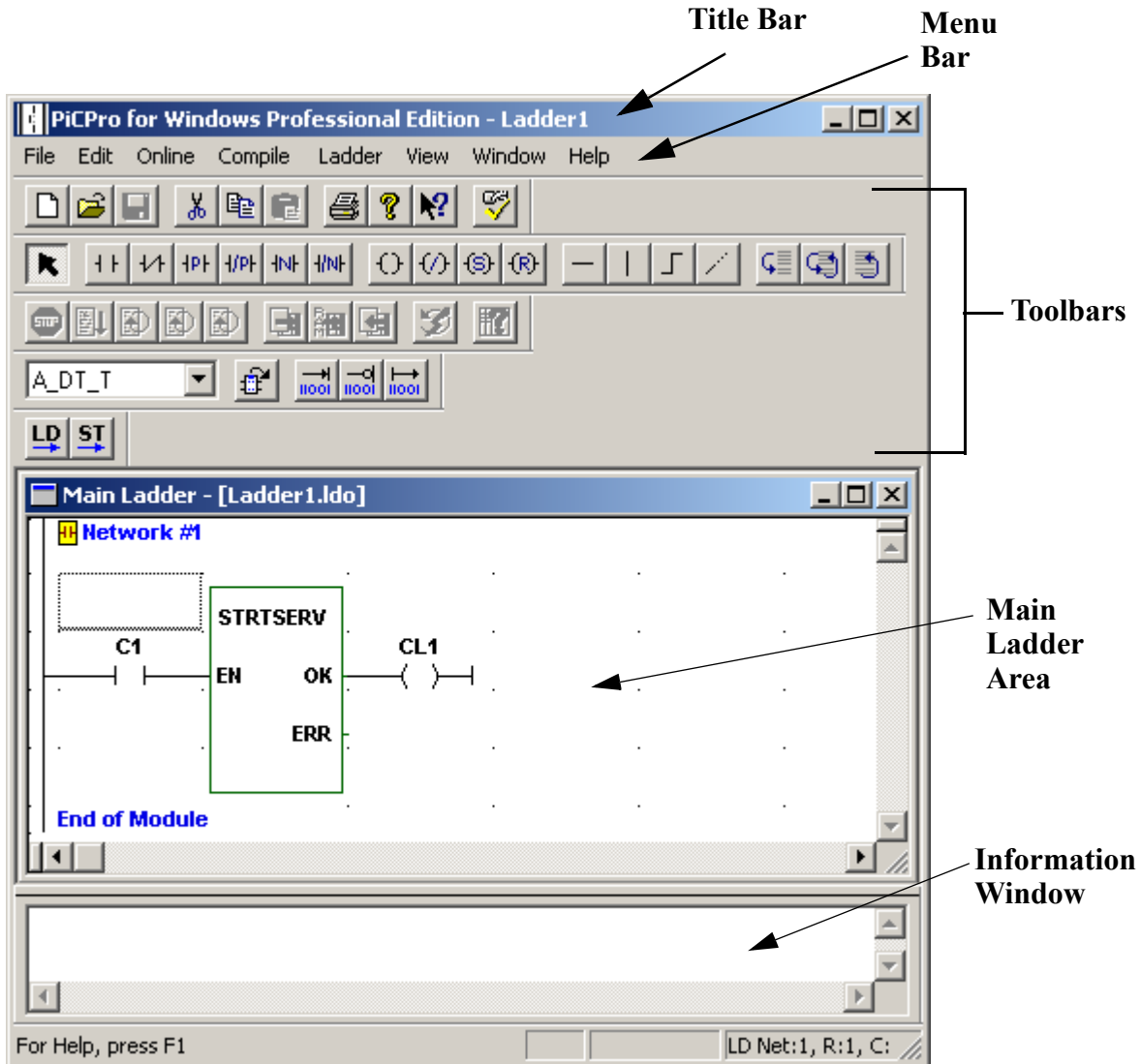
For Windows 95 Computers: PiCPro V13.0 requires Winsock.dll version 2.2 or higher. Most newer computers or operating system software (such as Windows 2000) use this file. To determine the Winsock.dll version on the computer that PiCPro V13.0 is installed on:

1. Use Windows Explorer and from the **T**ools menu, select **F**ind.
2. Select **F**iles or **F**olders from the flyout menu.
3. In the **N**amed: box, type in winsock.dll
4. In the **L**ook in: box, select C: and make sure **I**nclude **s**ubfolders is checked.
5. Right click the Winsock.dll file(s) that are displayed in the lower window.
6. Select **P**roperties, and then click the **V**ersion tab. The file version number is shown.

Newer Winsock.dll versions can be located on the Microsoft website.

The Work Area in PiCPro for Windows

When you open a file, its window appears in the work area. This is where you work on developing a new file or editing an existing file. An information window appears at the bottom of the work area. You can open multiple files and easily move between them in the work area.



The Main Ladder Area

The main ladder area is where you create or edit your LDO file. Usually only part of your file can be displayed on your screen and you have to scroll up and down or left and right to bring portions into view.

The Information Window








The information window provides a read only area that PiCPro uses to display data pertinent to your application. It is displayed when you open a file and anytime data has been placed in it by a process within PiCPro. The information remains in the window until it is replaced by data from another process, the file it pertains to is closed, or PiCPro is closed. With certain data (i.e. ladder compiler error messages) you can go to the location of the error in your file by double clicking on the message in the information window.

Other features of the information window include:

- You can undock the information window by dragging it away from the PiCPro edge it is docked to.
- You can dock the information window by dragging it to any edge within PiCPro.
- You can resize it vertically or horizontally, but not diagonally.
- You can copy the contents of the information window to the clipboard by selecting some or all of the text, right clicking, and then selecting **C**opy.
- You can print the contents of the information window by putting focus in the Info Window and doing a **F**ile | **P**rint.
- You can hide the information window from view in various ways:
 - Pressing <Esc>
 - Choosing **V**iew | **I**nformation Window from the menu
 - Right clicking the mouse and choosing **H**ide
- To view a hidden information window, choose **V**iew | **I**nformation Window.

Status Bar

The status bar appears at the bottom of the PiCPro screen and provides the following information

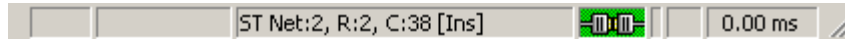
Indicator	Description
<text at far left>	Concise help information relevant to the selected window component.
	Indicates that a project is open and that the PiCPro Version and Edition configured for the project matches the one currently running. This icon has a green background.
	Indicates that a project is open and that the PiCPro Version and Edition configured for the project does NOT match the one currently running. This icon has a red background.
LDO CFG I/O	Indicates the Ladder Configurable I/O status: if LDO CFG I/O is displayed, then this option is enabled.
LD Net or ST Net: , R: , C:	The Structured Text (ST) network, row and column in the ladder that the cursor is in.
	PC is connected to control (configured for Ethernet-TCP/IP communication). This icon has a green background.
	PC is NOT connected to control (configured for Ethernet-TCP/IP communication). This icon has a red background.
	PC is connected to control (configured for serial communication). This icon has a green background.
	PC is NOT connected to control (configured for serial communication). This icon has a red background.
IP:	For Ethernet-TCP/IP communication, the IP address of the control.
IP: r	This is the same as IP: except that the “r” indicates relay mode
	This symbol appears only when forcing is on.
0.00ms	Dynamically updates with the scan time when the scan is running.

Example 1:



In Example 1, the PC is not connected to a control, but is configured to connect serially. The scan is not running (scan time is 0.00ms) and the focus in the ladder is in **LD Network #3, Row 1, Col 3**.

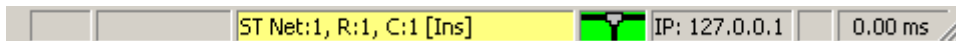
Example 2:



In the example status bar above, the PC is not connected to a control, but is configured to connect serially. The scan is not running (scan time is 0.00ms) and the focus in the ladder is in **ST Network #2, Row 2, Col 38**, and is in **Insert** mode.

Note: The element can be in either Insert (Ins) or Overstrike (OVR) mode. In insert mode, characters are inserted at the cursor position and existing characters past the inserted character are pushed to the right. In Overstrike mode, inserted characters will overwrite any characters already at that position.

Example 3:



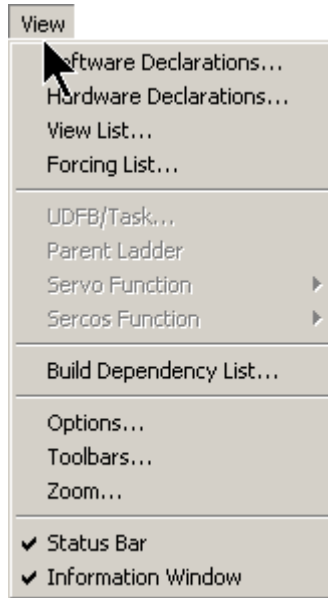
In Example 3, the PC is connected to a control (configured for Ethernet-TCP/IP communication), the scan is not running (scan time is 0.00ms) and the focus in the ladder is in **ST Network #1, Row 1, Col 1**, and is in **Insert** mode.

The color of “ST Net” turns yellow as the number of lines in the ST element reach the limit of 500 lines.

Displaying/Hiding the Status Bar

You can choose to display or hide the status bar.

- Choose **V**iew | **S**tatus Bar from the main menu. A check mark indicates that the status bar will be displayed.
- Click on Status Bar again to remove the check mark and hide the status bar.




Window Elements

If you are familiar with Windows, you will recognize basic screen elements common to most Windows applications. If you are new to Windows, this section will inform you about some of these elements.

The Title Bar

When you start PiCPro, a Title Bar extends across the top of the window. It displays the edition of PiCPro you are using, the name of the file you are working in and indicates whether it is the active window or not. The Title Bar of the active window is highlighted. The Title Bar of the inactive windows are dimmed.

You can drag the Title Bar to reposition the window within the work area. Clicking on the icon on the left end of the Title Bar drops down a menu that allows you to **R**estore, **M**ove, **S**ize, **M**inimize, **M**aximize, or **C**lose PiCPro.

There are three buttons on the right end of the Title Bar.  Use the first to minimize the window, use the second to maximize the window to fill the entire screen, and use the third to close PiCPro.

The Menu Bar

Directly below the Title Bar is the Main Menu Bar. It contains the names of the menus that group together similar commands. Clicking a menu name displays a list of commands that can be used to access PiCPro functions.

When PiCPro is started, the Main Menu Bar contains the names shown below.



File Online Compile View Help

Once you open a ladder file, the Menu Bar is expanded to include more items needed to program your ladder:





File Edit Online Compile Ladder View Window Help

Dialog Boxes

Dialog boxes can appear when you choose a menu item or a command. A dialog box presents the list of options available for the selected command and requires you to type in information or select options that PiCPro requires to proceed. Typically, you will see some or all of the following in a dialog box.

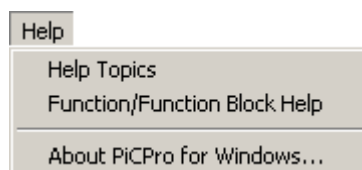
- OK button
- Cancel button
- Option button
- Check boxes
- Display boxes
- List boxes
- Spin boxes
- Tabs
- Edit fields
- Browse button

When using a dialog box, you can use the question mark button in the top right corner, to display **What's This?** help on a specific dialog component by doing the following:

1. Click on the  button. The cursor changes to a question mark: 
2. Click on the component you for which you want help.

Help Files

You can access Help files by using the **Help** menu. There you will have two types of Help available.



Help Topics will offer information on how PiCPro works, error messages, and general help topics.

Function/Function Block Help offers specific help on Functions/Function Blocks, ASFBs, and you can even add your own for UDFBs.


Additional details can be found later in this chapter.

Customizing PiCPro

PiCPro has several features that let you set up your PiCPro work space in a unique way for the way you want to work.

You can:

- Set preferences
- Choose animation color
- Customize the toolbars
- Choose Structured Text Keyword, Comment, Number, and Template colors.

Options for user preferences and color can be set under **View | Options** from the menu or by pressing the  button from the Standard toolbar.

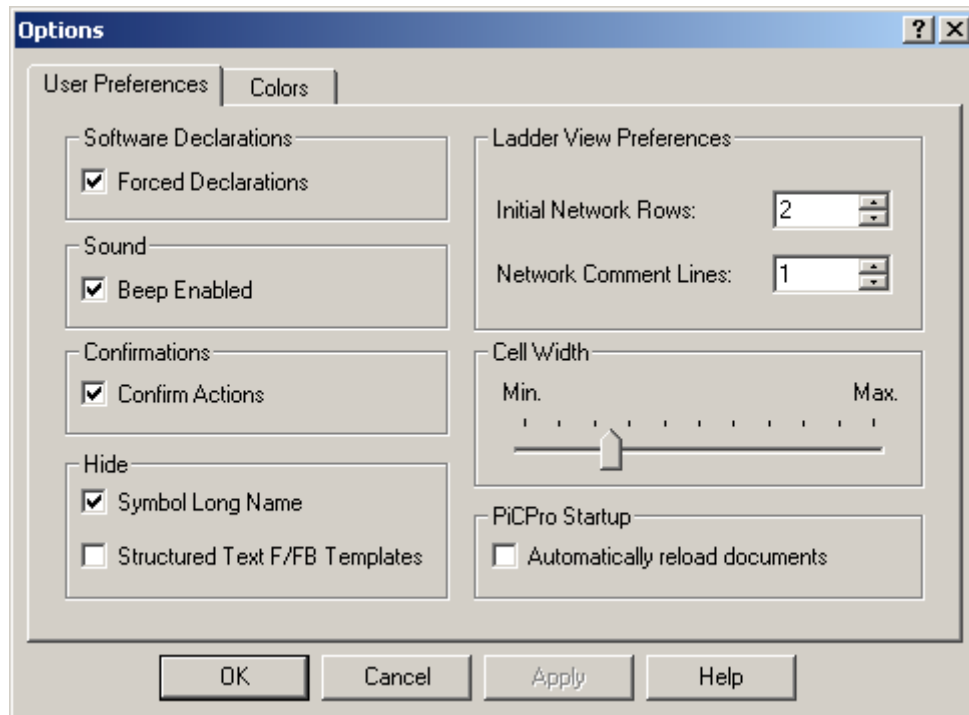
User Preferences

Under the User Preferences tab, you can set the following:

1. **Software Declarations - Forced Declarations:** Whether or not you want a message to appear that tells you to enter a software declaration for the ladder element you have just added to your ladder.
2. **Sound - Beep Enabled:** Whether or not you want a beep to sound after certain actions.
3. **Confirmations - Confirm Actions:** Whether or not you want a confirmation message to appear after certain actions.
4. **Hide:**
 - Symbol Long Name:** Whether or not you want to view long names you have entered in your program. Note that when you choose to view the long names all the cells in your ladder are enlarged.
 - Structured Text F/FB Templates:** Whether or not you want Function/Function block templates shown when you drop a Function/Function Block into a Structured Text element.
5. **Ladder View Preferences - Initial Network Rows and Network Comment Lines:** The number of network rows and the number of comment lines you want to view in your ladder.
6. **Cell Width:** How wide the cells in the networks appear on your screen.

7. **PiCPro Start-up - Automatically reload documents:** When you restart PiCPro, whether or not you want to reopen any documents (.ldo, .srv, .src) opened using **File | Open** or **File | New** that were still open when you exited PiCPro.

Note: Files that were opened using **View | UDFB/Task...**, **View | Servos Function** or **View | Servo Function** will not be reopened. The **Options** box below shows all the default settings on the User Preferences page.



Color

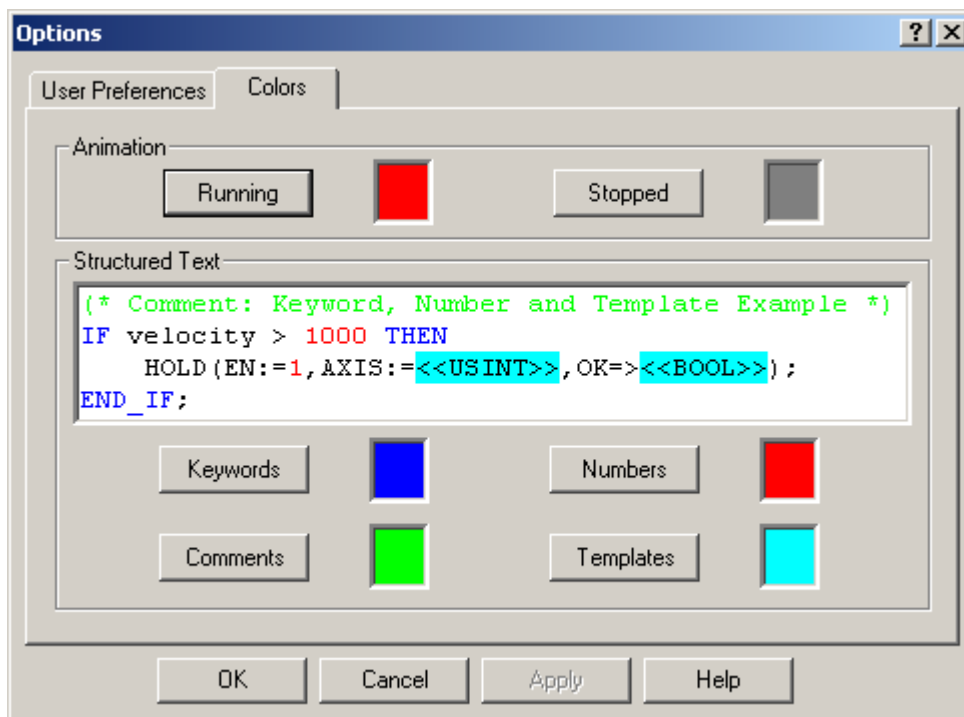
The Animation section shows the colors that will outline power flow when animation is on. The **Running** color is used when the control is scanning. The **Stopped** color is used when the scan is stopped.

In the **Structured Text** section, the colors used to display Keywords, Comments, Numbers and Templates are shown.

As changes are made to each of the ST element types, a sample of Structured Text code is displayed to preview how the changes will look. The default colors for **Keywords** is blue, for **Comments** green, for **Numbers** red, and for **Templates** cyan.

To change a color

1. Click on the associated button.
2. Choose a solid color from the Color box that appears. Click **OK** to save your choices.



Customizing Toolbars

The toolbars give you quick access to some of the frequently used tools. You can enable or disable toolbars from **View | Toolbars** under the main menu bar.

The type of toolbars available on-screen depends on the type of file opened on-screen. For example, if an LDO (Ladder) file is open and viewable on-screen, and **View | Toolbars** is selected from the menu, as many as nine different Toolbars are available. If there is no file open on-screen, only the Standard and Basic Online Operations toolbars are available. The tools available under each of the toolbars are listed under each toolbar title in Appendix I.

There is a lot of flexibility in displaying and arranging toolbars to fit your needs. Toolbars can be:

- Arranged within a toolbar region
- Docked horizontally or vertically on any or all four sides of the PiCPro screen
- Docked inside or outside the PiCPro screen
- Resized

In working with your PiCPro application you will discover which toolbar arrangement works best for you. Your final toolbar arrangement will become the default until you change it.

Moving Toolbars

When you install PiCPro or when you choose to display a toolbar from the menu, the toolbars appear at the top of your PiCPro screen under the menu bar in a toolbar region. Within this region you can move the toolbars by the drag and drop method. The toolbars are dockable and can be positioned in a variety of ways.

Note: The toolbar region that holds the toolbars that are displayed along any of the four sides of your PiCPro screen disappears when all the toolbars have been docked somewhere else.

Examples of Positioning the Compile Toolbar:

Docked Horizontally



Along the top or bottom of your PiCPro screen

Docked Vertically



Along the left or right edge of the PiCPro screen

Undocked (Labeled)



Anywhere within the PiCPro screen or on the desktop

Note: When the toolbar has a label, it is possible to resize it and close it.

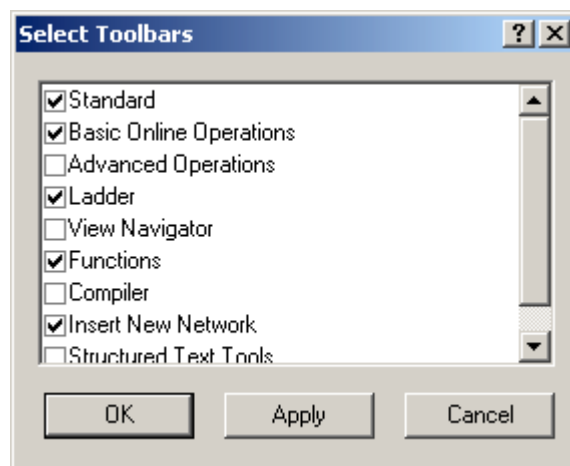
Docking Tips

- Always click in the dead space around a button to move the toolbar.
- When you want to dock the toolbar vertically, be sure that the edge of the PiCPro screen is in view. The orientation switches as you cross the edge of the PiCPro Screen.

Displaying/Hiding Toolbars

You can choose to display or hide any of the toolbars available in PiCPro.

- Choose **V**iew | **T**oolbars from the main menu. The **Select Toolbars** box appears. Click the box in front of the toolbar name to display it. A check mark appears in the box.
- Click on the box again to remove the check mark and hide the toolbar.



Organizing your PiCPro Files

When you install PiCPro for Windows, the default directory structure is:

C:\Program Files\Giddings & Lewis\PiCPro for Windows Vxx.x *editionname* Edition

- where Vxx.x is the version number (e.g. V13.0) and *editionname* is the name of the Edition (e.g. Professional). Hereafter, this directory is referred to as the PiCPro program directory.

Subdirectories under the Program Directory

Libraries - contains the standard function and function block library files.

Firmware - contains hex files.

Organizing

One method of organizing your files when working with PiCPro is to create a directory structure similar to the following

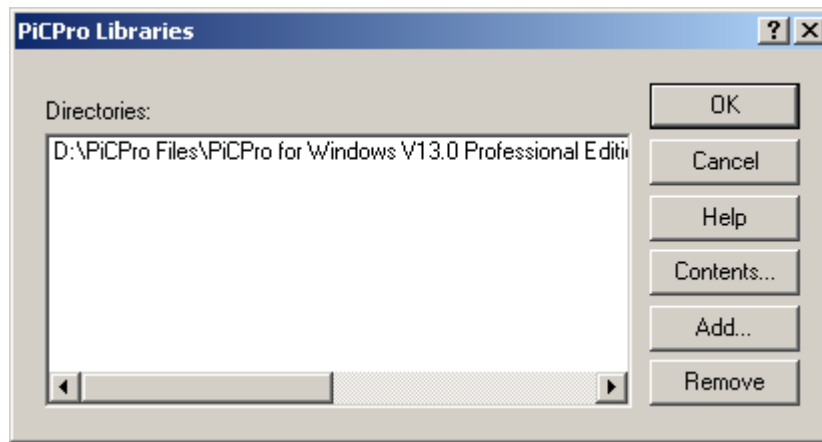
C:\PiCApplications\project1	to hold a project, keeping it separated from other projects
C:\PiCApplications\project1\myUDFBs	to hold any User Defined Function Blocks you may create.
C:\PiCApplications\project1\ASFBS	to hold any Application Specific Function Blocks you may be using.

To avoid confusion when installing new versions of PiCPro or uninstalling old versions, it is recommended that you NOT create your directories under the PiCPro program directory structure (where PiCPro is installed).

PiCPro Libraries

All PiCPro functions and function blocks are stored in libraries. There is a standard set of libraries that is installed with the PiCPro software. There are also libraries that can be created by the user to hold user-defined function blocks, set up functions, etc. You can add and remove the paths to these libraries using the dialog box found under **File | PiCPro Libraries**.

Note: To be able to compile with a function/function block, whether it's a standard one or a user-written one, the path to its library must be configured in the **PiCPro Libraries** list of directories. You can access this list directly by selecting **File | PiCPro Libraries** from the main menu or, if you are using a project, by adding directories from there.

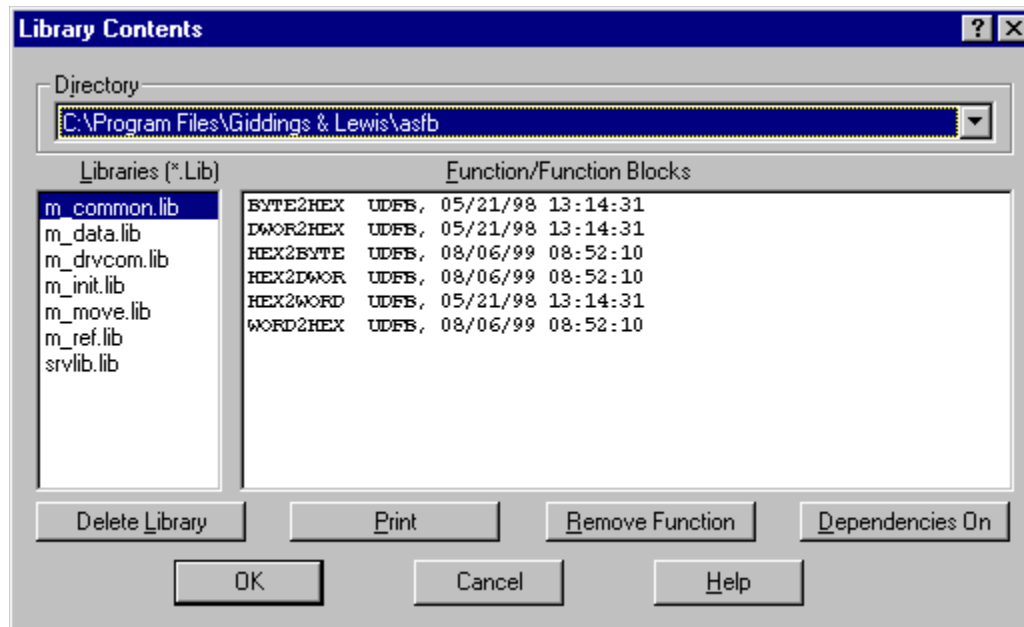


Library Contents

You may want to view a library's contents or delete libraries. You may also remove individual functions from within a library. Deleted libraries and removed functions are removed from the Function list in PiCPro (**Ladder | Functions**).

Note: You cannot delete any of the standard PiCPro libraries or functions.

With the **PiCPro Libraries** dialog box displayed, choose the **C**ontents button. The **Library Contents** box appears.



1. In the **D**irectory section, select the desired library path from the drop down list.
2. The list of libraries in that path will appear on the left in the **L**ibraries (*.Lib) section. Highlighting a library causes the functions included in that library to appear on the right in the **F**unction/**F**unction Blocks section.
3. To delete the entire library, highlight the library and choose the **D**elete **L**ibrary button. **Note:** The library is not deleted until the **O**K button is clicked.
4. To delete one or more functions from a library, highlight the function and choose the **R**emove **F**unction button. **Note:** The function is not deleted until the **O**K button is clicked.
In the **F**unction/**F**unction Block section, the name of the function/function block is listed followed by the type (standard, UDFB, SERCOS, servo, or internal) and the timestamp. The **D**elete **L**ibrary and **R**emove **F**unction buttons will be grayed if you highlight a function that is either a standard or internal (hidden) type or highlight a library that contains a standard or internal function.
5. If you want to print a list of all the libraries in the directory along with the functions in each, choose the **P**rint button. You can also choose to **P**rint to a **F**ile from within the print dialog box.
6. If you want to display the dependencies, if any, for each function/function block, choose the **D**ependencies **O**n button. The dependencies will be displayed right below the function name entry. With this button on, the dependencies will be included in the print out or print file if you choose to print.




Using On-line Help

The Help documentation includes an on-line Help system, a context-sensitive (What's This? system), and Function/Function Block Help system. Some of the ways you can access the Help system are by choosing it from the menu, by pressing <F1>, or by clicking on the question mark.

Using What's This? Help

The What's This? system displays context-sensitive information that is relevant to the component that you click on. The What's This? help appears in a popup box with a pale yellow background.

To use What's This? Help:

1. Click on the  button if no dialog is open, otherwise, click on the  button in the top right corner of the dialog box.
2. The cursor changes to: 
3. Click on the component that you want information on.
4. What's This? Help is displayed. It automatically disappears as soon as you click somewhere.

Retrieving On-line Help

The on-line Help system enables you to retrieve information you need quickly and return to your work within PiCPro. The Help appears in a separate window on your screen. You can keep the Help window displayed on top of your PiCPro application for quick access. You can also print Help topics.

Printing On-line Help

When running the on-line Help system, you can print specific topics or print entire sections from the **Contents** tab.

To print a specific Help topic

- Click the **P**rint button that appears along the bottom right-side of the Help window.
or
- While displaying the topic, right click the mouse and click **P**rint Topic

To print an entire section of Help

1. Display the **Contents** tab of the Help system and highlight the section you want to print.

Click the **P**rint button that appears along the bottom right-side of the window.

Note: Printing help windows that contain graphics using Windows 95 and Windows 98 may produce unexpected results. The problem does not occur when using Windows NT 4 or Windows 2000.

Using Function/Function Block Help

This Help system provides information on standard Functions and Function Blocks and also for ASFBS. You can access this Help for a specific Function or Function Block in the following ways:

- Click the **question mark** button and then click on the desired Function/Function Block in your ladder.
- Right click on the desired Function/Function Block in your ladder and then click **H**elp on the popup menu.

You can also start the Function/Function Block Help from the main menu: **H**elp | **F**unction/**F**unction **B**lock

Creating Function/Function Block Help

The Function/Function Block Help system also enables you to write help for your own UDFBs. Just do the following:

1. Make sure your UDFB's library is in your library path. (**F**ile | **P**iC**P**ro **L**ibrar**i**es)
2. Open the Function/Function Block Help system
3. Right click on your UDFB in either the **C**ontents list or **I**ndex list on the left side of the help window.
4. Select **E**dit. At the prompt, click **Y**es to create a new file.
5. Proceed with adding any plain text descriptions for your UDFB. The diagram of inputs and outputs is automatically generated for you based upon the inputs and outputs you enter in the corresponding tables that are provided. The initial information provided is based on the information in the library for this UDFB.
6. Click **O**K or **C**ancel. In either case, if you have changed anything, you are asked whether to save. This is to provide you one last chance at not losing any edits.
7. When you're finished, the help system takes your information and generates an html file named <your library name>.htm and deposits it the same directory as your library. **Note:** If you ever move or copy your library to a new location, you also need to move or copy it's corresponding .htm file. All of the UDFBs for a specific library are documented in that library's one .htm file. The more UDFBs you have in a library, the bigger the .htm file and the longer it takes to load when you want to view or edit it.
8. Once you've created help for your UDFBs, you can access it exactly the same way that you access the standard Function/Function Block Help.

Note: The help files for the standard libraries and the ASFB libraries are not .htm files, but rather .chm files. These are compiled html files. The same rule applies regarding their location and naming. They are named to match their corresponding library: <library name>.chm. Example: arith.chm goes with arith.lib. Both files must exist in the same directory, so if you're moving your libraries around, move their help files with them.

Starting PiCPro

Once you start PiCPro, you'll be able to create or edit your ladder program for your application.

With PiCPro running, you can easily switch between PiCPro programs (Ladder Diagram, Servo Setup, and SERCOS Setup). You can go back and forth between programs without exiting one and launching the other.

To begin working in PiCPro, start the PiCPro application.

To start PiCPro

1. Click the **Start** button in the Windows task bar.
2. Click the PiCPro for Windows entry in **Programs**.
3. Click the desired version and edition of PiCPro.

CHAPTER 2 Working with Projects

Overview: Using Projects

A project is a tool to help you manage your applications. A project consists of a project file (.prj) and, optionally, a compressed project file (.g&l). A project file contains descriptions of all the files (and their paths) that you have determined encompass your project. A compressed project file actually contains, in compressed format, a copy of all of the files described by name in the project file. PiCPro for Windows enables you to create, edit, compress, or print the project file as required. There is only one project described in a project file.

Once the files in the project have been defined in the project file, you can work on the development of your application using the tools found in PiCPro, Servo Setup, and SERCOS Setup. A compressed file (.g&l) containing all the files identified in the project file can be created. This file can be stored on the PC or on the FMSDISK on the control. **Note:** Use the DOS 8.3 format naming convention (8 characters, period, 3 characters) for compressed project files if you will be saving those on FMSDISK.

A project can play a role in simplifying your control maintenance. You can maintain the control by using a PC with only the PiC programming tools installed. All the files required for your application can be stored in the PiC on the FMSDISK. When you open a project from the FMSDISK, the compressed file is loaded onto the PC. The compressed file is expanded onto your PC according to the options you select. You can then use the software tools to maintain the project and then compress back to the FMSDISK (or even the PC).

A summary of the capabilities of project maintenance using PiCPro for Windows is listed below:

- Allows you to control the development of a project in a development, production, and archive version.
- Allows you to work with multiple versions of PiCPro.
- Simplifies creating a single disk containing all the files required for an application.
- Provides the tools needed to store a single compressed file containing all the files required for an application on the FMSDISK in the control.
- Allows you to update the FMSDISK on the control without stopping the scan.

Menu Commands that Manage Projects

The following is a list of commands from the main menu bar that enable you to manage your projects.

File | **N**ew – creates a new project tree. This project tree consists of various categories. To develop your project, you must add file definitions or other information to the applicable categories. **Note:** A project tree and a project file represent the same data; a project tree is the graphical representation of the project characteristics and file definitions in the opened project file.

File | Open – Opens an existing project. You can open a project file (.prj) or a compressed project file (.g&l). Opening a compressed project file uncompresses its contents to locations in your PC according to the options you select. Once open, you can edit, print, and compress a project file. The contents of an open project file is presented to you graphically as a project tree.

File | Open From Control - Opens a compressed project file on the control's FMSDISK.

File | Print – Prints the contents of an open project file.

File | Save As – Copies existing files defined in your project file to different directories according to your specifications and creates a new project file containing those new file definitions. After a **Save As**, the new saved-as project is the one that is left open.

File | Compress – Compresses the files defined by name in your project file into a single compressed project file (.g&l) and stores it on the control's FMSDISK, on the PC's hard disk or mapped network drive according to your specifications.

Note: Use the DOS 8.3 format naming convention for compressed project files if you will be saving those on FMSDISK.

File | Launch Project - Launches the version and edition of PiCPro for Windows as defined in the PiCPro VERSION category in the project tree.

File | Uppdate Project Tree - Updates the project tree file definitions based upon the file dependencies found in the main .LDO in the project. Update removes all items in the UDFBs/TASKs, SERVO SETUP FILES, SERCOS SETUP FILES and PROFILE SETUP FILES categories and adds file definitions based upon the determined dependencies AND the project's properties. If you manually add file definitions to any of these 4 categories, they could be lost on an update.

File | Validate Project Tree - Validates the project tree by verifying that all files and paths defined in the tree exist. If not found, the icon next to the file or path definition is altered to have a red circle with a diagonal slash through it.

Edit | Add - Enables you to add an item to a selected category such as LIBRARY PATHS or UDFBs/TASKs. Refer to the specific category or item description for information on the results because they depend upon which category or which item is selected.

Edit | Delete - Enables you to delete all items of a selected category or if an actual item is selected, enables you to delete that item from its category. Refer to the specific category or item description for information on the results because they depend upon which category or which item is selected.



Edit | Cut - Cuts the highlighted item from its category and places it on the clipboard. Refer to the specific category or item description for information on the results because they depend upon which category or which item is selected.

Edit | Copy - Copies the highlighted item or category of items to the clipboard. Refer to the specific category or item description for information on the results because they depend upon which category or which item is selected.

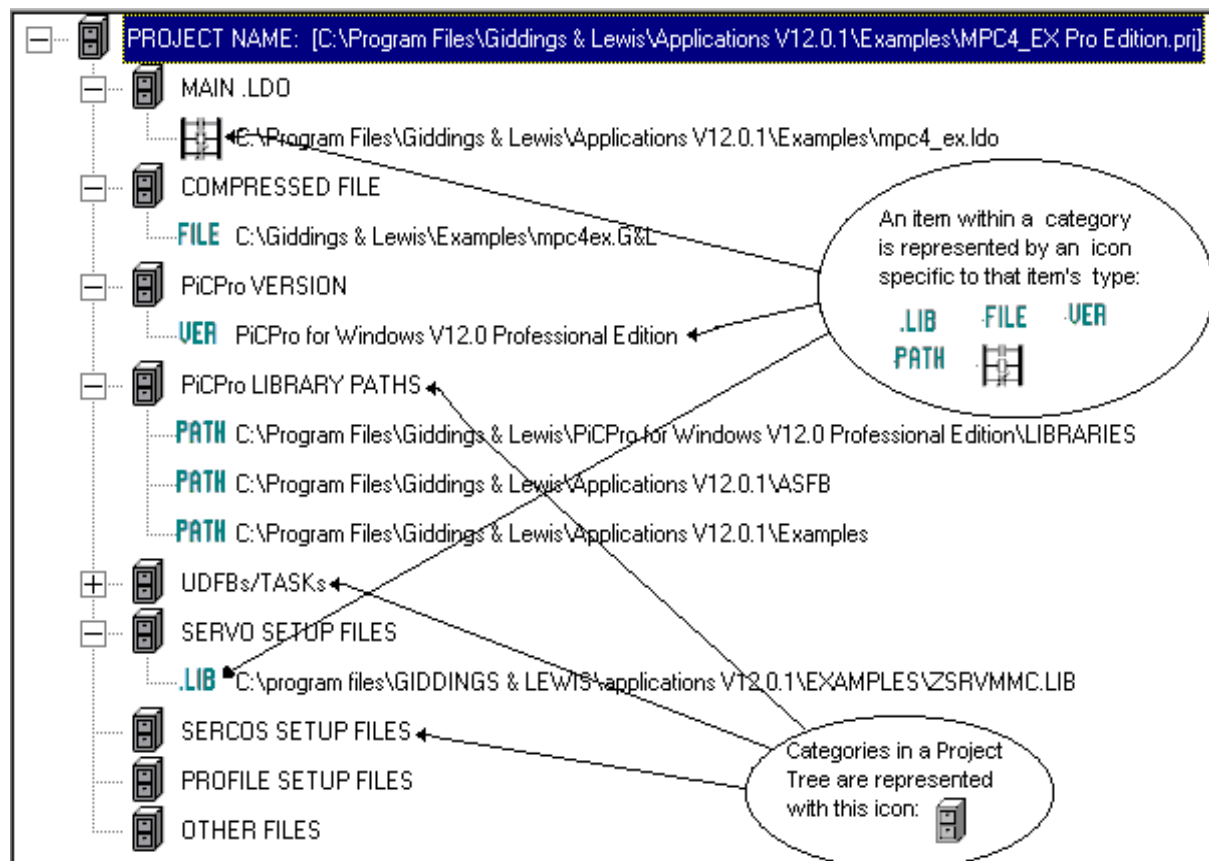
Edit | Paste - Pastes the contents of the clipboard into the selected category. Refer to the specific category or item description for information on the results because they depend upon which category or which item is selected.

The Project Tree and Its Components

A project tree is the graphical representation of the information configured in the project file. The project tree is displayed when you open a project file. It consists of several categories that are made up of items that describe different characteristics of the project or describe files and file locations that belong to the project.

You can see the items in a category by expanding the category. Do this by clicking on the  symbol to the left of the category name. If the symbol is a , click on it to collapse the category. You can also double click on the category name itself to toggle between expanded and collapsed. If neither symbol is displayed, no items are currently configured in that category.

Select a category or item in a category by single clicking on it. When selected, the category name or item is highlighted. Once highlighted, you can use main menu commands to manipulate an item or category or right click and use those menu items.



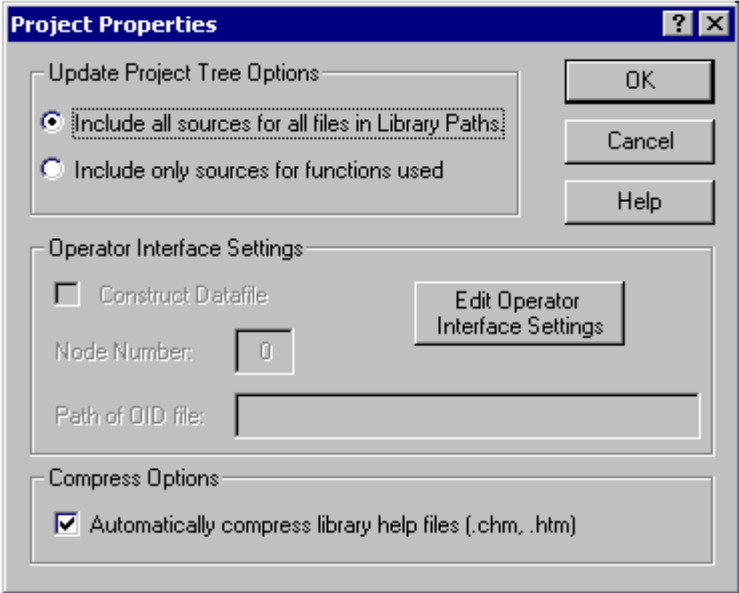
Project Tree Categories

Category: PROJECT NAME

The PROJECT NAME category defines the name of the project file. It's format is *<projectname>.PRJ* where *<projectname>* is whatever name you choose for this particular project. The path (or location) for the project file is also displayed with the name.

If you right click on the PROJECT NAME category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Compress	Allows you to compress the project file. This is the same as the command selected from F ile on the main menu.
Print	Allows you to print the information in the project tree. This is the same as the command selected from F ile on the main menu.
Launch Project	Allows you to start up the edition/version of PiCPro configured in the project if it is installed on your PC. This is the same as the command selected from F ile on the main menu.
Update Project Tree	Allows you to automatically update the entire project tree based upon dependency information. This is the same as the command selected from F ile on the main menu.
Validate Project Tree	Allows you to validate the contents of the tree categories. This is the same as the command selected from F ile on the main menu.

Popup Menu Selection	Description
<p>Properties</p>	<p>Enables you to change the properties for this project.</p>  <p>Update Project Tree Options:</p> <p>If you want to include all of the sources (all .LIB, .LDO, .SRV, .SRC, .PRO files) for all of the files found in the paths configured in the PiCPro LIBRARY PATHS category, then check the top option. This is the default option.</p> <p>Otherwise choose Include only sources for functions used. This means that when you select Update Project Tree, only those sources for the actual functions used will be configured in the project file. This will result in a smaller compressed project file (should you choose to compress your project). Be aware, however, that this could limit future development or alteration of the project if working from the compressed project file. For example, lets say that you only have the compressed project file available and a problem occurs. And you want to include a new-to-this-project ASFB in your ladder to help diagnose the problem. The source for that ASFB won't be in the compressed project file and you may or may not be able to find it elsewhere.</p> <p>Operator Interface Settings:</p> <p>Your project automatically inherits the current Operator Interface Settings. You can change these settings by pressing the Edit Operator Interface Settings button or by using the Compile Settings dialog. Both methods display the same dialog.</p>

Popup Menu Selection	Description
Properties (Continued)	<p>Compress Options</p> <p>When checked, help files associated with a project's libraries will automatically be included when the project is compressed. Help files include files with .chm and .htm extensions.</p> <p>Note: Projects created prior to V13.0 and opened in V13.0 will have this option de-selected by default. Projects created in V13.0 will have this option selected by default. Projects created in V13.0 and opened in V11.0 will have the checkbox marked, but it will have no effect on projects compressed in V11.0. Help files will not be included.</p>

Category: MAIN.LDO

The MAIN .LDO category can contain one item. This is the main .LDO file (ladder file) for this project.

If you right click on the MAIN .LDO category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Add	Enables you to add a main ladder file to your project. This option is only enabled if a main .ldo file is NOT already specified.
Delete All	Enables you to delete all of the items belonging to this category. In this case, that is only one: the main ladder file. Note: the actual file is not deleted; just the entry in the project tree is deleted. You are prompted for a confirmation first.
Cut	Cuts (removes) the main ladder file item from this category and places it on the clipboard.
Copy	Copies the main ladder file item from this category to the clipboard.
Paste	Pastes the contents of the clipboard into the MAIN .LDO category. This command is only available if there is not already a main ladder file item entered for this category and if the clipboard contains a main .ldo file item.
Update Project Items under this Category	Not enabled for this category.

Item: MAIN.LDO

If you right click on an item (the main ldo file) in the MAIN .LDO category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Open	Opens the main ladder file. If this ladder is already open, it makes it the active window. You can also open this file by double clicking on it.
Delete	Removes this item, the main ladder file, from this category. You are prompted for a confirmation first.
Edit	Allows you to manually edit this item's name and path by typing in the new information.
Cut	Removes this item, the main ldo file, from this category.
Copy	Copies this item to the clipboard.
Paste	Pastes a main ladder file into this category. This is only available if a main .ldo file item is on the clipboard. This overwrites the selected main ladder file item.

Category: COMPRESSED FILE

The COMPRESSED FILE category contains a file description for this project's compressed project file, if it has one. (**Note:** Use the DOS 8.3 format naming convention for compressed project files if you will be saving those on FMSDISK.) The compressed file contains the project's characteristics like version and properties. It also physically contains, in a compressed format, a copy of all of the files described by name in the project along with their supporting files if any exist:

- .REM - comment file for an .LDO file (main ladder, UDFB, etc.)
- .FRC - force list file for an .LDO file (main ladder, UDFB, etc.)
- .RTD - view list file for and .LDO file (main ladder, UDFB, etc.)
- .SVT - force/view (tuning) information for an .SRV file (servo setup file)
- .SCT - force/view (tuning) information for an .SRC file (SERCOS setup file)

These support files do not show up in the tree, but are the "behind-the-scenes" files that are necessary to support the usage and customization of the .LDO, .SRV, and .SRC files that are named in your tree.

If no file is configured in the COMPRESSED FILE category, you will not be prompted to do a compress when you do things like edit and then save and close a project file. But if there is a compressed file specified, you will be prompted to

update the compressed file with the changes you've made (i.e. compress the project again.). During your project's development process, it may be useful for you to postpone specifying a compressed file until closer to completion in order to avoid all of the prompts to compress.

If you right click on the COMPRESSED FILE category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Add	Enables you to add a compressed project file item to your project. This option is only enabled if a compressed project file is NOT already specified.
Delete All	Enables you to delete all of the items belonging to this category. In this case, that is only one: the compressed project file item. Note: the actual file is not deleted; just the entry in the project tree is deleted. You are prompted for a confirmation first.
Cut	Cuts (removes) the compressed project file item from this category and places it on the clipboard.
Copy	Copies the compressed project file item from this category to the clipboard.
Paste	Pastes the contents of the clipboard into the COMPRESSED FILE category. This command is only available if there is not already a compressed project file item entered for this category and if the clipboard contains a compressed project file item.
Update Project Items under this Category	Not enabled for this category.

Item: COMPRESSED FILE

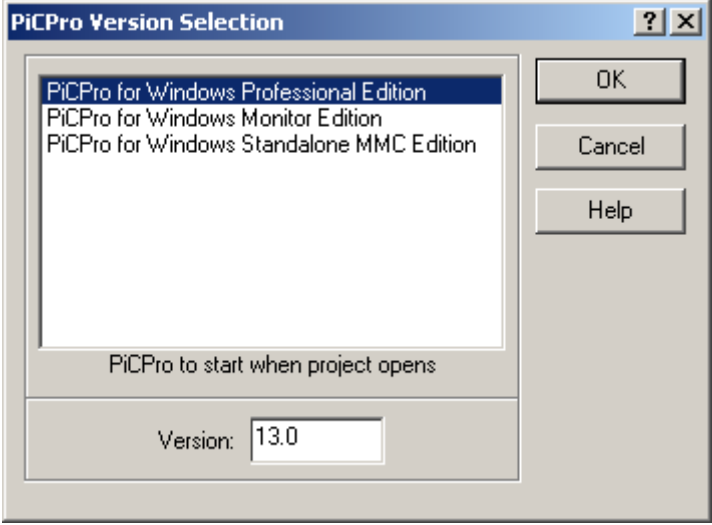
If you right click on an item (the compressed project file) in the COMPRESSED FILE category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Open	Not enabled for this category. Instead, to open a compressed file, use File Open and select a compressed project. This will close the currently open project and uncompresses the selected compressed project according to your specifications.
Delete	Removes this item, the compressed project file, from this category. You are prompted for a confirmation first.
Edit	Enables you to edit the compressed project file item, not the compressed project file itself. You may type in a new path and/or compressed project file name.
Cut	Removes this item, the compressed project file, from this category.
Copy	Copies this item to the clipboard.
Paste	Pastes a compressed project file into this category. This is only available if a compressed project file item is on the clipboard. This overwrites the selected compressed project file item.

Category: PiCPro VERSION

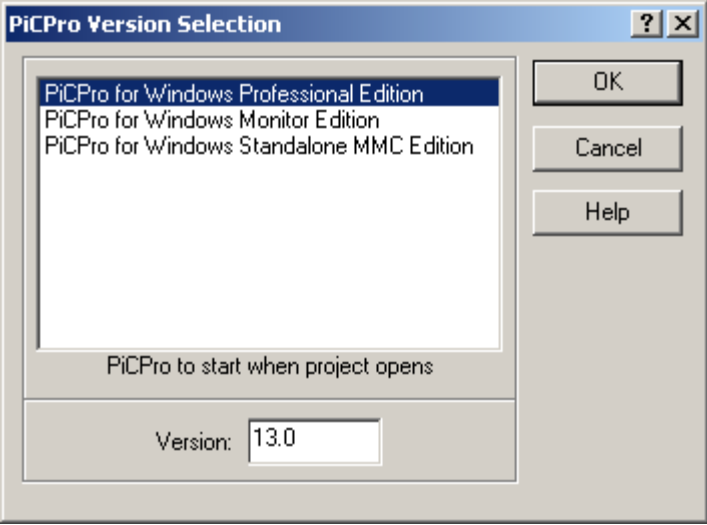
The PiCPro VERSION contains an item which specifies which version and edition this project is built for. If you open a project in a version or edition other than what is specified in its VERSION category, then you will not be able to compile and download the main ladder or any other ladder. In addition, you will not be able to compile servo or SERCOS functions. You must **Launch** the correct version to do so (**File | Launch Project**). Of course, you can only launch a particular version/edition of PiCPro if it is actually installed on your PC.

If you right click on the PiCPro VERSION category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Add	<p>Enables you to add a PiCPro version/edition item to your project by displaying the PiCPro Version Selection dialog. This dialog allows you choose both the edition and version of PiCPro that you want associated with this project. This option is only enabled if a PiCPro version/edition is NOT already specified.</p>  <p>In this example, the edition that is chosen is the Professional Edition and the version is 13.0.</p>
Delete All	<p>Enables you to delete all of the items belonging to this category. In this case, that is only one: the PiCPro version/edition item. You are prompted for a confirmation first.</p>
Cut	<p>Cuts (removes) the PiCPro version/edition item from this category and places it on the clipboard.</p>
Copy	<p>Copies the PiCPro version/edition item from this category to the clipboard.</p>
Paste	<p>Pastes the contents of the clipboard into the PiCPro VERSION category. This command is only available if there is not already a version item entered for this category and if the clipboard contains a PiCPro version item.</p>
Update Project Items under this Category	<p>Not enabled for this category.</p>

Item: PiCPro VERSION

If you right click on an item (PiCPro version/edition) in the PiCPro VERSION category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Open	Not enabled for this category.
Delete	Removes this item, the PiCPro version/edition, from this category. You are prompted for a confirmation first.
Edit	<p>Enables you to edit the PiCPro version/edition by displaying the PiCPro Version Selection dialog. You can choose the edition and the version of PiCPro that you want associated with this project.</p>  <p>In this example, the edition that is chosen is the Professional Edition and the version is 13.0.</p>
Cut	Cuts (removes) this item, the PiCPro version/edition, from this category.
Copy	Copies this item to the clipboard.
Paste	Pastes a PiCPro version item into this category. This is only available if a PiCPro version item is on the clipboard. This overwrites the selected PiCPro version item.

Category: PiCPro LIBRARY PATHS

This category contains all of the paths for all of the libraries that your project uses. You can edit this list by right clicking on the category or an actual item in the category and selecting **Modify**. The **PiCPro Libraries** dialog is displayed. Alternatively, you can select **File | PiCPro Libraries** while a project is open. Any changes made to the library path list there will be reflected in the PiCPro LIBRARIES PATHS category of the open project.

If you right click on the PiCPro LIBRARY PATHS category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Modify	Allows you to add or remove library paths in this category. Displays the same dialog that you get when you select File PiCPro Libraries .
Delete All	Deletes all of the libraries paths under this category. You are prompted for a confirmation first.
Cut	Cuts (removes) all of the library paths from this category and places them on the clipboard. A library path on the clipboard can be pasted into the PiCPro LIBRARY PATHS category if it does not already exist there. Note: if you do a cut on this category, you'll get a warning message stating that PiCPro couldn't find any library paths. Just click OK . Remember, you will need to add some library paths back in sometime.
Copy	Copies all of the library paths in this category and places them on the clipboard. A library path on the clipboard can be pasted into the PiCPro LIBRARY PATHS category if it does not already exist there.
Paste	If the clipboard contains library paths, this pastes the contents of the clipboard into this category. If a matching path already exists there, a duplicate one is not added. This menu selection is only available if the clipboard contains library paths.

Item: PiCPro LIBRARY PATHS

If you right click on an item (a library path) in the PiCPro LIBRARY PATH category, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Edit	Allows you to manually edit the library path by typing in the new information.
Modify	Allows you to add or remove libraries paths in this category. Displays the same dialog that you get when you select File PiCPro Libraries .
Delete	Deletes the selected library path from the list. You are prompted for a confirmation first.
Cut	Cuts (removes) the selected library path from the list and places it on the clipboard.
Copy	Copies the selected library path from the list and places it on the clipboard.
Paste	Pastes the contents of the clipboard into this category, replacing the selected item. This menu selection is only available if the clipboard contains library path items.

Source and Library File Categories

These categories list the source files and the libraries for your projects. These categories are updated by **Update Project Tree** and **Update Items in this Category**. If you manually change the contents of these categories, performing an update will potentially lose those changes. Update first removes all items from a category and then based on dependencies and the Properties settings for the project, adds items back in.

The following are Source and Library File Categories for your project:

Source and Library File Category	Source File Extension	Category Description
UDFBs/TASKs	.ldo	Typically contains entries for the library files (.lib) for the tasks and function blocks for this project. Also contains entries for their corresponding source files (.ldo).

Source and Library File Category	Source File Extension	Category Description
SERVO SETUP FILES	.srv	Typically contains entries for the library files (.lib) for the Servo Setup functions for this project. Also contains entries for their corresponding source files (.srv).
SERCOS SETUP FILES	.src	Typically contains entries for the library files (.lib) for the SERCOS Setup functions for this project. Also contains entries for their corresponding source files (.src).
PROFILE SETUP FILES	.pro	Typically contains entries for the library files (.lib) for the Profile Setup functions for this project. Also contains entries for their corresponding source files (.pro).

Category: Source and Library Files

If you right click on one of these categories, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Add	Allows you to manual add libraries files (.lib) and source files to the selected category. WARNING: any manual changes to this category could be lost if you perform an Update Project Tree or Update Project Items under this Category . To add files where they won't automatically be removed on an update, add them to the OTHER FILES category.
Delete All	Deletes all of the items in the selected category. You are prompted to confirm this first.
Cut	Cuts (removes) all of the items from this category and places them on the clipboard.
Copy	Copies all of the items in this category and places them on the clipboard.
Paste	Pastes all of the items in the clipboard into the selected category. This menu selection is only available if the sources and/or library items on the clipboard are compatible with the selected category.

Popup Menu Selection	Description
Update Project Items under this Category	Performs the Update Project Tree operation on just this category of the tree, leaving all of the other categories untouched. All items in the category are removed and new entries are added based on the configured properties for this project. The properties determines whether all sources and libraries in your configured paths are added or only those that are used.

Item: Source and Library Files

If you right click on an item (source or library file) in the one of these categories, a popup menu is displayed with the following menu selections:

Popup Menu Selection	Description
Open	Opens the selected source file in the appropriate PiCPro editor. This menu selection is not available if a library is selected. Note: This command is equivalent to File Open . It is <u>not</u> the same as View UDFB/Task , View Servo Function or View SERCOS Function . You will not be able to animate the file if it is a UDFB/Task, Servo function, or SERCOS function.
Delete	Deletes the selected item from its category. The entry for this file is removed from the project; the file itself is not removed from disk. You are prompted for a confirmation first.
Edit	Allows you to manually edit this item's name and path by typing in the new information.
Cut	Cuts (removes) the selected item from its category and places it on the clipboard.
Copy	Copies the selected item and places it on the clipboard.
Paste	Pastes the contents of the clipboard into the selected item's category, overwriting the selected item. The types of items on the clipboard must be compatible with this category, otherwise this menu selection is disabled.

Category: OTHER FILES

The OTHER FILES category provides you with the opportunity to include any type of files that you want to in your project. Perhaps you have a text file or spreadsheet with relevant notes or data. Or maybe you have some alternative UDFBs you want included but you're not using yet. Or maybe its important to you to have the online help files for the UDFBs or ASFBs available. Or maybe you would like to include the Drive IDN files associated with your SERCOS functions. You can include them all here and, unlike files entered in the UDFBs/TASKs, SERVO, SERCOS, and PROFILE categories, the **Update Project Tree** command will leave them alone.

If you right click on the OTHER FILES category, a popup menu is displayed with the following menu selections:

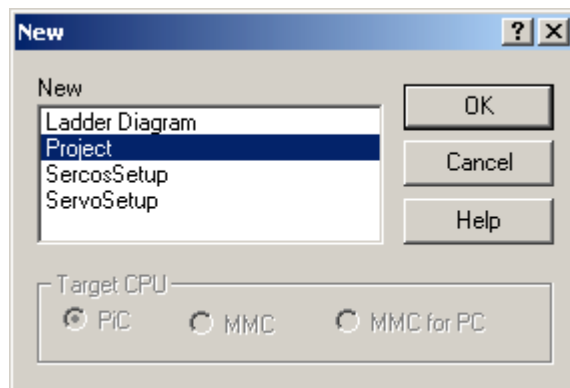
Popup Menu Item	Description
Add	Enables you to add any type of file to this category.
Delete All	Enables you to delete all of the items belonging to this category. You are prompted for a confirmation first.
Cut	Cuts (removes) all of the items belonging to this category and places them on the clipboard.
Copy	Copies all of the items in this category to the clipboard.
Paste	Pastes the contents of the clipboard into this category. This command is enabled only if the clipboard contains OTHER FILES items.
Update Project Items under this Category	Not enabled for this category.

Item: OTHER FILES

If you right click on an item in the OTHER FILES category, a popup menu is displayed with the following menu selections:

Popup Menu Item	Description
Open	If the selected file is a PiCPro source file, that file is opened in the appropriate PiCPro editor. Otherwise, this menu selection is not enabled.
Delete	Deletes the selected item from this category. You are prompted for a confirmation first.
Edit	Enables you to manually edit the name and location of the selected item.
Cut	Cuts (removes) the selected item from this category and places it on the clipboard.
Copy	Copies the selected item in this category to the clipboard.
Paste	Pastes the contents of the clipboard into this category, overwriting the selected item. This command is enabled only if the clipboard contains OTHER FILES items.

Creating a New Project



To create a new project:

1. Select **File | New** from the main menu. The **New** dialog box is displayed.

Note: If you are running PiCPro for Windows Monitor Edition, you will not be prompted with the **New** dialog box. Instead, if another project is open, you are prompted to close it, and then a new project tree is immediately created and displayed for you.

Note: In any of the PiCPro editions, if you already have a project open, you are prompted to close it. You can only have one project open at a time.

2. Choose **Project** from the list by clicking on it.

Note: The Target CPU choice is immaterial here and all choices are disabled. CPU type is not a characteristic of the project file itself; it is a characteristic of the source files configured in the project.

3. Click on **OK**.
4. A project tree is created and displayed for you.
5. Modify the project as desired.
6. Select **File | Save** to save the project file. The file name you enter becomes the name of the project.

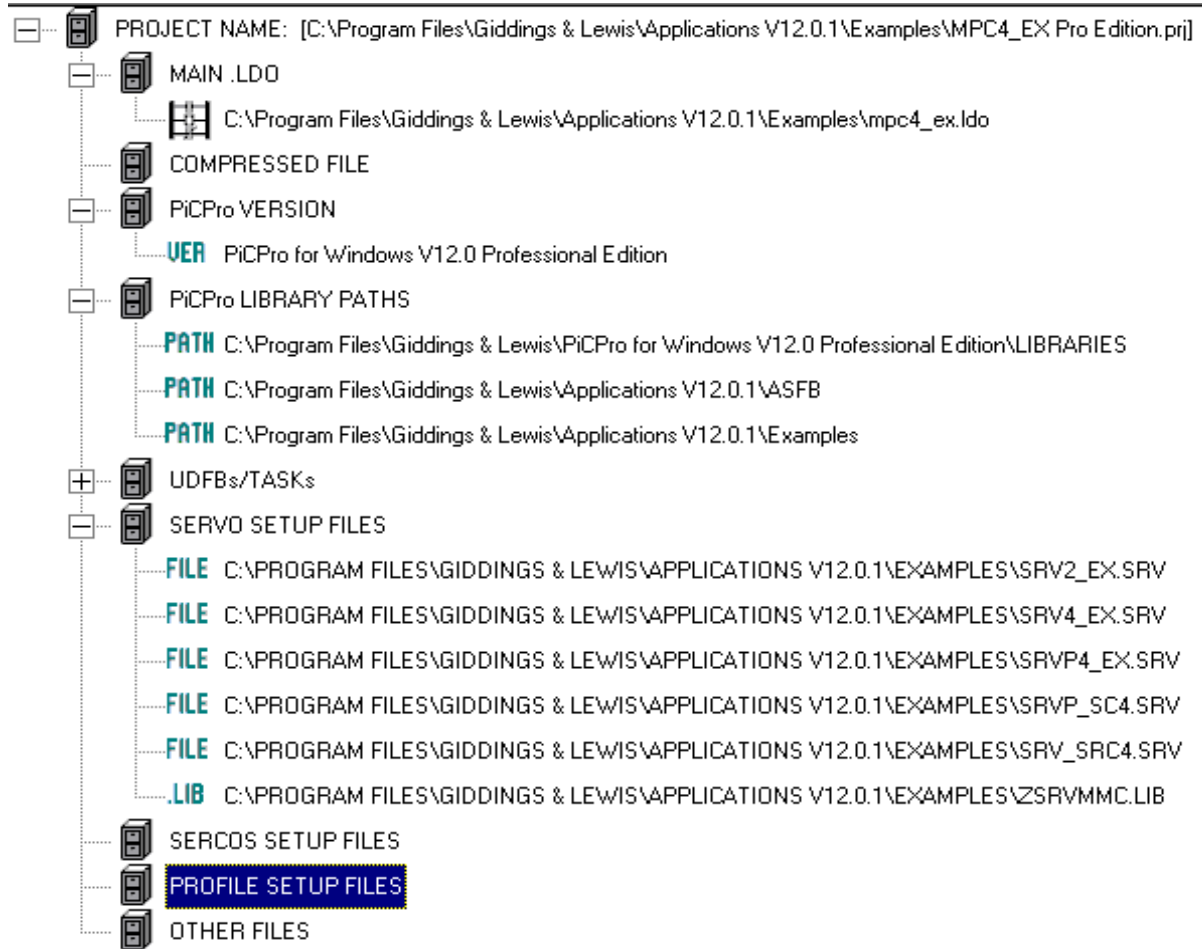
When a new project tree is displayed, all of the categories exist in the tree, but not all have default values.

- **PROJECT NAME** - No default. Specified when you save the project (**File | Save**).
- **MAIN .LDO** - If you have a ladder open and the ladder has focus when you create this new project, this category defaults to the name and location of the open ladder.
- **COMPRESSED FILE** - If you have a ladder open when you create this new project, the file item in this category defaults to <open ladder name and location>.g&l.
- **PiCPro VERSION** - Defaults to the edition and version of PiCPro that you are currently running.

- PiCPro LIBRARY PATHS - defaults to the library path list that is configured for the edition/version of PiCPro that you are running.
- UDFBs/TASKs - If you have a ladder open and the ladder has focus when you create this new project, this category defaults to include all of the dependencies for that ladder based on the project's properties, which default to ALL files in Library Paths, even if not used.
- SERVO SETUP FILES - If you have a ladder open when you create this new project, this category defaults to include all of the dependencies for that ladder based on the project's properties, which default to ALL files in Library Paths, even if not used.
- SERCOS SETUP FILES - If you have a ladder open when you create this new project, this category defaults to include all of the dependencies for that ladder based on the project's properties, which default to ALL files in Library Paths, even if not used.
- PROFILE SETUP FILES - No defaults.
- OTHER FILES - No defaults.

If an .LDO file is not opened and/or the .LDO does not have focus, the **New Project** dialog box will open with two defaults entered, the PiCPro VERSION and the PiCPro LIBRARY PATHs. All other entry fields will be blank.

You can modify or add to these defaults as applicable. Remember that any modifications made to the UDFBs/TASKs, SERVO SETUP FILES, or SERCOS SETUP FILES categories will be lost if you do an **Update Project Tree**. The OTHER FILES category allows you to enter additional files you want to include in the project and is not altered by an update.



Opening an Existing Project

A project is opened by using the **File | Open** command from the main menu. There are four types of project files that can be opened:

1. PiCPro for Windows project file (.PRJ or .prj).
2. PiCPro for Windows compressed project file (.G&L or .g&l).
3. PiCPro for DOS project file (.PRJ or .prj)
4. PiCPro for DOS compressed project file (.G&L or .g&l).

You can also open any of these from the FMSDISK, if you have one, on the control.

Opening a Project File (.PRJ)

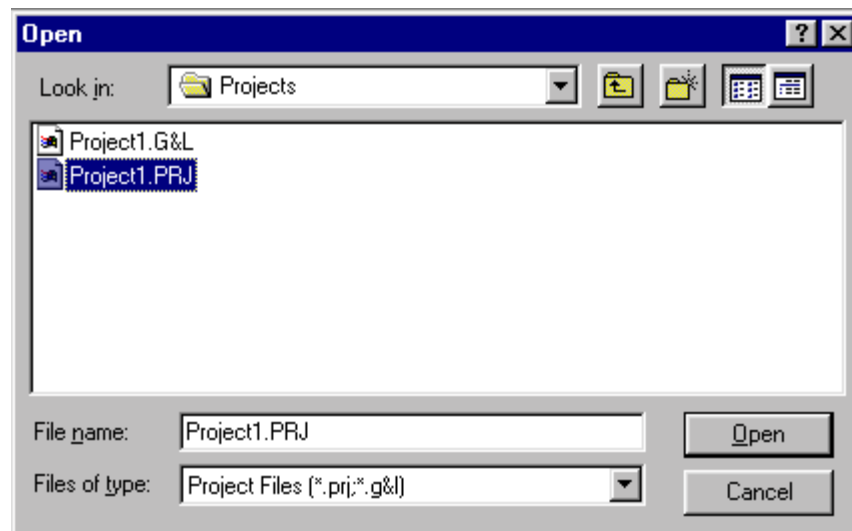
If you want to open an existing PiCPro for Windows project file:

1. Selecting **File | Open** from the main menu. Then the **Open** dialog box is displayed.
2. In the **Files of type** edit box, select Project Files (.prj,.g&l)
3. Browse to the desired project file.
4. Highlight the project file that you want to open
5. Click the **Open** button.
6. If any other project is open, you are prompted to close it because only one project can be open at a time.

Note: if the version and edition of PiCPro that is configured in the project is different than the version and edition you are currently running, you will be warned of this and can either continue opening the project or not. On the status bar, an icon with a red background indicates that the editions/versions don't match: **PRJ** . This indicates that to have full functionality, you should launch the correct edition/version of PiCPro for Windows (**File | Launch**). An icon with a green background indicates that a project is open and that the editions/versions match: **PRJ**

7. The project is now open for editing and if found, the main ladder file of the project is opened, also.

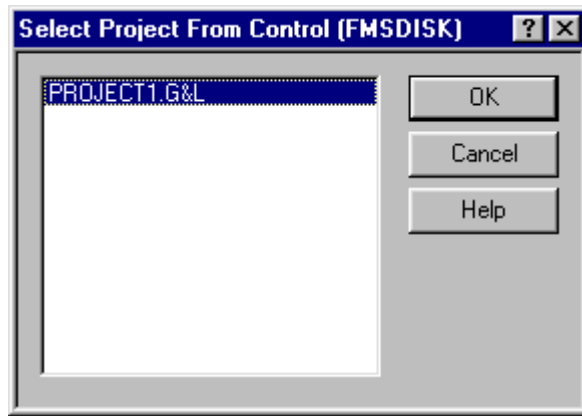
Alternatively, if the desired project has been recently open, you might find it in the “most recently used list” located near the bottom section of the **File** menu. Just select the entry that you want.



Opening a Compressed Project File (.G&L)

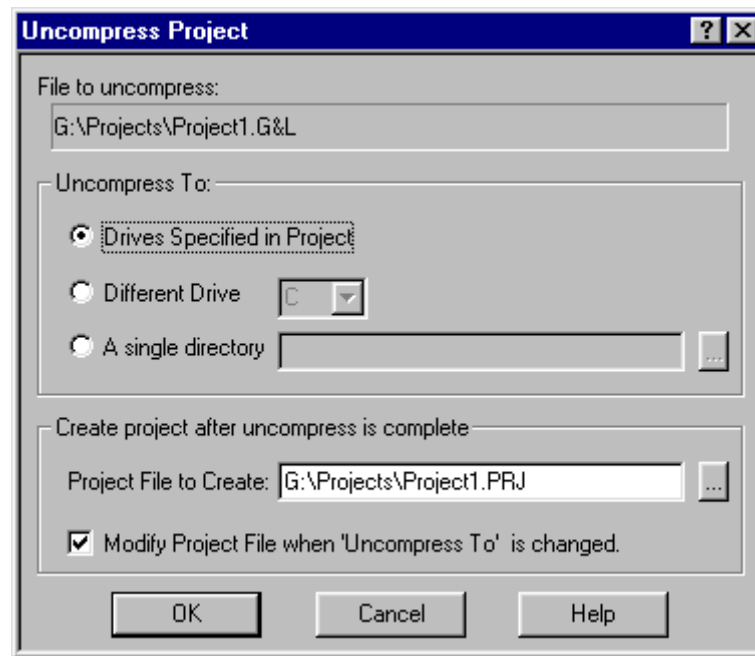
A PiCPro for Windows compressed project file that is stored on your PC or on the FMSDISK on your control can be uncompressed by opening it:

1. Select **F**ile | **O**pen from the main menu. Make sure the file type at the bottom of the dialog is set to “Project Files (.prj,.g&l)”. Browse to the compressed project file (.G&L) that you want to open and select it and click on **O**pen.
2. Alternatively, if the compressed project file is on the FMSDISK on your control, select **F**ile | **O**pen **F**rom **C**ontrol instead. A list of compressed files on the control’s FMSDISK is displayed to you. Realistically, there should only be one there. Select the file by clicking on it and then click **O**K. The file is then copied to a temporary directory on your PC before the **U**ncompress **P**roject dialog is displayed to you.



3. If any other project is open, you are prompted to close it because only one project can be open at a time.

4. The **Uncompress Project** dialog is displayed.



5. Choose the locations you want to uncompress to:
 - **Drives Specified in Project** - this option uncompresses all of the files in the compressed project file into the exact locations (drive and directory paths) that were specified in the project when the compressed file was made.
 - **Different Drive** - this option uncompresses all of the files in the compressed project file into the directory location specified in the original project file, BUT substitutes the drive with the one you select here.
 - **A single directory** - this option uncompresses all of the files in the compressed project file into a single drive and directory location. Use the button to the right to browse to the desired location or manually type in the desired location.
6. The project name will default to the uncompressed project file name with a .PRJ extension instead. You can modify the name and path.
7. If you have chosen an **Uncompress To:** option other than **Drives Specified in Project**, selecting the **Modify Project File when 'Uncompress to' is changed** will generate the new project file with file entries whose locations match your **Uncompress To:** choice. Exception: The drive and location of standard libraries in the LIBRARY PATH is not altered.
8. Click **OK**.

Note: if the version and edition of PiCPro that is configured in the project is different than the version and edition you are currently running, you will be warned of this and can either continue opening the project or not. On the status bar, an icon with a red background indicates that the editions/versions don't match: **PRJ** .

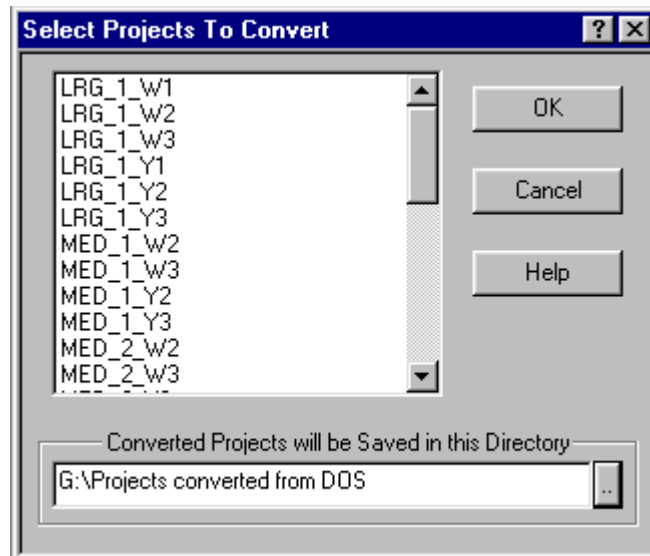
This indicates that to have full functionality, you should launch the correct edition/version of PiCPro for Windows (**File | Launch**). An icon with a green background indicates that a project is open and that the editions/versions match: **PRJ**

9. When done, a message saying “Uncompress project is complete” appears. All files have then been uncompressed to the location(s) you specified and the specified project file has been created according to your specifications and is opened for editing.

Opening a DOS Project File(.PRJ)

If you want to open an existing DOS project file:

1. Select **File | Open** from the main menu. In the **Files of type** edit box, select Project Files (.prj,.g&l)
2. Browse to the desired DOS project file.
3. Highlight the project file (.PRJ) that you want to open.
4. Click the **Open** button.
5. If any other project is open, you are prompted to close it because only one project can be open at a time.
6. Since you chose a DOS project file (.PRJ), you will be prompted whether you want to convert to the PiCPro for Windows project file format. Select **Yes** to continue opening. **Note:** PiCPro for Windows cannot save a project to the DOS format.
7. Now the **Select Projects to Convert** dialog is displayed.



8. Select one or more of the DOS project files listed. Remember that DOS project files can hold more than one project, whereas Windows project files can only hold one. Any selected file is shown as highlighted. To select a file, click on it. Alternatively, click on a selected file to deselect it. Click and drag to select

multiple, continuous files from the list. Or use Shift-click to select multiple, contiguous files from the list. Initially, the dialog has all of the files selected by default.

9. Click **OK**.
10. The DOS project(s) you chose are converted to PiCPro for Windows project file format. No projects are left open after the conversion is done.
11. To edit or look at one of these new projects, select **File | Open** from the main menu.

Note: If a DOS project uses the "@" feature to specify a list of files, those files are automatically deposited into the OTHER FILES category when the PiCPro for DOS project is converted to a PiCPro for Windows project. Any other use of the "@" feature to list files in a project is not supported in PiCPro for Windows.

Opening a DOS Compressed Project File(.G&L)

If you want to open an existing DOS compressed project file:

1. Select **File | Open** from the main menu.
2. In the **Files of type** edit box, select Project Files (.prj,.g&l)
3. Browse to the desired DOS compressed project file.
4. Highlight the compressed project file that you want to open.
5. Click the **Open** button.
6. If any other project is open, you are prompted to close it because only one project can be open at a time. Then the **Open** dialog box is displayed.
7. Since you chose a DOS compressed project file (.G&L), you are prompted whether you want to convert to the PiCPro for Windows project file format. Select **Yes** to continue opening.
8. Now the **Uncompress Project** dialog is displayed. **Note:** Just like PiCPro for Windows, a PiCPro for DOS compressed project file only contains one project.
9. Just like when you uncompress a PiCPro for Windows compressed project file, you are prompted for information on where to expand the contents.
10. The project is now open for editing and is now in PiCPro for Windows project format. It cannot be saved in PiCPro for DOS project format.

Note: If a DOS project uses the "@" feature to specify a list of files, those files are automatically deposited into the OTHER FILES category when the PiCPro for DOS project is converted to a PiCPro for Windows project. Any other use of the "@" feature to list files in a project is not supported in PiCPro for Windows.

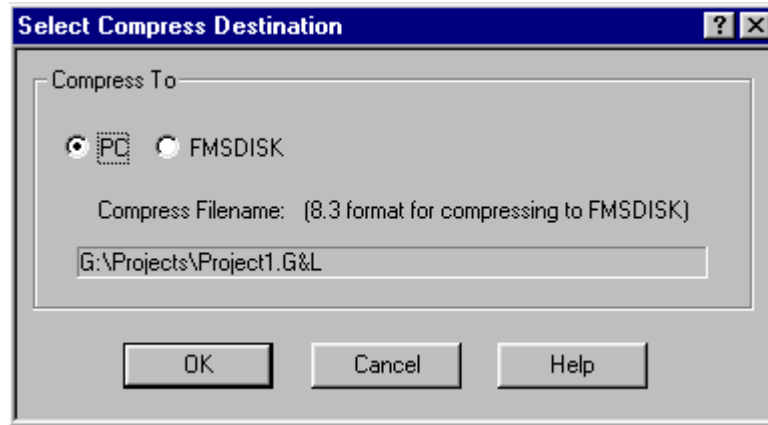
Compressing a Project

When you create a project using PiCPro for Windows, you create a project file. This file has entries that describe the name and location of the various source files and library files and others that constitute your project. The actual files are not a part of the project file. However, if you compress your project, a compressed project file is created. This file physically contains, in compressed format, a copy of all of the files described by name (plus their supporting files) in the project file. You can now easily distribute your project as a single file or deposit it on an appropriate control's FMSDISK.

Note: Compressed project files do not include the standard libraries.

To compress a project file:



1. Open or create a project.
2. Make sure a compressed project file name is configured in the COMPRESSED FILE category of the project.
3. Select **File | Compress** or right click on PROJECT NAME category and select **Compress**
4. The project is automatically **Validated**. This checks that the files that are referenced in the project tree actually exist in the configured location. It does not validate the compressed project file or verify the PiCPro edition/version.
5. You are also prompted to save your project if it has changed. You must save it to continue with the compress.
6. You are asked whether or not to **Update** your project. Updating checks the dependencies for the files in your project and automatically includes source and library files according to the properties selected for this project. Remember that entries for files that were manually added to the UDFBs/TASKs, SERVO SETUP FILES, and SERCOS SETUP FILES could be lost if other files are not dependent on them.
7. The **Select Compress Destination** dialog is displayed. Your choices for destination are the PC or the FMSDISK, if you have one. If you select the PC, the location for the compressed project file is exactly what you had configured in the project file. However, if you select FMSDISK, the location for the compressed project file is FMSDISK:\. (**Note:** you cannot edit the destination from here, only from the COMPRESSED FILE category in the project tree.)
8. Click **OK**.



FMSDISK:\Project1.G&L

Note: Even if you compress to your PC, you can copy the compressed project file to the FMSDISK on your control by using **Online | Disk Operations | Flash**. Remember that depositing any file(s) to FMSDISK completely removes anything that was already there. You do get a prompt warning you of this, allowing you to **Cancel**, if desired.

Launching a Project

At times, you may open a project that is configured with a version and edition of PiCPro for Windows that is different than the version and edition you are currently running. In these cases, you are warned of this and can either continue opening the project or not. If you choose to continue the open, the status bar will display an icon with a red background that indicates the edition/version mismatch:  . If no mismatch, this icon is displayed in green:  .

If you have a red (version mismatch) icon displayed, PiCPro for Windows provides a mechanism for you to automatically reopen your project in the correct edition/version of PiCPro for Windows (provided it is installed on your PC).

- Select **File | Launch Project**

OR


- Right click on the PROJECT NAME category of the project tree and select **Launch Project**.

The **Launch Project** feature first checks to see if the configured edition/version of PiCPro for Windows is installed on your PC. If it isn't, a message is displayed and the launch is canceled. Otherwise, the currently running PiCPro is closed, the configured one is opened and the project is also automatically opened.

Note: It is important to be running the same edition/version of PiCPro for Windows as is configured in the project if you need to edit and download any changes to the control. To avoid unexpected results, a particular edition/version of PiCPro for Windows should only be used with the libraries distributed with it. For example, if a project called for PiCPro for Windows 10.2 Standalone MMC Edition and you are currently running PiCPro for Windows 11.0 Professional Edition a version mismatch occurs. In fact, if your editions/versions mismatch, the **Compile and Download** feature will report the mismatch to you and will then cancel.

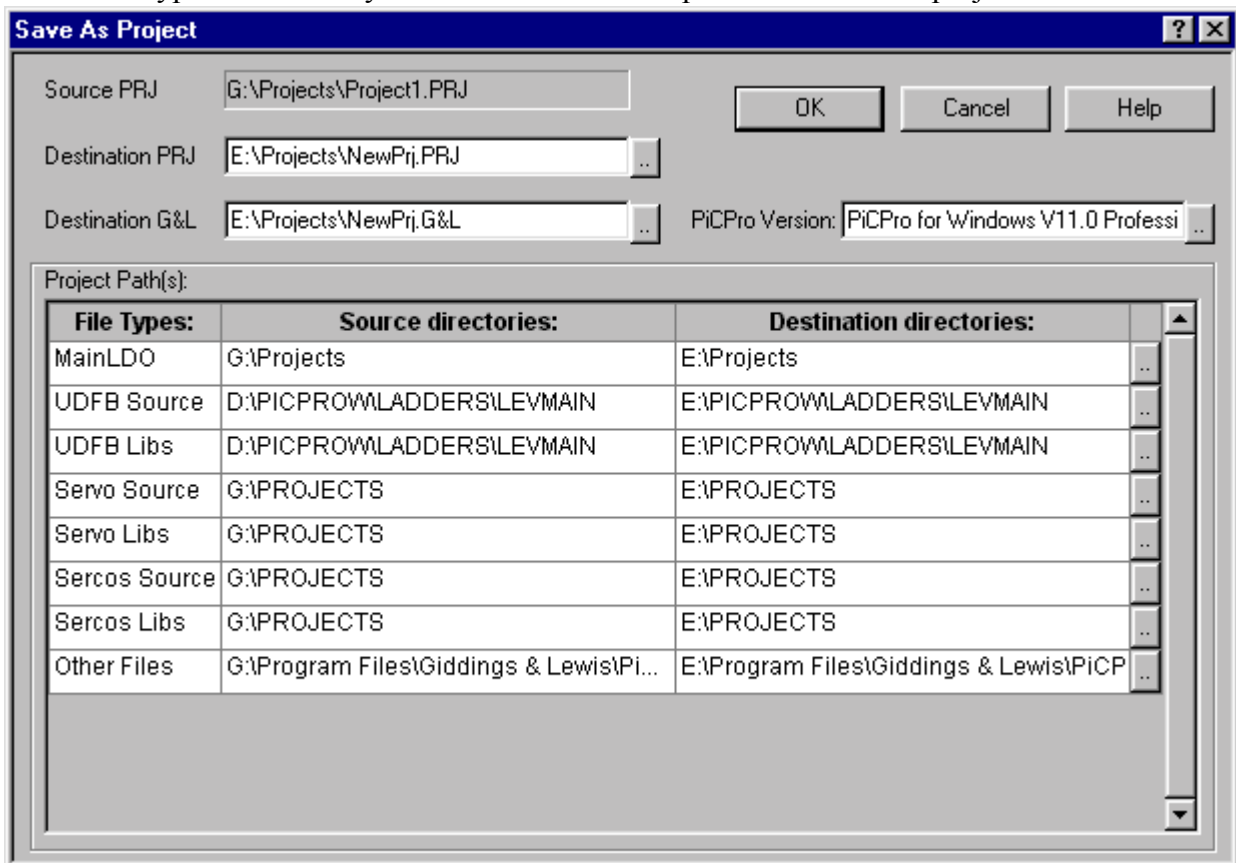
Printing a Project

To print the contents of a project file:

1. Open the desired project file.
2. Once the project is open, select **File | Print** from the main menu or click the print toolbar button: 
3. The standard **Print** dialog box is displayed.
4. Make appropriate changes for printer properties, etc. if needed.
5. Click **OK**.

Copying a Project (Save As)

The **Save As** command allows you to copy all of the files that make up an entire project to new locations. Here you have total control over where each different file type and directory combination will be copied to for the new project.



The **Save As Project** feature can be useful as a tool to help you move your project from development directories to production directories.

Source PRJ

This is the name and location of the project you are making a copy of. You cannot alter this field.

Destination PRJ

Fill in the name and location for the new project file. The name and/or location is usually different than the Source PRJ name. You can use the button to the right of this field to browse to the location you want and then type in the new name. In the new project, this will be entered into the PROJECT NAME category.

Destination G&L

Fill in the name and location for the new compressed project file. The name can be different than what is configured in the Source PRJ project file, although that is what it defaults to. In the new project, this will be entered into the COMPRESSED FILE category.

PiCPro Version

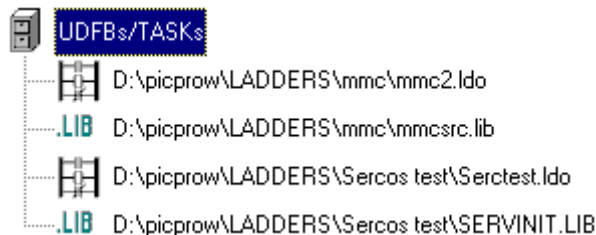
Enter the edition/version for the new project. Pressing the button to the right of this field brings up the **PiCPro Version Selection** dialog. Choose the desired edition and version from this box and click **OK**. In the new project, this will be entered into the PiCPro VERSION category.

Project Paths

This section is made up of file types and paths that are used to build the following categories in the new project:

- UDFBs/TASKs
- SERVO SETUP FILES
- SERCOS SETUP FILES
- PROFILE SETUP FILES
- OTHER FILES

For every different type of file (Servo Setup source, UDFB library, etc.) and path combination that exists in the Source PRJ project file, a corresponding entry exists in the **Project Path(s)**: in the dialog. For the example below, if for some reason this is how you've arranged your UDFB files in the Source PRJ project file...



...then this will be the information the **File Types** and **Source directories** columns of the **Save As Project** dialog:

UDFB Source	D:\picprow\LADDERS\mmc
UDFB Source	D:\picprow\LADDERS\Sercos test
UDFB Libs	D:\picprow\LADDERS\mmc
UDFB Libs	D:\picprow\LADDERS\Sercos test

Note: If a category in the Source PRJ is empty, then the corresponding file types will not be listed in this dialog's **Project Path(s):**. For example, if the Source PRJ has no items in the OTHER FILES category, then **Other Files** will not be listed in the **File Types:** of this dialog. If the Source PRJ has source files in the PROFILE SETUP FILES categories, but no library files, then the **File Types:** in this dialog will have **Profile Setup Source** listed but not **Profile Setup Libs**.

The **File Types:** and **Source directories:** columns cannot be edited from this dialog, but the entries in the **Destination directories:** column can be. Edit this field for every file type's destination that you want changed in the new project. By default, the **Destination directories:** column is initialized with the same information as in the **Source directories:** column.

To begin editing a **Destination directories:** field, single click at the point in the text where you want to start typing. A blinking cursor should appear. Any characters typed will be inserted before the cursor. If any portion of a field is highlighted, any character you type will replace that highlighted portion. If you click in the field where there isn't any text, the entire field will be highlighted.

Note:

- <Ctrl+X> cuts any highlighted portion to the clipboard.
- <Ctrl+C> copies any highlighted portion to the clipboard.
- <Ctrl+V> pastes clipboard contents over any highlighted portion otherwise pastes before the blinking cursor in the field.

You can also use the browse button to the right of each **Destination directories:** field to browse to the desired location.

When done click the **OK** button. The source PRJ file is automatically closed, the files are copied to their new locations, and the new project file is opened in the PiCPro for Windows that is currently running.

NOTES

CHAPTER 3 Working with Ladders

A PiCPro Session

This section covers information about basic operations you need to be familiar with to use PiCPro. A PiCPro session begins when you launch the application. Next you choose what you want to do in the session.

- Create a new ladder program (LDO)
- Open an existing LDO file
- Create a new servo or SERCOS setup program (SRV, SRC)
- Open an existing SRV or SRC file.
- Create or edit a Project (.PRJ)


Once the file you've chosen is open, you can begin to add/change the file using the editing tools. Then you save the file to a location specified by you. Finally, you can exit PiCPro or, if completed, compile and download your file to the PiC.

Creating New Files

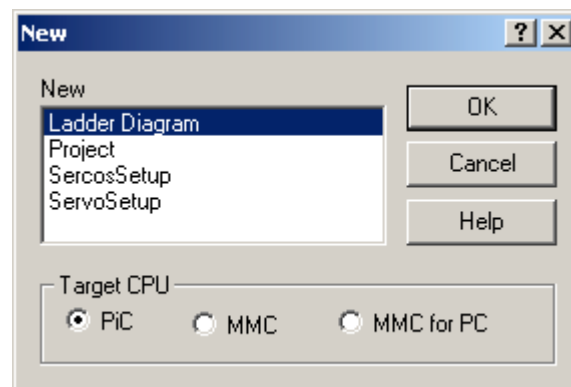
Once PiCPro is started, you can create a new LDO file or a new setup file or a new project depending on which program type you choose. Do one of the following:

Choose **F**ile | **N**ew from the menu.

OR

Click the  button in the Standard toolbar.

The New box appears and you can choose Ladder Diagram, Project, SERCOS Setup, or Servo setup. When creating new files in PiCPro for Windows Professional Edition, you can also select the CPU type (**P**iC, **M**MC, or **M**MC for **P**C). The default selection is the last selection made.



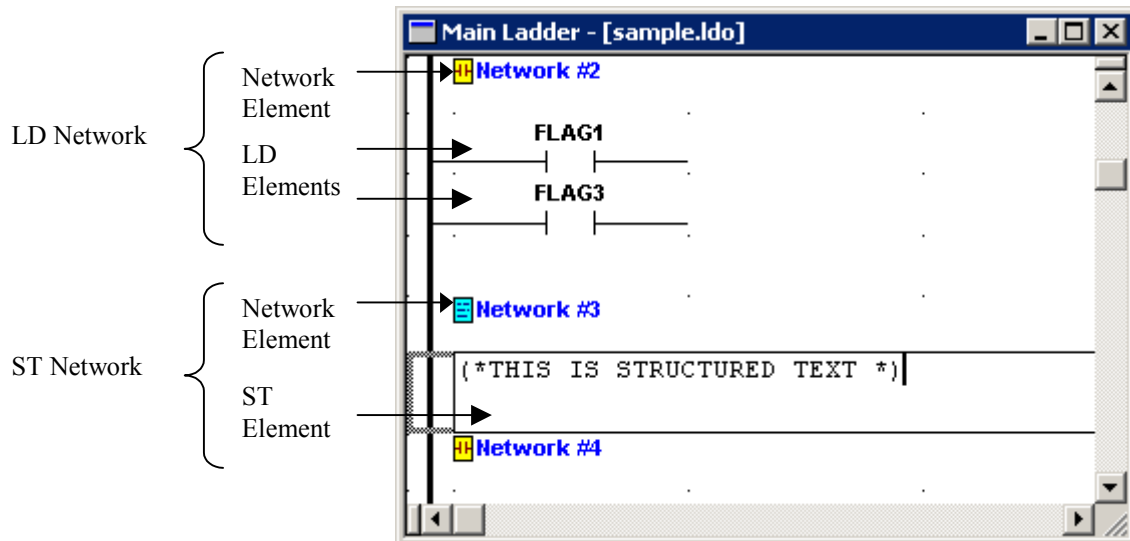
Note: In the PiCPro for Windows Standalone MMC Edition, the CPU type is always MMC and, therefore, is not an option on this dialog.

PiCPro displays a new ladder diagram, project tree, or setup window. You can now create your ladder, project, or setup program with the tools and features available in PiCPro and then save your file.


Opening a New Ladder File

When you open a new ladder file, Network #1 appears and you can begin entering your program. Additional networks can be added as you develop your program.

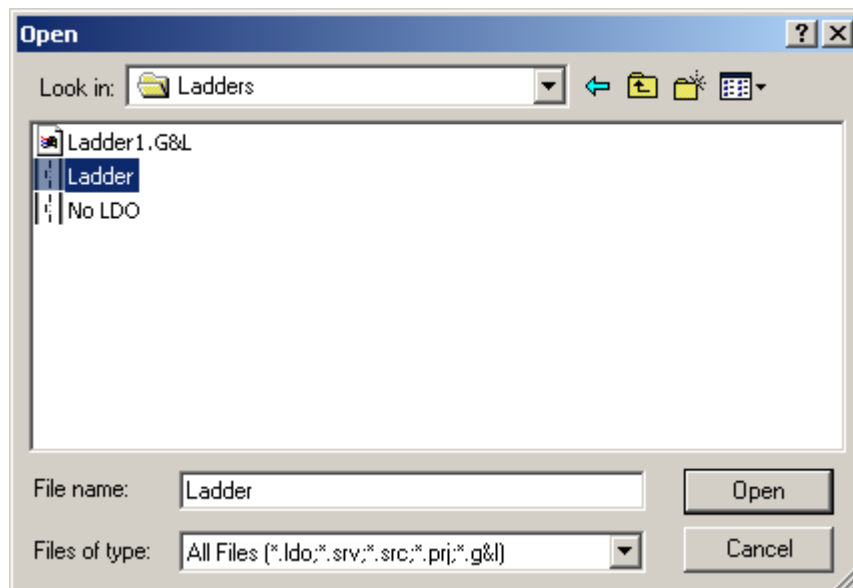
The location of elements for both an LD Network and a ST Network are illustrated below.



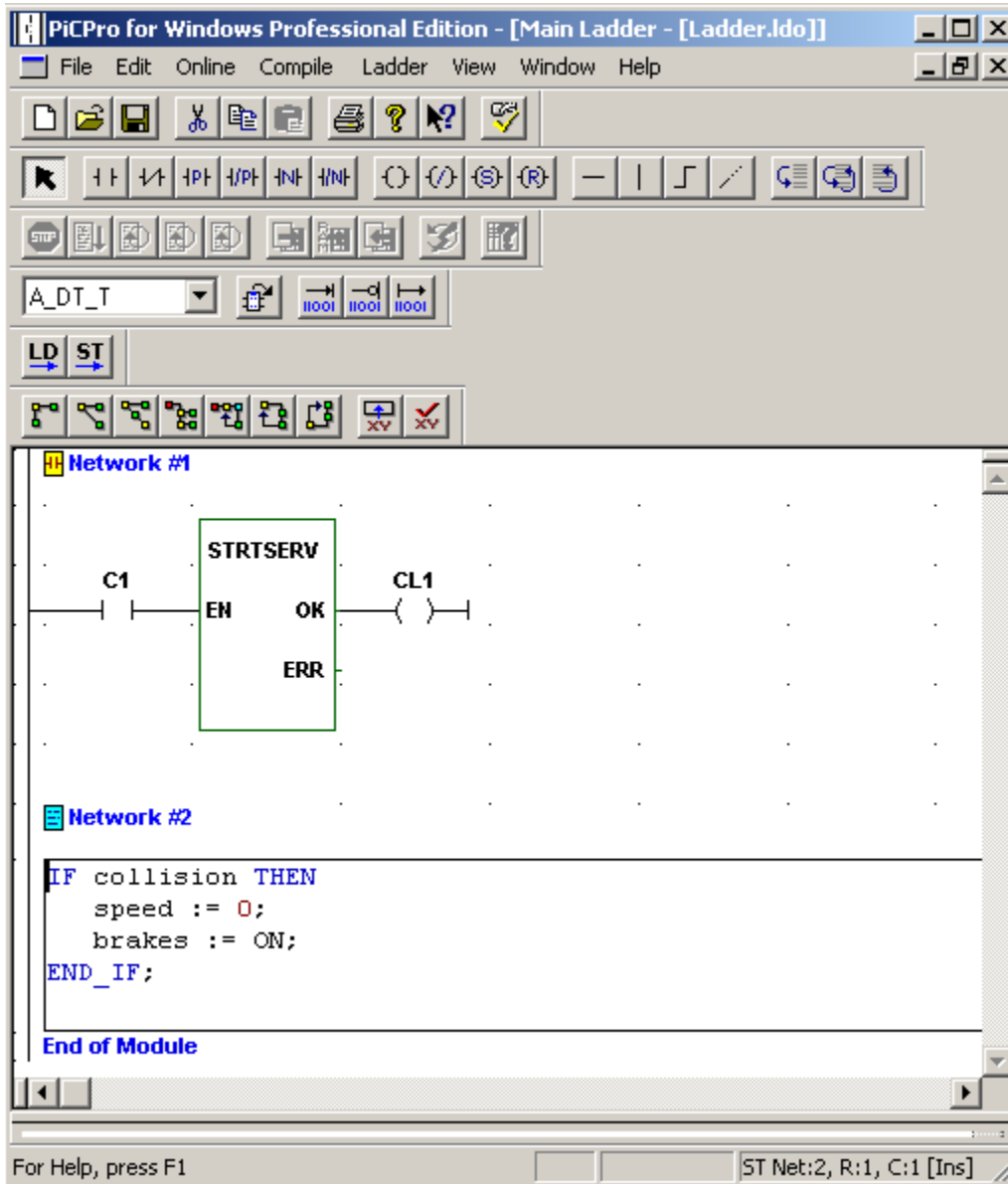
Opening an Existing Ladder File

The Open command opens files that have already been saved. Choose **File | Open** or Click the  button:

1. In the **Look In:** list box, choose the drive where the file is located.
2. Double-click the folder where the file is located.
3. Double-click the filename of the file you want to open.



An example PiC CPU ladder file is illustrated below.



Opening an LDO file with a PiC CPU in PiCPro for Windows Standalone MMC Edition will result in a confirmation prompt stating the following:

The ladder you have just opened has a PiC cpu. Only ladders with an MMC cpu can be downloaded in this version of PiCPro. The ladder could be converted to use a MMC cpu. All I/O in the main and remote racks will be removed. Because of these changes, I/O points defined in Software Declarations should be checked to ensure that they are mapped to the correct locations. Do you want to convert the file so that it can be downloaded in this version of PiCPro?

Choose **Yes** or **No**. If you choose **No**, the file will be opened but cannot be downloaded, compiled etc.

Opening a Recently Opened File

1. Choose **F**ile from the main menu.

A list of the most recently opened files appears at the bottom of the menu. These will be either LDO, PRJ, SRV, or SRC files.

2. Click on the file you want to open.


Opening an LDO file with a PiC CPU in PiCPro for Windows Standalone MMC Edition will result in a confirmation prompt. If you choose not to convert the LDO, the file will be opened but cannot be downloaded, compiled etc.

PiCPro Tools

The toolbars give you quick access to some of the frequently used tools. These are the toolbars available in PiCPro (refer to Appendix I).

- Standard
- Basic Online Operations
- Advanced Operations
- Ladder
- View Navigator
- Functions
- Compiler
- Insert New Network
- Structured Text Tools

Using the Pointer Tool

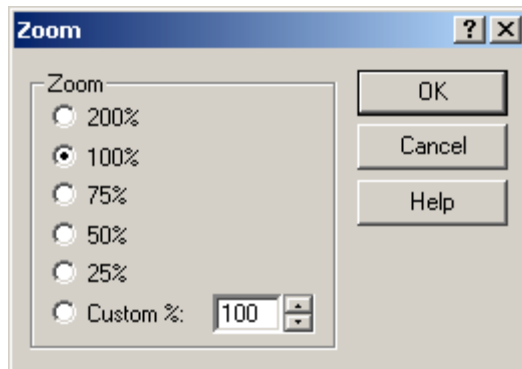
The Pointer tool  button is found on the Ladder Toolbar. It is the default selection allowing you to point and click to select locations in the ladder. You can exit the drop mode for any of the other Ladder toolbar buttons by selecting the Pointer tool.

Zooming In and Out

Zooming in and out allows you to view and work on your program from 5% to 400% magnification.



Using Zoom Command from View Menu

1. Choose **V**iew | **Z**oom. The **Zoom** dialog box appears:



2. Choose a predefined zoom rate by clicking a button or pressing a hot key (i.e. **Z** for 75%)
or
Enter a custom rate by incrementing or decrementing the scroll box in increments of 1%.
Note: that choosing 200% will double the size and choosing 50% will decrease the size by one half. The default is 100%.
3. Click **OK** to accept the zoom rate or **Cancel** to ignore it.

Using Zoom Command from Navigator Toolbar

- To decrease the size of the display, click on the  repeatedly until you reach the desired magnification.
- To increase the size of the display, click on the  repeatedly until you reach the desired magnification.

Tip

When you zoom in or out, the cell in the top left-hand corner remains in that location after zooming.

Selecting and Deselecting Items

Focus

A cell or group of cells in your ladder will always have focus. Focus is indicated by a dashed rectangular border around the area. A cell that has focus can be copied, cut, pasted, and deleted. You can also select a cell or group of cells.

The quickest way to set focus is by using the Pointer tool to single click on a cell. If you place focus on the power rail in your ladder, the whole row will have focus. If you place focus on a Function/Block, the whole Function/Block will have focus. However, if you placed focus on the power rail and there is a Function/Block in the row, the entire Function/Block is not selected. You can also move the cell focus from the keyboard as follows.

Keys	Focus Moves:
LD (Ladder) and ST (Structured Text)	
<Arrow>	From current position to new position in the direction of the arrow
<Home>	To first column of current row (not on the power rail)
<End>	To the last filled cell of the current row
<Ctrl + Home>	To Network #1 in your ladder if focus is already in first row, first column of current network or else to first row, first column of current network
<Ctrl + End>	To the End of Module marker if focus is already on the last occupied cell in the last row of the network or else to the last occupied cell of the current network
<Page Down/Up>	One vertical viewable page down/up
<Ctrl + Page Down/Up>	One horizontal viewable page left/right
LD Only	
<Tab>	One element to the right of the current element
<Shift + Tab>	One element to the left of the current element

Selection

Selection is indicated by highlighting the area. Selected cells can be copied, cut, pasted, deleted, dragged and dropped.

Selection is set by first placing focus on the cell and then clicking within the cell. The cell is highlighted.

A row is selected by placing focus on the power rail and then clicking on the power rail within the focus row.

Multiple cells are selected by clicking on the first cell and then dragging the mouse over the area you want to highlight.

If the power rail is not included in the selection, the selection must be contained within a network.

If the power rail is included in the selection, the selection can contain sequential rows from more than one network.

Note: An ST element takes up only 1 row but can contain up to 500 lines of text.

You can also select cells using the keyboard.

Keys	Selects
<Shift + Arrow>	All cells within the rectangular area defined by the arrow keys (Within a network if focus is not on the power rail and over multiple networks if focus is on the power rail.)
<Shift + Home>	The focus cell and cells to the left of it
<Shift + End>	The focus cell and cells to the right of it until the last filled cell
<Shift + Ctrl + Home>	The focus cell and all cells to the upper left-hand cell in the network if focus cell is not on the power rail. The focus cell and all cells to the first row in the module if the focus cell is on the power rail of the first row of the current network or else selects the focus cells and all cells through the first row of the current network
<Shift + Ctrl + End>	The focus cell and all cells to the last row in the network if focus cell is <u>not</u> on the power rail. The focus cell and all cells to the last row in the module if the focus cell is on the power rail of the last row of the current network or else selects the focus cell and all cells through the last row of the current network
<Ctrl + A>	All cells in the ladder if all cells in the current network are selected. If all cells in the current network are not selected, all cells in the current network will be selected.

Deselecting

If you click on any cell, all cells will be de-selected.

If you move focus with an arrow key, all cells will be de-selected.

Entering Ladder Networks

Inserting a Network

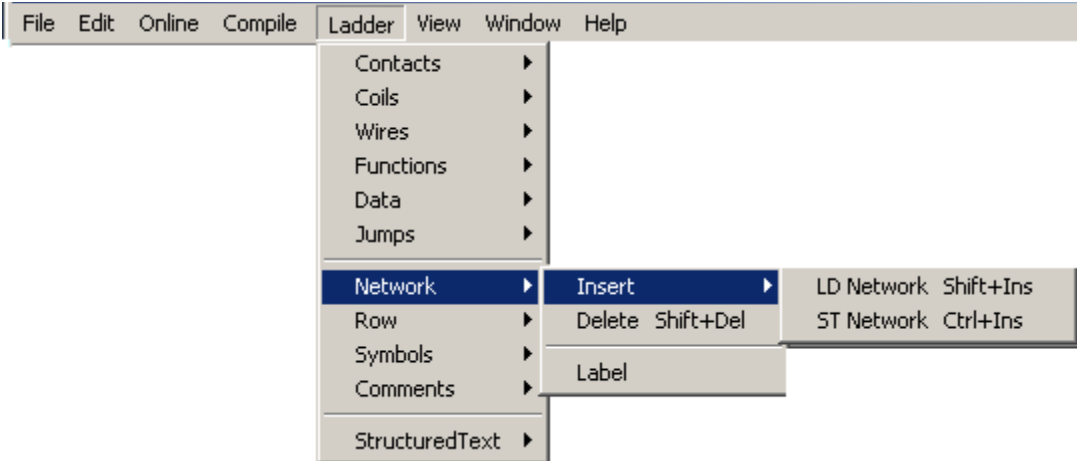
When a network is inserted it will be inserted above the focus cell. A new network appears above the focus cell. All the networks below the insertion are renumbered.

Right Mouse Click Method

1. Position the focus where you want to insert a network.
2. Right click to see a Pop-up menu and choose **Insert | LD Network** or **ST Network**.

Menu Pull-Down Method

From the Main Menu, select **Ladder | Network | Insert** and select either **LD Network** or **ST Network**.



Hotkey Method

Press a hotkey combination; **<Shift + Insert>** for a LD Network or **<Ctrl + Insert>** for a ST Network.

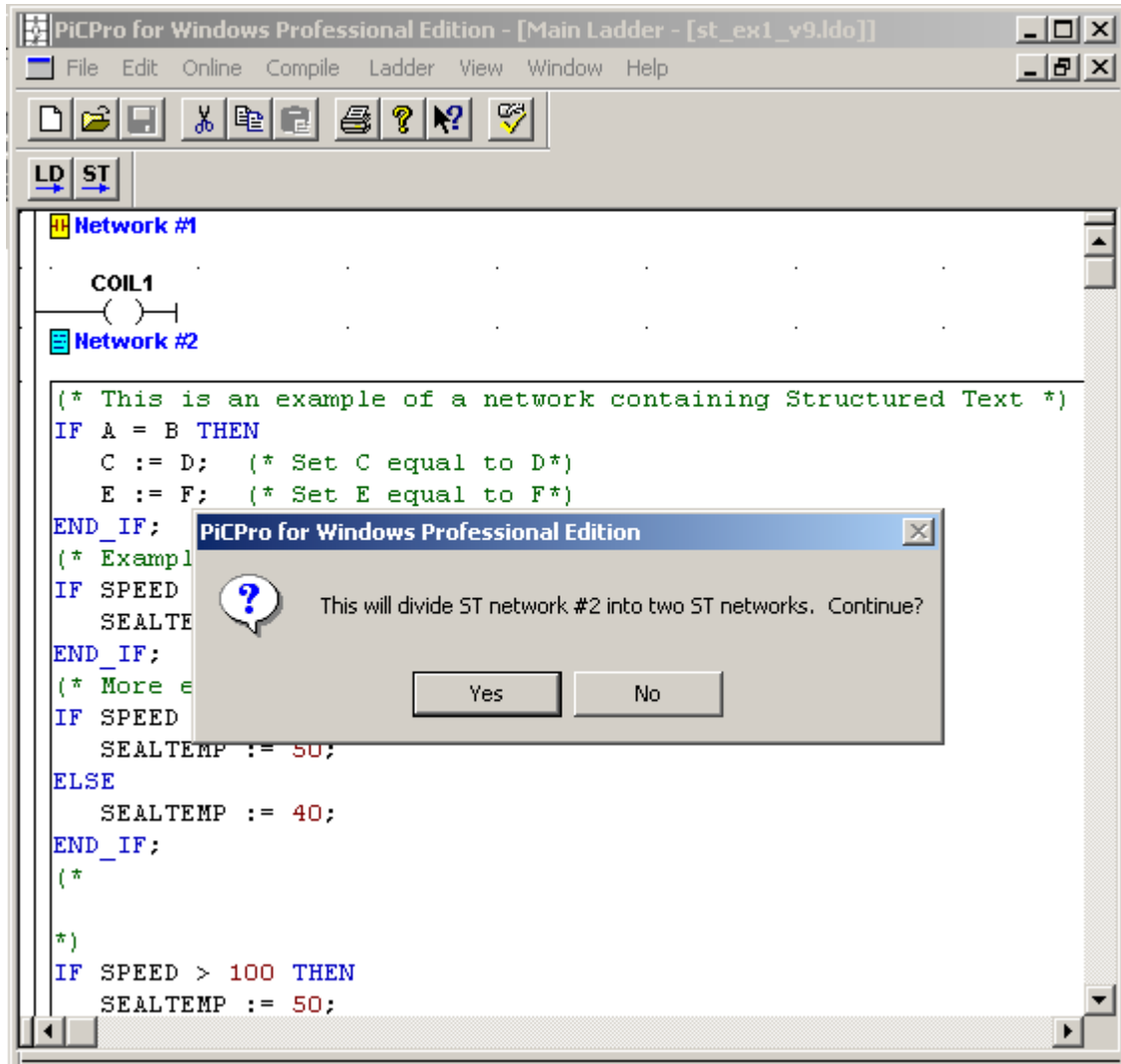
Toolbar Method

Select a toolbar button from the “Insert New Network” toolbar. Select **LD** for a Ladder Network or **ST** for a Structured Text Network.



Inserting a ST network with focus in a ST element

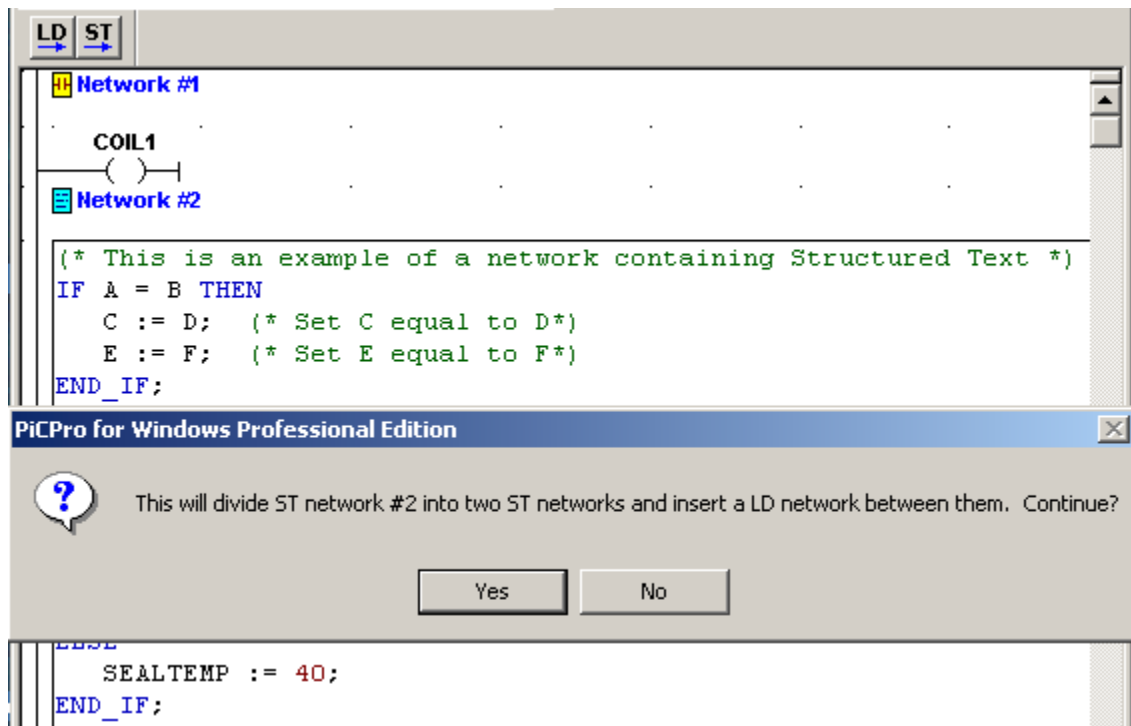
If focus is in a ST element and you insert a ST network, the ST element will be divided at the cursor position. The following prompt will appear:



Select Yes and the ST network will divide into two ST networks. The line with the cursor will appear in the second ST network. The networks will be renumbered if this is not the last network.

Inserting a LD network with focus in a ST element

If focus is in a ST element and you insert a LD network, the ST element will be divided at the cursor position. The following prompt will appear:



Select **Yes** and the ST network will divide into two ST networks at the cursor location and a LD network will be inserted between them. The focused element will be the new LD element. The networks will be renumbered if this is not the last network.

Inserting a ST network with focus in a LD element

If focus is in a LD element and you insert a ST network, the LD network will be divided into two LD networks at the cursor position. A ST network will be inserted between them. The focused element will be the new ST element. The networks will be renumbered if this is not the last network.

Overview - Cut, Copy, Delete and Paste

The Cut, Copy, and Paste commands allow you to use the clipboard to make copies of LD or ST Network Elements, LD Elements or Structured Text items from your ladder.

The **C**opy command places a copy of the selected item on the clipboard.

The **C**ut command removes the item from your ladder and places it on the clipboard.

Once an item is on the clipboard, you can use the **P**aste or **P**aste **I**nsert commands to place the item in a ladder. The item remains on the clipboard until you cut or copy another item. Only one item can be placed on the clipboard at a time.

Note: The **D**elete command removes the selected item from your ladder. It does not place it on the clipboard.

If animation is running when you begin these procedures, it will be halted.

Delete

Deleting LD or ST Networks

To delete an LD or ST network do the following:

1. Position the focus anywhere in the network you want to delete.
2. Choose **L**adder | **N**etwork | **D**elete or press <Shift + Delete>.
3. A confirmation message will appear. If you select **OK**, the network will be deleted. All the networks after the deleted network are renumbered.
4. You can delete multiple networks by highlighting all the rows within the networks and choosing **L**adder | **R**ow | **D**elete or press <Delete>.

Deleting LD or ST Network Elements

Deleting a LD or ST network element results in the merging of two networks. However, only like networks can be merged. To delete a network element, do the following:

1. Position the focus on a network element.
2. Choose **E**dit | **D**elete or press <Ctrl + D>, or press **Delete**.
3. A confirmation message will be displayed.

Deleting LD Elements or Structured Text

The Delete command removes the selected LD Element or Structured Text. It does not place it on the clipboard.


To use the Delete command

1. Click on the LD element or highlight the Structured Text.
2. Choose **E**dit | **D**elete or press <Ctrl + D>, or press **Delete**.

Cut, Copy, and Paste


Cutting Network Elements

Cutting a network element results in the merging of two networks. However, only like networks can be merged (i.e. a LD network cannot be merged with a ST network and vice versa). To cut a network element, do the following:

1. Position the focus in the network element you want to cut.
2. Choose **E**dit | **C**ut, or **Ctrl-X**, or  from the toolbar menu, or right click and select **C**ut from the popup menu.
3. A confirmation prompt will be displayed if the network element is the same type as the previous network element. An error message will be displayed if the network element is not the same type as the previous network element (an LD network cannot be merged with a ST network).


Cutting LD Elements or Structured Text

When you cut a LD Element, an ST Element or a portion of an ST element, it is removed from the file and placed on the clipboard.

1. Select what you want to cut.
2. Choose one of the following methods to issue the Cut command:
 - Click the  button from the standard toolbar.
 - Choose **E**dit | **C**ut from the main menu.
 - Press <Ctrl + X>.
 - Right-click the mouse and select **C**ut from the popup menu.

Copying LD Elements or Structured Text

When you copy LD Elements or Structured Text Elements, they remain in the file and are placed on the clipboard.

1. Select what you want to copy.
2. Choose one of the following methods to issue the Copy command:
 - Click the  button from the standard toolbar.
 - Choose **E**dit | **C**opy from the main menu.
 - Press <Ctrl + C>.
 - Right-click the mouse and select **C**opy from the popup menu.


Copy Rules

- If focus is on a ST element and nothing is selected, nothing will be copied.
- If focus is on a ST network element, only the ST network element will be copied.
- If focus is on a LD element, the LD element will be copied.
- When a portion of a source ladder is copied and pasted into a different target ladder, the software declarations that were used in the copied portion of the source ladder are also placed on the clipboard and pasted in the target ladder. Undeclared variables will remain undeclared. This is true for LD elements and ST elements.

Pasting LD Elements or Structured Text

To paste LD Elements or Structured Text into a file from the clipboard, you can use either the Paste or the Paste Insert commands.

To use the Paste command

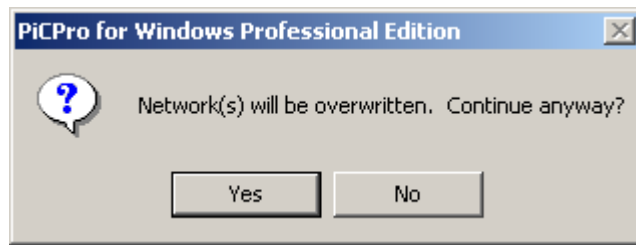
1. Select a location for the paste.
2. Choose one of the following methods to issue the Paste command:
 - Click the  button from the standard toolbar.
 - Choose **E**dit | **P**aste from the main menu.
 - Press <Ctrl + V>.
 - Right-click the mouse and select **P**aste from the popup menu.

Paste Rules

- If the data copied to the clipboard contains a power rail, it must be pasted on a power rail.
- Rows must be pasted on rows. The only exception is if focus is on the End of Module (EOM). If focus is on EOM, rows are inserted.
- Pasting a non-network element on a network element has the effect of merging two like networks.

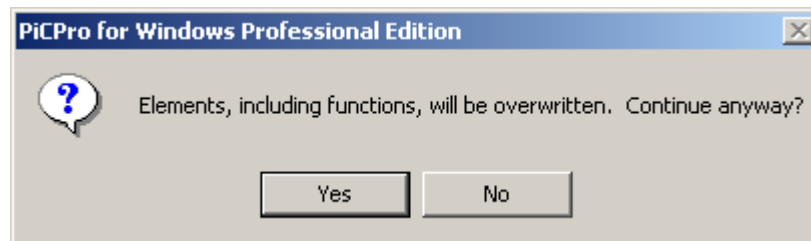
- A paste operation that does not include the power rail does a replace as follows:

If focus is on a LD network element, the following prompt will be displayed:



If you click Yes to continue, data in the network will be overwritten with the data from the clipboard.

If focus is on a ladder element and the number of rows affected is less than or equal to the number of rows in the network, the following prompt is displayed:



If you click Yes to continue, elements and functions in the network will be overwritten with the data from the clipboard.

If focus is on a ladder element and the number of rows affected is more than the number of rows remaining in the network, rows are added to the network.

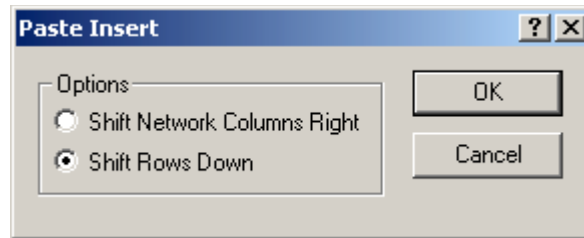
If focus is on the EOM, rows are inserted.

- A paste operation that includes the power rail, does an insert above the line with focus.
- The maximum number of cells in an LD network is 255.
- An error message will be displayed if an attempt is made to paste text into a ST network that would cause the number of lines to exceed 500 or the number of characters in a line to exceed 500.
- A ST element (not a ST network) or portion of a ST element that is copied can be pasted as plain text into Microsoft Word™, Notepad™ or Wordpad™. In this case, software declarations information is not pasted.
- Text copied from Microsoft Word™, Notepad™ or Wordpad™ can be pasted into a ST element.
- A portion of a ladder can be copied and pasted into the view and force lists.

- Within a LD network, highlighting ladder elements makes no difference when pasting. Only ladder elements actually replaced when the paste takes place will be deleted.

Paste Insert Rules

- If **Edit | Paste Insert** is selected while focus is on a ladder element, the following prompt is displayed asking if rows should be shifted down or network columns should be shifted right:



- If the clipboard data contains a power rail, it must be pasted on a power rail. The only exception is End of Module (EOM). If focus is on the EOM, rows can be pasted.
- The maximum number of cells in an LD network is 255.
- Within a ST element, a paste is always a paste insert operation. If text is highlighted when paste is selected, the highlighted text will be deleted when the paste occurs.

Drag and Drop

You can drag a highlighted area by clicking and holding down the mouse, moving to the desired area, and drop the selection by releasing the mouse.

If a power rail is not included in the selection, the selection must be contained within a network.

If the power rail is included in the selection, the selection can contain sequential rows from more than one network.

Note: An ST element takes up only 1 row but can contain up to 500 lines of text.

Drag and Drop Copy

1. Select an item.
2. Hold down the **Ctrl** key and hold down the left mouse button at the same time.
3. Drag the selection to a new location. A + character will appear by the mouse pointer.
4. Release the left mouse button and the item will be copied to the new location.

Drag and Drop Copy Rules

When a portion of a ladder is Drag and Drop copied into a different ladder, the software declarations that were used in the copied portion of the ladder are added to software declarations. Undeclared variables will remain undeclared.

Drag and Drop Move

1. Select an item.
2. Hold down the left mouse button.
3. Drag the selection to a new location.
4. Release the left mouse button and the item will be moved to the new location.

Drag and Drop Move Rules

When a portion of a ladder is Drag and Drop moved into a different ladder, the software declarations that were used in the moved portion of the ladder are added to software declarations. Undeclared variables will remain undeclared.

If a LD network element is dragged and dropped, one of the following will occur based on the location of the element to be moved.

After the move:

- If the network above the element being moved is a LD network, the remaining rows in the network will be merged with the network above.
- If the network above is a ST network, an error message will be displayed.
- If the network is the first network in the ladder and the network is empty, a prompt will be displayed asking if the entire network should be deleted.
- If the network is the first network and the network is not empty, an error message will be displayed along with a prompt asking if the entire network should be deleted.

If a ST network element is dragged and dropped, one of the following will occur based on the location of the element to be moved.

After the move:

- If the network above the element being moved is a ST network, the ST element below the network element being moved will be merged with the ST element above the network element being moved.
- If the network above is a LD network, an error message will be displayed.
- If the network is the first network in the ladder and the network is empty, a prompt will be displayed asking if the entire network should be deleted.
- If the network is the first network and the network is not empty, an error message will be displayed along with a prompt asking if the entire network should be deleted.

Drop Rules

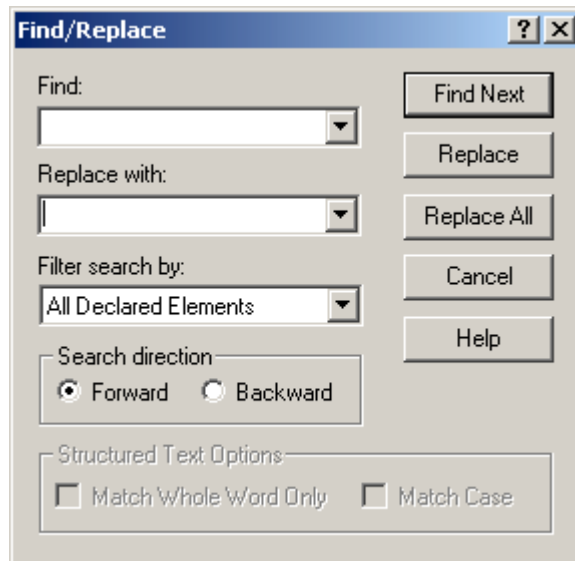
- The maximum number of lines in a ST network is 500. The maximum number of characters in a line in a ST network is 500.
- The maximum number of cells in an LD network is 255.
- If the data to be dropped contains a power rail, it must be dropped on a power rail (rows must be dropped on rows). The only exception to this rule is the EOM. If focus is on the EOM, rows can be dropped.
- Dropping a non network element on a network element has the effect of merging two **like** networks.
- A drop operation, which does not include the power rail, does a replace as follows:
 - If dropping on a network element, the following prompt is displayed: **Network(s) will be overwritten. Continue anyway?**
 - If dropping on a ladder element, and the number of rows affected is less than or equal to the number of rows in the network, the following prompt is displayed. **Element(s) will be overwritten. Continue anyway?**
 - If dropping on a ladder element, and the number of rows affected is more than the number of rows remaining in the network, rows are added to the network.
 - If dropping on EOM, rows are inserted.
- A drop operation, which includes the power rail, does an insert above the line with focus.

Other Notes - Drag and Drop

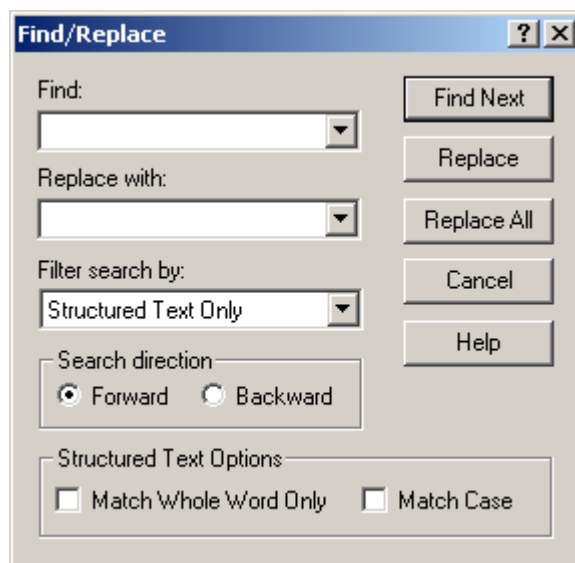
- Within a ST element, a drop is always an insert operation.
- Drag and drop can be used to copy/move a ST element (not a ST network) or portion of a ST element into Microsoft Word™. In this case, software declarations information is not dropped.
- Text dragged from Microsoft Word™ can be dropped into an ST element.
- Notepad™ and Wordpad™ do not support drag and drop.
- Drag and drop can be used to copy or move a portion of a ladder or structured text into the view and force lists.

Finding and Replacing

In a ladder in PiCPro, you can search for declared contacts, declared coils, declared function blocks, declared elements, function/function block titles, network numbers, or structured text.



The search can be refined by applying a filter that limits the search (e.g. **Structured Text Only**):



Control	Description
Find:	Enter Item to search for. An item can be chosen from the list or typed in. Valid entries are filled in when a filter is chosen. The dropdown list contains valid entries for the specified filter. If the Structured Text Only filter is selected, the Find: drop down list contains the most recently searched for “Structured Text Only” strings.
Replace with:	Enter the item you want to replace the Find: item with or choose an item from the list. The drop down list will contain items from a previous find/replace.
Filter search by:	An item can be chosen from the list to filter the search with. The filter will describe the type of item being searched for.
Search direction	Check the appropriate box to search either forward or backward from where the focus is in your ladder. The search will proceed in the direction you specify and then auto-wrap when the top/bottom of the ladder is reached.
Structured Text Options	<p>Match Whole Word Only - Enabled for Structured Text Only filter. When selected, finds strings that are whole words and not part of a larger word.</p> <p>Match Case- Enabled for Structured Text Only filter. When selected, distinguishes between uppercase and lowercase characters and will match only those instances where the capitalization matches the text typed in the Find: field.</p>
<u>F</u>ind Next	Finds the next occurrence of the entry in Find:
<u>R</u>eplace	Replaces the entry in Find: with the entry in Replace with:
Replace <u>A</u>ll	Replaces all occurrences of the entry in Find: with the entry in Replace with:
<u>C</u>ancel	Closes this dialog without saving any changes.
<u>H</u>elp	Accesses the Help system.

Notes:

- You must enter the complete name of the item you want to search for.
- When searching for declared contacts, coils, Function Blocks or Function Block titles in a ST network, text may not be recognized as a declared element if there are errors while compiling. **Replace** will be disabled in a ST Network that contains compile errors.
- When searching in a ST network, Function Blocks of a specific type can only be found if the Title is left in the function.
- Choosing the **Structured Text Only** filter causes a search to be done only inside ST networks. LD networks will not be searched. Any filter selected other than **Structured Text Only** causes a search to be done in both ST and LD networks.
- You can find and replace items individually or find and replace all occurrences of items by entering a declared variable in the **Replace with:** box and choosing the **Replace** or **Replace All** button.

Finding and Replacing - Procedures

To search for an element in the ladder

1. Choose **E**dit | **F**ind/Replace or press <Alt F3> to see the **Find/Replace** box.
2. Enter the name of the ladder or structured text element you want to search for in the **Find:** field or use the dropdown list and select the name of the ladder element you want to search for. If the focus in your ladder is on an element that can be searched for, it will automatically appear in the **Find:** field.
3. Be sure the correct filter is selected in the **Filter search by:** box.
4. Choose the **Search direction**.
5. Select the **F**ind Next button to move either forward or backward to the next location of the element in the ladder.
6. Continue selecting the **F**ind Next button or press <Enter> if you want to continue to find occurrences of the element. A message will inform you after the ladder has been completely searched.

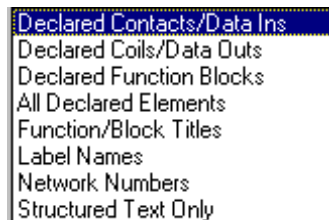
To search and replace an element in the ladder

1. Choose **Edit | Find/Replace** or press <Alt F3> to see the **Find/Replace** box.
2. Enter the name of the ladder element you want to search for in the **Find:** box. If the focus in your ladder is on an element that can be searched for, it will automatically appear in the box.
3. Enter the name you want to replace the variable with in the **Replace with:** box or use the dropdown list and select the name you want to replace the variable with. If the replacement name has not been added to the software declarations table, you will be prompted to do so now.
4. Be sure the correct filter is selected in the **Filter search by:** box.
5. Choose the **Search direction**.
6. First choose the **Find/Next** button to find the element.
7. Choose either **Replace** to replace an occurrence of the element or **Replace All** to replace all occurrences

Filtering Find and Replace

You can filter your search by the criteria listed below in the **Filter search by:** box within the **Find/Replace** box. Some things to keep in mind include:

- If focus in your ladder is on a contact, coil, data input, data output, function, function block, network label, or a jump label, the name will automatically be entered in the **Find:** box.
- If the focus is on a wire or a return label, things that cannot be searched for, you must enter a name in the **Find:** box and the correct filter in the **Filter search by:** box.
- If you want to find a network, enter the number in the **Find:** box and select Network Numbers for the **Filter search by:** box.



Contact or data input
Coil or data output
Function Block instance
Contact, coil, data input, or data output
Functions
Network or jump label
Numbered network
Structured Text

- All the filter selections except **Structured Text Only**, **Function/Block Titles** and **Network Numbers** require entries into the **Find:** box that are defined in software declarations and that match the current filter. If **Structured Text Only** is selected, any text can be typed into the **Find:** box.

- If the **Declared Contacts/Data Ins** filter is selected, the entire ladder will be searched for the contents of the **Find:** field. A match will be found in a ST network anywhere the contents of the **Find:** field is used and not set. **Note:** When this filter is selected, the **Find:** field must contain a valid variable name.
- If the **Declared Coils/Data Outs** filter is selected, the entire ladder will be searched for the contents of the **Find:** field. A match will be found in a ST network anywhere the contents of the **Find:** edit field is set and not just used. **Note:** When this filter is selected, the **Find:** field must contain a valid variable name.
- If the **Declared Function Blocks** filter is selected, the contents of the **Find:** field represent a function block instance name. In an ST element, a function block is represented in the following format: **Instance:Title(...)**. If found in a ST network, only **Instance** will be highlighted.
- If the **All Declared Elements** filter is selected, all declared elements will be searched.
- If the **Function/Block Titles** filter is selected, the contents of the **Find:** field represents a function/block title. In an ST element, a function/block title is represented in the following format: **Instance:Title(...)**. If found in a ST network, only **Title** will be highlighted. If ‘:Title’ is not typed in (it is optional), the search will not relate the instance name back to the title being searched.
- If the **Label Names** filter or the **Network Numbers** filter is selected, ST elements will not be searched since these items are not relevant to a ST element.
- If the **Structured Text Only** filter is selected, the drop down list for the **Find:** box will contain a list of the last **Structured Text Only** search entries.

Finding Duplicates

When debugging a ladder problem, it can be useful to identify all symbols that are modified at multiple locations within a ladder. **Find Duplicates** provides this functionality. When **Find Duplicates** is selected, PiCPro searches for function blocks, data outs, and coils that appear more than once in the ladder. When PiCPro finds duplicated function blocks, data outs, or coils, a message is displayed in the information window.

To search for duplicated elements

1. Choose **Edit | Find Duplicates**.
2. PiCPro searches the ladder and puts a message in the information window when finished either saying no duplicates were found or listing the ones that were found.

Note: Duplicated jump labels, contacts, data ins, and functions are not included in this search.

Saving, Closing, and Exiting

When you close your ladder, PiCPro asks if you want to keep any changes that have not yet been saved. You have three choices:

- **Yes** to save the latest changes
- **No** to lose the changes and close the ladder
- **Cancel** to allow you to change your mind and continue to work on the ladder.

If you close a ladder that you have not yet assigned a name to, PiCPro asks if you want to save changes to the ladder diagram. If you choose **Yes**, the **Save As** box appears and you can type in the name. PiCPro assigns the name to the ladder, saves the ladder, and then closes.

Note: A ladder saved in V13.0 is saved in a format that cannot be read by versions of PiCPro prior to V13.0 if software declarations contains any mixed case variable names, variable names longer than 8 characters, the ladder contains any ST networks, or scan with ASIU errors has been enabled in hardware declarations.


Saving Files

It is a good idea to save your file at regular intervals as you work on it.

Using the **S**ave command, you can save the file under its existing name.

Using the **S**ave **A**s command, you can specify a new filename and/or a location where you want the file stored.

To save a new file

1. Choose **F**ile | **S**ave or the  button.
2. In the **S**ave **A**s box, choose a drive and folder where you want to save your LDO.
3. Enter a name in the **F**ile **n**ame: box.
4. Click **S**ave.

To save an existing file

- Choose **F**ile | **S**ave or the  button.

To save all files open within PiCPro

- Choose **F**ile | **S**ave **A**ll.

If any of the open files have not been saved previously, PiCPro prompts you to choose a location to save the file to. Type in a name in the **F**ile **n**ame: box and click **S**ave.

Note: This command will update the Windows time stamp on all of the open files regardless of whether changes were actually made.

Saving Files with a Different Name

You can use the **Save As** command to change the name and/or the location of a file. By giving the file a different name when you save it, you create a copy of the existing file while keeping the original intact.

To save a file under a different name

1. Choose **File | Save As** from the menu.
2. Select the location you want to save the file to in the **Save in:** box.
3. Enter the new file name in the **File name:** box.
4. Click **Save**.

Closing Files

If the file has changed but has not yet been saved, PiCPro asks if you want to save the file before closing it. Choose **Yes** to save the file before closing, choose **No** to close the file without saving it.

To close a file

- Choose **File | Close**

Exiting PiCPro

Exiting means shutting down PiCPro. It marks the last step in a PiCPro session.

To exit

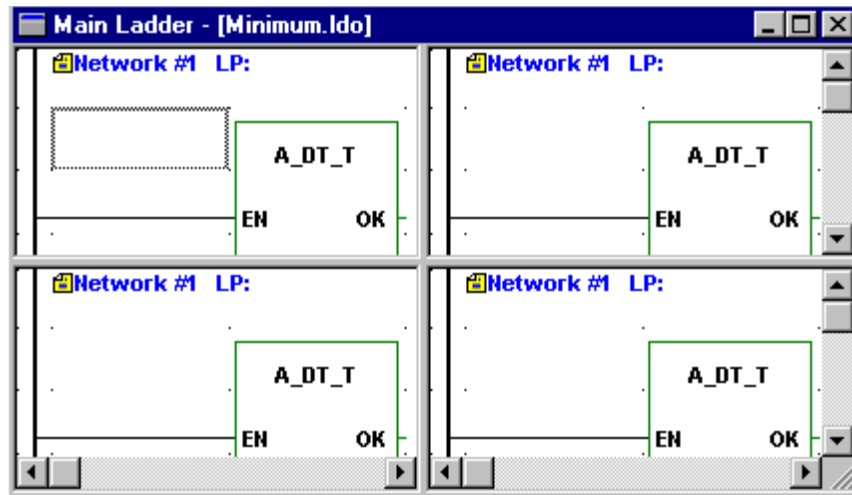
- Choose **File | Exit**.
- or
- Click on the corner symbol on the title bar to drop down a list. Choose **Close** from this list
- or
- Press **<Alt + F4>**.
- or
- Click the **X** button in the upper right corner.
PiCPro asks if you want to save any unsaved changes in any open files.
 - Click **Yes** to save changes before exiting.
 - Click **No** to exit without saving changes.
 - Click **Cancel** to exit the dialog box and continue with your application.

Working with a Split Screen

You may find it helpful to use the split screen feature when you want to view sections of a ladder at the same time. Each section will have its own scrolling capability so you can move to any ladder location you want.

Note: Click within the quadrant you want to scroll before using the scroll bar.

The screen can be split horizontally and/or vertically.



To split the screen

1. To split the screen horizontally, move the arrow to the top section (encircled below) of the vertical scroll bar.



2. When the arrow head changes to the double line/double arrow, click the mouse and drag the horizontal line into your ladder and release.
3. To split the screen vertically, move the arrow to the left section (encircled below) of the horizontal scroll bar.



4. When the arrow head changes to the double line/double arrow, click the mouse and drag the vertical line into your ladder and release.

To unsplit the screen

- Double click on the vertical or horizontal split line to remove the split screen and return to your normal view.

Hardware Declarations

Hardware declarations will differ based on whether you declare a PiC CPU, standalone MMC CPU, or MMC for PC CPU. If you declare a CPU that differs from what is actually installed in the control, you will not be able to download your LDO file to the control.

The hardware configuration in the hardware declarations table is compared to the I/O points defined in the software declarations table when the LDO is compiled. PiCPro can detect an error such as an output that is defined at an input location. An error message will be displayed when you issue the **Compile Only** or the **Compile & Download** command.

In the hardware declarations table, you define what type of hardware module occupies each slot in the master rack. Based on the type of CPU selected, you can also define the other types of I/O in your system such as remote I/O expansion racks, block I/O, ASIU I/O, Ethernet, or DeviceNet modules.

If you later change the CPU type from what was originally declared, you will need to review the I/O points defined in software declarations to ensure that your I/O points are correctly assigned.

Hardware Declaration Restrictions

When making hardware declarations, you must first determine that the CPU supports the I/O in your system.

	PiC 90 CPU	PiC 900 CPU	Standalone MMC	MMC for PC
Remote I/O Expansion Rack	NO	YES	NO	NO
Block I/O Module	YES	YES (1)	YES	YES
ASIU I/O Module	NO	NO	NO	YES
Ethernet	YES	YES	YES	YES (2)
DeviceNet	YES	YES	YES	YES

(1) Not all PiC 900 and PiC90 CPUs support block I/O modules. Check the hardware manuals for these CPUs for complete information.

(2) MMC for PC does not require a separate Ethernet module. PiC and standalone MMC require separate module.


For CPUs that accept Remote I/O Expansion racks, up to 7 racks can be added.

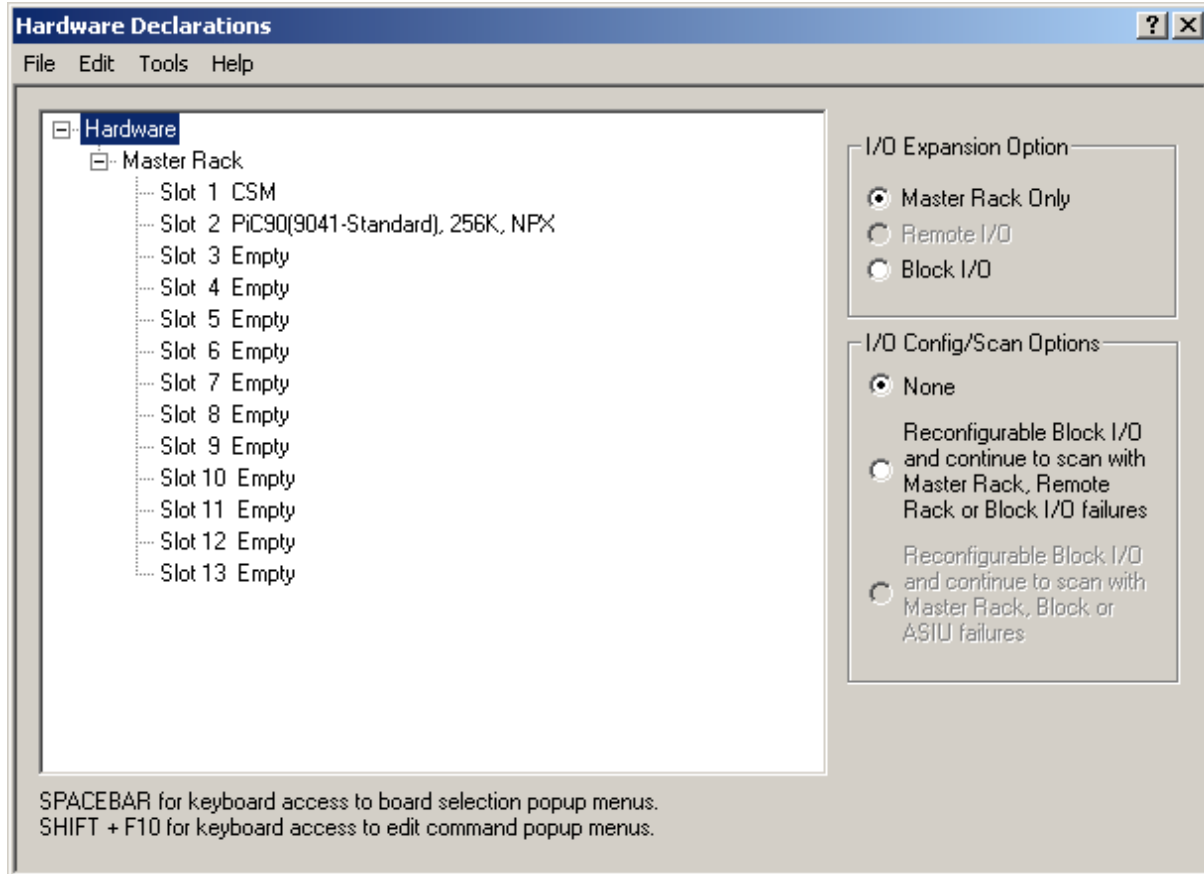
For CPUs that accept Block I/O modules, up to 77 modules can be added.

For CPUs that accept ASIU I/O modules, up to 8 modules can be added.

Entering Hardware Declarations

To bring up the Hardware Declarations table


Choose **V**iew | **H**ardware Declarations or select the  hardware button from the View Navigator toolbar. The following dialog is displayed:



The following describes the Hardware Declarations dialog:

Control	Description
<i>I/O Expansion Option</i>	
Master Rack Only	Select this option if your hardware configuration consists of a master rack only.
Remote I/O	Select this option if your hardware configuration contains remote I/O.
Block I/O	Select this option if your hardware configuration contains Block I/O.
<i>I/O Config/Scan Options</i>	
None	Select this radio button to disable Ladder Configurable I/O and prevent continued scan of a ladder with I/O errors.
Reconfigurable Block I/O and continue to scan with Master Rack, Remote Rack or Block I/O failures	Select this radio button to enable Ladder Configurable I/O and allow the ladder to continue scanning with Master Rack, Remote Rack or I/O errors. Note: If an ASIU error is present, the ladder will not continue to scan. The function block IO_CFG must be used to allow the ladder to react to I/O errors.
Reconfigurable Block I/O and continue to scan with Master Rack, Block or ASIU failures	Select this radio button to enable Ladder Configurable I/O and allow the ladder to continue scanning with Master Rack, Block or ASIU errors. Note: This option is only available if an MMC for PC Analog Servo CPU is selected.

To enter hardware declarations - master rack (all CPU types)

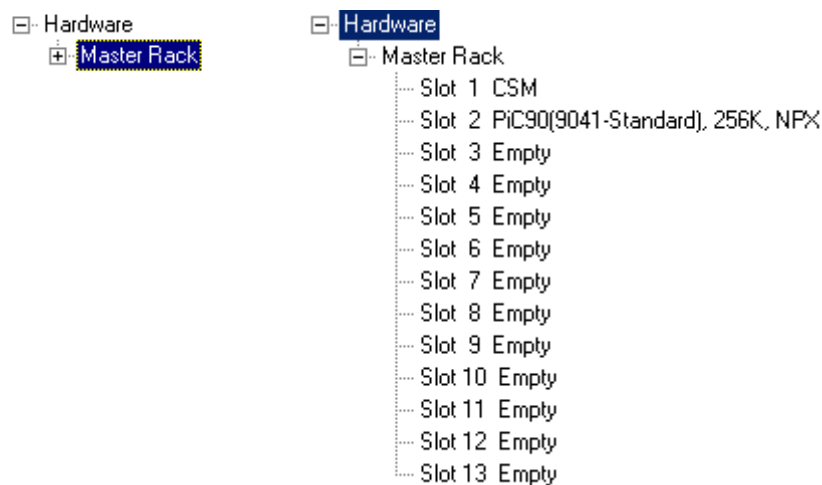
1. Position the cursor on the slot you want to declare a hardware module in and left click. A fly-out box appears. If an arrow  is at the end of the item on the fly-out list, there are more choices.
2. Slide the mouse over each fly-out list highlighting the correct description of the module you want to declare.
3. When you are at the final fly-out list, left click the mouse to make your selection. Your choice will be listed in the master rack list. **Note:** if you want a brief description of the module, press <F1> before making your selection.
4. To add remote I/O (PiC CPU only) or block I/O, click the checkbox. You can then use the fly-out lists to configure the I/O.
5. To enable **I/O Config/Scan Options**, select the appropriate radio button.
6. To save your declarations or changes and return to your ladder, use **File | Save & Close** or <F10> or <Enter>. To exit without saving your changes, use **File | Close** or <Esc>.

You can access various editing commands (Copy, Paste, Insert, etc.) with a right click of the mouse or from the **Edit** menu.

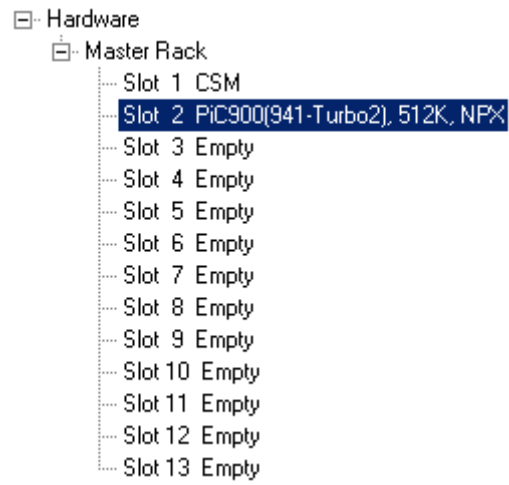
You can also use the keyboard for selecting modules by pressing the spacebar. Then use the up and down arrow keys to scroll, and press <Enter> to select a module. You can use the keyboard to access the editing commands by pressing <Shift + F10>.

Viewing Hardware Declarations

You can change the view of the hardware declarations list using the plus (+) and minus (-) symbols to expand and contract the branches of the tree.



Hardware Declarations Table - PiC CPU



The Hardware Declarations Table is described on the following pages.

Master Rack

Slot 1 is reserved for the Central Services Module

Slot 2 is reserved for the CPU module as follows:

PiC90	▶	9041-Standard
PiC900	▶	9043-Turbo
MMC	▶	
MMC for PC	▶	

The five memory types are available for all four PiC900 models

PiC90	▶		
PiC900	▶	941-Turbo2	▶ 384K
MMC	▶	943-Turbo2	▶ 512K
MMC for PC	▶	945-Turbo3	▶ 640K
		947-Turbo3	▶ 704K
			▶ 768K

PiC90	▶	
PiC900	▶	
MMC	▶	MMC 2-Axis Analog Servo
MMC for PC	▶	MMC 4-Axis Analog Servo
		MMC Digital Servo (1 Ring)

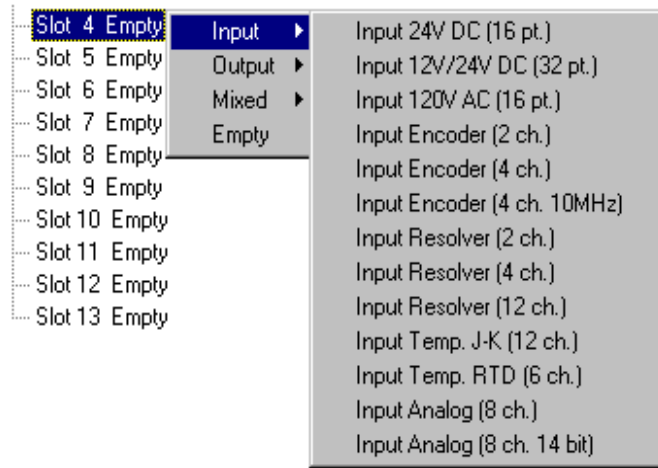
PiC90	▶		
PiC900	▶		
MMC	▶		
MMC for PC	▶	MMC for PC - Digital Servo (1 ring - 8 Slaves)	▶ 384K
		MMC for PC - Digital Servo (1 ring - 16 Slaves)	▶ 512K
		MMC for PC - Digital Servo (1 ring - 32 Slaves)	▶ 640K
		MMC for PC - Analog Servo	▶ 704K
			▶ 768K

The five memory types are available for all four MMC for PC models

Slots 3 through 13 define the type and location of the I/O modules.

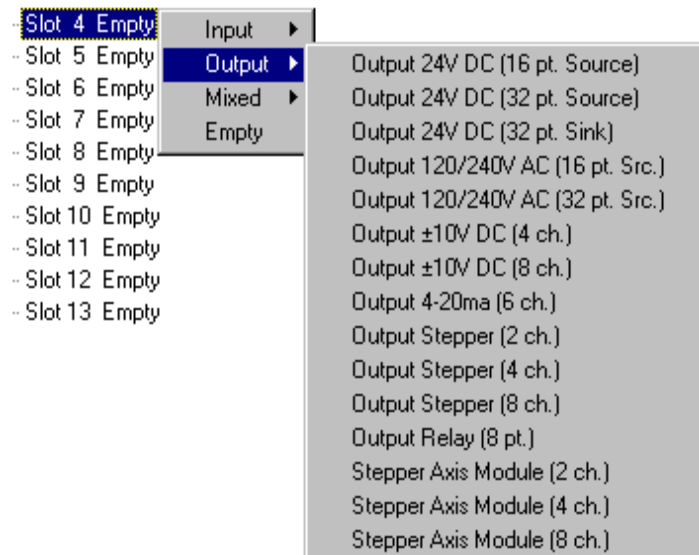
PiC I/O -- Inputs

Slots 3 - 13 Define the type and location of the I/O modules.



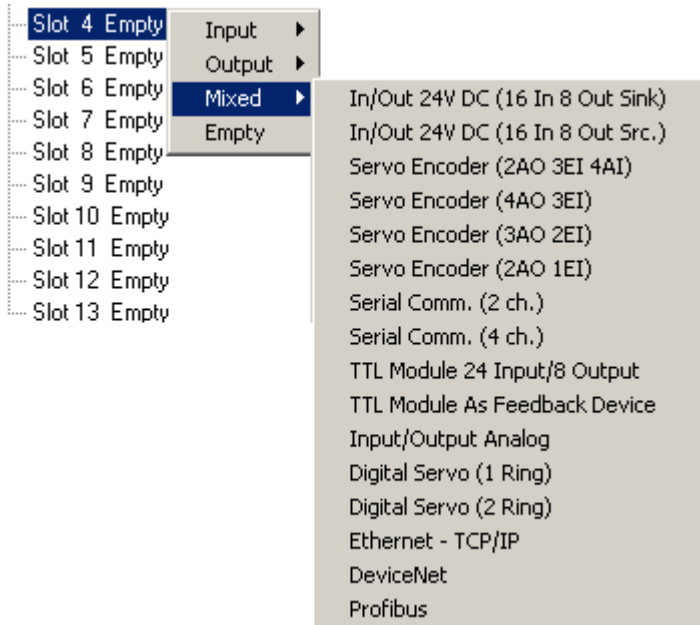
PiC I/O -- Outputs

Slots 3 - 13 Define the type and location of the I/O modules.



PiC I/O -- Mixed I/O

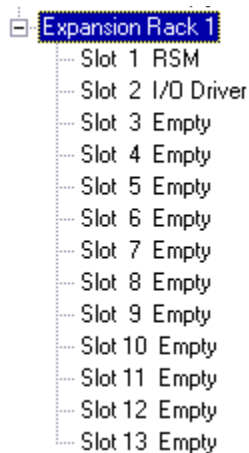
Slots 3 - 13 Define the type and location of the I/O modules.



Hardware Declarations Table - Remote I/O Expansion Rack

Expansion Rack

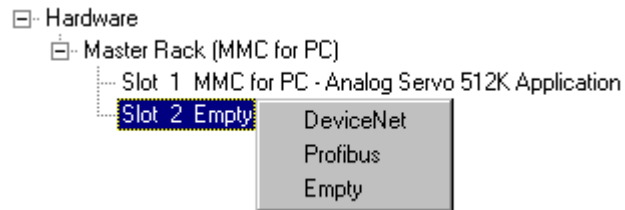
Slot 1 Reserved for Remote Services Module
Slot 2 Reserved for I/O Driver Module
Slots 3 - 13 Define the type and location of the additional I/O modules.



Hardware Declarations Table - MMC for PC CPU

Master Rack (Board mounted in PC)

Slot 1	MMC for PC CPU (Analog or Digital Servo)
Slot 2	Reserved for the DeviceNet or Profibus module

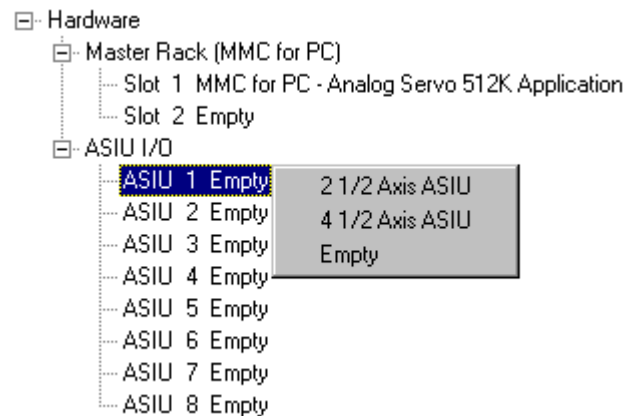


MMC for PC ASIU I/O

Analog Servo Interface Units

Slot 1 - 8 (Only available when an Analog Servo CPU is specified)

A left click on an ASIU "Slot" will display the selection menu shown. The ASIU "slot" number refers to the physical switch setting on the hardware.

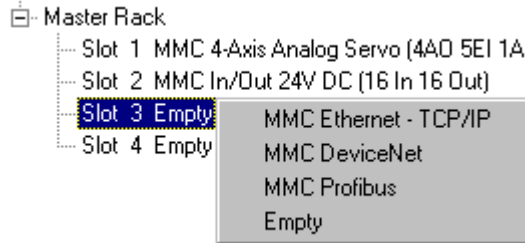


Note: If an MMC for PC Digital Servo is selected, the ASIU I/O branch will not be shown on the hardware tree.

Hardware Declarations Table - Standalone MMC CPU

Master Rack

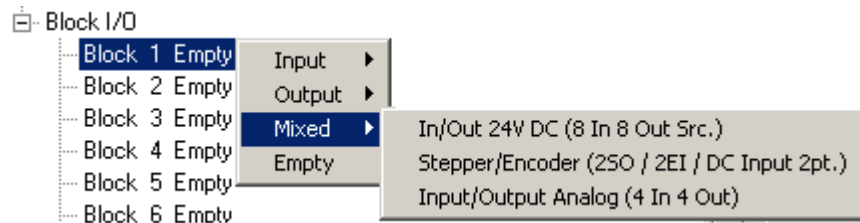
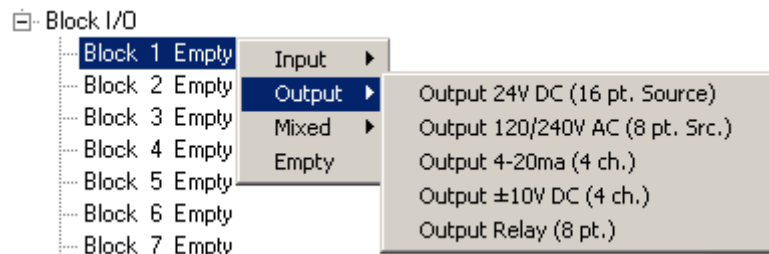
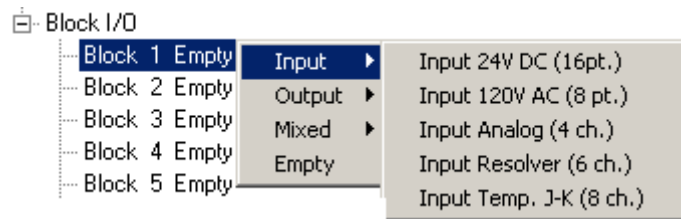
Slot 1	2 1/2 or 4 1/2 Axes Servo
Slot 2	MMC CPU and General I/O
Slots 3 - 4	Reserved for Ethernet-TCP/IP, DeviceNet, or Profibus modules



Hardware Declarations Table - Block I/O Module

Block I/O Modules

Blocks 1 - 77 Define the type and location of the I/O modules. Select the module type and then the correct module from the fly-out list provided



Editing Hardware Declarations

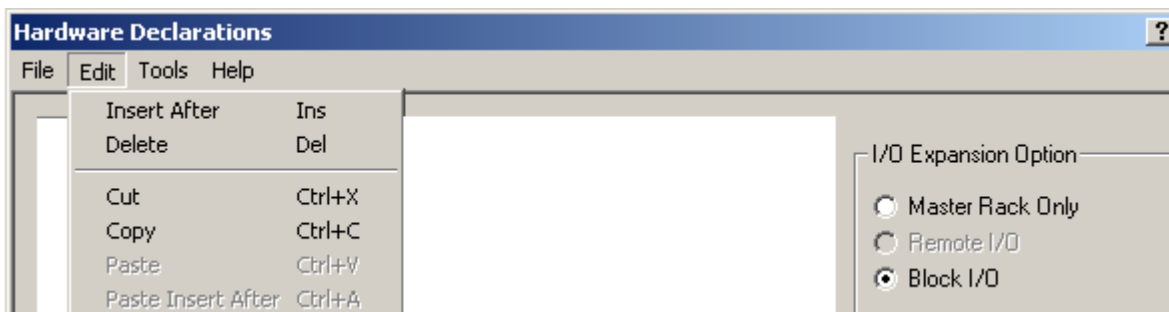
The editing commands in the hardware declarations table are insert, delete, cut, copy, and paste.

After entering the first Remote I/O expansion rack, you can add additional expansion racks by using the **Insert After** command.

To insert additional remote I/O expansion racks or block I/O

1. Select the device after which you want to insert another device.
2. Choose **Edit | Insert After** or **<Insert>**. The new device will be placed directly after the one you selected. If there are any racks or blocks after it, they will be renumbered.

Note: Block I/O will only be incremented up to the first empty block. For example, if you have Block I/O modules in blocks 1 through 5, and 6 through 9 are empty, and 10 through 15 contain Block I/O modules, and you **Insert After** at block 2, only 3, 4, and 5 will shift becoming 4, 5, and 6. The block modules in 10 through 15 are unaffected. You will also be prompted to have the corresponding Block I/O points adjusted in Software Declarations. For this example, all I/O points for block 3 would be changed to block 4, etc. You cannot insert Block I/O that would cause a block defined at block 77 to be replaced. You will be prompted to remove that block first.



To delete a remote I/O expansion rack or a block I/O

1. Select the object you want to delete.
2. Choose **Edit | Delete** or **<Delete>**. A warning message appears. The object will be deleted if you choose **OK**. With a remote I/O expansion rack, if there are any racks after the one you deleted, they will be renumbered.

Note: When you delete a Block I/O module, remaining modules are moved up in the list to the first empty block. For example, if blocks 1 through 6 are filled with Block I/O modules, 7 through 9 are empty, and 10 through 15 contain Block I/O modules, and you delete block 4, then 5 and 6 become 4 & 5, 6 is empty, and 10 through 15 are unaffected. Any Software Declarations specifying I/O points for block 4 will be modified to remove the I/O points. You are prompted to have the Software Declarations automatically adjusted. For this example, all I/O points for block 5 will be changed to block 4, etc.

Cutting and Copying Hardware Declarations

The **C**opy and **C**ut commands let you use the clipboard to create copies of some selected objects in the declarations table. The commands are context sensitive and will be grayed if they do not apply to where you are in the table.

The **C**opy command places a copy of the selected object on the clipboard.

The **C**ut command removes the object from the table and places it on the clipboard.

Copy command **can** be used on these selected objects:

- Entire Hardware Declarations Table
- Master Rack
- Expansion Rack(s)
- Block I/O Branch
- ASIU I/O Branch
- CPU, I/O, Modules

Copy command **cannot** be used on:

- Modules that cannot be changed

Cut command **can** be used on these selected objects:

- Expansion Racks
- I/O and Block I/O Modules
- Block I/O Branch
- ASIU I/O Modules

Cut command **cannot** be used with:

- Entire Hardware Declarations Table
- Master Rack
- CPU and modules that cannot be changed
- I/O in slot 1

To copy an object in the hardware declarations table:

1. Select the object.
2. Choose **E**dit | **C**opy or <Ctrl + C> to copy the selected object to the clipboard.

To cut an object in the hardware declarations table:

1. Select the object.
2. Choose **E**dit | **C**ut or <Ctrl + X> to cut the selected object.

Things to note about cutting objects from the hardware declarations table:

- When you delete a Block I/O module, remaining modules are moved up in the list to the first empty block. For example, if blocks 1 through 6 are filled with Block I/O modules, 7 through 9 are empty, and 10 through 15 contain Block I/O modules, and you delete block 4, then 5 and 6 become 4 & 5, 6 is empty, and 10 through 15 are unaffected. Any Software Declarations specifying I/O points for block 4 will be modified to remove the I/O points. You are prompted to have the Software Declarations automatically adjusted. For this example, all I/O points for block 5 will be changed to block 4, etc.
- When you cut an expansion rack or a block branch, the rack or block branch is removed. With expansion racks, any remaining racks are renumbered.

Pasting and Paste Insert in Hardware Declarations

Once an object is on the clipboard, you can use the **P**aste command or the **P**aste **I**nsert **A**fter command to place the object back into the table. The object remains on the clipboard until you cut or copy another object onto the clipboard. Only one object can be placed on the clipboard at a time.

Using the Paste command

1. Select the location where you want the paste to occur.
2. Choose **E**dit | **P**aste or <Ctrl + V>. The object on the clipboard will replace the selected object.

Using the Paste Insert After command

1. Select the location where you want the paste insert after to occur.
2. Choose **E**dit | **P**aste **I**nsert **A**fter or <Ctrl + A>. The object on the clipboard will be inserted after the location you selected.

Things to note about pasting objects in the hardware declarations table

- With the **Paste** command, the object on the clipboard must match the selected object in the table.
- With the **Paste Insert After** command, the object on the clipboard must be an expansion rack or a block module. The selected object can be the master rack, an expansion rack, the block I/O root, or an individual block module.
- With the **Paste Insert After** command, Block I/O will only be incremented up to the first empty block. For example, if you have Block I/O modules in blocks 1 through 5, and 6 through 9 are empty, and 10 through 15 contain Block I/O modules, and you **Paste Insert After** at block 2, only 3, 4, and 5 will shift becoming 4, 5, and 6. The block modules in 10 through 15 are unaffected. You will also be prompted to have the corresponding Block I/O points adjusted in Software Declarations. For this example, all I/O points for block 3 would be changed to block 4, etc. You cannot insert Block I/O that would cause a block defined at block 77 to be replaced. You will be prompted to remove that block first.

Pasting to Software Declarations

You can copy I/O information for a standalone MMC or MMC for PC to the clipboard and paste it in Software Declarations.

Right click on a standalone MMC module (slot 1 or 2) or an MMC for PC ASIU module, select **Copy I/O to clipboard**. Declarations for all I/O points on that module will be placed on the clipboard.

Printing Hardware Declarations

The hardware declarations table can be printed using the **P**rint command from the **F**ile menu for the application. The table will be printed in its expanded form regardless of how you have it displayed when the command is issued.

To print your hardware declarations

1. Choose **F**ile | **P**rint or <Ctrl + P>.
2. The **P**rint dialog box appears. Choose **OK** to proceed or **Cancel** to exit.

Closing and Saving Hardware Declarations

You can close the hardware declarations table and save any changes or close without saving changes.

Closing and saving the hardware declarations table

1. If you choose **F**ile | **S**ave & **C**lose or <F10>, your table will close and any changes will be saved.
2. If you choose **F**ile | **C**lose or <Esc>, your table will close and no changes will be saved.

Software Declarations

In the software declarations table, you enter the following information for your ladder:


- The name and data type of every variable, contact, coil, function block, structure and array used.
- The location of every physical input or output used.
- The initial or default value of any entry if required.
- The retentive, global, external, or UDFB in or out attribute of any variable if required.
- A descriptive long name if desired.

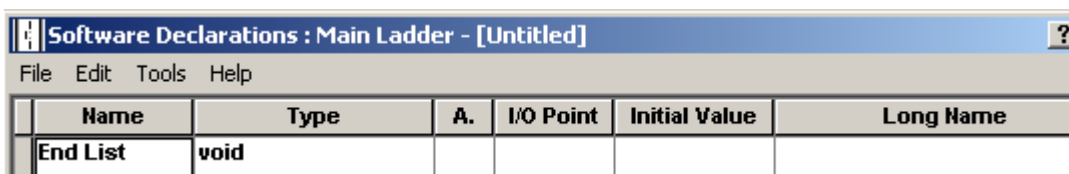
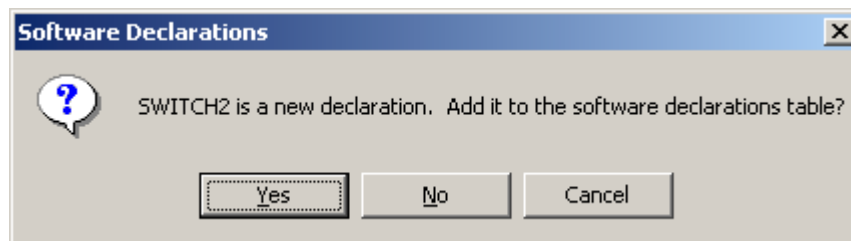
You can perform the following editing functions in the software declarations table:

- Inserting and deleting entries
- Finding declared entries
- Cutting, copying, and pasting entries
- Purging the table of entries that are not used in your ladder

Entering Software Declarations

There are several ways to access the software declarations table in order to declare entries.

- Select **Software Declarations** from the **View** menu.
- Select the software declarations button  from the view navigator toolbar.
- When you enter a new variable in your ladder and it has not been declared yet, answer **Yes** to this box and the software declarations table will appear.

A screenshot of the software declarations table in a window titled "Software Declarations : Main Ladder - [Untitled]". The table has a menu bar with "File", "Edit", "Tools", and "Help". The table has the following structure:

Name	Type	A.	I/O Point	Initial Value	Long Name
End List	void				

You can add the variable to the declarations. There are six columns in the table in which you enter information.

- The name and data type of every variable, contact, coil, function block, structure and array used must be entered in the Name and Type columns.
- If required, the retentive, global, external, or UDFB in or out attribute of any variable is entered in the A. (or Attribute) column.
- The location of every physical input or output used is entered in the I/O Point column.
- The initial or default value of any entry if required is entered in the Initial Value column.
- A descriptive long name if desired can be entered in the Long Name column.

Names and Long Names of Variables

Names

In the **Name** column of the software declarations table, type in up to 63 alphanumeric/underscore characters.

- The first character must be alpha (A-Z, a-z) or the underscore (_).
- The 2nd through 63rd character can be alpha, numeric (0-9), or underscore.
- Mixed case characters are allowed.

Note: No distinction is made between names having the same characters with different case (e.g. Switch1 and SWITCH1). If there is an existing variable named SWITCH1 and a new variable named Switch1 is inserted, an error message box will be displayed indicating there is already another variable with the same name.

If the name is changed in this column, every occurrence of the variable in your ladder will be changed. The only exception is an ST Network that contains syntax errors. In this case the name may not be changed.

Long Names

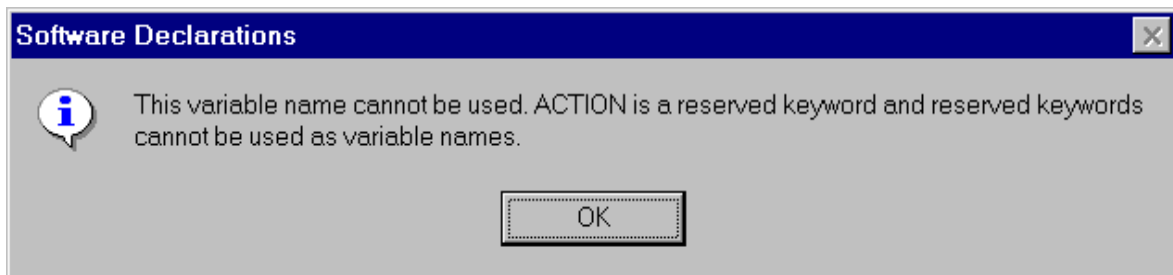
In the Long Name column, you can add a long name to any variable used in a network. Long names can have up to 40 ASCII characters on four lines (10 per line). You can view, edit or add a long name to a variable in a LD network or to the Long Name column in the software declarations table. When you add or change a long name for a variable in a network, PiCPro automatically updates the Long Name column in the software declarations table and vice versa. You can choose to display or hide the long names in your LD Networks.

Reserved Keywords

The following are reserved Keywords that are not allowed as variable names:

ACTION	ADD	AND	ANDN
ARRAY	AT	BOOL	BY
BYTE	CAL	CALC	CALCN
CASE	CONFIGURATION	CONSTANT	D
DATE	DATE_AND_TIME	DINT	DIV
DO	DS	DWORD	ELSE
ELSIF	END_ACTION	END_CASE	END_CONFIGURATION
END_FOR	END_IF	END_FUNCTION	END_FUNCTION_BLOCK
END_PROGRAM	END_REPEAT	END_RESOURCE	END_STEP
END_STRUCT	END_TRANSITION	END_TYPE	END_VAR
END_WHILE	ENO	EQ	EXIT
FOR	FROM	FUNCTION	FUNCTION_BLOCK
F_EDGE	GE	GT	IF
INITIAL_STEP	INT	JMP	JMPC
JMPCN	L	LD	LDN
LE	LINT	LREAL	LWORD
LT	MOD	MUL	N
NE	NOT	OF	ON
OR	ORN	P	PROGRAM
R	READ_ONLY	READ_WRITE	REAL
REPEAT	RESOURCE	RET	RETURN
R_EDGE	S	SD	SINT
SL	ST	STEP	STN
STRING	STRUCT	SUB	TASK
THEN	TIME	TIME_OF_DAY	TO
TRANSITION	TYPE	UDINT	UINT
ULINT	UNTIL	USINT	VAR
VAR_ACCESS	VAR_EXTERNAL	VAR_GLOBAL	VAR_INPUT
VAR_IN_OUT	VAR_OUTPUT	WITH	WHILE
WORD	XOR	XORN	

If you attempt to enter a variable name that is a reserved keyword (e.g. ACTION), the following message box will appear:



Press OK and enter a variable name that is not a reserved keyword.

I/O Points

In the **I/O Point** column, enter the location of physical inputs and outputs only.

- For hardware modules located in the master or expansion rack, the rack and slot location of the module and the channel location of the input or output are defined. The hardware module must be declared in the Hardware Declarations table.
- For block I/O modules, the block number and the point number are defined.
- For ASIU I/O modules, the ASIU number and the point number are defined.

The data type in the **Type** column must be boolean in order to enter information in the I/O Point column.

The format of I/O points defined in software declarations is different based on the CPU type declared in hardware declarations. The definition of Block I/O points is unaffected by CPU choice.

Standalone MMC I/O Points

Both PiCPro Professional and the Standalone MMC Edition allow you to copy and paste the I/O information to the software table from the Hardware Declarations table. This can be done for the I/O on the MMC CPU module and the MMC analog module.

The information can be copied by:

- Selecting the CPU (Slot 2) or Analog Module (Slot 1) section in the hardware declarations table.
- Right clicking and choosing **Copy I/O to Clipboard**.
- Closing the hardware table and viewing the software table.
- Pasting the information into the software declarations table.

All necessary information will be entered in the software declarations table.

Below is a list of the I/O Point labels for the general connector on the MMC CPU module and for the axis and auxiliary connectors on the analog module. The information in column 2 will be pasted in the **Name** column and the information in column 3 will be pasted in the **I/O Point** column of the software declarations table.

Discrete point	Declared Name	Software Declaration I/O assignment (MMC I/O Point)
16 general DC Inputs	GENI1 –GENI16	IGEN.1 through IGEN.16
16 general DC Outputs	GENO1 –GENO16	OGEN.1 through OGEN.16
2 short circuit Inputs	SHORT1-SHORT2	ISGEN.1 through ISGEN.2
6/12 auxiliary DC Inputs	AUXI1-AUXI12	IAUX.1 through IAUX.12
Axis 1 DC input	AX1READY	IA1.1 (Axis 1, Input 1)
Axis 2 DC input	AX2READY	IA2.1
Axis 3 DC input	AX3READY	IA3.1
Axis 4 DC input	AX4READY	IA4.1
Axis 1 DC output	AX1ENABL	OA1.1 (Axis 1 Output 1)
Axis 2 DC output	AX2ENABL	OA2.1
Axis 3 DC output	AX3ENABL	OA3.1
Axis 4 DC output	AX4ENABL	OA4.1
Axis 1 DC output	AX1RESET	OA1.2 (Axis 1 Output 2)
Axis 2 DC output	AX2RESET	OA2.2
Axis 3 DC output	AX3RESET	OA3.2
Axis 4 DC output	AX4RESET	OA4.2

Note: If you later change the CPU type in hardware declarations, the I/O points might be automatically changed in software declarations. Refer to Appendix B, Conversion References for more information.

Fast Inputs

The following I/O points can be **manually entered** to software declarations if desired.

Axis 1 fast input	AX1FINPT	IFAUX.1 (aux port, Axis 1, Fast Input)
Axis 2 fast input	AX2FINPT	IFAUX.2
Axis 3 fast input	AX3FINPT	IFAUX.3
Axis 4 fast input	AX4FINPT	IFAUX.4
Axis 49 fast input	DIGFINPT	IFAUX.49 (aux port, Digitize 49, Fast Input)
Expansion input (future)		I3.1 – I3.x

MMC for PC I/O Points

PiCPro Professional Edition is used to program the I/O points for MMC for PC CPU types. If an MMC for PC CPU is specified in the hardware declarations, all I/O points must be MMC for PC I/O points.

I/O points can be manually entered or copied from one ladder to another. The ASIU I/O points can be copied to the clipboard from Hardware Declarations, and pasted into the Software Declarations. **Note:** Declarations for the Fast Inputs available for each axis must be manually entered into Software Declarations. They cannot be copied to the clipboard from Hardware Declarations.

ASIU I/O

The following table describes the MMC for PC, Analog Servo Interface Unit (ASIU) I/O points. The information in column 2 is the default declared name which will be pasted into the **Name** column if the ASIU I/O is copied to the clipboard from Hardware Declarations. The information in column 3 will be pasted in the **I/O point** column. The ASIU number is included in both the declared name and the I/O assignment. In the following table, “#” represents the ASIU number.

Discrete point	Declared Name	Software Declaration I/O assignment
16 general DC Inputs	GENI#_1 - GENI#_16	IGEN#.1 - IGEN#.16
16 general DC Outputs	GENO#_1 - GENO#_16	OGEN#.1 - OGEN#.16
2 short circuit Inputs	SHORT#_1 - SHORT#_2	ISGEN#.1 - ISGEN#.2
6/12 auxiliary DC Inputs	AUXI#_1 - AUXI#_12	IAUX#.1 - IAUX#.12
Axis 1 DC Input	AX1RDY#	I#A1.1 (ASIU #, Axis 1, Input 1)
Axis 2 DC Input	AX2RDY#	I#A2.1
Axis 3 DC Input	AX3RDY#	I#A3.1
Axis 4 DC Input	AX4RDY#	I#A4.1
Axis 1 DC Output	AX1EN#	O#A1.1 (ASIU #, Axis 1, Output 1)
Axis 2 DC Output	AX2EN#	O#A2.1
Axis 3 DC Output	AX3EN#	O#A3.1
Axis 4 DC Output	AX4EN#	O#A4.1
Axis 1 DC Output	AX1RES#	O#A1.2 (ASIU #, Axis 1, Output 2)
Axis 2 DC Output	AX2RES#	O#A2.2
Axis 3 DC Output	AX3RES#	O#A3.2
Axis 4 DC Output	AX4RES#	O#A4.2

Note: If you later change the CPU type in hardware declarations, the I/O points might be automatically changed in software declarations. Refer to Appendix B, Conversion References for more information.

MMC for PC Fast Inputs

The following I/O points can be **manually entered** to software declarations if desired.

Axis 1 fast input	AX1FIN#	IFAUX#.1 (aux port, Axis 1, Fast Input)
Axis 2 fast input	AX2FIN#	IFAUX#.2
Axis 3 fast input	AX3FIN#	IFAUX#.3
Axis 4 fast input	AX4FIN#	IFAUX#.4
Axis 49 fast input	DIGFIN#	IFAUX#.49 (aux port, Digitize 49, Fast Input)

Numbering

The manner in which I/O points are displayed changes based on the type of CPU selected in Hardware Declarations (PiC, Standalone MMC, or MMC for PC).

PiC CPU

The master or CPU rack is #0. Expansion racks are numbered 1 - 7, where #1 is the rack connected to the master, #2 is the rack connected to #1, etc. Slots are numbered left to right when facing the PiC rack. Slot 1 and slot 2 are reserved for the CSM/CPU module. On an expansion rack, slot 2 is reserved for the I/O driver module.

Master Rack, PiC CPU

Enter four to six characters.

1 st	I or O	Input or Output
2 nd	0 - 1	First digit of module slot number* (can omit if 0)
3 rd	0 - 9	Second digit of the module slot number*
4 th	. (point)	Used as a separator
5 th	0 - 3	First digit of channel number** (can omit if 0)
6 th	0 - 9	Second digit of channel number**

* Valid slot numbers are 3 - 13.

**Valid channel numbers are 1 - 64.

Example:

If the input is in the master rack at slot 4, channel 3, enter: I4.3

Expansion Rack I/O

Note: Expansion Rack I/O is only available if PiC CPU is chosen.
Expansion Rack I/O is not available for a standalone MMC or MMC for PC CPU.

Enter six to eight characters.

1 st	I or O	Input or Output
2 nd	1 - 7	Expansion rack number
3 rd	. (point)	Used as a separator
4 th	0 - 1	First digit of module slot number* (can omit if 0)
5 th	0 - 9	Second digit of the module slot number*
6 th	. (point)	Used as a separator
7 th	0 - 3	First digit of channel number** (can omit if 0*)
8 th	0 - 9	Second digit of channel number**

*Valid slot numbers are 3 - 13. **Valid channel numbers are 1 - 64.

Example:

If the output is from expansion rack #7 at slot 12, channel 10, enter: O7.12.10

Master Rack Standalone MMC CPU

1 st	I or O	Input or Output
2 nd	GEN, AUX, A1, A2, A3, A4, FAUX	Connector/Type
3 rd	. (point)	Used as a separator
4 th	0 - 4	First digit of channel number
5 th	0 - 6	Second digit channel number

ASIU I/O for MMC for PC CPU

(Two Options)

1 st	I or O	Input or Output
2 nd	GEN, AUX, FAUX	Connector/Type
3 rd	1 - 8	ASIU number
4 th	. (point)	Used as a separator
5 th	0 - 4	First digit of channel number
6 th	0 - 6	Second digit channel number

1 st	I or O	Input or Output
2 nd	1 - 8	ASIU number
3 rd	A1, A2, A3, A4	Connector/Type
4 th	. (point)	Used as a separator
5 th	1 - 2	First digit of channel number

Block Expansion Rack I/O

Note: Block Expansion Rack I/O is only available for certain CPUs.

Enter five to seven characters.

1 st	B	Block
2 nd	I or O	Input or Output
3 rd	0 - 7	First digit of module number* (can omit if 0)
4 th	0 - 9	Second digit of the module number*
5 th	. (point)	Used as a separator
6 th	0 - 6	First digit of point number** (can omit if 0)
7 th	0 - 9	Second digit of point number**

*Valid block module numbers = 1 - 77.

**Valid point numbers = 1 - 64.

Example

If the input is in block I/O module 33, point 5, enter: BI33.5

Blown Fuse Status

The status of up to four fuses on an AC Output, DC Output, or a combination I/O module can be made available to your ladder program. You declare the fuses as inputs in the software declarations table.

On modules having both inputs and outputs, the points are numbered sequentially (starting at 1 for inputs and starting at 1 for outputs) in the software declarations table as shown in the two examples below.

The 24 V DC Output 16 point module with four fuses			The 24V I/O 16/8 source module with two fuses		
Name	Type	I/O Point	Name	Type	I/O Point
OUT1	BOOL	O4.1	IN1	BOOL	I5.1
OUT2	BOOL	O4.2	IN2	BOOL	I5.2
.
.
OUT16	BOOL	O4.16	IN16	BOOL	I5.16
FB1	BOOL	I4.1	OUT1	BOOL	O5.1
FB2	BOOL	I4.2	OUT2	BOOL	O5.2
FB3	BOOL	I4.3	.	.	.
FB4	BOOL	I4.4	OUT8	BOOL	O5.8
			FB1	BOOL	I5.17
			FB2	BOOL	I5.18

Short Circuit Detection

The status of the short circuit detection feature of the general DC outputs for the standalone MMC, the MMC for PC ASIU, and the block I/O output modules can be made available to the ladder diagram. There is one circuit for each group of outputs. The module's hardware description defines how many output points are in a common electrical group. You declare the circuit as an input in Software Declarations.

	<u>16 point DC out</u>	<u>Mixed (8 in/ 8 DC out)</u>
Block I/O	BI#.1 or 2	BI#.9
Standalone MMC	ISGEN.1 or 2	not applicable
MMC for PC	ISGEN#.1 or 2	not applicable

For example: If the short circuit input is from a 24VDC input and output block I/O, module number 33, there is only one group of outputs, the detection input is 9, enter: BI33.9

If the short circuit input is from the GEN outputs for a standalone MMC, there are two groups of outputs (1 to 8 and 9 to 16), there can be two detection inputs: ISGEN.1 and ISGEN.2. The detection input from an ASIU would also include the ASIU number.

Fast Inputs

PiC CPU

The fast inputs available on the encoder, resolver, and servo encoder hardware modules can be declared as inputs in the software declarations table. The inputs are:

For Channel 1	For Channel 2	For Channel 3	For Channel 4
X.1	X.3	X.5	X.7

Standalone MMC CPU

The fast inputs available on the standalone MMC can be declared as inputs in the software declarations table. The inputs are:

IFAUX.1	Channel 1	(AXIS 1)
IFAUX.2	Channel 2	(AXIS 2)
IFAUX.3	Channel 3	(AXIS 3)
IFAUX.4	Channel 4	(AXIS 4)
IFAUX.49	Channel 5	(AXIS 49)

MMC For PC CPU

The fast inputs available on the MMC for PC can be declared as inputs in the software declarations table. The inputs are (where # is the ASIU number):

IFAUX#.1	Channel 1	(AXIS 1)
IFAUX#.2	Channel 2	(AXIS 2)
IFAUX#.3	Channel 3	(AXIS 3)
IFAUX#.4	Channel 4	(AXIS 4)
IFAUX#.49	Channel 5	(AXIS 49)

Initial Values

- All variables have a default value of zero. By entering a value in the **Initial Value** column, you can change the default value. Default values go into effect:
 - For all variables when a cold restart is performed
 - For variables that are not retentive, when a warm restart is performed

Note: An initial value cannot be set if the variable has an I/O point assigned to it.

Prefixes for initial values

The following prefixes are required for values in the formats listed.

Prefix	Value Format
2#	Binary
8#	Octal
None	Decimal
16#	Hexadecimal
D#	Date
TOD#	TIME_OF_DAY
DT#	DATE_AND_TIME
T#	TIME

Working with Data Types in Software Declarations

When a variable is declared in the software declarations table, you assign a data type to it in the **Type** column by entering in the type or selecting it from the drop-down list. The data type categories and associated data types are listed below.

Bitwise	Numeric	String	Time of day	Time duration
BOOL	SINT	STRING	DATE	TIME
BYTE	INT		TIME_OF_DAY	
WORD	DINT		DATE_AND_TIME	
DWORD	LINT			
LWORD	USINT			
	UINT			
	UDINT			
	ULINT			
	REAL			
	LREAL			
Function/Function block (Any Data Type)		Structures (Any Data Type)		

Bitwise

Bitwise variables/constants are used to represent binary values. When manipulated, these data types are treated as a series of binary digits.

Description	Data Type	# of Bits	Range
A boolean value is a bit. Contacts and coils are always represented by boolean variables. 0 = off/no 1 = on/yes	BOOL boolean	1	0 or 1
Byte, word, double word, and long word values are groups of bits that are used when logical and bit manipulation operations are performed.	BYTE	8	eight 0s - 1s
	WORD	16	sixteen 0s - 1s
	DWORD double word	32	thirty-two 0s - 1s
	LWORD long word	64	sixty-four 0s - 1s

Bitwise values can be entered in binary, octal, decimal, or hexadecimal. The format for entering these values is shown below for the number 23. (2#, 8#, and 16# are required.)

Binary = 2#10111 **Octal** = 8#27 **Decimal** = 23 **Hexadecimal** = 16#17

Note: Bitwise constants are limited to 32 bit values. A constant can be entered for any bitwise data type. For data types 32 bits or less, the constant value must be within the range of the data type. For data types greater than 32 bits, the 32 bit constant value will be extended. Zeros will be placed in the 32 most significant bits. The constant value will be held in the least significant 32 bits.

When booleans are entered as constants, enter the following.

For: Enter:
On 1, (hex) 16#80, or (decimal) 128 can be used interchangeably.
Off 0

For boolean initial values, enter a 0 or 1.

Numeric

Numeric variables/constants are used to represent integers and real or floating point numbers. When manipulated, these types of data are treated as numbers.

Description	Data Type	# of Bits	Range
Integers are whole numbers or zero.	SINT short integer	8	-128 to +127
	INT integer	16	-32,768 to +32,767
	DINT double integer	32	-2,147,483,648 to +2,147,483,647
	LINT long integer	64	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
Unsigned integers are greater than or equal to zero. PiCPro will prevent you from entering a negative number for an unsigned integer.	USINT unsigned short integer	8	0 to 255
	UINT unsigned integer	16	0 to 65,535
	UDINT unsigned double integer	32	0 to 4,294,967,295
	ULINT unsigned long integer	64	0 to 9,223,372,036,854,775,807
Real numbers are floating point numbers. They contain a decimal point and an exponent indicating the power of ten the number is to be multiplied by to obtain the value represented.	REAL	32	significant digits 6-7
	LREAL long real	64	significant digits 15-16

String

String variables are used to represent a sequence of zero or more characters. The characters are ASCII and /or extended ASCII.

Description	Data Type	Length in bytes (internal)	Range
String type variables are used to read or write messages. Every string has two extra bytes that hold information about the string. The second byte tells the actual length of the string. Note: STRINGS cannot be entered as constants.	STRING	Two plus number of declared characters	1 to 255 ASCII characters

String values are keyed in exactly as they are to be interpreted. A three character combination of the dollar sign (\$) followed by two hexadecimal digits is interpreted as the hex representation of an eight-bit ASCII character code. A two character combination of the dollar sign followed by a character is the ASCII interpretation given below.

Character String	Interpretation
\$\$	dollar sign
\$'	single quote
\$L	line feed
\$N	new line
\$P	form feed (page)
\$R	carriage return
\$T	tab

Time

Time of Day

Time of day values represent the time of day, the date, or both.

Description	Data Type	# of Bits	Range	Examples
Date and time values are used when the date or time is to be printed, or when an event is to be triggered at a given moment in time. Values can be assigned manually or extracted from the PIC clock (with functions).	DATE	16	Jan.1, 1988 - Dec. 31, 2051	Year/month/day D#1997-10-23
	TIME_OF_DAY	32	00:00:00 - 23:59:59	Hours/minutes/ seconds TOD#23:59:59
	DATE_AND_TIME	32	Same as the two above, combined	Year/month/day/hours/ minutes/ seconds DT#1997-10-23-23:59:59

Time duration

A time (duration) value represents an amount of time.

Description	Data Type	# of Bits	Range	Examples
Time duration values serve as inputs to timer functions for counting up/down for a specified amount of time. They also serve as outputs into which elapsed time values are entered.	TIME	32	0 - 49d17h2m47s295ms	Days/hours/minutes/ seconds/milliseconds
			0 - 1193h2m47s295ms	
			0 - 71582m47s295ms	T#1d2h3m4s5ms
			0 - 4294967s295ms	
			0 - 4294967295ms	

Values are entered as shown in the examples below where d = day, h = hour, m = minute, s = second, ms = millisecond. T# is required and d, h, m, s, and ms are required if the value for that increment is not zero.

T#1d3h7m16s45ms

T#14d

T#459871ms

Function/Function Block

Constants can be inputs to any Function/Function Blocks except functions that operate on STRINGS (string values must be put into variables). A constant value must be in the range and format it would be in if it were a variable value.

Generally, all input variables and the output variable must have the same data type. However, this is not true for a function whose purpose is to change a data type i.e., converting numeric type data into bitwise type data. It is also not true for inputs that are providing extraneous data for the operation, and outputs that are providing information about extraneous data for the operation. In all other cases though the output data type must match the input data type.

Although input and output data must be of the same type, input and output variables usually do not have to be unique variables. For instance, you could add the constant 1 to a variable call VAR1, and place the result in VAR1.

IMPORTANT

If an output variable is unique from an input variable, the input variable is unaltered by execution of the function/block. Only the output variable is changed.

Functions which return a string type variable as an output have a unique characteristic. The output variable must be assigned on the input (left) side of the function. This is necessary because STRINGS require memory allocations before the operation is performed.

Groups of data - Structures and Arrays

Variables can be grouped to create entities called arrays and structures. This ability to group data enables you to keep various types of data together that have a common link. Values can be read into or written from these groups with I/O functions. Also, individual variables in structures and arrays can serve as inputs to and outputs from Function/Function Blocks.

Groups of variables can be handled by arrays, structures, structures with arrays, an array of structures, and an array of structures with arrays.

Arrays

An array is a group of variables. Each variable in an array must be of the same data type. Any data type is acceptable. An array can have from 2 to 999 variables. These variables are called elements. Arrays are useful for handling large groups of like data items.

- Only single dimensioned arrays are supported in PiCPro
- Array variables are identified with square brackets.
- Array subscripts follow the same rules in ST and LD. An array subscript can be any variable or constant that evaluates to USINT or UINT. Array subscripts can be nested if the variable name used in the subscript is another array.

To declare an array in the software declarations table

1. Enter the name and data type for the array you want to create.
2. If all the elements in the array will have the same initial value, enter the initial value.
3. Choose **Tools | Make Array**, press <Alt + A>, or right click the mouse and select **Make Array**.
4. Enter the array length in the dialog box. The range is from 2 to 999.
5. The index numbers (0...x-1) will appear behind the data type of the variable where x is the defined array length.
6. The word ‘...ARRAY...’ will be displayed in the initial value field if the initial value for any element in the array has been entered.
7. To initialize or change the initial values of the elements in the array, move the cursor to the initial value field and press the <Enter> key. If no initial value was entered before the array was declared, then there will be no value for any elements. If an initial value was entered before the array was declared, that initial value will be duplicated for all elements in the array.

To resize an array in the software declarations table

1. Position the selected cell anywhere on the variable declaration you want to change.
2. Press <Alt + A> or the **Make Array** command.
3. Enter a new array length.
4. The index numbers displayed behind the data type of the variable will be updated.
5. To initialize or change the initial values of the elements in the array, move the cursor to the initial value field.
If the array length is increased the added elements will have the same value as the first element (index 0).

To remove an array from the software declarations table

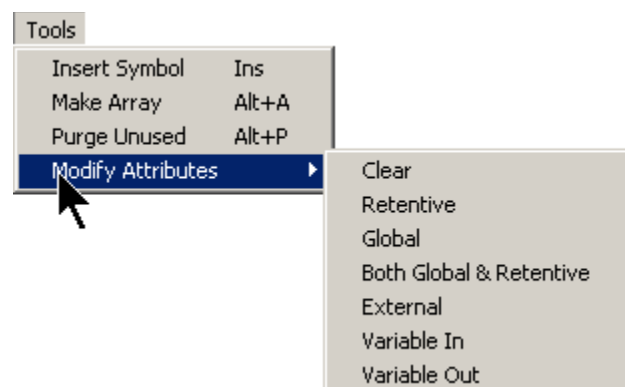
1. Position the selected cell anywhere on the variable declaration you want to change.
2. Press <Alt + A> or the **M**ake **A**rray command.
3. Enter an array length of 1.
4. The index numbers previously displayed behind the data type of the variable will be removed.
5. The initial value, if any, will be displayed in the initial value column instead of ‘...ARRAY...’.

Structures

A structure is a group of variables where the variables can be of any data type except another structure. Each variable that comprises a structure is called a member.

Attributes

You can assign a retentive, global, global and retentive, external, variable in or variable out attribute to a selected variable in the software declarations table. The global, global and retentive, and external attributes are used when the DOS LDO Merge feature is used to merge multiple LDOs. The external attribute is also used for TASKS. Variable in and variable out attributes are used with UDFBs. You access attributes for a selected variable by tabbing to the “A” column and clicking to bring up the Attribute menu or from the **T**ools | **M**odify **A**tttributes command or by right clicking and selecting Modify Attributes. The **C**lear choice will remove an attribute from a selected variable as will **D**elete. In addition, if the selected cell is in the Attributes column, you can enter **E** for **E**xternal, **I** for **V**ariable **I**n, **O** for **V**ariable **O**ut, **G** for **G**lobal, **R** for **R**etentive or **B** for **B**oth **G**lobal and **R**etentive.



Retentive Attribute

The retentive attribute makes the selected variable retain its value upon a warm restart or power cycle, but not on a cold restart. Physical I/O points and individual structure members cannot be made retentive. All other elements, including entire structures, can be made retentive. **Note:** Retentive memory is limited to 24K.

Global Attribute

The global attribute is used when an operator interface device or DOS LDO merge is being used.

When an operator interface device is used, the global attribute identifies the variable as one to be used by the operator interface device.

When DOS LDO merge is used, it identifies the selected variable as the master or global variable. Any variable with the same name in other LDOs will inherit this variable's definition (data type, I/O point, initial value) when the LDOs are merged.

Both Global and Retentive Attribute

The both global and retentive attribute is used when DOS LDO merge is being used. It identifies the selected variable as the master or global variable and as retentive.

External Attribute

The external attribute is used when DOS LDO merge is being used or when the TASK feature is used. It signals PiCPro that this variable is used in more than one LDO, and that the variable's global definition is in another LDO. If the properties of this variable (data type, I/O point, initial value) differ from the properties of its global counterpart, the variable will acquire the properties of the global variable when LDOs are merged.

The external attribute is also applied to a TASK variable that is to be shared with other tasks. Only mark the variable as External in the TASK.LDO, never in the main LDO in which the TASK is called.

In and Out Variable Attributes for UDFBs

Variable In designates the variable as an input to a UDFB. When the LDO you are creating will be converted to a UDFB, every variable you want to use as an input to the UDFB must have this attribute.

Note: The first variable with the I attribute in the software declarations table will become the EN input of the UDFB. Its data type must be BOOL. Additional UDFB inputs should be entered in the order you want them to appear on the left side of the function block.

Variable Out designates the variable as an output to a UDFB. When the LDO module you are creating will be converted to a UDFB, every variable you want to use as an output from the UDFB must have this attribute.

Note: The first variable with the O attribute in the declarations table will become the OK output for the UDFB. Its data type must be BOOL. Additional UDFB outputs should be entered in the order you want them to appear on the right side of the function block.

Editing Software Declarations

If you need to add additional declarations or edit existing declarations, you can use the following tools:

- Inserting and/or deleting declarations
- Cutting/copying or pasting declarations
- Searching for existing declarations
- Purging the table of unused declarations

Inserting/Deleting Software Declarations

You can insert and/or delete entries to the software declarations table when it is active.

To insert software declarations

1. Click in the table and press **<Insert>** or choose **Tools | Insert Symbol** from the menu. A new row will appear above the row the focus was in. If focus is on the end list, you can either press **<Insert>** or simply begin typing. A new row will be inserted.
2. Enter the name of the new entry in the **Name** column and press **<Enter>**. This accepts the name and moves the focus to the **Type** column.
3. Typically, the **Type** column will automatically have the same data type as the previous entry or, if there is no entry, the type defaults to **BOOL**. If the previous entry is a **BOOL** type with an I/O point assigned, the new entry will have an incremented I/O point assigned to it. If the new entry has a different data type you must enter the new data type.
4. Enter information in the remaining columns if required.
5. To accept changes to the software declarations table, choose **File | Save and Close** from the menu or press **<F10>**. To exit without saving any changes, choose **File | Close** or press **<Esc>**.

To delete software declarations

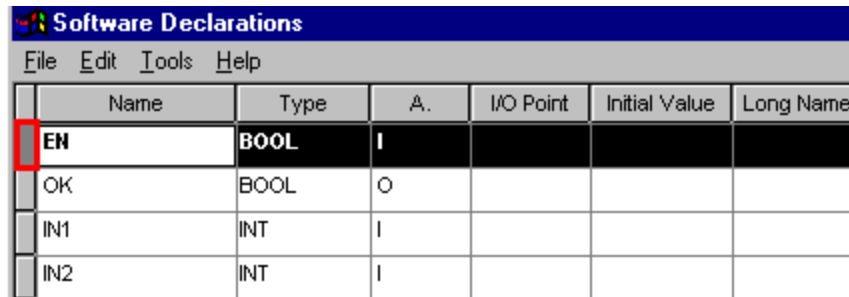
1. Click in the **Name** column of the table (or highlight the entire row) of the entry you want to delete.
2. Choose **Edit | Delete** or press **<Delete>**. A confirmation box will appear. You can choose **Yes**, **No**, or **Cancel**. If you choose **Yes**, the entry will be deleted if it is not used in the LDO.

Cutting, Copying, and Pasting Software Declarations

You can cut or copy and paste items within the software declarations table or from the software declarations table of one LDO into the software declarations table of another LDO.

To cut software declarations

1. Make the software declarations table active.
2. Highlight the entry (or entries) you want to cut. You can highlight a row in the table by placing the arrow in the small rectangle region (marked below) until the pointer arrow becomes a horizontal black arrow pointing to the row. Then click to select the row.



	Name	Type	A.	I/O Point	Initial Value	Long Name
	EN	BOOL	I			
	OK	BOOL	O			
	IN1	INT	I			
	IN2	INT	I			

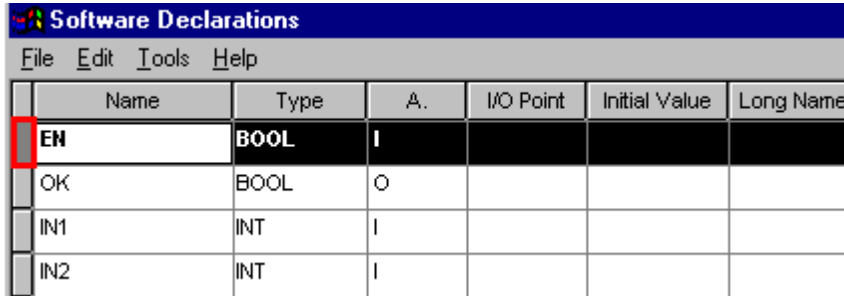
3. Choose **E**dit | **C**ut from the menu
OR
Press <Ctrl + X> on the keyboard
The cut entry is placed on the clipboard.

If the entry is not referenced in the LDO, the entry is removed from the table.

If the entry is referenced in the LDO, the entry is grayed and is not removed from the table. If the entry is a structure and the structure or any of its members are referenced in the LDO, the entire structure will be grayed. The entry will remain grouped until it is pasted into a new location in the same declarations table or until Software Declarations is closed.

To copy software declarations

1. Make the software declarations table active.
2. Highlight the entry you want to copy. You can highlight a row in the table by placing the arrow in the small rectangle region (marked below) until the pointer arrow becomes a horizontal black arrow pointing to the row. Then click to select the row.



	Name	Type	A.	I/O Point	Initial Value	Long Name
	EN	BOOL	I			
	OK	BOOL	O			
	IN1	INT	I			
	IN2	INT	I			

3. Choose **E**dit | **C**opy from the menu
OR
Press <Ctrl + C> on the keyboard
The copy entry is placed on the clipboard. ACTION

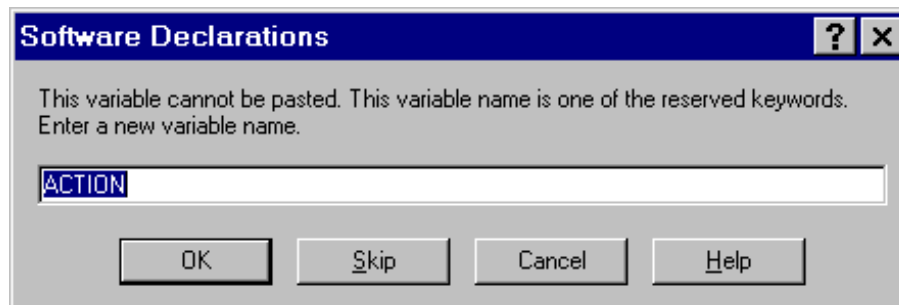
To paste software declarations in another table

1. Make the software declarations table that you want to paste in active.
2. Click anywhere in the table. When you issue the paste command, the paste will be inserted in the row above where you click. If you want to replace a row, highlight the row and then paste.
3. Choose **E**dit | **P**aste from the menu.
OR
Press <Ctrl V> on the keyboard.

Note: If a ladder has a different CPU declared than the one into which you are pasting the software declaration, the I/O points will be changed. Refer to Appendix B, Conversion References for more information.

Reserved Keywords when pasting

If you attempt to paste a variable name that is a reserved keyword, the following error message box will appear:

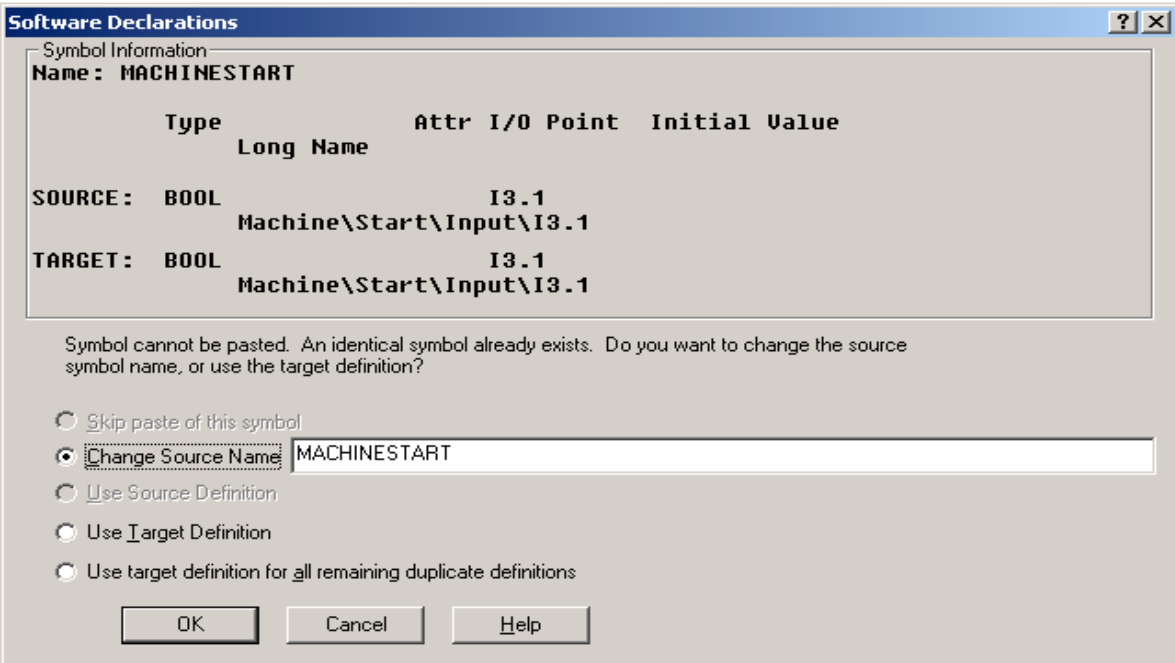


This message displays the variable that cannot be pasted. Enter a variable name that is not a reserved keyword and use the appropriate control button.

Control Button	Description
OK	Press this button and the dialog box closes. The variable name entered is validated.
<u>S</u> kip	Press this button and the dialog box closes. The variable is removed from the paste list and is not pasted.
Cancel	Press this button and the dialog box closes. The variable is removed from the paste list and is not pasted.
<u>H</u> elp	Press this button and Help information is displayed.

Duplicate Symbols when Pasting

If you attempt to paste software declarations or networks into another ladder and the symbols have the same name, type, and array status, the following box appears with the first symbol in the paste listed in the Symbol Information area. You can choose to skip the paste of this symbol, use the source (where you are copying from) definition, the target definition (where you are pasting to), or change the source name.



If the symbols have the same name, type, and array status and you choose to use the target definition, the resulting array size could be smaller. Initial values could change. No attempt is made to make sure one of the array elements being removed is not referenced in the ladder.

Symbol Information

This area displays information about a duplicate symbol that was encountered while pasting. It shows the name, type, Attributes, I/O point, initial value, and long name of the duplicate symbol from the source and from the target.

Source:

The source is the ladder you are cutting or copying from.

Target:

The target is the ladder you are pasting into.

Note: The source and target can be the same ladder if you are cutting, copying, and pasting in that ladder's software declaration table otherwise they are two distinct ladders.

Skip paste of this symbol

Select this option to skip the paste of this symbol. This will leave the pasted ladder element unnamed.

Change Source Name

Select this option to alter the name of the duplicate symbol in the source ladder. Enter the new name in the field provided. Select **Change Source Name** and enter the new name here.

Use Source Definition

Select this option to use the source ladder's definition for this duplicate symbol.

Note: If the symbols have the same name, type and array status and you choose to use the source definition or the target definition, the resulting array size could be smaller. Initial values could change. No attempt is made to make sure one of the array elements being removed is not referenced in the target ladder.

Use Target Definition

Select this option to use the target ladder's definition for this duplicate symbol.

Note: If the symbols have the same name, type and array status and you choose to use the source definition or the target definition, the resulting array size could be smaller. Initial values could change. No attempt is made to make sure one of the array elements being removed is not referenced in the target ladder.

Use target definition for all remaining duplicate definitions

Select this option to the target ladder's definition for this and all remaining duplicate symbols found.

Note: If the symbols have the same name, type and array status and you choose to use the source definition or the target definition, the resulting array size could be smaller. Initial values could change. No attempt is made to make sure one of the array elements being removed is not referenced in the target ladder.

OK

Closes this dialog window and saves any changes you have made.

Cancel

Closes this dialog without saving any changes

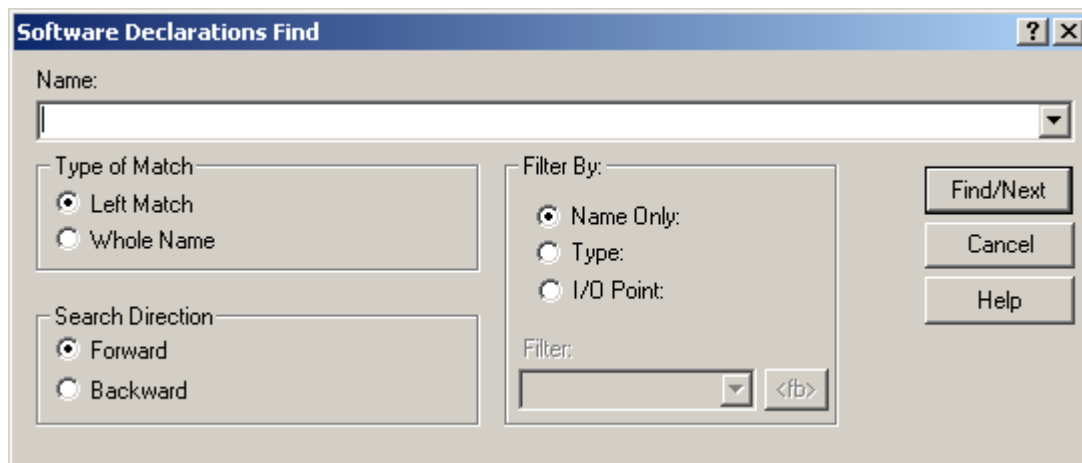
Help

Accesses the Help system.

Searching in the Software Declarations Table

You can search for entries in the software declarations table using the Find command. The search will begin at the row that has focus and proceed in the specified direction.

Choose **E**dit | **F**ind or press <Alt + F3>. The box below appears and you enter the criteria you want to base your search on.



Find by Name Only

The default is to search by name only.

1. In the **F**ilter **B**y: area, select the **N**ame **O**nly: button if it is not already selected.
2. Enter the name or portion of a name with Left Match selected (Left Match is not case sensitive) in the **N**ame: box. The list accessed through the down arrow will hold all the declared variable names. You may choose from this list if applicable.

Find by Type

To search by data type:

1. In the **Filter By:** area, select the **Type:** button. This will enable the **Filter:** box.
2. Choose a data type from the drop down list in the Filter box. If you are searching for a function block, click on the **<fb>** button and select a function block from the displayed menu.

Find by I/O Point

To search by I/O point:

1. In the **Filter By:** area, select the **I/O Point:** button. This will enable the **Filter:** box.
2. Enter an I/O point in the Filter box. The list accessed through the drop down arrow will hold the most recently searched for points. You may choose from this list if applicable.

Once you have entered the criteria by which you want to search, click on the **Find/Next** button to begin. The focus will move to the row with the first occurrence of the declaration.

Using the Find Next Command

You can find the next occurrence of the declaration by clicking the **Find/Next** button again or by pressing **<F3>**.

Purging Unused Variables from the Software Declarations Table

You can remove any variables from your software declarations table that are not being used in the ladder with the **Purge Unused** command.

To purge unused variables

Choose any of the following:

Tools | **P**urge Unused

OR



<Alt + P>

OR

Right click the mouse and select **P**urge Unused

Working with Networks and Network Elements

Your LDO consists of executable networks entered by you. They hold the ladder logic commands and/or structured text that run your application. They are numbered sequentially by PiCPro as you enter them. The networks are executed in this numerical order unless a jump command forces execution to be out of sequence.

PiCPro supports two types of networks: LD (Ladder Diagram)  and ST  (Structured Text). A network can be either LD or ST but not both.

Execution of the ladder logic and commands occurs left to right, top to bottom within the network. When entering network elements, you must enter them in the order you want them to carry out their function, starting from the left and proceeding to the right. The split screen feature can be used to view large networks.

A label can be added to a network element to facilitate jumps. Comments can be added to networks to allow for easier identification, or convey other information about the network both on screen and when the ladder is printed.

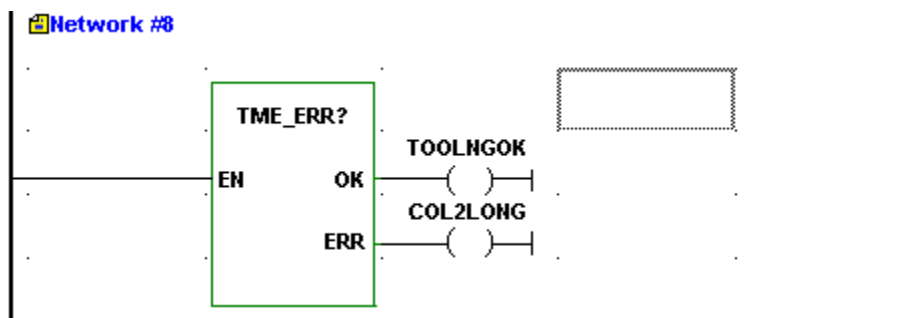
Network Size

LD Network Size

A LD network is comprised of a matrix of rectangles (outlined by dots) called cells. Each cell is approximately the size of the cursor. Each element in the network occupies at least one cell. Elements, like functions, can occupy several cells.

A PiCPro LD network accepts a network with an area of 1 to 255 cells. The area of the network is the sum of the product of its width (number of columns in widest row, excluding last row) and its length (number of rows - 1) plus the width of the last row.

A portion of a LD network is shown below. The cursor on the right occupies one cell in the network. Note the dots that mark each cell. The function on the left occupies four vertical cells.

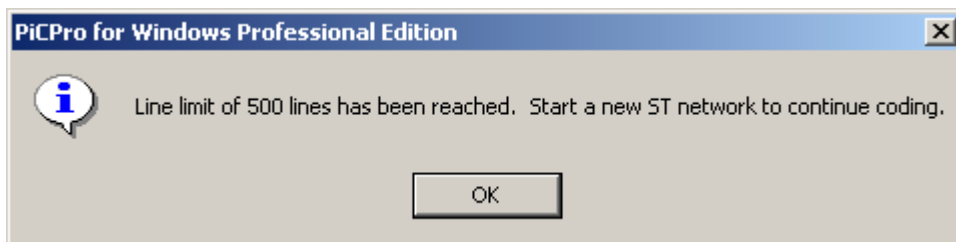


ST Network Size

The ST element is dynamically resized to fit the viewable grid.

- If <Enter> is pressed to move the caret to a new line, the Structured Text element will expand its vertical size to accommodate one additional line of text.
- Entering and deleting rows of characters along with cut and paste and menu selections that insert text (e.g. constructs) will cause resizing of the ST element.

The maximum number of lines in a ST network is 500. Each line is limited to 500 characters. Furthermore, a ST network is limited by the code it generates. If a line exceeds the 500 character limit, a message will be displayed in the Information Window indicating the character limit.



At this time the network will have to be split and a new ST network started. If you want to continue writing the current structured text network it must be continued in the new network. Any loop or IF/THEN constructs must be closed in the first ST network; they can not be continued in the next network.

Network Labels in Networks

You can label a network with the Label command found in the **L**adder menu under the **N**etwork flyout. You must assign a label to any network you want to jump to. If a Jump command is in a network and there is no network with the designated label, an error message will appear when you attempt to download the module.

A label can be from one to eight [alpha, numeric or _ (underscore)] characters long. Note: The first character cannot be numeric.

To assign or edit a label on a network

1. Place the cursor in the network you want to label.
2. Under the **L**adder menu, select **N**etwork.
3. From the Network flyout, choose **L**abel.
4. A text box appears next to the network number. Type in the label you want or edit an existing one.

Comments in Network Elements

Up to 100 lines (80 characters long) of comments (documentation) can be added to a network element. The comments appear directly under the network number line. You can choose to display one or more lines as you work in your ladder.

Add comments to your ladder

1. Under the **E**dit menu, select **P**roperties.
2. Select the **N**etwork tab. Type in the comments you want for the network you are working in.

Choose the number of comment lines to display

1. Under the **V**iew menu, select **O**ptions.
2. Select the **U**ser **P**references tab. In the **L**adder **V**iew **P**references box, enter the number of comment lines you want displayed at each network.

Note: Although you can choose to display up to 100 comment lines, they will not all display at the same time. To see all the lines you must double click in the comment display box. This takes you to the comment editor where you can see all the comments. When you exit the comment editor, choose **C**ancel.

Constants and Variables in Network Elements

All data elements in your ladder that can be read from or written to are in the form of variables or constants. A read element can be a variable or a constant. A write element can only be a variable.

Constants

- Constants can be used as input values to function/function blocks.
 - Note:** In a LD network, constants cannot be entered anywhere else in the network.
- Constant values must be entered in the same form and within the same range a variable value would be entered in the software declarations table.
- Strings cannot be represented as constants. String constants are defined in Software Declarations via a variable and then referenced within the ladder file.

Variables

Variables are used in four ways:

- As input values to function/function blocks
- As output values from function/function blocks
- As values that represent the states of contacts and coils in a LD network.
- In expressions in a ST network.

Variables must be declared in the software declarations table. There you define the following:

- A name for the variable
- The data type the variable is
- The initial value for the variable if it is different than the default 0
- The hardware module location if the variable is a physical input or output
- Any attributes that apply to the variable

Variable Names in Networks

Names

See “Names and Long Names of Variables” on page 41 in the **Software Declarations** section of this chapter for an explanation of variable names.

In a LD network, if a variable name does not fit within its cell and there are empty cells containing only horizontal wires to the left, the variable name will extend into the cells to the left (this is not true for Function Blocks and Task names). The cells to the left will not be included when the variable name is selected. If a single cell is selected and it contains a variable name, the variable name will be displayed in the status line.

Long Names

See “Names and Long Names of Variables” on page 41 in the **Software Declarations** section of this chapter for an explanation of long variable names.

When editing a long name, **Ctrl-Tab** can be used to move focus from the long name edit control to the **OK** button.

Adding/Editing Long Names in a LD Network

1. Place the cursor on the variable you want to add/edit a long name.
2. Under the **E**dit menu, choose **P**roperties.
3. Select the **S**ymbol tab.
4. Enter/edit the long name and click **OK**.

Viewing Long Names in a LD Network

1. Use the **L**ong **N**ame button on the View Navigator toolbar to view long names or proceed with steps 2 through 5.
Note: All cells in your LDO increase in size when you display Long Names.
2. Under the **V**iew menu, choose **O**ptions.
3. Select the **U**ser **P**reference tab.
4. In the **H**ide box, be sure the **S**ymbol **L**ong **N**ame is not checked.
5. Click **OK**.

LD Elements

The elements that go into creating a LD network include contacts, coils, wires, functions/function blocks, data, jumps, variables and constants.

Contacts

Contacts are boolean elements that represent hard wired circuitry or internal logic in the ladder. Each boolean element must be assigned a boolean variable declared in the software declarations table. The variable holds the value of the state of the input. If the input is OFF or deenergized, the value is 0. If the input is ON or energized, the value is 1. If the element is a physical input, its location must also be declared.

Buttons



Contact Types

Normally Open

Normally Closed

Positive transition, normally open

Positive transition, normally closed

Negative transition, normally open

Negative transition, normally closed

Note: Transition contacts pass power until the next update of the coil. Depending on the logic in your ladder, this may not equal one scan.

Normally Open Contact

Variable = 1 The power flow/logic continuity is provided.

Variable = 0 The power flow/logic continuity is not provided.

Normally Closed Contact

Variable = 1 The power flow/logic continuity is not provided.

Variable = 0 The power flow/logic continuity is provided.

Normally Open Positive Transition Contact

If the last write to the variable caused it to go from 0 to 1, power flow/logic continuity is provided.

If the last write to the variable did not cause it to go from 0 to 1, continuity is not provided or is dropped.

Normally Closed Positive Transition Contact

If the last write to the variable caused it to go from 0 to 1, power flow/logic continuity is not provided or is dropped.

If the last write did not cause the variable to go from 0 to 1, continuity is provided.

Normally Open Negative Transition Contact

If the last write to the variable caused it to go from 1 to 0, power flow/logic continuity is provided.

If the last write did not cause the variable to go from 1 to 0, continuity is not provided or is dropped.





Normally Closed Negative Transition Contact

If the last write to the variable caused it to go from 1 to 0, power flow/logic continuity is not provided or is dropped.

If the last write did not cause the variable to go from 1 to 0, continuity is provided.

Coils

Coils or control relays are boolean elements that represent hardwired circuitry or internal logic in the ladder. Each boolean element must be assigned a boolean variable declared in the software declarations table. The variable holds the value of the state of the output. If the output is OFF or deenergized, the value is 0. If the output is ON or energized, the value is 1. If the element is a physical output, its location must also be declared.

Buttons	Coil Types
	energize
	deenergize
	set (latch)
	reset (unlatch)

Energize Coil

If power flow/logic continuity to this coil occurs, it is turned on.

If power flow/logic continuity to this coil drops, it is turned off.

Deenergize Coil

If power flow/logic continuity to this coil occurs, it is turned off.

If power flow/logic continuity to this coil is dropped, it is turned on.

Set (latch) coil

If power flow/logic continuity to this coil occurs, it is turned on.

If this coil is on and power flow/logic continuity is dropped, it stays on.

If power flow/logic continuity to this coil does not occur, it does not turn on.

Reset (unlatch) coil




If power flow/logic to this coil occurs, it is turned off. If this coil is off and power flow/logic continuity drops, it stays off.

If power flow/logic continuity to this coil does not occur, it does not turn off.


Wires

Wires are used in your network to connect other elements. They create data paths along which data is transferred from one element to another. They can also be used to broaden or lengthen the spacing of elements on your screen.

There are three types of wires available.

Buttons	Wire Types
	Horizontal
	Vertical
	Both (combination)

There are several ways to add wires to your network.

1. Choose the toolbar button of the type of wire you want to use and drop it in the appropriate cell. Note that the cursor changes into the shape of the selected wire. You are now in the wire drop mode and can continue to place the chosen wire in your network until you cancel the wire drop mode by choosing another item or pressing <Esc>.
2. Choose the point to point wire button . Position the cursor on the wire starting point and drag the cursor to the wire terminal point. Release the mouse button. Note that you must be in the same row (horizontal wires) or column (vertical wires) for this to work.
3. Choose **Ladder | Wires** from the menu. A flyout appears from which you can choose **V**ertical, **H**orizontal, or **B**oth.
4. Use the following hot keys for the following wires:

Wire	Hot Key	Position of Focus After Wire is Placed
Vertical	<Ctrl + Shift + V >	Remains over vertical wire
	<Ctrl + I>	Moves to cell above vertical wire
	<Ctrl + M>	Moves to cell below vertical wire
Horizontal	<Ctrl + Shift + H>	Remains over horizontal wire
	<Ctrl + J>	Moves to cell to left of horizontal wire*
	<Ctrl + K>	Moves to cell to right of horizontal wire*
Combination	<Ctrl + Shift + B>	Remains over the combination wire

* Unless you are next to a function/function block in which case focus moves to the cell below the vertical wire.

The horizontal wire provides a left-to-right horizontal connection between elements in the network. The vertical wire provides a top-to-bottom vertical connection between elements in the network. The combination wire provides a branch connection between elements in the network. The branch must follow this same left-to-right, top-to-bottom scanning.

Functions/Function Blocks in LD Network




Functions are network elements that allow you to perform operations such as arithmetic or motion control.

Function blocks are network elements that allow you to perform operations that must retain data for a period of time, such as timing or counting operations. Function blocks must be declared in the software declarations table.

For detailed information on functions/function blocks, please refer to the PiCPro Function/Function Block Reference Guide.

Data

When entering functions or function blocks into a network, you can choose from the three categories of data below to enable the connections to these elements.

Data Flyouts	Descriptions
 In	Selected to enable the connection and entering of inputs to functions/function blocks.
 In Inverted	Selected to enable the connection and entering of inverted inputs to functions/function blocks.
 Out	Selected to enable the connection and entering of outputs to functions/function blocks.




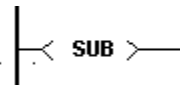

Enable Function/Function Block Connections

1. After inserting the function/function block into the network, place the cursor at the first connection you want to make.
2. Under the **Ladder** menu choose **Data** and open the flyout. Make your choice. Or choose the correct button from the Function toolbar.
3. In the variable scroll box that appears, enter a new or select an existing variable. If the variable is new and has not been declared in the software declarations table, you will need to do that.
4. Move onto the next input or output you want to connect and follow step 2 and step 3.

Jump in LD Network

The jump command causes the execution of the ladder to jump or move to the beginning of a specified network. The specified network must have a label assigned by you in order for the jump command to find it.

You can jump to a label (with no return) or jump to a subroutine (with return).

Jump Flyouts	Descriptions
To <u>L</u> abel	Places the symbol for the jump to label command into the network and prompts you to enter the label of the network to which execution should jump.  
To <u>S</u> ubroutine	Places the symbol for the jump to subroutine command into the network and prompts you to enter the label of the network to which execution should jump.  
<u>R</u> eturn	Places the symbol for the return command (<return>) into the network. Can be placed: <ul style="list-style-type: none">■ In a subroutine where it returns you to the network from which the jump originated■ In the main module (typically at the end of the program before any subroutines) where it takes you to the <End Of Module> in the ladder■ In a UDFB source ladder (typically at the beginning) where it takes you to the <End Of the Module> if the EN (enable) is not set. 

Notes:

It is recommended that subroutines be placed at the end of your main module program. A return command should be placed not only at the end of the subroutine, but also at the end of the main module directly before the subroutine network. The return at the end of the subroutine takes you back to the original jump to subroutine command network. The return at the end of the module takes you to the end of the module. If it were not included, the program would execute the subroutine again.

To insert a jump in your ladder

1. Place the cursor within the LD network you want to insert the jump into.
2. Under the **L**adder menu, select **J**umps. From the Jumps flyout choose either **T**o **L**abel or **T**o **S**ubroutine. Or choose the button from the Ladder toolbar.

ST Elements and Structured Text

ST networks are composed of ST statements and comments. The structured text statements that are supported in PiCPro are assignment, conditional, iteration, and Function/Function Blocks. Many of the ST statements are created using expressions.

Structured Text is a higher level programming language whose syntax is similar to other higher level programming languages such as C, C++, Java, Pascal, Basic, etc...

Expressions

Expressions are used in many of the ST statements and are one of the fundamental building blocks for many of the ST statements. They are used for calculations and they resemble mathematical formulas. An expression is a combination of operands and operators that are used to form an algebraic expression that evaluates to a single value and data type. An operand can be a variable, constant, or function call. Valid expression operators are listed in the table **ST Operators-Evaluation Precedence** located in the next section titled **Expressions - Operators and Precedence Order**. Examples of very simple to more complex expressions are shown in the following table:

Example	Explanation/Description
SPEED	Expressions can be as simple as a constant or a variable.
MAX(IN0 :=A, IN1 :=B)	They can also be a function call. The result from this expression is either A or B depending on which is larger.
A * (B + MIN(IN0 :=C, IN1 :=D))	Expressions can be as complex as needed. Result: B is added to the minimum value of C or D, the result is then multiplied by A.

Comments:

- All operands in an expression must be of the same data type. A compile error will result when an operand does not match the data type of the expression.

- If it is necessary to change the data type on an operand use the data type conversion functions found in the Function/Function Block Reference Guide.
- To detect run time errors such as overflow and divide by zero, in expressions, call the OK_ERROR function. Refer to the Function/Function Block Reference Guide for a description of this function.

Expressions - Operators and Precedence Order

Expressions are evaluated based on a given precedence order. Valid operators and their evaluation precedence order are given in the table below. The precedence order is from highest to lowest. If operators have the same precedence, they are evaluated from left to right.

ST Operators - Evaluation Precedence		
Operator	Description	Precedence
(...)	Parenthesized expression	Highest
Function (...)	Parameter list of a function, function evaluation	
NOT -	Boolean Complement Arithmetic Negation	
* / MOD	Multiplication Division Modulus operation	
+ -	Addition Subtraction	
< > <= >=	Less Than Greater Than Less Than or Equal To Greater Than or Equal To	
= <>	Equal To Not Equal	
AND &	Boolean AND Both operators are used interchangeably	
XOR	Boolean exclusive OR	
OR	Boolean OR	Lowest

Example	Explanation/Description
$P * R / Q + W / X - Y$ <p style="text-align: center;"> 1 2 4 3 5 </p>	<p>When P :=10, R := 5 , Q := 2, W := 15, X := 3, Y := 12</p> <p>1 10 * 5 is 50</p> <p>2 50 / 2 is 25</p> <p>3 15 / 3 is 5</p> <p>4 25 + 5 is 30</p> <p>5 30-12 is 18, which is the final result</p>

Comments:

Many of the ST operators have an equivalent function that can be called. Each of these functions provide the same functionality/behavior as their ST operator equivalent with the following exceptions:

- Functions are evaluated before their ST operator equivalent.
- Functions provide status via OK. This can be used to check for overflow, divide by zero, etc.

Comments in ST Elements

In addition to the network comments that can be entered there are two types of ST comments that can also be entered.

The first type, also known as line comments, begin with a // and end with the enter key. A line comment can be written on a line by itself or at the end of a line containing an ST statement.

The other comment style begins with (* and ends with *) and can be placed wherever it is acceptable to insert one or more spaces within the text itself. These comments can span multiple lines.

Example of (* and end with *)

```
(* Title: Determines if limit switch set.  
Date: 4/17/02 *)
```

Example of Line Comment

```
Area := 3.1416 * Radius * Radius; //Computes area of a circle
```

Comments:

- (* *) comments cannot be nested (e.g., (* (* comment *) *)).
- Neither type of ST comment can be exported or imported.

Assignment Statements

Assignment statements are used to change the value stored within a variable.

Generic Format	
<code><<Variable>> := <<expression>>;</code> Expression is evaluated and results are assigned or stored in variable.	
Example	Explanation/Description
<code>A := B + C * (10 / D);</code>	Results: When A := 1090, B :=10, C := 5, D := 2 Expression evaluates to 35 and A is then set to 35.

Comments:





- The data type of the `<<expression>>` must match the data type of the `<<variable>>`.
- In its simplest form, an assignment statement is equivalent to the MOVE function, the assignment statement can be used to copy an entire structure but cannot be used to copy an entire array. For example, if A and B are of type struct, then `A := B;` will move the entire contents of struct B into A.
- If a underflow/overflow or divide by 0 occurs when evaluating `<<expression>>`, then the value stored in `<<Variable>>` will be unpredictable. To determine if this occurred, call the OK_ERROR function. Refer to the Function/Function Block Reference Guide for a description of this function.

Conditional Statements

A conditional statement selects and executes one or a group of statements based on a specified condition.

A conditional statement can be typed into a ST Element directly or can be selected from **Ladder | Structured Text** or by clicking the appropriate icon on the Structured Text Toolbar.

Structured Text in PiCPro allows four conditional statements, IF-THEN, IF-THEN-ELSE, ELSIF-THEN-ELSE and CASE.

Button	Statement
	IF-THEN
	IF-THEN-ELSE
	ELSIF-THEN-ELSE
	CASE

IF-THEN

Generic Format	
<p>IF <<boolean expression>>THEN</p> <p>Section 1</p> <p>This section will execute if the above <<boolean-expression>> is true. If the above <<boolean-expression>> is not true, then Section 1 will not execute.</p> <p>END_IF;</p>	
Example	Explanation/Description
<pre>IF A > 10 THEN C := 20; END_IF;</pre>	<p>Results:</p> <p>When A is greater than 10, the value of C will not be changed.</p> <p>When A is less than or equal to 10, then C will be set to 20.</p>

Comments:

- Section 1 must contain at least one ST statement.

IF-THEN-ELSE

Generic Format	
IF <<boolean-expression>> THEN Section 1 Section 1 will execute when the above <<boolean-expression>> is true. ELSE Section 2 If the above <<boolean-expression>> is not true, then this Section will execute. END_IF;	
Example	Explanation/Description
<pre>IF A > 10 THEN C := 20; ELSE C := 100; END_IF;</pre>	Results: When A is greater than 10, then C will be set to 20. When A is less than or equal to 10, then C will be set to 100

Comments:

- Section 1 and Section 2 must each contain at least one ST statement.

ELSIF-THEN-ELSE

Generic Format	
<p>IF <<boolean-expression #1>>THEN</p> <p>Section 1 Section 1 will execute when <<boolean-expression #1>> is true.</p> <p>ELSIF <<boolean-expression #2>>THEN</p> <p>Section 2 When <<boolean-expression #2>> is true and <<boolean-expression #1>> is false, Section 2 will execute.</p> <p>ELSE</p> <p>Section 3 If none of the previous <<boolean-expressions>> are true, then Section 3 will execute.</p> <p>END_IF;</p>	
Example	Explanation/Description
<pre>IF A > 11 THEN C := 20; ELSIF A := 5 THEN C := B + X; ELSIF A := 6 THEN C := B + 1; ELSE C := 100; END_IF;</pre>	<p>Results:</p> <p>When A is equal to 11, then C will be set to 20.</p> <p>When A is equal to 5, then C will be set to B+X.</p> <p>When A is equal to 6, then C will be set to B+1.</p> <p>When A is equal to, 2 then C will be set to 100.</p>

Comments:

- Each section must contain at least one ST statement.
- Additional ELSIF sections can be added if needed.
- The ELSE and Section 3 are optional.

CASE

Generic Format	
<p>CASE << integer-expression>>OF</p> <p>Section 1 This section is evaluated first. Statements (in the form <<int-constant #n>> : <<statement>>) that have an integer value that matches the value of the <<integer expression>> above are executed.</p> <p>ELSE</p> <p>Section 2 If no matches to the above <<integer-expression>> are found, then Section 2 will execute.</p> <p>END_CASE;</p>	
Example	Explanation/Description
<pre> CASE SPEED_SETTING OF 1: SPEED := 10; 2: SPEED := 20; 3: SPEED := 30; 4,5: SPEED := 50; 6..10: SPEED := 60; 11,14,16..18: SPEED := 70; ELSE SPEED := 0; END_CASE; </pre>	<p>Results:</p> <p>When SPEED_SETTING is equal to 1 then Speed will be set to 10</p> <p>When SPEED_SETTING is equal to 2 then Speed will be set to 20</p> <p>When SPEED_SETTING is equal to 3 then Speed will be set to 30</p> <p>When SPEED_SETTING is equal to 4 or 5 then Speed will be set to 50</p> <p>When SPEED_SETTING is equal to 6, 7, 8 9 or 10 Speed will be set to 60</p> <p>When SPEED_SETTING is equal to 11, 14, 16 17 or 18 then SPEED will then be set to 70</p> <p>When SPEED_SETTING is not equal to one of the previous values then Speed will be set to 0</p>

Comments:

- The data type of the <<integer-expression>> must be SINT, INT, DINT, or LINT.
- The ELSE and Section 2 are optional.
- The form of <<int-constant #n>> can be defined as a single, several or a range of values. A single value is simply an integer constant. A range of values is defined by separating 2 integer constants by “..”. Several values are defined by putting together any combination of single or range of values and separating them with commas (i.e 1..5, 10, 13:). The <<int-constant #n>> is terminated by a “:” .

Note: When specifying a range of values, the first number must be less than the last number. A compile error will result if the last number is less than the first number.

- The <<int-constant #n>> cannot overlap. Overlapping <<int-constant #n>> will result in a compile error.
- All sections must each contain at least one ST statement.




Iteration Statements

Iteration statements are provided for situations where it is necessary to repeat one or more statements a number of times depending on the state of a particular variable or condition.

Note: An iteration statement must have a finite limit. It is the user/programmer's responsibility to insure that the iteration statement does not take too long to execute which may cause a scan loss.

An iteration statement can be typed directly into a ST element or can be selected from **Ladder** | **Structured Text** or by clicking the appropriate icon on the Structured Text Toolbar.

Structured Text in PiCPro allows three iteration statements, WHILE-DO, FOR-DO, and REPEAT-UNTIL.

Button	Statement
	WHILE-DO
	FOR-DO
	REPEAT-UNTIL.

WHILE-DO

Generic Format	
<p>WHILE <<boolean-expression>> DO</p> <p>Section 1</p> <p><<boolean-expression>> is evaluated. If <<boolean-expression>> evaluates to true, Section 1 will execute. This process repeats until <<boolean-expression>> evaluates to false.</p> <p>END_WHILE;</p>	
Example	Explanation/Description
<pre> ArrayIndex := 0; WHILE ArrayIndex < 10 DO PartReject[ArrayIndex] := 0; ArrayIndex := ArrayIndex + 1; END_WHILE; </pre>	<p>Results:</p> <p>The first time through the loop, ArrayIndex is 0 and PartReject[0] is set to 0. The second time through the loop ArrayIndex is 1 and PartReject[1] is set to 0. This process repeats and the loop terminates when ArrayIndex is 10.</p> <p>At the end of this loop PartReject[0] through PartReject[9] will be set to 0.</p>

Comments:

- If <<boolean-expression>> initially evaluates to false, Section 1 will not be executed.
- To prevent an endless loop, ensure that Section 1 includes a ST statement that will cause <<boolean-expression>> to evaluate to false. In the example above, the ST statement **ArrayIndex := ArrayIndex + 1;** causes the <<boolean-expression>> **ArrayIndex < 10** to evaluate to false when **ArrayIndex** is greater than or equal to 10 and causes the loop to terminate.
- Section 1 must contain at least one ST statement.

FOR-DO

A FOR-DO statement is used to execute a set of ST statements a fixed number of times. It is generally used for counting up or counting down. A FOR-DO statement consists of:

- a starting condition that sets the initial value for the counter of the FOR-DO statement,
- a limit expression that defines the termination value of the FOR-DO statement,
- a set of ST statements that are to be executed in the FOR-DO statement,
- a step expression is the value that the counter will be incremented each time through the FOR-DO statement.

Note: The FOR-DO statement does not provide the same functionality as the CTD, CTU, or CTUD function blocks and should not be used in place of these function blocks.

Generic Format
<p>FOR <<count-variable>> := <<initialization-expression>> TO <<limit-expression>> BY <<step-expression>> DO</p> <p>Section 1 This is the body of the FOR loop. This contains one or more ST statements.</p> <p>END_FOR;</p> <p><u>Description</u></p> <ol style="list-style-type: none">1. The <<initialization-expression>> is evaluated and the results stored in <<count-variable>>. This sets the starting value of the loop.2. The <<limit-expression>> and the <<step-expression>> are evaluated.3. The <<limit-expression>> is then compared to the <<count-variable>>. If the <<count-variable>> is less than or equal to (greater than or equal to, if counting down) the <<limit-expression>>, continue to Step 4, otherwise the loop terminates.4. The ST statements in Section 1 are executed.5. The <<step-expression>> is then added to the <<count-variable>> and the results are stored in the <<count-variable>>.6. Go to Step 3.

Example 1 - Count Up
<pre> Production_Total := 0; FOR Index := 0 TO 9 BY 1 DO Production_Total := Production_Total + Lane [Index].Production; END_FOR; </pre>
Explanation/Description
<p>Results:</p> <p>The first time through the FOR-DO statement, Index is 0 and Lane [0].Production will be added to Production_Total. The second time through the FOR-DO statement, Index is 1 and Lane [1].Production will be added to Production_Total. This process repeats and the FOR-DO statement terminates when Index is 10.</p> <p>At the end of this FOR-DO statement the Index will be 10 and Production_Total will be the sum of Lane [0].Production through Lane [9].Production.</p>
Example 2 - Count Down
<pre> FOR Station_Number := 9 TO -1 BY -1 DO Station_Pointer := INT2UINT (IN := Station_Number); IF Station_Available [Station_Pointer] THEN EXIT; END_IF; END_FOR; </pre>
Explanation/Description
<p>Results:</p> <p>Starting with Station_Number 9 we are looking backwards for the first station that is available. When an available station is found, the FOR-DO statement terminates and Station_Number will contain the available station. If no stations are found, then Station_Number will be -1. Each time through the FOR-DO statement Station_Number is decremented by 1.</p>

Comments:

- The data type for the <<count-variable>>, <<limit-expression>>, and <<step-expression>> must all be the same and must be of the data type SINT, INT, DINT, LINT, USINT, UINT, UDINT, or ULINT.
- BY and <<step-expression>> are optional. If omitted, 1 is assumed to be the step value.
- Specifying a <<step-expression>> that is negative causes the FOR-DO statement to count down. The only data types that are valid with count down FOR-DO statements are SINT, INT, DINT, or LINT. Using USINT, UINT, UDINT, or ULINT with a count down FOR-DO statement will result in a logic error in your program.
- Modifying the <<count-variable>> within Section 1 is not recommended and may produce unpredictable results.
- After the FOR-DO statement is terminated the <<count-variable>> will contain the value that caused the FOR-DO statement to exit.
- Section 1 must contain at least one ST statement.

REPEAT-UNTIL

Generic Format
<p>REPEAT</p> <p>Section 1</p> <p>This section of ST statements are executed.</p> <p>UNTIL <<boolean-expression>> END_REPEAT;</p> <p>The <<boolean-expression>> is evaluated. If <<boolean-expression>> evaluates to False, the ST statements in Section 1 are re-executed. When the <<boolean-expression>> evaluates to True, the loop terminates.</p>
Example 1
<pre> ArrayIndex := 0; REPEAT PartReject[ArrayIndex] := 0; ArrayIndex := ArrayIndex + 1; UNTIL ArrayIndex > 9 END_REPEAT;</pre>
Explanation/Description
<p>Results:</p> <p>The first time through the loop, ArrayIndex is 0 and PartReject[0] is set to 0. The second time through the loop ArrayIndex is 1 and PartReject[1] is set to 0. This process repeats and the loop terminates when ArrayIndex is 10.</p> <p>At the end of this loop PartReject[0] through PartReject[9] will be set to 0.</p>
Example 2
<pre> REPEAT FIND (IN1 := Vision_X_String, IN2:= Space_Char, OUT =>Loc_of_Space); IF Loc_of_Space > 0 THEN DELETE (OUT := Vision_X_String, IN := Vision_X_String, L :=1, P := Loc_of_Space); END_IF; UNTIL Loc_of_Space = 0 END_REPEAT;</pre>
Explanation/Description
<p>The first time through this loop, find the first space in the string Vision_X_String. If a space is found, delete it from Vision_X_String. The second time through this loop find the next space in the string Vision_X_String. If a space is found, delete it from Vision_X_String. This process repeats until all spaces are removed from Vision_X_String.</p> <p>At the end of the loop, all spaces from Vision_X_String will be deleted.</p>

Comments:

- The ST statements in Section 1 always execute at least one time.
- To prevent an endless loop, ensure that Section 1 includes an ST statement that will cause <<boolean-expression>> to evaluate to True. In Example 1 above, the ST statement **ArrayIndex := ArrayIndex + 1;** causes the <<boolean-expression>> **ArrayIndex > 10** to evaluate to True when ArrayIndex is greater than 10 and causes the loop to terminate.
- Section 1 must contain at least one ST statement.

Other Statements in ST

EXIT

This statement can only be used within iteration statements and allows iteration loops to end prematurely.

Generic Format
EXIT; When this statement is executed, the iteration statement is immediately terminated.
Example
<pre>FOR Station_Number := 10 TO 0 BY -1 DO Station_Pointer := INT2UINT (IN := Station_Number); IF Station_Available [Station_Pointer] THEN EXIT; END_IF; END_FOR</pre>
Explanation/Description
Results: When Station_Available [Station_Pointer] is true, the EXIT statement is executed and the FOR-DO statement is terminated.

Comments:

- When the EXIT statement is located within nested iteration statements, exit shall be from the innermost iteration statement in which the EXIT is located, that is, control shall pass to the next statement after the innermost iteration statement that is terminated.

RETURN

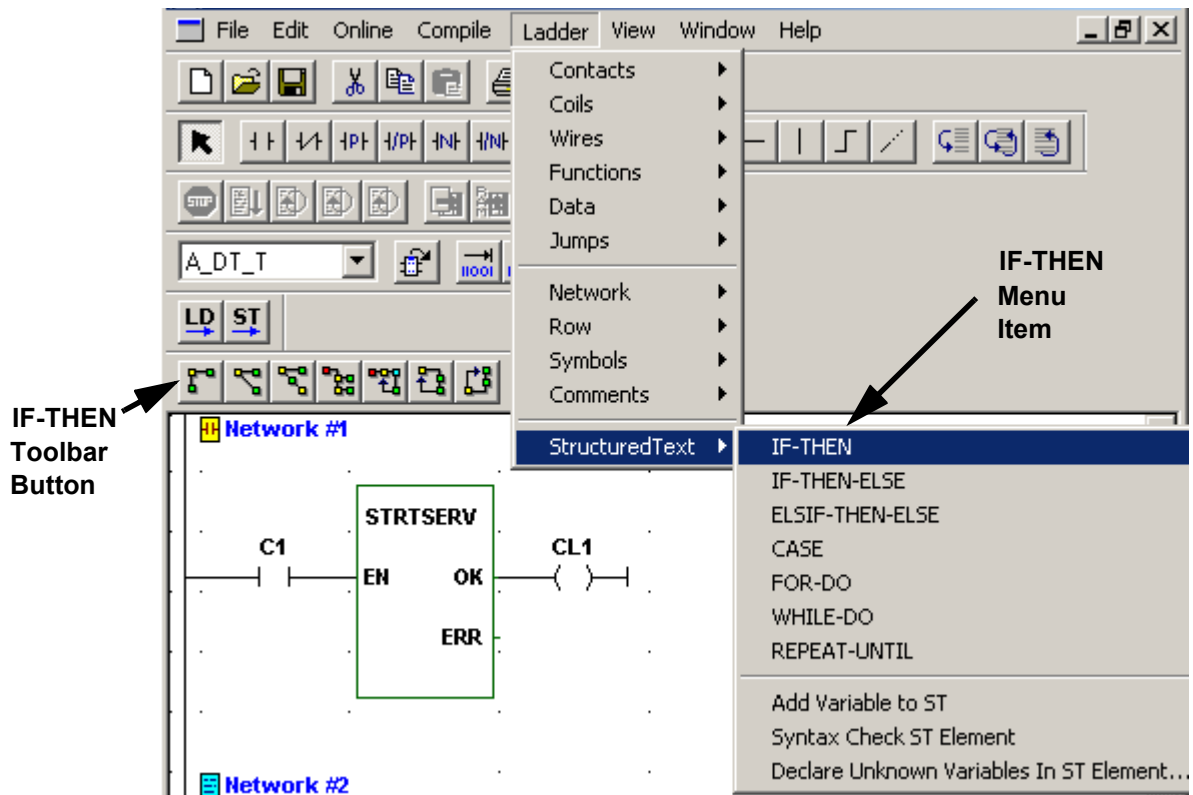
Generic Format	
RETURN; This statement, when executed, will cause the subroutine or UDFB to return to the calling module.	
Example	Explanation/Description
<pre>IF NOT EN THEN RETURN; END_IF; OK := 1;</pre>	Results: Assuming this example is in a UDFB and the EN is false, the RETURN statement causes a return to the calling module.

Comments:

- This statement has the same functionality in ST as it does in LD.
- For more information about this statement refer to the section in this chapter titled **Jump in LD Network**.

Inserting Conditional and Iteration Statements in ST

When focus is in a ST network, you can use a toolbar or pull-down menu to enter conditional and iteration statement (e.g. an IF-THEN statement) templates. Replace the template fields with the desired text.



The cursor will be placed on the first template field. The text is located inside << >> and will be highlighted. The first character entered replaces the template field.

Rules for a Structured Text Template

A Structured Text Template Field (e.g. <<bool-expression>>) will act as a single character.

- If the cursor is located inside a template field, a character entry will overwrite the template field with that character (even though the template field is not highlighted).
- When dragging a selection with a mouse, if the beginning or end of the selection ends in the middle of the template field, the entire template field will be selected.
- When deleting or backspacing over a template field, the entire template field will be deleted.
- When typing a character and 'insert mode' is off, if the character overwrites the start of a template field, the entire template field will be replaced with the character.
- You cannot create a template field from the keyboard.

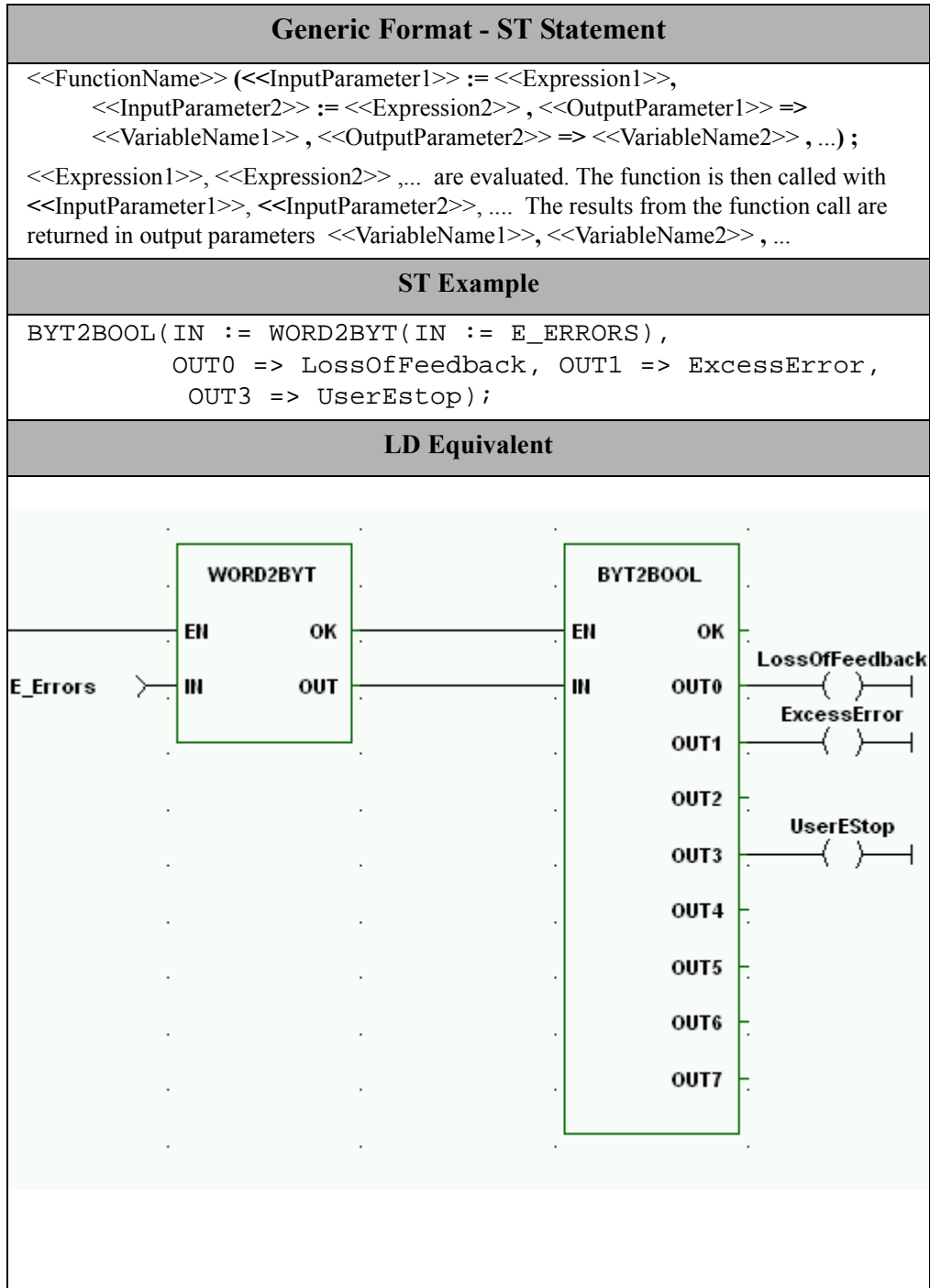
Functions in ST

Functions allow you to perform operations such as arithmetic or motion control.

Functions can be called from an ST statement or from within an expression. See the Generic Format and Examples below.

For detailed information on Functions, please refer to the PiCPro Function/Function Block Reference Guide.

A function block can be typed into an ST element directly or can be selected from the **Ladder | Functions** menu.



Explanation/Description
<p>Results:</p> <p>When E_ERRORS is 9 then LossOfFeedback and UserEstop will be enabled.</p> <p>When E_ERRORS is 3 then LossOfFeedback and ExcessError will be enabled.</p>
Generic Format - ST Expression Operand
<pre><<FunctionName>> (<<InputParameter1>> := <<Expression1>>, <<InputParameter2>> := <<Expression2>> ,...);</pre> <p><<Expression1>>, <<Expression2>>, ... are evaluated. The function is then called with <<InputParameter1>>, <<InputParameter2>>, The results from the function are used in the ST expression.</p>
ST Example
<pre>JogSpeed := SEL(G:=JogSlowFastSwitch, IN0:=SlowJogRate, IN1:=FastJogRate);</pre>
LD Equivalent
Explanation/Description
<p>Results:</p> <p>When JogSlowFastSwitch is not set, then JogSpeed := SlowJogRate.</p> <p>When JogSlowFastSwitch is set, then JogSpeed := FastJogRate.</p>

Comments:

- Selecting **Ladder | Functions** will insert a Function template that includes the input and output parameters into your ST Element. Input parameters are identified using := and output parameters are identified using =>. If more than one parameter is used, they are separated by commas. Parameters can be specified in any order.
- Output parameters cannot be specified in a function call used in an ST expression. Output parameters are optional in a function call used in an ST statement. However, if no output parameters are specified, no results will be returned from the function.

- Functions that do not return a value cannot be used in a ST expression. They must be called from an ST statement (e.g. CLOSLOOP).
- Functions that return multiple values, cannot be used in an ST expression. They must be called from an ST statement (e.g. MOVE, and BYT2BOOL).
- Although EN is always specified as the first input parameter to a Function in an LD network, EN cannot be specified as an input parameter to an ST function.
- The data type for each function input and output parameter must match the data type of the function definition for that parameter.

Function Blocks in ST

Function Blocks allow you to perform operations that must retain data for a period of time, such as timing or counting operations. Function Blocks must be declared in the software declarations table.

For detailed information on Function Blocks, please refer to the PiCPro Function/Function Block Reference Guide.

A Function Block can be typed into an ST element directly or can be selected from the **Ladder | Functions** menu.

Generic Format
<pre> <<FunctionBlockInstance>> : <<FunctionBlockTitle>> (<<InputParameter1>> := <<Expression1>> , <<InputParameter2>> := <<Expression2>> , <<OutputParameter1>> => <<VariableName1>> , <<OutputParameter2>> => <<VariableName2>> , ...) ; </pre> <p><<Expression1>>, <<Expression2>>, ... are evaluated. The function block is then called with <<InputParameter1>>, <<InputParameter2>>, The results from the function block call are returned in output parameters <<VariableName1>>, <<VariableName2>>, ...</p>
ST Example
<pre> CTU1: CTU(CU := PE, R := FULL, PV := 6, Q => FULL, CV => CURRENT) ; </pre>
LD Equivalent
<p>The diagram illustrates the LD equivalent of the ST example. It features a central CTU1 function block. On the left, there are three normally open contacts: 'PE' connected to the 'CU' input, 'FULL' connected to the 'R' input, and a constant value '6' connected to the 'PV' input. On the right, there are two outputs: 'Q' is connected to a coil labeled 'FULL', and 'CV' is connected to a coil labeled 'CURRENT'.</p>

Explanation/Description

Results:

The PE input detects a product on the line. When six products have been completed, the full output at Q will energize. Current will represent the number of products in the package. Full would be used to move the finished package out and reset the counter to zero.

Comments:

- Function Block titles are limited to 8 characters.
- Function Blocks/tasks can not be used in expressions.
- The “:FunctionBlockTitle” is optional and is not part of the standard IEC1131-3. The “:FunctionBlockTitle” has been included so modules are easier to read and understand.
- There is a very significant difference between when Function Blocks are called in ST versus LD. In LD, every Function Block is invoked with each and every scan. It does not matter what ladder logic surrounds the Function Block. In ST, if a Function Block is in a condition or iteration statement, the Function Block will not be invoked if the logic that surrounds it has not been satisfied.
- Selecting **Ladder | Function** will insert a Function Block template that includes the input and output parameters into your ST Element. Input parameters are identified using := and output parameters are identified using =>. If more than one parameter is used, they are separated by commas. Parameters can be specified in any order.
- Input and output parameters are optional and only need to be specified if needed.
- The data type for each Function Block input and output parameter must match the data type of the Function Block definition for that parameter.


ST Network Considerations

- In a LD network, Function Blocks are called each and every scan. In a ST network, the Function block will not be called every scan but will be called only when the logic surrounding the Function Block supports it.
- There are no equivalent ST statements for transitional contacts (i.e., Normally Open Positive Transition Contact, Normally Closed Positive Transition Contact, Normally Open Negative Transition Contact, and Normally Closed Negative Transition Contact).
- ST networks, as do LD networks, execute every scan. Therefore, it is important to take this into consideration when creating a ST network.
- Conditional and iteration ST statements can be nested within other conditional and iteration ST statements to create more complex ST networks.
- An iteration statement must have a finite limit. It is the user/programmer’s responsibility to insure that the iteration statement does not take too long to execute because this may cause a scan loss.

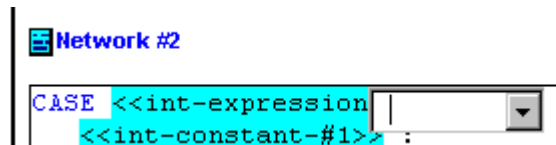
Variables in ST Networks

Adding Variables to a ST Network

With focus in an ST Element do one of the following:

- From the Main menu in PiCPro, choose **Ladder | Structured Text | Add Variables to ST**.
- Press **Ctrl-Enter**.
- Click on  from the Structured Text toolbar.

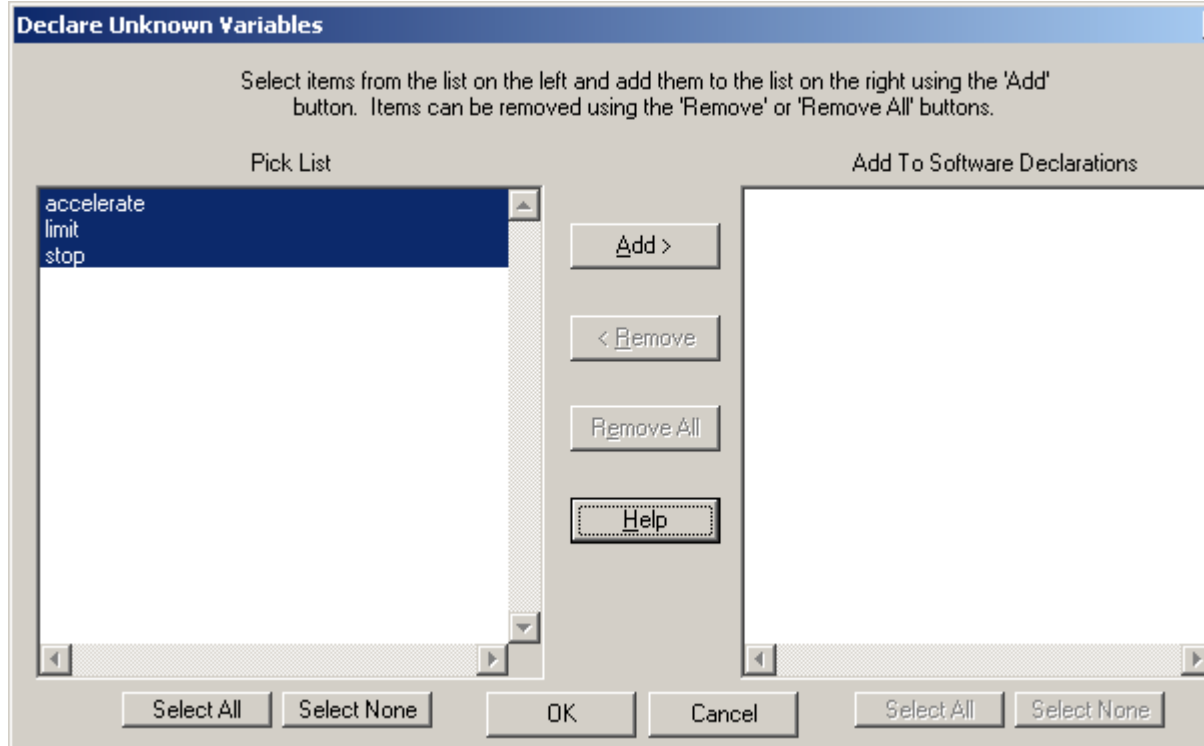
A combo box will appear in the ST network.



If the cursor is on an ST template that specifies a certain type of variable (e.g. integer), only that type of variable is available in the combo box.

Adding Undefined Variables to Software Declarations - ST Network

1. Enter a new variable in a ST network and either click outside of the network or use the arrow keys to move out of the ST network. A list of the undefined variables will be displayed under **Pick List**:



Note: If an * is displayed before the variable name, the variable has been used more than once in the ST network and the compiler detects a conflict with the variable type.

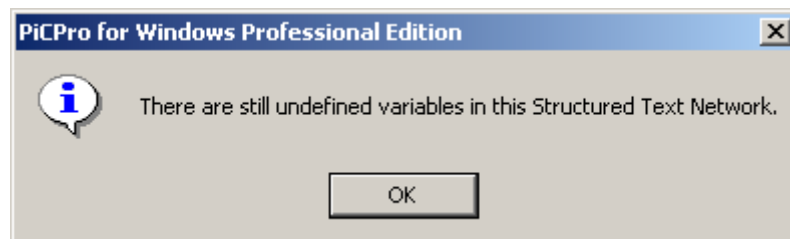
- Click **Add >** to move the variable(s) to the **Add to Software Declarations** list.
- Click **OK** and the Software Declarations dialog will be displayed:

Name	Type	A.	I/O Point	Initial Value	Long Name
C1	BOOL				
CL1	BOOL				
speed	BOOL			0	
brakes	BOOL			1	
collision	BOOL				
End List	void				

- Put focus on the line in the **Software Declarations: Add Unknown ST Variables** dialog.
- Press the **Insert** key or select **Tools | Insert Symbol**. The following dialog will be displayed.

- Enter the datatype in the **Select Datatype** field

After variables have been added to software declarations, the ST network will be recompiled. If the **Forced Declarations** field is checked in the **User Preferences** tab of the **Options** dialog, and if any unknown variables still exist, you are prompted to add the variables to software declarations.



Note: If the **Forced Declarations** field is checked you will be forced to enter software declarations for unknown variables when:


- Ladder focus has changed from an ST element to another grid or row.
- **Save**, **Save All**, or **Save As** is chosen and a change has been made to the ladder program.
- Closing the current ladder program or closing PicPro causes a **Save** to occur.

The following is a description of all control buttons for the Declare Unknown Variables dialog

Control Button	Description
<u>A</u> dd	Press this button to move undefined variables highlighted in Pick List to Add to Software Declarations List
< Remove	Press this button to move undefined variables highlighted in Add to Software Declarations List to Pick List.
< Remove All	Press this button to move everything in Add to Software Declarations List to Pick List .
Select All	Press this button to select (highlight) all undefined variables in the list above.
Select None	Press this button to deselect (remove the highlight) from highlighted undefined variables in the list above.
OK	Press this button after selecting undefined variables to be added software declarations. The Software Declarations dialog will be displayed. Press the insert key to insert undefined variables.
Cancel	Press to close the dialog without adding any undefined variables to software declarations.
<u>H</u> elp	Press to display information related to S-Curve.

Checking Syntax in an ST Element

With focus in an ST Element do one of the following:

- From the Main menu in PiCPro, choose **Ladder | Structured Text | Syntax Check ST Element**.
- Press **Ctrl-Alt-S**.
- Click on  from the Structured Text toolbar.

A compile and syntax check will occur. Upper and lower case spelling errors of variables will be corrected automatically. Compiler errors will be displayed in the Information Window.

Note: This menu item is enabled anytime an ST element is edited.

Using the Right Click Menu in an ST Element

Various tasks can be performed using the right mouse button while focus is on an ST Element. The availability of the menu items used to perform these tasks depends on the location of the cursor. A description of these menu items is as follows:

Menu Item	Description
Help	Click on this menu item to view the on-line Help topics.
Cut	Click on this menu item to cut a highlighted selection.
Copy	Click on this menu item to copy a highlighted selection.
Paste	Click on this menu item to paste an item where the cursor is located.
Insert	Provides the choice of inserting a LD or ST network where the cursor is located. Networks are renumbered if the cursor is not at the end of the module.
Delete Network	Click on this item to delete the network the cursor is located on.
Info:	The information displayed will vary based on the location of the cursor.
Software Declarations	Click on this menu item to open Software Declarations.
Declare All Unknown Variables in Network	Click on this menu item to add unknown variables to Software Declarations.
Add to View List	If the cursor is on a known variable, clicking on this menu item adds the variable to the view list. If multiple variables are highlighted, this item becomes “Add Selected Var(s) to Force List” and all variables selected and declared will be added to the Force List.

Menu Item	Description
Add to Force List	If the cursor is on a known variable, clicking on this menu item adds the variable to the view list. If multiple variables are highlighted, this item becomes “Add Selected Var(s) to Force List” and all variables selected and declared will be added to the Force List.
View UDFB/Task	If the cursor is on a UDFB or task, clicking on this menu item opens the UDFB or task.
View Servo Function	If the cursor is on a servo function, clicking on this menu item opens the servo function.
View SERCOS Function	If the cursor is on a SERCOS function, clicking on this menu item opens the SERCOS function.

Compiling and Downloading

Compiling

When the development is complete, the LDO can be compiled into one of the following:

- A BIN file which can either be downloaded directly to a PiC or stored in a directory or on a disk and later loaded into a PiC with the Restore command
- A HEX file which allows your LDO to be copied to an EPROM or to ladder flash memory
- A TASK creating a UDFB for use in a main LDO
- A UDFB creating a customized UDFB for use in other LDOs

Downloading

Downloading sends the active LDO on your PC to the CPU memory after it has been compiled into an executable program.

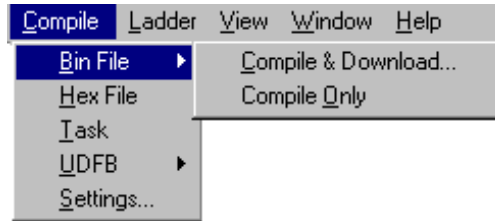
The source form of your LDO remains on the PC. You can edit the source form, run animation, etc. Any off-line changes you make to the source file must be downloaded to the control before they take effect. Always backup this source LDO to ensure that you will have a copy of your LDO in its downloaded form. Include in the backup the filenames for the LDO with any of the following extensions:

LDO, REM, SRV, LIB, SRC, FRC, RTD, SVT, SCT

You can backup this executable form from the PiC to the PC by using the **Online | User Program | Backup | Application Program** command or the button on the Basic Online Operations toolbar. You can restore any BIN file to the control memory with the **Online | User Program | Restore** command or the button on the Basic Online Operations toolbar.

Compiling a Bin File


There are two options under the **Compile | Bin File** command; **Compile and Download...** and **Compile Only**.

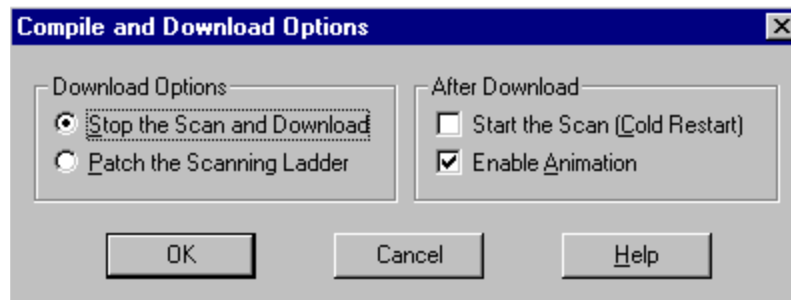


These commands can also be accessed from the **Compile and Download** button or the **Compile Only** buttons on the Compiler toolbar.

Compile and Download

The Compile and Download command allows you to create a binary file out of the active LDO on your PC and download it to the control.

1. Ensure that your PC is connected to the control.
2. Choose the **Compile | Bin File | Compile & Download...** command or the  button on the Compiler toolbar. The **Compile and Download Options** box appears.



You have two download options. You can choose to stop the scan and download the LDO or, if you were able to make on-line changes to the LDO, you can choose to patch the scanning ladder. You can also choose to restart the scan if it was stopped and have animation enabled after the download is complete.

3. The binary file is compiled and downloaded and the status is reported in the information window. If the download is successful without any errors, the memory usage information is shown in the information window.

```
Control Memory Usage
 109 of 64k data bytes
 308 ladder code bytes, 1008 total code bytes
Download Complete!
```


This displays:

- The amount of memory used for data bytes. (64K limit)
- The amount of memory used for ladder code bytes.
- The total code bytes for your application ladder plus libraries*.

*The total code bytes must not exceed the application memory available on the CPU module.

Compile Only

The **Compile Only** command allows you to create a binary file without downloading it to the control. This provides an opportunity to check the LDO for errors and make any corrections or to build a binary file that can be stored on disk or in a directory and later restored to the control.

1. Choose the **Compile | Bin File | Compile Only** command or the  button on the Compiler toolbar.
2. The binary file is compiled and the status is reported in the information window. If the compile is successful without any errors, the memory usage information is shown in the window.

```
Control Memory Usage
109 of 64k data bytes
308 ladder code bytes, 1008 total code bytes
```

Just like when you perform a compile and download, this displays:


- The amount of memory used for data bytes. (64K limit)
- The amount of memory used for ladder code bytes.
- The total code bytes for your application ladder plus libraries*.

*The total code bytes must not exceed the application memory available on the CPU module.

Compiling a Hex File

An LDO file that has been opened on your PC can be saved in Intel 8086 Hex format. The name of the LDO becomes the name of the hex file with a.HEX extension. This file may then be copied to an EPROM programmer to program your Proems or loaded to FLASH memory on processors that support this option.

To Compile a Hex File


1. Open the LDO you want to compile as a hex file.
2. Choose **C**ompile | **H**ex from the menu or choose the  button from the compiler toolbar.
3. If there are no errors, the hex file details are listed in the Information Window and the compile is complete.
4. The hex file is saved in the directory of the current LDO file with the LDO file-name and the HEX extension.
5. If any errors occur during the compile process, they will be reported to the Information Window. The compile process will stop and you must edit the ladder to resolve the errors and compile again.

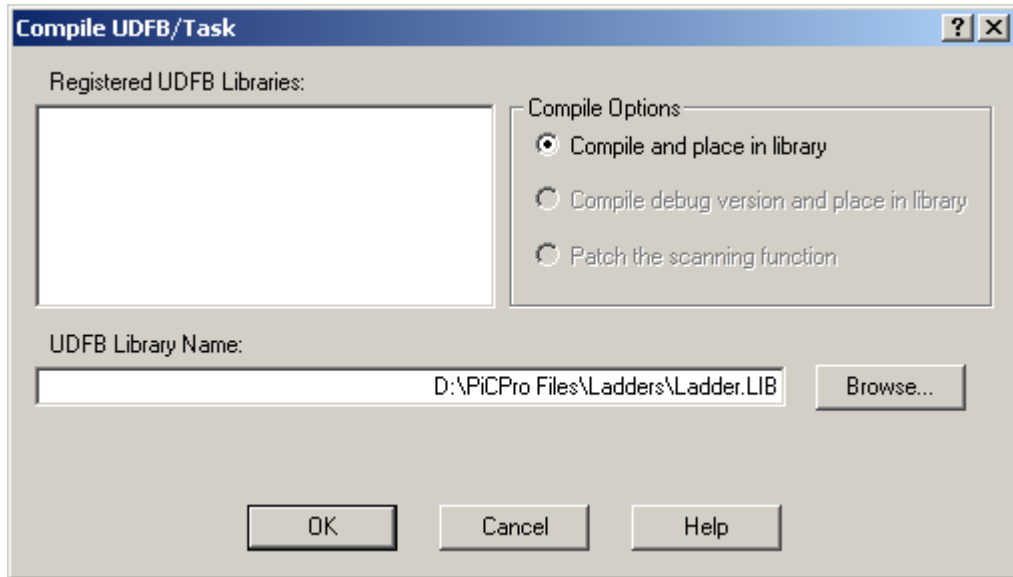
The **D**ownload Hex command from the **O**nline menu is used to load the file to flash memory.

Compiling a Task

The logic you enter in a task LDO is converted into a TASK function block by compiling it.

To Compile a Task

1. Open or create the LDO you want to compile as a TASK.
2. Choose **Compile | Task** from the menu or choose the  button from the compiler toolbar. The **Compile UDFB/Task** dialog box appears.



Under **Compile Options**, **Compile and place in library** is selected by default.

Note: The other options are grayed and not available for TASKs.


You must select the library file in which to insert the TASK. You can enter the name of a new library file in the **UDFB Library Name:** box. A confirmation message will appear. If there are existing libraries listed in the **Registered UDFB Libraries:** box, you may select one of them. If the task already exists in one of the registered UDFB libraries, this library will be selected by default and cannot be changed.

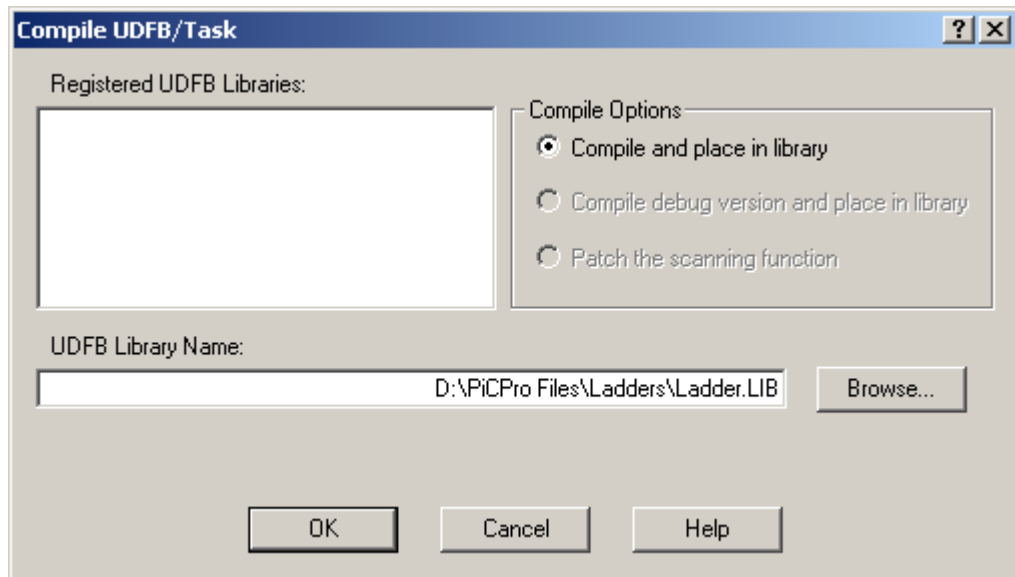
Note: You cannot place a TASK in any of the standard library files.

Compiling a UDFB

The logic you enter in a UDFB LDO is converted into a function block by compiling it.

To Compile a UDFB

1. Open or create the LDO you want to compile as a UDFB.
2. Choose **Compile | UDFB** from the menu or choose the  button from the compiler toolbar. The **Compile UDFB/Task** dialog box appears.



You may choose from the **Compile Options** to:

Compile and place in library

or

Compile debug version and place in library

The debug version reserves some space for patching. (Adds additional data bits (40), data bytes (80), and function/jump links (20) to each instantiation of the UDFB to allow you to perform more extensive on-line changes.)

Note: Minor changes that do not add things like new functions, declarations, or jump labels can be made on-line without creating a debug version.

You also must select the library file in which to insert the UDFB.

You can enter the name of a new library file in the **UDFB Library Name:** box. A confirmation message will appear.

If there are existing libraries listed in the **Registered UDFB Libraries:** box, you may select one of them.

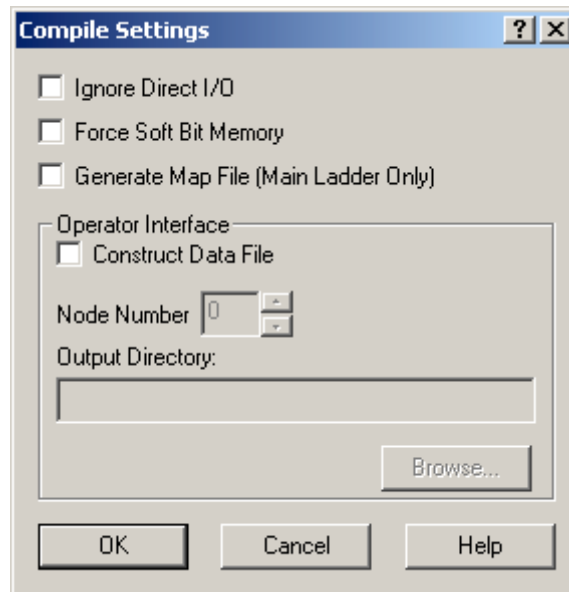
You are not allowed to place a UDFB in a standard library file.

If the UDFB already exists in one of the registered UDFB libraries, this library will be selected by default and cannot be changed.

Settings

The Settings command under the **C**ompile menu allows you to choose settings in four areas:

1. **I**gnore direct I/O
2. **F**orce soft bit memory
3. **G**enerate Map file
4. Use an **O**perator Interface



Ignore direct I/O

If this setting is selected, the compiling of the binary file will occur with all the direct I/O disabled. Typically, this is used for testing where the current I/O does not match the specified I/O.

Force soft bit memory

If this setting is selected, soft bit memory is enabled in all Turbo CPUs that do not have standard soft bit memory. Typically, this is used when data memory is low.

Generate Map File (Main Ladder Only)

If this option is selected, a symbol map file will be generated for the main ladder, whenever you do a compile. The map file is saved in the directory of the current LDO file with the LDO filename and the MAP extension. Subsequent compiles of the main ladder will automatically delete and regenerate the map file. Typically, the MAP File is used for debug purposes and to try and optimize data memory when data memory exceeds 64K.

Operator interface

Within the operator interface box, there are several settings.

When the **Construct Data File** checkbox is checked, an operator interface data file (OID) is created when you compile the BIN file. This file contains the necessary information for the operator interface software.

The Operator Interface **Node Number** is usually set to 0. But it can be used to create multiple displays for a single machine. Available numbers are 0 to 255.

The Operator Interface **Output Directory** allows you to put the OID file in the directory of your choice. If you leave this box empty, the OID file will be placed in the same directory as the LDO file.

Extended Data Memory Feature

If you have a PiC control with a 486 processor running firmware dated August 1, 1999 or later and at least 512K of application memory, data memory can be extended beyond the standard 64K. Two additional data memory areas have been added bringing the total to three, each containing 64K providing a total of 192K of data memory for your control.

To take advantage of this feature, you do not have to do anything. But you should be aware of how the processor handles this extended memory feature. When you select the **Compile Only** or **Compile & Download** command, the first data memory area of 64K is allocated. When it is filled and more data area is needed, a second data memory area is automatically allocated. This memory is taken from the 512K of application memory. When the second area is filled, a third data memory area is allocated. When that is filled, an error message will warn you that the PiC is out of data memory.

After a Compile command, the information window reflects how much data memory is being used and how much is allocated for each data memory area. **Note:** The allocated memory may not match 64K exactly. If all three data memory segments are filled, you may want to optimize your memory usage by getting all three segments as close to 64K as possible. Call our technical support department for help with this. You can produce a readable symbol map file using the **Compile | Settings** box. Check the **Generate Symbol Map** check box so that a text file is generated when you compile. **Note:** This can only be done from the main LDO.

IMPORTANT

You must rebuild all UDFBs and TASKs if this is the first time you are using a version of PiCPro and firmware that have the Extended Data Memory feature.

Tips on Working with Extended Data Memory

- The extended data memory feature requires version firmware dated August 1, 1999 or later. The version can be checked using **Online** | **Status**.
- Retained variables and discrete I/O are restricted to the first data segment. This means you cannot have more than 64K of them. You will get an “out of data memory” error message if this is exceeded.
- UDFBs are restricted to 64K of data memory.
- No structure or array may be larger than 64K. An error message will be issued if one is found.
- Patching of data is allowed up until the last data memory area is filled. At that point, a full compile and download is required.

IMPORTANT


UNPREDICTABLE RESULTS MAY OCCUR if you group arrays in software declarations so that they are treated as one large array using an index value that exceeds the first array's limit! It is NOT recommended that you do this.


It is possible to change the memory configuration in PiC controls with 486DX processors and the MMC for PC. The default configuration is 256K of RAMDISK and 512K of Application memory. If you find it necessary to change the default configuration, use the **Download Hex** command from the **Online** menu available before any files are open.

Animating the Ladder

Animation allows you to monitor the power flow in the LD networks of a ladder diagram that has been compiled and downloaded to the control and in one or multiple UDFB and/or TASK LDOs that are displayed. Variables used in ST networks must be animated in the View List.

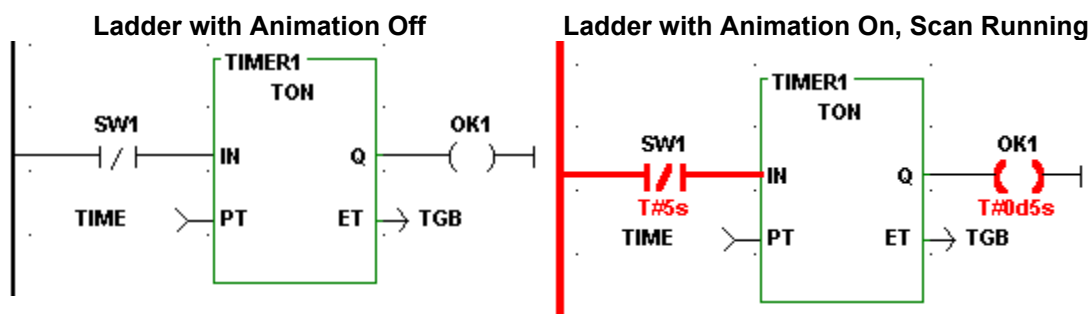
To Turn Animation On

1. Compile and download the ladder.
2. Display any UDFB or Task LDOs you want to animate using the **View | UDFB/Task** command.
3. Choose **Online | Animate** from the menu or with focus on the window to be animated, select the  button on the Advanced Operations toolbar to turn animation on.

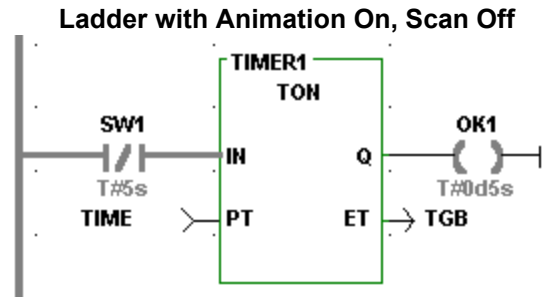
- Power flow is indicated by a heavy wire in a customized color. The default is red. Use the Options  button to change the color. Any energized element (contact, coil, etc.) in the ladder will be highlighted in the color chosen.
- Numeric variables are displayed above the variable name. These variables are updated dynamically as the contents change. You can view things like timers or counters changing. If the results of working with floating point numbers yield an infinite, indefinite, or not-a-number result, the OK on the function will not be set and animation will display the following labels:

Value	Output
+Infinity	1.#INF
-Infinity	-1.#INF
Indefinite	<i>digit.#IND</i>
Not-a-number	<i>digit.#NAN</i>

- All variable lengths are dependent on cell size for display. If a string is too long for the cell, you can view it in the view list.




If the scan is stopped while animation is on, the color of the power flow is grayed to indicate that the control is no longer scanning. When the scan is started again, animation will resume.



To Turn Animation Off

Any of the following will turn animation off.

- De-select **Online** | **Animate** | **LDO**.
- Toggle the  button on the Advanced Operations toolbar when the window which should no longer be animated has focus.
- Enter the edit mode and make a change to the LDO. A message will tell you that the PC version is no longer identical to the PiC version. The LDO must be compiled and downloaded after an edit.
- Access any dialog other than **View List** or **Forcing List**.

When animation is turned off, any numeric values shown above the variables are no longer displayed.

Forcing & Viewing Variables

Forcing is a way to alter the contents of up to 61 variables while the program is running. It is a powerful tool in isolating a problem. Variables can be forced individually or as a group. Grouping is an extension of the forcing feature. One example of how it could be used is to allow you to toggle the state or value of a variable by entering one value or state in grouping and the other in forcing and then switching between them.

When testing or debugging a software module, you can Force a value into the following:

- An input element to simulate the data input from an application device which has not been connected
- An element that enables a function or function block you want to check
- A preset timer variable to check what happens when it times out

When troubleshooting an application, you can Force a value into the following:

- An output element to locate a problem when troubleshooting an application
- Selected elements between an input that works and an output that does not to isolate an internal problem.

WARNING

Do not force any variable in a program that is running an application until you are sure you understand all the effects the changes may cause. A malfunctioning program may cause injury or damage the machinery. **Note:** Forcing is turned off automatically when power is cycled. If you use the power reset button in PiCPro, a warning message will inform you of this.

If you have a forcing list open but inactive when you attempt to turn forcing on, you will get a message that asks if you want to activate forcing. When you switch from the main ladder force list to a UDFB force list, you turn off the main force list and vice versa.

Briefly, to force a variable, you will do the following:

1. Enter the variable name and value in the Force List.
2. Mark the variable in the Force List to enable it.
3. Turn Force on.
4. If needed, edit the Force List and update the forcing information to the control.

IMPORTANT

Animation may not reflect forced conditions. Physical inputs are forced at the beginning of the scan. All other variables are forced at the end of the scan. The writing of variable values by the LDO is not inhibited by forcing. Therefore, it is possible to change the state of a coil during the scan but not see the change in animation.

Entering Variables in the Force List

With the control running and your LDO file open, you are ready to enter variables into a Force list. Only one Force List is created for each LDO file. It is given the FRC extension and the same filename as the LDO file.

To bring up the Force list, choose **View | Forcing List** from the menu or choose the



button from the View Navigator tool bar. The **Force List Window** appears with focus on the **Name** column. The window and columns are sizable.

Force Enable	Group Enable	Name	Value

You have four ways to enter a variable into the **Force List Window**.

Copy/Paste

1. Copy the element in the LDO file that you want to place in the Force List.
2. Open or activate the **Force List Window**.
3. Position the focus in the Force List Window grid and paste the selection.
If you are in a blank row, the variable you are entering will fill that row.
If you are in a row that already contains a variable, the new variable will appear in a row above the existing one.
If you are in a row that is highlighted, the new variable will replace that row.
4. Enter a value in the **Value** column. An error message will appear if the value you enter is out of range, incompatible, etc.
5. Check the **Force Enable** and/or **Group Enable** column.

Drag-n-Drop

1. Open or activate the **Force List Window**.
2. Select the variable in the ladder to add to the Force List.
3. Drag the selection from the ladder to the Force List row you want to insert the variable in and release the mouse.
4. Enter a value in the **Value** column. An error message will appear if the value you enter is out of range, incompatible, etc.
5. Check the **Force Enable** and/or **Group Enable** column.

Right Click Menu in LDO

1. Select the variable in the ladder to add to the Force List.
2. Right click the mouse and choose **Add symbols to Force List**. The variable will be added to the end of the Force List.
3. Enter a value in the **Value** column. An error message will appear if the value you enter is out of range, incompatible, etc.
4. Check the **Force Enable** and/or **Group Enable** column.

Manual Entry

1. Open or activate the **Force List Window**.
2. Place the focus in the **Name** column of the Force List.
3. Type in the name of the variable you want to add or click on the down arrow to bring up a list of all variables in the software declarations table except function blocks. If the symbol is an array, it will be entered into the list as an ARRAY(). You must enter the index number in the parenthesis.
4. Enter a value in the **Value** column. An error message will appear if the value you enter is out of range, incompatible, etc.
5. Check the **Force Enable** and/or **Group Enable** column.

Turn Forcing/Grouping On/Off

Turn Forcing/Grouping On

In order to activate variable forcing/grouping, the following conditions must be met:

- The LDO file is open.
- The LDO file has been compiled and downloaded to the control.
- If you make any changes after the download, be sure to compile and download again or the time stamps will be different. If the time stamps are different, an error message will appear and you will not be able to activate forcing.
- The **Force List Window** is open.
Note: The window does not have to be active.



If the above conditions are met, you can turn forcing on.

Turning On Forcing/Grouping from the On-Line Menu

1. Check all variables that you want to force and/or group in the **Force Enable** column or in the **Group Enable** column respectively of the Force List.
2. Choose **Online | Force | Forcing** from the menu to activate forcing. Select **OK** from the OK/Cancel confirmation that appears.

3. Choose **Online | Force | Grouping** from the menu to activate grouping. Select **OK** from the OK/Cancel confirmation that appears. Turning off grouping updates forcing. Turning off forcing turns off grouping. The symbols for forcing and grouping will appear on the status bar when they are on.

Turning On Forcing/Grouping from the Tool Bar Buttons



1. Check all variables that you want to force and/or group in the **Force Enable** column or in the **Group Enable** column respectively of the Force List.
2. Toggle forcing on by choosing the  button from the Advanced Operations toolbar.
3. Toggle grouping on by choosing the  button from the Advanced Operations toolbar.

The symbols for forcing and grouping will appear on the status bar when they are on.

Turning Off Forcing/Grouping from the On-Line Menu


1. Choose **Online | Force | Forcing** from the menu to deactivate forcing.
Note: If grouping is on, it will be turned off when forcing is deselected.
2. Choose **Online | Force | Grouping** from the menu to deactivate grouping.

Turning Off Forcing/Grouping from the Tool Bar Buttons

1. Toggle forcing off by choosing the  button from the Advanced Operations toolbar.
Note: If grouping is on, it will also be toggled off.
2. Toggle grouping off by choosing the  button from the Advanced Operations toolbar.

Updating the control after Force List is edited

While forcing, you can change any entries in the Force List. These changes can then be updated in the control.


1. Edit the Force List.
2. Choose **Online | Force | Update Forcing List** from the menu or choose the  button from the Online menu.

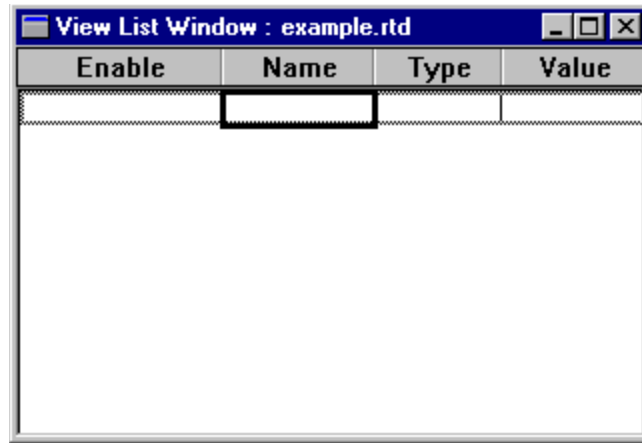
Note: If you delete any or all entries from your force list without updating the control, the entries remain active.

Viewing Enabled Variables

With the control running and your LDO file open, you are ready to enter variables into a View List. Only one View List is created for each LDO file. It is given the RTD extension and the same filename as the LDO file.

To bring up the View List, choose **View | View List** from the menu or choose the

 button from the View Navigator toolbar. The View List appears with focus on the Name column. The Window and columns are sizable.



You have four ways to enter a variable into the **View List Window**:

- Copy and Paste
- Drag-n-Drop
- Right Click Menu in LDO
- Manual Entry

Copy/Paste

1. Copy the element in the LDO file that you want to place in the View List.
2. Open or activate the **View List Window**.
3. Position the focus in the View List grid and paste the selection.
If you are in a blank row, the variable you are entering will fill that row.
If you are in a row that already contains a variable, the new variable will appear in a row above the existing one.
If you are in a row that is highlighted, the new variable will replace that row.
4. Check **Enable**.

Drag-n-Drop

1. Open or activate the **View List Window**.
2. Select the variable in the ladder to add to the View List.
3. Drag the selection from the ladder to the View List row you want to insert the variable in and release the mouse.
4. Check **Enable**.

Right Click Menu in LDO


1. Select the variable in the ladder to add to the View List.
2. Right click the mouse and choose **Add symbols to View List**. The variable will be added to the end of the View List.
3. Check **Enable**.

Manual Entry

1. Open or activate the **View List Window**.
2. Place the focus in the Name column of the View List.
3. Type in the name of the variable you want to add or click on the down arrow to bring up a list of all variables in the software declarations table except function blocks. If the symbol is an array, it will be entered into the list as an ARRAY(). You must enter the index number in the parenthesis.
4. Check **Enable**.

Turning Animation On in the View List

In order to view the variable values which are updated dynamically in the View List, you must turn animation on in the View List.

1. Ensure that the View List has focus.
2. Choose **Online | Animate** from the menu or select the  button on the Advanced Operations toolbar to turn animation on in the View List.

Controlling the Scan

When the control is running, the CPU repeatedly scans the application program in its memory. A scan is the cyclical process of:

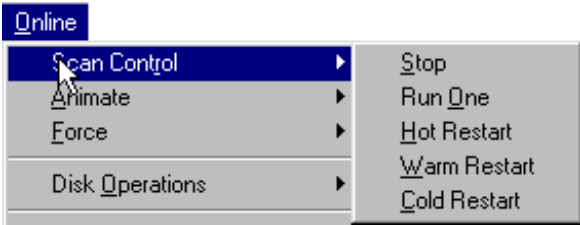
1. Reading data from the hardware input modules
2. Using the forcing list to modify input data
3. Executing ladder logic and using the data to update program variables*
4. Using the forcing list to modify output data
5. Sending commands to the hardware output modules
6. Sending data to the PC to be displayed in animation and/or view modes

Note: When you created your ladder program, the PiC hardware modules were declared under Hardware Declarations. Each input/output point was assigned to a specific logic element through the Software Declarations.

* Step 3 above is referred to as the ladder scan and the remaining steps are referred to as the system overhead. How long a scan takes depends on the length and complexity of the program.

IMPORTANT
The PiC or MMC systems shut down as a safety measure if either the ladder scan or the overhead takes more than 200 ms.

Note: You can display the **Scan Control** menu from the **Online** menu.




WARNING
Do not use any commands in the Scan Control menu until you understand how the control system runs and exactly how the command will affect the application. Starting or stopping the scan at random may cause injury or damage to machinery.

Stopping the Scan

When you stop the scan, the current scan is completed, but the logic commands sent to the output modules are all zero. Power is still on to the system so all elements in the program - timers, counters, input contacts, output coils, etc. keep their values until the scan starts again.


To Stop the Scan

- Choose **Online | Scan Control | Stop** from the menu.
- or
- Select the  button from the Basic Online Operations toolbar.

Running One Scan

The CPU will run through one complete scan and then send zeros to all the outputs when this command is issued.


To Run One Scan

- Choose **Online | Scan Control | Run One** from the menu.
- or
- Select the  button with the BLUE arrow from the Basic Online Operations toolbar.

Doing a Hot Restart

If you restart the scan with a hot restart, all the element values are kept as they were when the scan stopped. The application continues as if the Stop scan command had not been sent.

To do a Hot Restart

- Choose **Online | Scan Control | Hot Restart** from the menu.
- or
- Select the  button with the circular RED arrow from the Basic Online Operations toolbar.


Doing a Warm Restart

If you restart the scan with a warm restart, all elements that were declared with the retentive attribute will keep the values they had when the scan stopped. All other elements return to the initial values they had when the module was first downloaded. Scanning resumes and the application continues normally.

To do a Warm Restart

- Choose **Onl**ine | **Scan Control | **Warm Restart** from the menu.**

or

- Select the  button with the circular YELLOW arrow from the Basic Online Operations toolbar.


Doing a Cold Restart

If you restart the scan with a cold restart, all elements are returned to their initial values declared by you as if the module had just been downloaded.

To do a Cold Restart

- Choose **Onl**ine | **Scan Control | **Cold Restart** from the menu.**

or

- Select the  button with the circular BLUE arrow from the Basic Online Operations toolbar.

On-line Editing

In normal program development, you edit your ladder and then download the entire module to the control to incorporate the changes. The scan is stopped during this download and you must restart the scan when the download is complete.

There may be times when you need to edit or patch the ladder on-line without stopping the scan. If you make changes to the ladder that is currently being scanned in the control and choose compile and download, you are given two choices.

- **Stop the Scan and Download** (off-line edit)
or
- **Patch the Scanning Ladder** (on-line edit)

When you choose **Patch the Scanning Ladder**, only the changes you have just made will be downloaded and the scan is not stopped. The changes will take effect at the beginning of the next scan. These changes become a permanent part of the program in your control. They are included in any backup from the PiC. When you are working with UDFBs, you can patch the main module and the source ladders of any UDFB. Animation is turned off when patching but forcing remains on.


When the patch download is complete, the information window will summarize the resources listed below that you still have available.

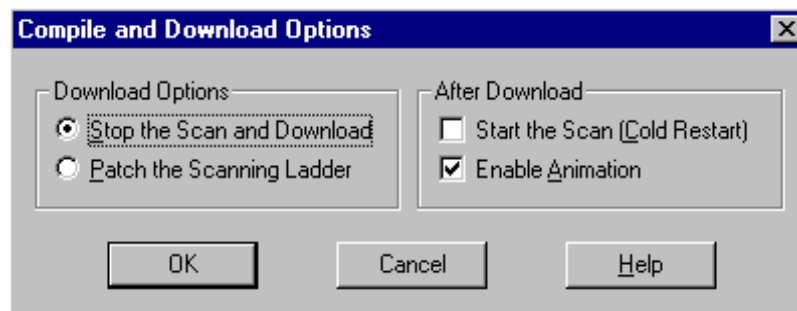
- Data Bits (in hard bit systems only)
- Data Bytes
- Code Bytes
- Number of Patches (out of 100)
- Number of Label/Function Links (out of 27)

Patch the Scanning Ladder is not available in the following situations.

- If any changes have been made to the hardware declarations, you must do a full download stopping the scan to incorporate them into your ladder.
- If more than 40 internal tasks have been added/modified in the current ladder.
- If more than 100 patches have been added to the current ladder
Every time a network is modified, it is considered one patch but could include several internal tasks.
- If more than 27 label or function links have been established in the current ladder
- If the network to be patched contains a TASK.
- If the application program is in EPROM memory.
- If you have ignored a previous Save request from PiCPro.
- If you change or add initial values to existing variables, you must do a full download stopping the scan to incorporate these into your ladder. You can add new variables with initial values with on-line edit.

To perform an on-line edit or patch



1. Enter your changes into the ladder. Save the file.
2. Choose **C**ompile | **B**in File | **C**ompile & Download from the menu or press the  button on the Compile toolbar. The **Compile and Download Options** box appears with the defaults shown



3. Select **P**atch the Scanning Ladder and click **OK**.

To abort a patch

You can undo the last patch downloaded to the control with the Abort Last Patch command. This will remove the patch from the control but does not remove it from your ladder. You can redo the patch in your ladder and do another patch download. The Abort Last Patch command remains enabled after patching a ladder until one of the following occurs:

- The next full download is performed.
 - The ladder doc is closed.
 - PiCPro is closed.
1. After a patch has been downloaded to the control, choose **Online | Abort last patch** or press the  button on the Online toolbar.
 2. The last patch is removed from the control. Edit your ladder to remove it from your ladder program.
 3. Choose **Compile | Bin File | Compile and Download** or press the  button from the Compile toolbar to download the edited ladder to the control.

IMPORTANT

Aborting the last patch removes the operation but not necessarily the effect of that operation from the control. For example, if you add a new network that turns on a new coil and do a patch download followed by an Abort Last Patch command, the new network is removed from the control but the coil will remain on.


Printing

You can print your ladder file, the View list, Forcing list, or the Information Window. Put focus, by clicking, on the window you want printed, and choose **File | Print**.

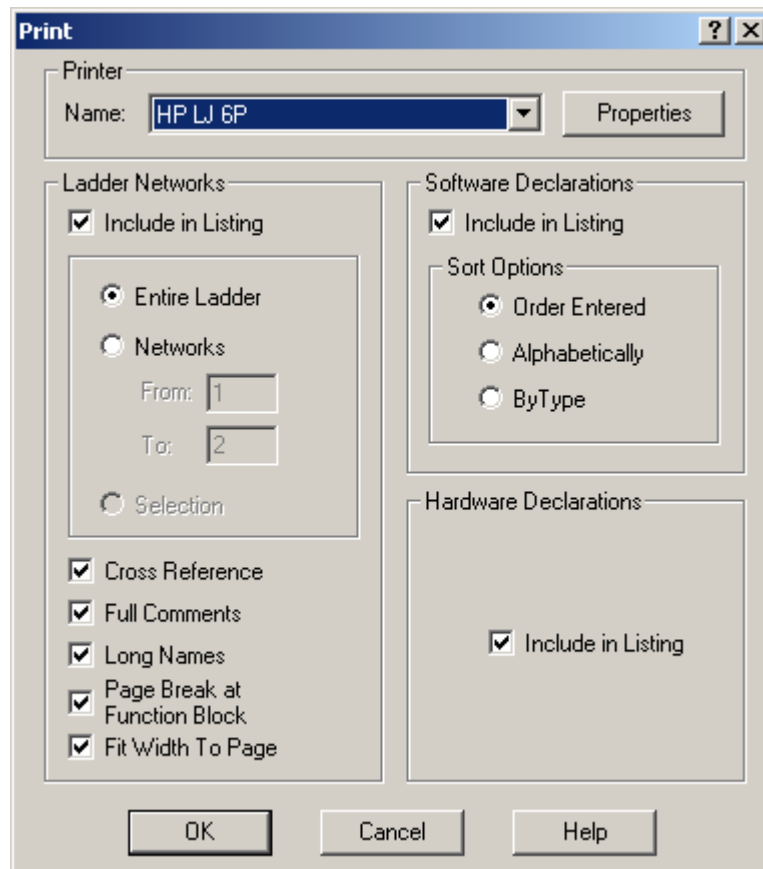
Printing the Ladder

Once your ladder program is developed, you can print it out using the print command.

To print the ladder file

1. Make the window that contains the ladder active.
2. Choose any of the following to bring up the print dialog box:
 - **File | Print**
 - **<Ctrl>**
 - Right click the mouse and then select the **Print**
 - Use the  button
3. Make your selections or accept the defaults and click **OK** in the print dialog box.

Selections in the Ladder Network Section of the Print Dialog



Include in Listing

The **Include in Listing** box is checked (default) and the **Entire Ladder** selected (default). If you do not want to print the entire ladder you have two choices.

- You can print a range of networks by selecting **Networks** and entering a range in the **From:** and **To:** boxes. The From/To boxes initially contain the range of networks for the entire ladder.
- You can print a previously highlighted ladder selection by choosing **Selection**.

Cross Reference

When the **Cross Reference** box is selected (default), all cross reference information on all elements in the ladder is printed. The listing includes:

- Name
- Type
- Source (Where written)
- Used (Where)

The listing is sorted alphabetically by the variable name. At the end of each network, a Network Coil Usage list will be printed listing all the outputs along with any other networks which reference the same output. Each contact/coil element will also display cross reference information. Variable names longer than 22 characters are displayed on a separate line followed by the ('used in') information.

- Beneath contacts/coils representing direct inputs/outputs, the I/O point is displayed.
- Beneath all other contacts, the source network(s) is displayed.

If a variable name is longer than 21 characters, the name is displayed on a separate line. The line(s) following the name display the type, attribute, source and used information.

Full Comments

When the **Full Comments** box is checked (default), the entire comment for each network is printed. If unchecked, the first line of each comment is printed.

Long Names

When the **Long Names** box is checked (default), the long name is printed below each element in a ladder network for which you have entered a long name in software declarations. This has no impact on ST networks.

Page Break at Function Block

When the **Page Break at Function Block** box is checked (default), a page break is inserted to prevent a function block from being printed on two pages if possible.

Fit Width to Page

When the **Fit Width to Page** box is checked (default), the widest network determines the font, etc. so that the network will print on the page. If this is not checked, printing can span multiple pages, both vertically and horizontally. Print size is affected by current zoom settings and cell width options.

Selections in the Software Declarations Section of the Print Dialog

Include in Listing

When the **Include in Listing** box is selected (default), the software declarations table is printed with the ladder. You can choose to sort the software declarations by one of the following:

- **Order Entered** (default)
- **Alphabetically**
- **By Type**

Note that when you choose to print a portion of your ladder, only the software declarations connected to the printed portion will be included.

If a variable name is longer than 8 characters, the name is displayed on a separate line. The line(s) following the name display the type, attribute, I/O point, initial value and long name.

Selections in the Hardware Declarations Section of the Print Dialog


Include in Listing

When the **Include in Listing** box is selected (default), the hardware declarations are printed with the ladder.

Printing View List, Force List, or Information Window

To print the view list, force list, or information window

You can also open a file and print the View List, Force List, or the Information Window from the standard Windows print dialog box. Choose:

- **V**iew | **V**iew List and then choose **F**ile | **P**rint
 - or
 - **V**iew | **F**orcing List and then **F**ile | **P**rint
 - or
 - **V**iew | **I**nformation Window and then **F**ile | **P**rint
1. Ensure that the list or window you want to print is active by clicking it. If you want to print part of the list or window, highlight that portion before choosing the print command.
 2. Choose any of the following to bring up the print dialog box:
 3. **F**ile | **P**rint
 4. <Ctrl+P>
 5. Click on the  toolbar button
 6. Make selections or accept the defaults and click **OK** in the print dialog box.

Printout Symbols

When you print out a copy of your LDO, these symbols may appear.

Symbol	Description
&	Appears in the cross reference section of the printout and in the Network Coil Usage section at the end of each network telling you that the output/coil is sourced in more than one place.
/	Appears in the cross reference section of the printout and also in the Network Coil Usage section at the end of each network telling you that the contact is normally closed.
@	If a network contains a complex variable name, an alias will be substituted in the printout. The alias format is @1 for the first substitution, @2 for the second substitution... up to @255. Numbering is restarted within each network.

Appears under a direct input to indicate the following:

I04.01 Input in master rack (I), slot 4 (04), channel 1 (01)
 or
 I2.04.01 Input in expansion rack 2 (I2), slot 4 (04), channel 1 (01)
 or
 A standalone MMC or MMC for PC specific Input point

Appears under a direct output to indicate the following:

O03.01 Output in master rack (O), slot 3 (03), channel 1 (01)
 or
 O2.03.01 Output in expansion rack 2 (O2), slot 3 (03), channel 1 (01)
 or
 A standalone MMC or MMC for PC specific Output point

Troubleshooting the Print Option

If your file does not print, check the following:

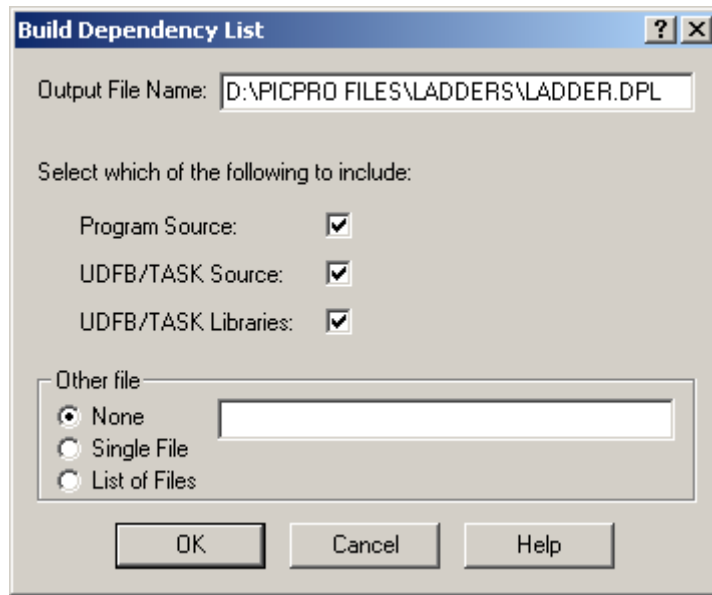
- Printer is on-line and supplied with paper.
- The computer end of the cable is plugged into the correct port. Make sure the connections are secure on both ends.
- Run the printer's diagnostic tests, then try to print the file again.
- If the printer still does not print, check these possibilities:
 - An incorrect or damaged cable
 - A malfunctioning port on the workstation
 - A malfunctioning port on the printer

Building a Dependency List

When you build a dependency list, you create a complete list of all the files that are related to the module you are currently running in PiCPro. The list can include all the LDO source, REM source, UDFB/TASK sources, and UDFB/TASK libraries required to run the program. If you have FLASH memory installed on your control, the dependency list can be used to send all the listed files to the FMSDISK using the Flash Disk Operations under the online menu.

To build a dependency list

1. Choose **Build Dependency List** from the **View** menu.
2. The **Build Dependency List** box appears. It defaults to include the program source, the UDFB/TASK source, and the UDFB/TASK libraries. The **Other file** section allows you to include additional files in your dependency list.



3. When you click **OK**, the list is generated and saved to the current folder of the active ladder. The filename is that of your ladder with a DPL extension.
4. To view your dependency list, open the .dpl file in a text editor program.

Comment Import / Export

Importing Comments

You can import a text file of exported comments (or a text file of comments you have created) into your LDO file.

1. Click on the **L**adder menu.
2. From the drop down menu, choose **C**omments.
3. From the flyout, select **I**mport.
4. Browse to the location on your PC of the text file you want to import and select it so that it appears in the entry field.
5. Select either the option to update the comments or the options to replace all comments with the text file.
Update comments with text file option: This is the default. Only the comments whose index or network number is in the text file will be modified in the LDO.
Replace ALL comments with text file ... option: ALL comments in the LDO file will be removed and replaced with the comments defined in the text file. In other words, any comment in the LDO file that has no counter part in the text file is removed.
6. Select the **S**tart button. The Action box shows the progress.

Note: After importing comments, the LDO must be saved to keep these changes.

Exporting Comments

You can export the LDO comments to a text file. You can then open and edit the text file using a text editor. This text file can then be imported back into the same LDO or into a different one.

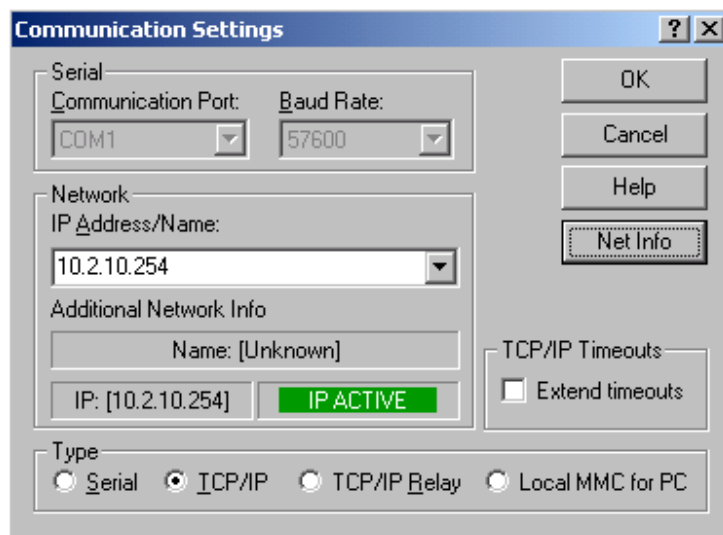
1. Click on the **L**adder menu.
2. From the drop down menu, choose **C**omments.
3. From the flyout, select **E**xport.
4. Enter the path and filename for the text file which will hold the comments. The default will be the path and filename of the LDO file with a .txt extension.
5. Select the option to use index numbers (#n) used by the software to keep track of each network comment or to use the actual network numbers (@n). The symbol and number appear above your comment in the text file.
IMPORTANT: Never change the number or use the # (or @) symbol in the first column of any comment you edit in the text file.
6. Select the **S**tart button. A bar shows the progress.

Note: Comments in a ST network that begin with (*) and end with *) are not exported.

Communications

Communications Settings

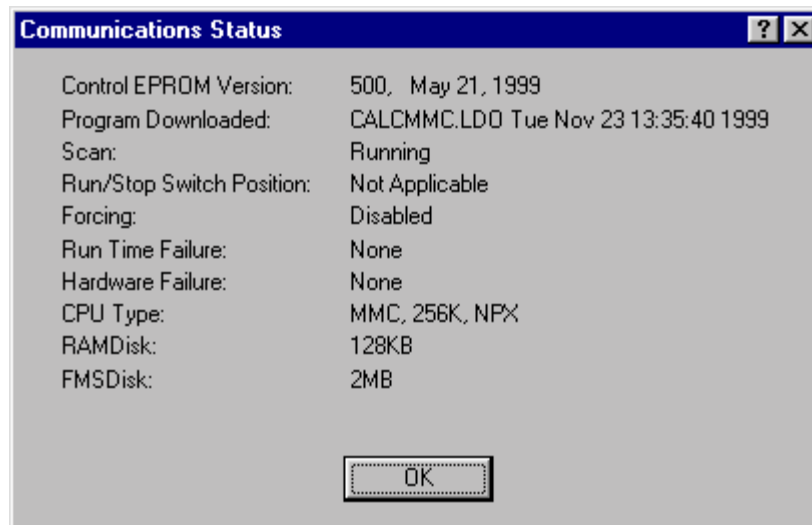
Your PC and control must be set up to communicate with each other. The **Comm Settings** command found under the **Online** menu allows you to set the serial communications port, the baud rate, the IP address if communicating via Ethernet, and the type of communications you are doing: serial or Ethernet-TCP/IP.



- The Type section specifies the type of communicating you are doing: **Serial**, **TCP/IP** (if you have an Ethernet card in your PC), or **TCP/IP Relay** (if you do not have an Ethernet card in your PC), or **Local MMC for PC** (if you have an MMC for PC in your PC). With TCP/IP Relay, you select an RS232 connection to a PiC and then an Ethernet connection to another PiC that has an Ethernet-TCP/IP module installed. Use the G&L Ethernet-TCP/IP configuration tool to set up the module. See Chapter 7.
- The Serial section is active if you choose **Serial** or **TCP/IP Relay** as a communication type. You select an RS232 communication port that the control is connected to and a baud rate to communicate at.
- If you choose **TCP/IP** or **TCP/IP Relay**, the Network section is active. You enter the **IP Address** in dotted decimal format, name, or internet format. This is the address of the control you are communicating with over Ethernet.
- If you click the **Net Info** button, additional information associated with the IP Address/Name is displayed. An indication is also provided if the destination IP Address/Name is on the network and this computer can communicate with it.
- The TCP/IP Timeouts section is available if you choose either **TCP/IP** or **TCP/IP Relay**. Selecting **Extend timeouts** increases the timeout value for TCP/IP network and internet connections between PiCPro and the control. This means that PiCPro will wait longer for a response from the control before issuing a communications timeout message. **Note:** If you are experiencing timeouts with your control that is connected via your TCP/IP network or the internet, try running with this option selected. Be aware, however, that because PiCPro will now wait longer for messages from the control, you will probably experience a slowdown in the responsiveness of PiCPro (especially animation, which relies on the receipt of messages from the control to supply the animation information.)

Communication Status

The current status of communications with the control, as well as additional information about the control can be displayed.



To access this information, from the **Online** menu, choose **Status**

OR use the button from the basic online operations toolbar.



From the dialog, click the question mark in the top right corner and then click on any of the status comments displayed to get further information. You cannot change any of the information from this dialog.

Time Read / Set

The PiC has its own battery-powered clock located in the CSM. This dialog allows you to edit the current time and date inside the control.

Get System Time fills the edit fields with the data from the PC settings.

Note: To ensure that the time for PiCPro in Windows and PiCPro for DOS match, you must add a line in your CONFIG.SYS file.

The following example applies to a time zone setting of Central Standard Time with a six hour difference between UTC and local time and Central Daylight Savings time.

Set TZ=CST6CDT

Refer to the DOS documentation regarding the environment variable TZ and how to set it for your time zone.

Downloading a Hex File

The Download Hex File command allows you to do any of the following;

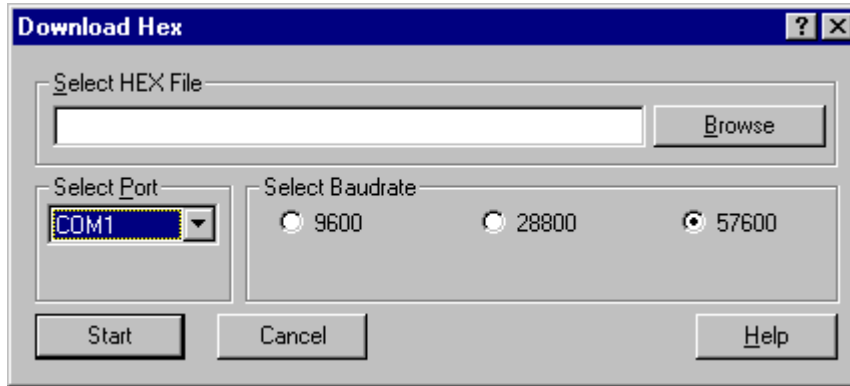
- Configure application/RAMDISK size.
- Update the control's firmware.
- Update TCP/IP firmware.
- Update the SERCOS module firmware.
- Load the ladder flash memory on the CPU.
- Clear the ladder flash memory.
- Clear the ladder application memory.

NOTE
Back up your RAMDISK and ladder before downloading new firmware.

To Download a Hex file for a PiC or Standalone MMC

To use the Download Hex command, PiCPro must be running on your PC and communicating with a PiC or MMC control. No files can be open. If a file is open, this command will be grayed and inaccessible. The following procedure is used to download a hex file:

1. Click **O**nlone.
2. Click **D**ownload Hex.
3. The **Download Hex** dialog box appears. Use **B**rowse to select the Hex file you want to download. The default location for these files is the Utilities folder for PiCPro for Windows.
4. Click on the appropriate **B**audrate.
5. Select the **P**ort if different from the default (PiCPro port).
Click on **S**tart to begin downloading the hex file. Read and follow each prompt carefully. They will tell you to turn the control off and then back on again.

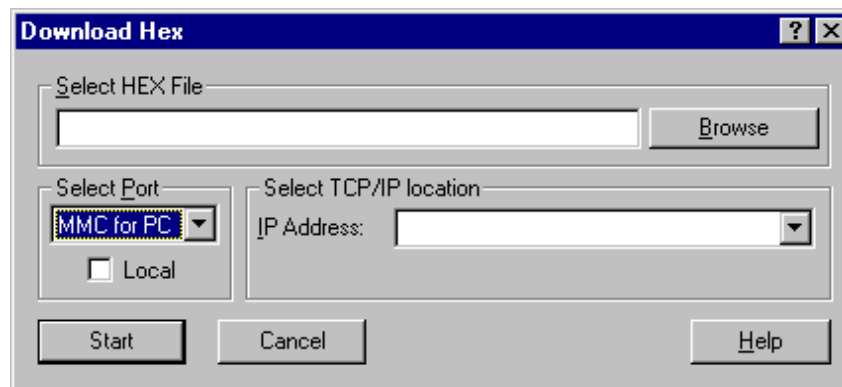


A status bar will show you the progress of the download. If any errors occur, they will be reported in the Information Window.

To Download a Hex file to an MMC for PC

To use the Download Hex command, PiCPro must be running on your PC and communicating with an MMC for PC control. No files can be open. If a file is open, this command will be grayed and inaccessible. The following procedure is used to download a hex file:

1. Click **O**nline.
2. Click **D**ownload **H**ex.



Enter the name of the HEX file, or use the **B**rowse button to select the HEX file to download.

Use **S**elect **P**ort drop down list to select the com ports available on the computer. Select the MMC for PC port. Use the **L**ocal checkbox to indicate that the MMC for PC is in the local machine. If the MMC for PC is not in the computer you are currently using, then uncheck the box. Use the **S**elect **T**CP/**I**P **l**ocation drop down lists to enter the **I**P **A**ddress. Click the **S**tart button to begin the download process.

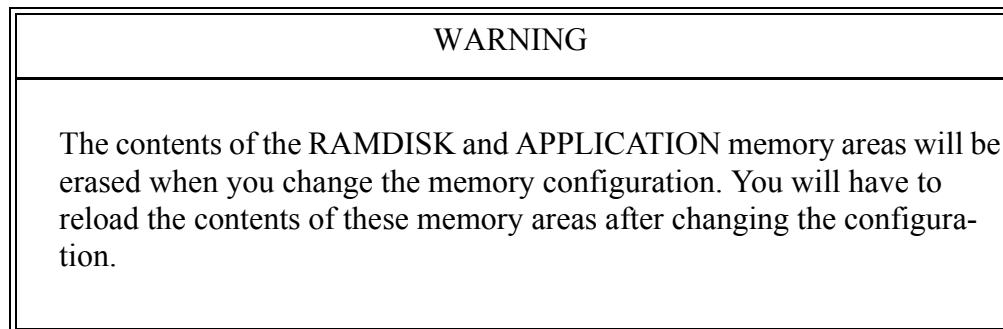
A status bar will show you the progress of the download. If any errors occur, they will be reported in the Information Window.

Configure Application / RAMDISK Size

It is possible to change the memory configuration in PiC controls with 486DX processors and the MMC for PC. The default configuration is 256K of RAMDISK and 512K of Application memory. If you find it necessary to change the default configuration, use the **Download Hex** command from the **Online** menu available before any files are open.

The three memory configuration hex files are found in the Firmware folder in your PiCPro for Windows folder. (Memory sizes shown are approximate) They are:

- Con512.hex (256K of RAMDISK and 512K of application memory) default
- Con384.hex (384K of RAMDISK and 384K of application memory) optional (Cannot be used with extended memory feature)
- Con640.hex (128K of RAMDISK and 640K of application memory) optional
- Con704.hex (64K of RAMDISK and 704K of application memory) optional
- Con768.hex (0K of RAMDISK and 768K of application memory) optional



Update CPU's Firmware

To update the CPU firmware, connect the PC to the controller.

Using the **Online | Download Hex** command, select the appropriate .hex file from the Firmware folder. Click the **Start** button once you have set the proper port and baudrate (if different from the default settings).

- For PiC900 systems, select P94X##.hex
- For PiC90 systems, select P904X##.hex
- For standalone MMC, select MMC##.hex
- For analog MMC for PC, select MMCPA##.hex
- For SERCOS MMC for PC, select MMPCS##.hex

Where ## is a version number.

Update TCP/IP Module Firmware

To update the Ethernet TCP/IP module firmware, connect the PC to the RS232 Com 2 Port on the module.

Using the **Online | Download Hex** command, select the tcpip##.hex file from the Firmware folder. Click the **Start** button once you have set the proper port and baudrate (if different from the default settings).

Update SERCOS Module Firmware

To update the SERCOS module firmware, connect the PC to the RS232 Port on the SERCOS module.

Using the **Online | Download Hex** command, select the SERCO##.hex file from the Firmware folder. Click the **Start** button once you have set the proper port and baudrate (if different from the default settings).

Download Ladder Hex File

A ladder can be downloaded into the CPU flash memory using **Online | Download Hex** after you have created the .hex file using **Compile | Hex File**.

If the battery in the controller were to become low on power (or run out), when the CPU was next powered up, the ladder would be taken from the flash memory and loaded to the application memory.

Clear Flash Memory

To clear the flash memory in the controller, from the **Online** menu, select **Download Hex**. Select the clrflash.hex file from the Firmware folder. Click the **Start** button once you have set the proper port and baudrate (if different from the default settings).

This procedure should be done before using the Clear Application procedure.

Clear Application Memory

If you are using either a standalone MMC or an MMC for PC control, the Clear Application Memory command provides a software solution to clearing out a ladder application that may not be scanning correctly. When you choose the command from the **Online** menu, the **Download Hex** dialog box appears. The entry field for a hex file is grayed because PiCPro automatically enters a file called clrapp.hex. This file is downloaded to the standalone MMC or MMC for PC when you choose **Start**. If connected to an MMC, you will be prompted to turn the control off and on again.

Some examples of common programming errors that require you to do this include:

- Programming a backwards jump and losing the scan
- Declaring the wrong Block I/O and getting a Block I/O error.

When these programming errors occur, communication to the control is lost. By using the **Clear Application Memory** command to clear the ladder currently in the standalone MMC or MMC for PC control, communication is re-established and you can proceed to download a ladder file.

To clear application memory with any MMC CPU

1. Click on **O**nl ine on the main menu.
2. Click on **C**lear Application Memory.

IMPORTANT

An action that affects the contents of the application memory, does not affect the contents of the flash memory, and vice versa.

If a ladder is downloaded to flash memory, and no ladder is loaded in application memory, the ladder in flash memory is scanned. If subsequently, a ladder (the same or another) is downloaded to application memory, the ladder in application memory is scanned.

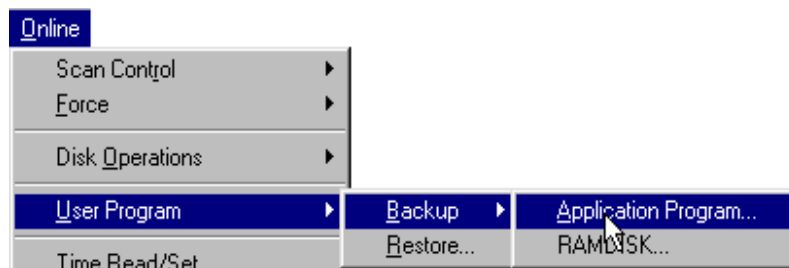
The clrapp.hex file does not clear the flash memory, and the clrflash.hex file does not clear the ladder in application memory.

Backing Up and Restoring User Programs

Backing Up User Programs

Your program in the control memory may be backed up as a binary file directly to a computer disk. With your PC connected to the control and communications established, follow these steps:

1. Click on the **Online** menu.
2. Choose **User Program** from the drop down menu.
3. Select **Backup** from the submenu. Backup will be bold and selectable if there is an existing communication link to your PC.
4. Select **Application Program**.



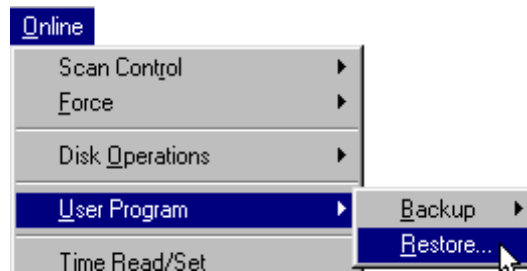
5. The **Backup File to PC** dialog box appears. The filename of the LDO currently in the control memory with a .bin extension will appear in the entry field.
6. Choose the location where you want to save the binary file and choose **Save**. A progress box will tell you the file is being saved to the PC.

The file can later be restored to the control using the **Restore** command.

Restoring User Programs

A program file saved in binary format with a .bin extension can be restored to the control memory by doing the following:

1. Click on the **Online** menu.
2. Choose **User Program**.
3. Select **Restore** from the submenu. The **Restore File to the Control** dialog box appears. Restore will be bold and selectable if there is an existing communication link to your PC.



4. On your PC, find the location of the binary file you want to restore and click on it so that it appears in the entry field. When the correct bin file is in the entry field, choose **Restore**. A progress box will tell you the file is being downloaded to the control.

PiC Restore

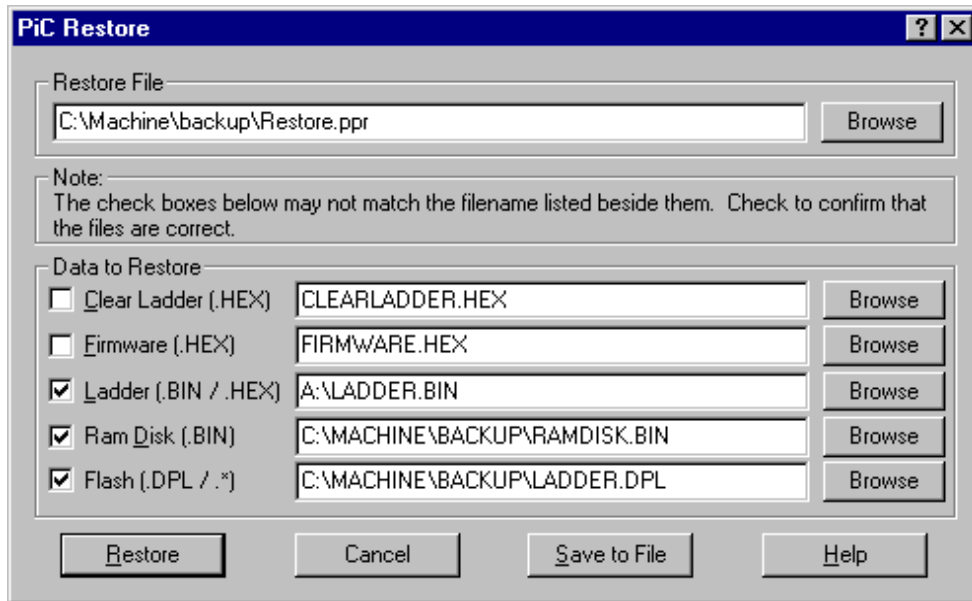
You can restore your PiC or MMC controls with the firmware, ladder (hex or binary), RAM disk data (binary), and flash memory if you ever lose the data in your control. You will need the required files either previously saved by you or supplied by your OEM.

PiC Restore is available only if an application ladder is not open. From the **Online** menu, choose **PiC Restore**. The following dialog appears. To restore or save files, enter the path and name of the restore file (.ppr).

The five check boxes allow you to check the files you want sent to the control. (The check boxes will default to any settings currently in an existing restore file.)

Selecting **Restore** starts execution of the restore file. When the download is complete without any errors, the information window will indicate the results by listing the files sent to the control.

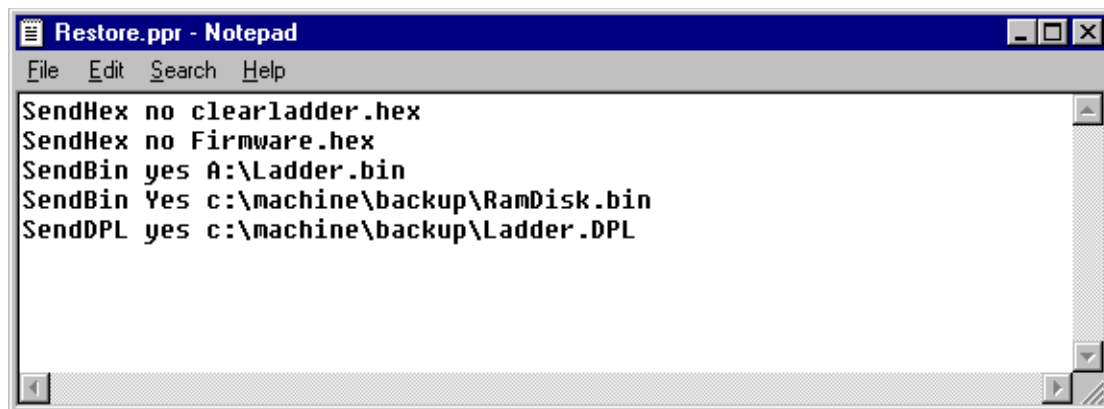
The **Save to File** button allows you to save the path and file names of the hex, bin, or dpl files and the check box settings to the restore file.



Note: Generic filenames are shown above. Your filenames and locations will be different.

In addition to using PiC Restore, a standard text editor can be used to create the restore file. The format for the restore file is shown below. The filenames are listed in the order they are to be sent to the control and shown in the PiC Restore dialog. **Note:** Generic filenames are shown below.

The first field is the type of send: either SendHex, SendBin, SendDPL, SendFlash, or Empty. The second field contains either a “yes” or a “no” and defines the default value used for the checkbox. The third field is the name and path of the file. Make sure the type of file corresponds with the type of send. For example, use SendHex for sending a hex file. If there is a file type you aren’t using, type in “Empty No Null” (without the quotes) to so indicate.



SendDPL can also be entered as: SendFlash yes C:\filelocation\Ladder.ldo, and any extension can be used.

CHAPTER 4 Servo Setup and Tuning

Servo Setup Overview

The Servo Setup program in PiCPro is used to enter setup data for all digitizing and servo axes used in your application. Each axis you insert appears in a list sorted by the axis type. It also allows you to read servo setup parameters in the servo view list and write selected parameters to an axis in the servo force list.

A general overview of the procedure to incorporate servo setup into your application program follows:

1. Use servo setup to enter setup data for your application.
2. Save the setup file (*name.SRV*).
3. With the compile command, make a function containing all the setup data. Assign an appropriate name to the function. This function is placed in a library file you create (*name.LIB*) which PiCPro can find. This library file will hold all servo functions defined by you for your application.
4. Use PiCPro to create an application program (*name.LDO*).
5. Include the setup function you made, along with the STRTSERV standard motion function, in a network of your ladder program.

Note: When setup data is called in a ladder, it is copied into the RAM memory of the CPU.


6. Download the application to the control. The setup data for your application is sent to the CPU in the form of the function.
7. Read and write setup parameters to fine tune each axis in your application.

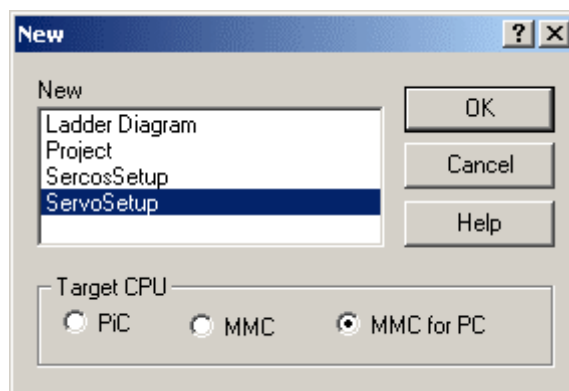
Note: This creates a .SVT file for the Servo View List and the Servo Force List.

Accessing Servo Setup

There are several ways to access the Servo Setup program.


Create a New Servo Setup File

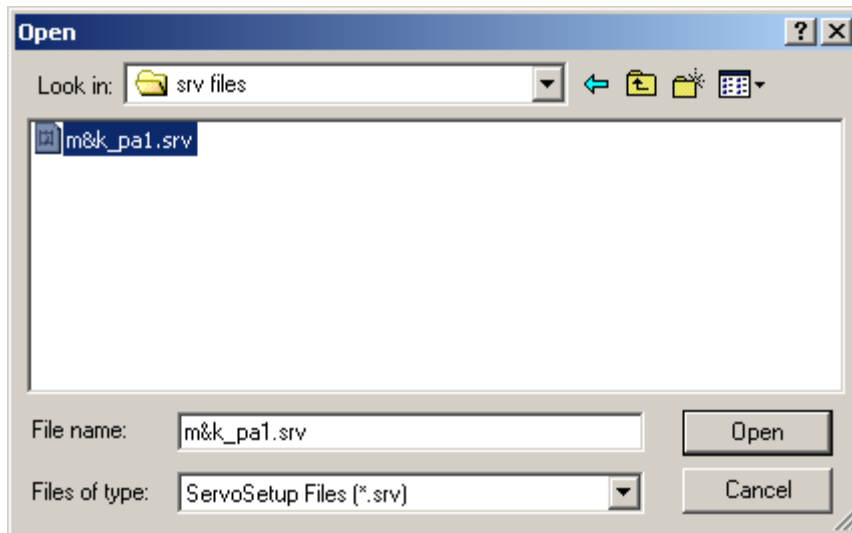
1. Choose **F**ile | **N**ew from the menu or click on the  button on the standard toolbar.
2. Select **ServoSetup** from the dialog box that appears.
3. Select the CPU type (**P**iC, **M**MC or **M**MC for **P**C).
Note: the default selection is the last selection made.
4. Choose **OK**.



Note: In the PiCPro Standalone MMC Edition, the CPU type is always MMC and is therefore not an option on this dialog box.

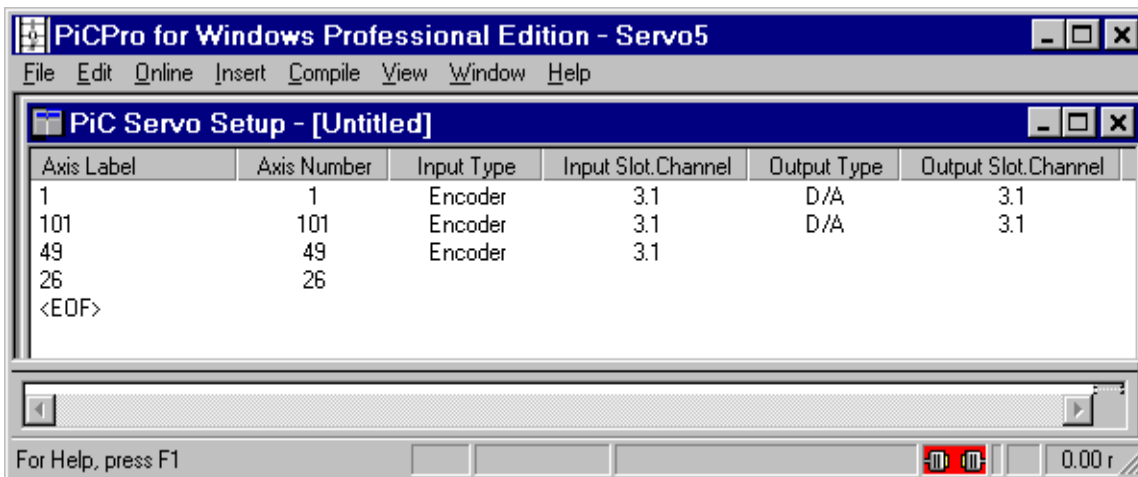
To Open an Existing Servo Setup File

1. Choose **File | Open** from the menu or click on the  button on the standard toolbar.
2. The **Open** dialog box appears. At the bottom, choose ServoSetup Files (*.srv) from the drop down list in the **Files of type:** box. This brings up a list of all existing servo setup files.
3. Highlight the file you want to open



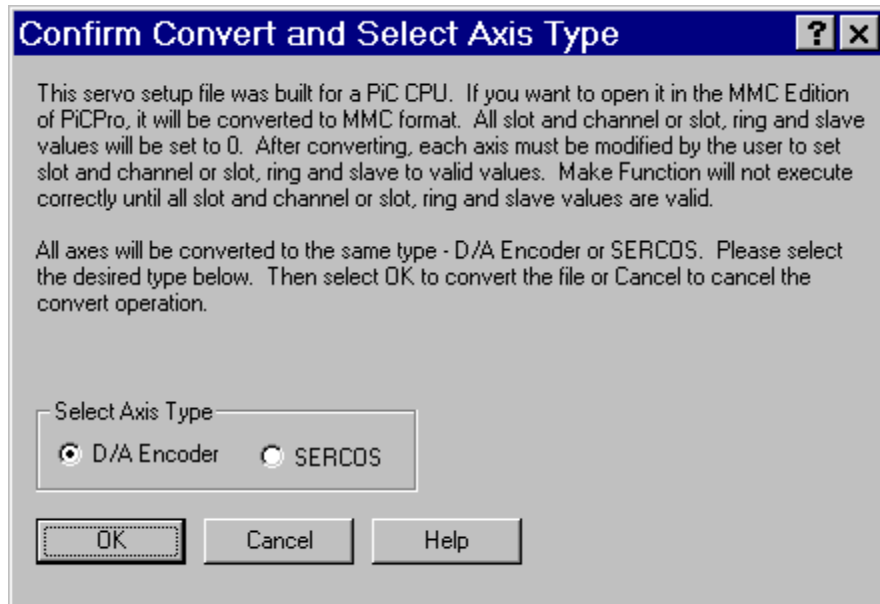
4. Double click on the file, click **Open** or press **<Enter>**.

The following window will appear:



Note: If Time axes are included, they will be displayed after digitizing axes in the Servo Setup list view (e.g Axis Label 26 below). Input and Output data are not included for Time axes (refer to **To insert a Time Axis** on page 4-17).

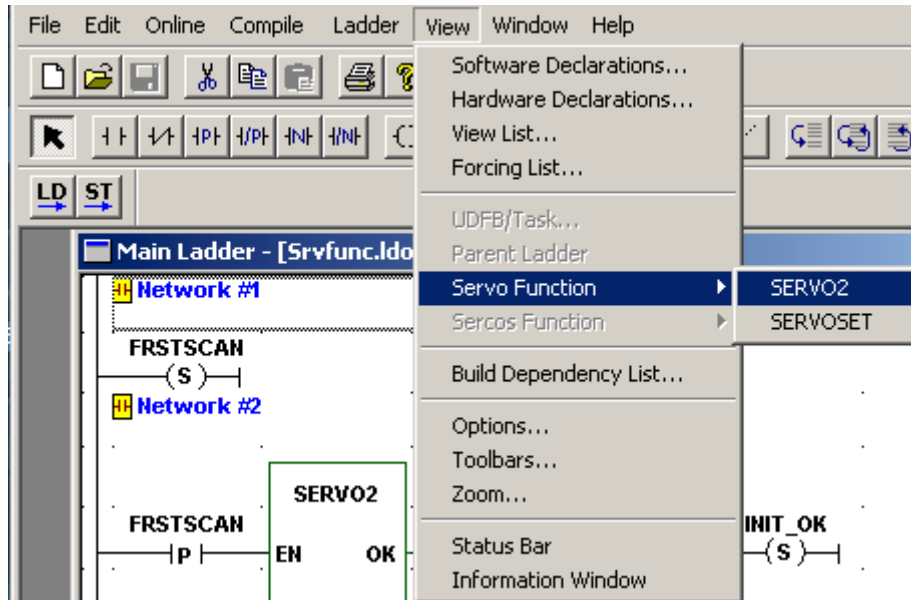
If you attempt to open an existing SRV file in the PiCPro Standalone MMC Edition, and the SRV file was previously saved for a PiC or MMC for PC CPU, the following confirmation prompt will be displayed that indicates the types of changes that will be made to the servo setup information for this file.



To Open file from the setup function in your ladder

If you have already created a servo setup function with your setup data and have included it in your ladder, you can access the servo setup file from there.

Place focus on the setup function in your ladder. Right click and select **View Servo Function** or choose **View | Servo Function** from the menu and then select the desired setup function from the flyout list.



Note: In order to open a servo setup file from within your ladder, the servo file and the servo function must have the same name. Any servo setup file created with PiCPro for Windows automatically has the same name as the servo setup function. However, any DOS servo setup file may not have the same name as the servo setup function. Change the name of the .srv file to match the name of the servo setup-function if you want to be able to open the file from within the ladder and/or be able to force axis values using the Servo Force List.

To Open file from Windows explorer

1. Open Windows Explorer and find the .srv file you want to open.
2. Choose the .srv file that you want to open and double click on it. The Servo Setup window is displayed.

Inserting an Axis in Servo Setup

An axis is added to Servo Setup by using the Insert command from the menu, pressing the **Insert** key, or right clicking and selecting one of the insert menu items. You can insert a servo, digitizing (read-only) or time axis. Digitizing axes have no output type. Time axes have no input or output type.

Servo setup will number the axes sequentially as they are inserted.

Each time you insert an axis you must enter the axis properties for that axis.

Servo Axis Limitations

In PiCPro for Windows V13.0 Professional Edition, regardless of the CPU type selected, the limitations are:

- The maximum number of servo axes is 32. The first 16 servo axes are numbered from 1 to 16 and the second 16 are numbered from 101 to 116.
- The maximum number of digitizing axes is 32. The digitizing axes are numbered from 49 to 80.
- The maximum number of time axes is 4. The time axes are numbered from 25-28.

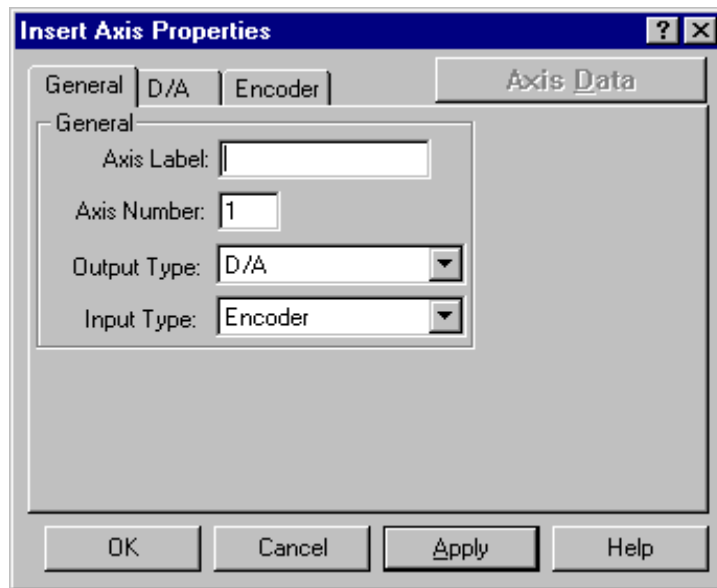
In PiCPro for Windows V13.0 Standalone MMC Edition, the standalone MMC CPU is limited to:

- The maximum number of servo axes is 4 if axes are D/A Encoder. The servo axes are numbered from 1 to 4.
- The maximum number of servo axes is 8 if axes are SERCOS. The axes are numbered from 1-8.
- The maximum number of digitizing axes is 1 if axis is Encoder. The digitizing axis is numbered 49.
- The maximum number of digitizing axes is 8 if axes are SERCOS axes. The axes are numbered 49-56.
- The maximum number of time axes is 4. The time axes are numbered from 25-28.
- The input and output channel numbers of a servo D/A Encoder axis must be the same as the axis number.
- The input channel number of a digitizing Encoder axis must be 5.

Note: All axes must be the same type in a standalone MMC or MMC for PC Servo Setup function regardless of which PiCPro Edition it is created in.

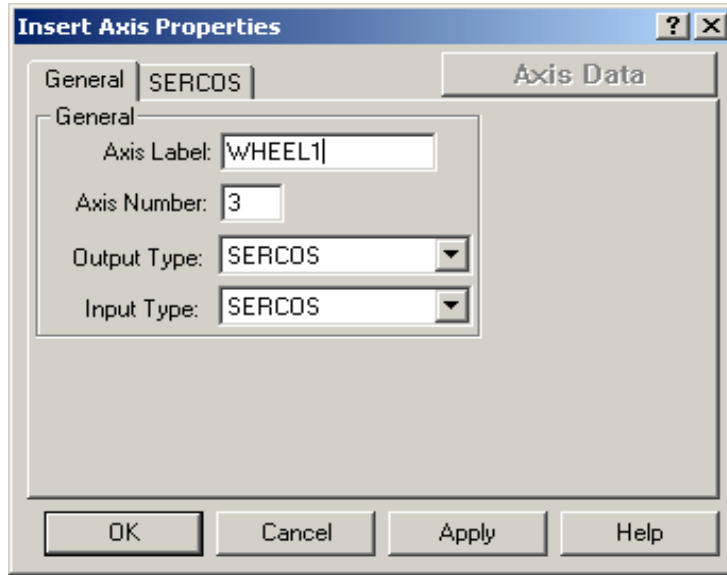
To insert a Servo Axis

1. Choose **Insert | Servo Axis**. The **Insert Axis Properties** box appears.



2. Enter a name for the axis in the **Axis Label:** box.
3. The **Axis Number:** is entered by PiCPro in sequential order. These can, however, be changed.
4. Enter the **Output Type:** and **Input Type:** in those boxes if they are different from the defaults you see above. Typically, a servo axis has a D/A output and an Encoder input.
Note: If you need to change the defaults, change the Output Type first since the Input Type choices vary based on your output selection.

The **Insert Axis Properties** dialog when using a SERCOS Output appears as:



5. Click on the tabs that appear to configure the output and input devices for your application. Typically, you will need to specify at least the input and output slot and channel for each device.
6. When you are done configuring your input and output properties, click on **Apply** to activate the **Axis Data** button for this axis OR click on **OK** to continue inserting axes.

Input / Output Types

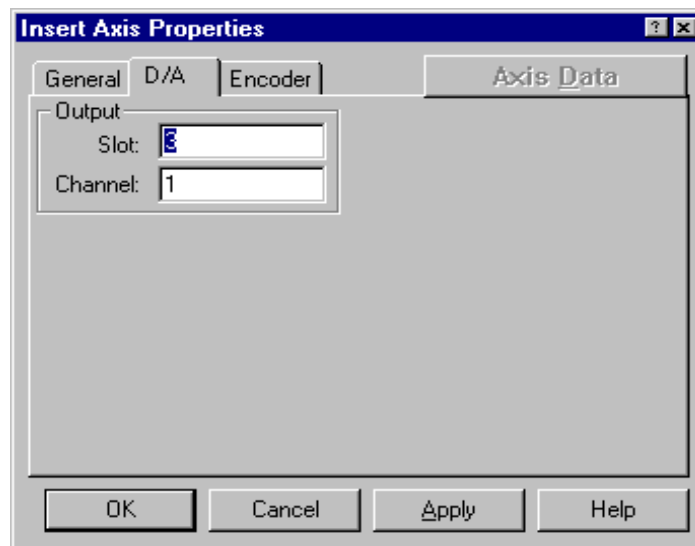
The various combinations of Input and Output types are listed below.

CPU Type	For a Servo Axis		For a Digitizing Axis	
	Inputs	Output	Inputs	Output
PiC	Encoder Resolver Analog TTL SERCOS No Input	D/A (+/- 11V) D/A D/A D/A SERCOS Stepper	Encoder Resolver Analog TTL	No Output
Standalone MMC & MMC for PC	Encoder SERCOS	D/A (+/- 10V) SERCOS	Encoder SERCOS	No Output

D/A Output Setup

To configure the D/A Output

1. Enter the **Slot:** and **Channel:** location of the D/A module.
 - For a PiC CPU: Slot is between 3 – 13, Channel is 1– 8
 - For a standalone MMC: Slot is limited to 1, Channel is 1 – 4.
Note: In PiCPro Standalone MMC Edition, Channel is limited to axis number. If the axis number is later changed, you must modify the channel number.
 - For an MMC for PC: Slot is 1 – 8, where it represents the ASIU number, Channel is 1– 4.
2. Click **OK**.



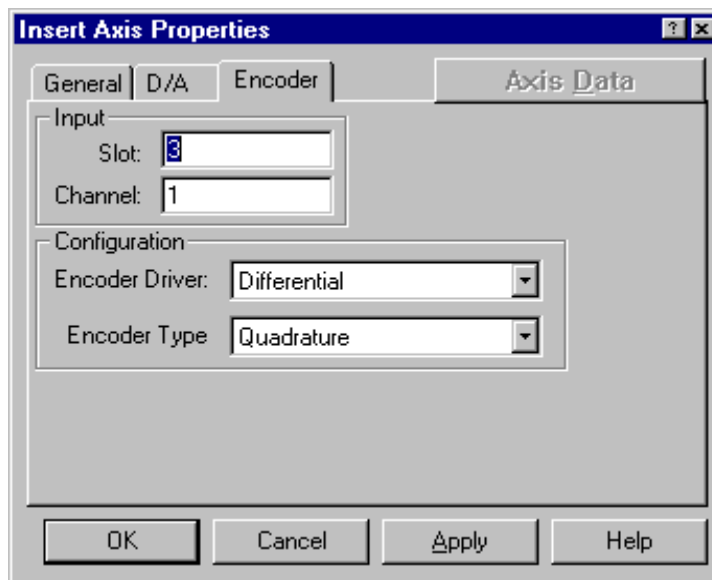
Encoder Input Setup

To configure the Encoder Input

1. Enter the **Slot:** and **Channel:** location of the encoder module.
2. Select the correct **Encoder Driver:**
3. Select the correct **Encoder Type:**

Note: Quadrature type assumes that there are four counts for each quadrature cycle. The rising and falling edges of both channel A and channel B are counted. Pulse type counts either channel A or channel B so that one count is recorded per cycle.

4. Click **OK**.



The options available on the **Encoder** tab will vary depending on CPU type.

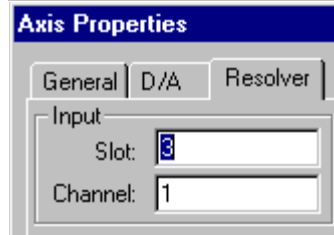
CPU Type	Slot	Channel	Driver	Type
PiC	3-13	1-4	Differential Single-ended	Quadrature Pulse
Standalone MMC	1	1-4 Servo 1-5 Digitizing	Differential	Quadrature
MMC for PC	1-8 (ASIU)	1-4 Servo 1-5 Digitizing	Differential	Quadrature

Note: In PiCPro Standalone MMC Edition, channel is limited to axis number for servo axes. When a new axis is inserted, channel will be set to axis number. If axis number is later changed, you must modify the channel number too. Channel is limited to 5 for a digitizing axis.

Resolver Input Setup (Only for PiC CPU)

To configure the resolver input

1. Enter the **Slot:** and **Channel:** location of the resolver module.
2. Click **OK**.

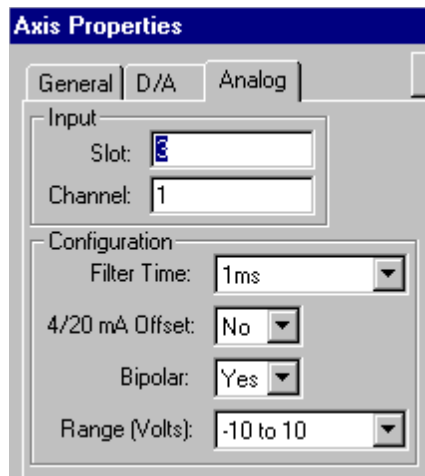


Note: An inductosyn device may also be used. Choose resolver if you will be using an inductosyn.

Analog Input Setup (Only for PiC CPU)

To configure the analog input

1. Enter the **Slot:** and **Channel:** location of the analog input module.
2. Select the **Filter Time:** from the drop down list.
3. If you are using the module in the **4/20 mA Offset:** mode, select **Yes** from the drop down list.
Note: The **Bipolar:** box will be changed to **No** and the unipolar input **Range (Volts):** will be changed to 0 to 5. These defaults cannot be changed when in the 4/20 mA mode.
4. If you are not using the module in the **4/20 mA Offset:** mode, select the Bipolar or Unipolar voltage range from the drop down list.
5. Click **OK**.

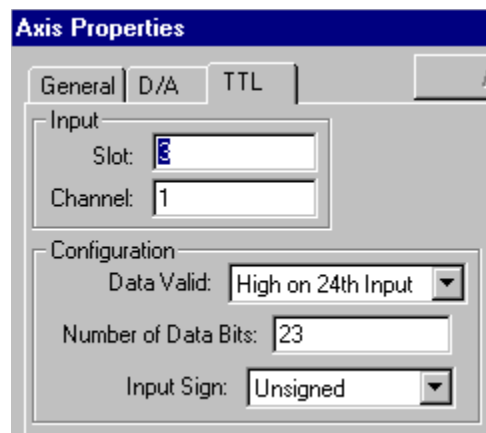


TTL Input Setup (Only for PiC CPU)

To configure the TTL input

Selections for the TTL configuration are based on the type of feedback device.

1. Enter the **Slot:** and **Channel:** location of the TTL input module.
2. Select the **Data Valid:** from the drop down list. (See the note below).
3. Enter the **Number of Data Bits:** your device requires.
 - A minimum of eight bits is required.
 - A maximum of 16 and a minimum of eight bits can be used if Same or Gray code is selected. The 24th input is not used.
 - A maximum of 23 bits can be used if High or Low is selected with the 24th bit used as an indicator of valid data.
4. The input unsigned/signed defaults to unsigned. Most TTL devices return an unsigned pattern with one end of travel all 1's and the other end all 0's. If the device returns a signed pattern with all 0's in the center of travel, change the default to signed.
5. Click **OK**.



Note: TTL data is defined as valid depending on what is selected from the list.

Binary	High	Whenever input 24 is high, the inputs are not allowed to change.
	Low	Whenever input 24 is low, the inputs are not allowed to change.
	Same	Whenever two consecutive reads of the inputs are the same, the 24 th input is not used.
Gray Code		This is an alternative to binary encoding. Only one bit of a gray code number changes at a time. The 24 th input is not used.

SERCOS Setup

When a SERCOS slave is connected to a servo axis, SERCOS is chosen as the feedback module.

To configure a SERCOS axis

1. Enter the **Slot:** and **Ring:** numbers for the SERCOS axis. The slot number identifies the slot in which the SERCOS module is installed in the control. The ring number identifies whether this slave axis is on the first or second ring on a SERCOS module.
2. Enter the **Slave Number:** for the SERCOS axis. The number entered here must match the address of the slave.

Note: The slot number, ring number, and slave number correspond to the SRS input on the SERCOS functions which identify this slave axis.

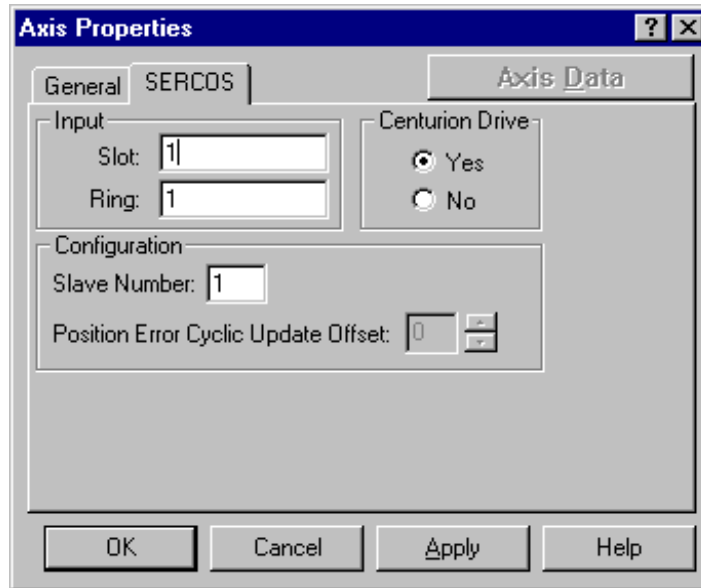
3. Indicate whether or not the drive is a Centurion Drive.

Different SERCOS drives use a different number of update cycles to accept a command position and return an actual position. This number of update cycles affects the accuracy of the approximate position error in the control.

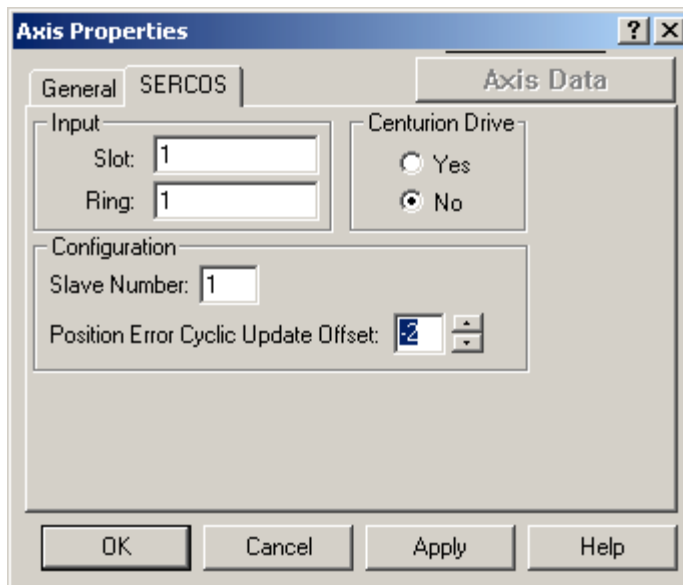
The **Position Error Cyclic Update Offset:** allows the position error approximation to be optimized for the SERCOS drive being used. A default value of zero provides the optimum approximation for a Centurion drive. When the drive type is Centurion (Yes is checked), Position Error Cyclic Update Offset: is set to 0 and cannot be changed.

If a different drive is used (Centurion Drive is set to No), this number may be changed (within the range [-5,2]) to achieve a more accurate position error approximation in the control. Reducing the number will increase the approximate position error value. Increasing the number will reduce the approximate position error value. The control's approximate position error is only used for display (READ_SV and Servo View List), In Position calculations, and Excess Following Error calculations. Changing this value does not affect the actual position error in the drive used to maintain closed loop control of the axis.

When the drive type is Centurion (**Yes** is checked), **Position Error Cyclic Update Offset:** is set to 0 and cannot be changed.



When the drive is not Centurion (**No** is checked), **Position Error Cyclic Update Offset:** defaults to 0 but can be changed to any value between -5 and 2:



Note: for a SERCOS Digitizing Axis, **Position Error Cyclic Update Offset:** fields are always hidden because they do not apply to digitizing axes.

SERCOS SETUP					
CPU Type	Number of Rings	Slaves per Ring	Slot Number	Ring Number	Slave Number
PiC	16	8	3-13	1-2	1-8
standalone MMC	1	8	1	1	1-8
MMC for PC	1	32	1	1	1-32

Stepper Setup (PiC CPU only)

To configure the stepper module

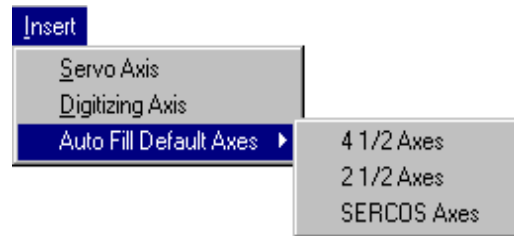
1. Enter the **Slot:** and **Channel:** location of the stepper module.
2. Enter the **Type:** of stepper you are using, CW/CCW or Step/Direction. The default is CW/CCW.

The image shows a software configuration window for a stepper module. It has two tabs: 'General' and 'Stepper'. The 'Stepper' tab is active. The window is divided into two sections: 'Output' and 'Configuration'. In the 'Output' section, there are two input fields: 'Slot' with the value '3' and 'Channel' with the value '1'. In the 'Configuration' section, there is a dropdown menu for 'Type' which is currently set to 'CW/CCW'.

Default axes using Auto Fill (Standalone MMC)

Auto Fill is only available in PiCPro when a standalone MMC is the Target CPU.

1. Choose **I**nsert | **A**uto Fill Default Axes. Select the appropriate menu for **4 1/2 Axes**, **2 1/2 Axes**, or **SERCOS Axes**.



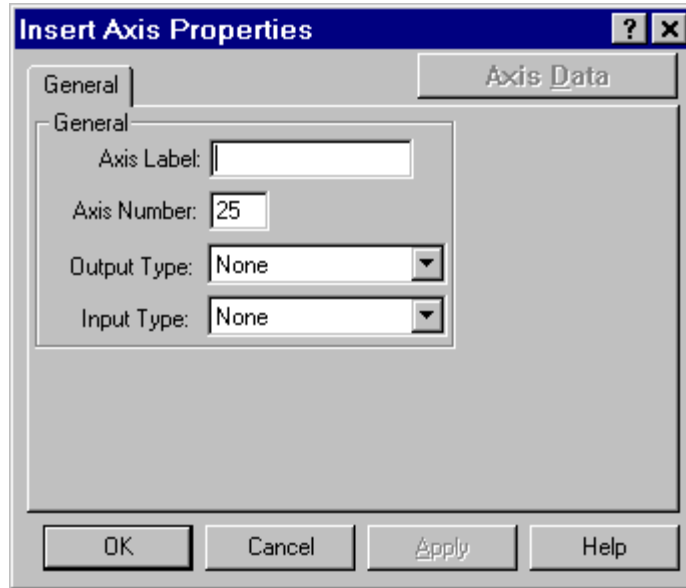
If axes are present when one of these menu items is selected, a message will be displayed to indicate that all axes must be removed before default axes can be inserted. If the axis count is 0 and 4 1/2 Axes or 2 1/2 Axes is selected, 4 or 2 servo axes will be added and 1 digitizing axis will be added. The servo axes will have an input type of Encoder and an output type of D/A. The digitizing axis will have an input type of encoder. The servo axes will be named AXIS1-4. The digitizing axis will be named AXIS49.

If the axis count is 0, and SERCOS Axes is selected, 8 SERCOS servo axes and 8 SERCOS digitizing axes will be added. Slot and ring will be set to 1. The slave numbers will be 1-8. The servo axes will be named AXIS 1-8. The digitizing axes will be named AXIS 49-56.

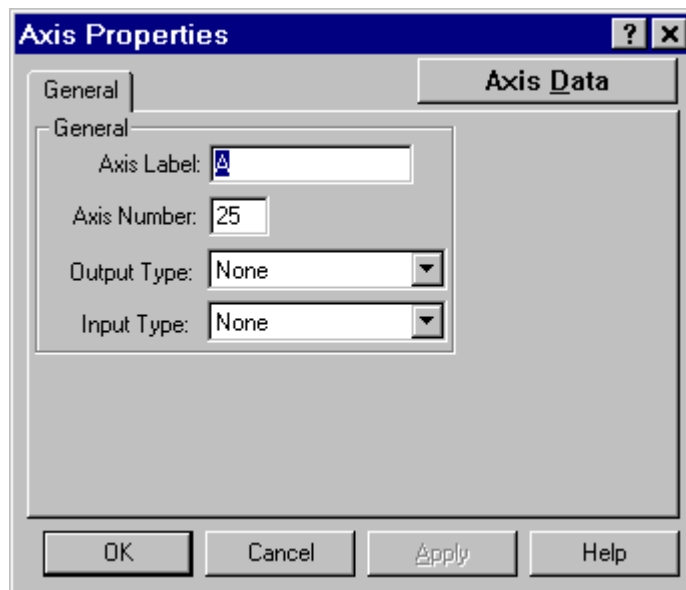
To insert a Time Axis

1. Choose **Insert | Time Axis** or press the insert key and select **Time Axis**, or right click the mouse button and select **Insert | Time Axis** from the context sensitive menu. Up to 4 time axes can be inserted.

The **Insert Axis Properties** box appears.



2. Enter a name for the axis in the **Axis Label:** box.
3. The **Axis Number:** is entered by PiCPro in sequential order. These can, however, be changed. The axis number for a time axis is limited to 25 - 28.
Note: Output Type: and **Input Type:** are not available for a Time Axis and appear as **None**.
4. Click on **Apply** to activate the **Axis Data** button for this axis OR click on **OK** to continue inserting axes.



Editing a Servo Setup File

Your servo setup file contains the data for one or more axes. When you open a servo setup file, the axes are ordered by axis type (Servo, Digitizing, Time) . You entered the properties and data for each axis when you inserted them into the file. You can edit that information.

Editing Axis Properties

You can access the axis properties box in one of the following ways.

- With the axis selected, choose **E**dit | **A**xis **P**roperties from the menu or press the <**E**nter> key.
- With the axis selected, right click and choose **A**xis **P**roperties.
- Double click on the axis.

Make the necessary changes and choose **OK**.

Editing Axis Data

You can access the axis data box in one of the following ways.

- With the axis selected, choose **E**dit | **A**xis **D**ata from the menu or press the <**A**lt + **E**nter> key.
- With the axis selected, right click and choose **A**xis **D**ata.

Make the necessary changes to Scaling, Iterator, or Position Loop Data and choose **OK**.

Servo / Digitizing Axis Setup Data Categories

There are three categories of servo axis setup data in the Servo Setup program:

1. **Scaling data** - sets up the ratio between feedback units, ladder units, and axis units. This allows you to enter axis units instead of feedback units in the appropriate places for iterator and position loop data in setup. The three type of units are defined below:
 - Feedback units - the units the servo software uses to perform its calculations and issue its commands in.
 - Ladder units - the units used in the ladder program. They must be integers.
 - Axis units - the units of measurement (inches, millimeters, degrees) for the system. They may be integers or non-integers (decimals) and are used in the Servo setup program to enter certain types of setup data.
2. **Iterator data** - specifies how data such as limits, ramps, filters, and rollovers will be handled by the move iterator.
3. **Position loop data** - provides the servo software with information on how the axis is set up for the position loop.

Servo Axis Data - [Axis #1]

Scaling Data | Iterator Data | Position Loop Data

Input Scaling

Feedback Units	1
Ladder Units	1
Ladder Units/Axis Units	1

Output Scaling

Commanded Voltage (mV)	10000
Motor RPM at Voltage	4000
Counts/Motor Revolution	8000

Calc Defaults

OK Apply Cancel Help Export Import

Entering Scaling Data

Input Scaling Feedback units, ladder units, and ladder units/axis units establish a scaling ratio between the feedback units the servo software uses to perform its calculations and issue its commands, the units used in the ladder, and the units entered in setup.

Note: When using SERCOS feedback, it is recommended that the units be entered in a 1:1 ratio.

Feedback Units - These are the units the servo software uses to perform its calculations and issue its commands.

Ladder Units - These are the units used in the ladder program. Ladder units must be integers and cannot be fractional values.

The ratio of the feedback device signal to the PiC feedback units is:

One revolution of a resolver = x feedback units

One pulse of an encoder = one feedback unit

Note: One cycle of a quadrature type encoder equals four pulses.

Ladder Units/Axis Units - Axis units are the units of measurement (inches, millimeters, degrees) used in your system. They are used to enter values for several setup parameters.

Enter the ratio of ladder units to axis units. (1, 10, 100, 1,000, 10,000, or 100,000)

IMPORTANT

Keep in mind that certain iterator and position loop data is entered in axis units. The default values are also in axis units and are sometimes at the upper limit of an acceptable value (ramps and software limits). If you choose a ratio for your ladder units/axis units that is greater than 1, then these default values will exceed the limit. Change the default value to a lower value for those parameters.

Output Scaling These three parameters provide the information the controller needs to be able to calculate how much voltage the analog output will need to operate the axis at a certain speed and how many feedback units will be received for each motor revolution.

Commanded Voltage - Enter commanded voltage in millivolts.

Motor RPM at Voltage - Enter the motor RPM at the Commanded Voltage entered.

Counts/Motor Revolution – Enter the number of feedback units that occur for each motor revolution.

Note: The Commanded Voltage, Motor RPM at Voltage, and Counts/Motor Revolution parameters are used for scaling servo calculations by the servo software. They do not represent limits. They do not apply to Stepper, SERCOS, digitizing or time axes.

Calculate Defaults

The **Calc Defaults** button can be used to calculate new default values of Iterator Data and Position Loop Data as long as the ratio of feedback units to ladder units (FU/LU) is less than 65535. (Not available for Digitizing Axes)

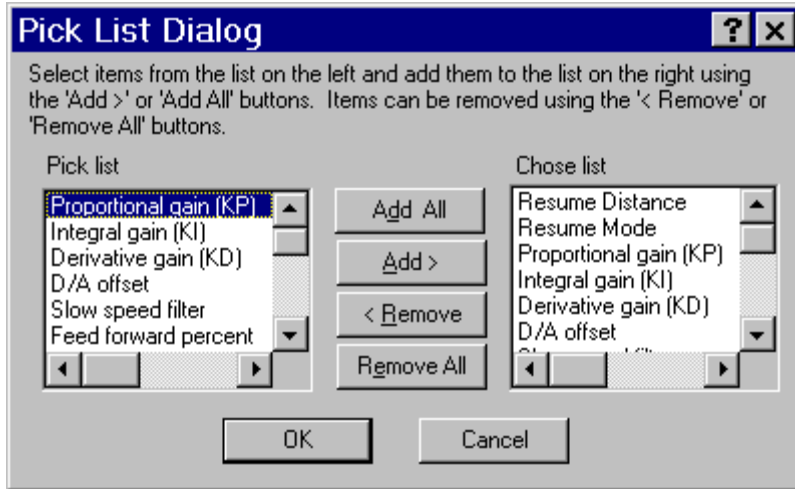
Data Field	Default Calculation
Velocity Limit (AU/min)	Counts/Motor Rev x Motor RPM at Voltage x AU/LU x LU/FU
Acceleration Ramp (AU/min/sec)	Velocity Limit / 2 sec.
Deceleration Ramp (AU/min/sec)	Velocity Limit / 2 sec.
Controlled Stop Ramp (AU/min/sec)	Velocity Limit / .2 sec.
Software Limit (Upper) (AU)	536870911 / (FU/LU x LU/AU)
Software Limit (Lower) (AU)	-536870912 / (FU/LU x LU/AU)
Following Error Limit (AU)	(Counts/Motor Rev / 8) x (AU/LU) x (LU/FU)
In Position Band (AU)	10 LU x AU/LU
(+) Integral Error Limit (AUFE x update)	In Position Band x 10
(-) Integral Error Limit (AUFE x update)	-In Position Band x 10
Max Acceleration (AU/min/sec)	(3/2) x Velocity Limit / 3 sec.
Constant Jerk	(3) (Max Acceleration) / 3 sec.

Note: If the result of a calculation would exceed the upper limit or fall below the lower limit of a given field, the field will be set to that limit.

Determining Scaling Information

To determine what scaling information to enter for the number of FU/min/volt of the D/A, you can use the Servo Force list (write) and the Servo View List (read).

1. Highlight the axis in Servo Setup and open the **View | Servo Force List**.
2. Click the **Add/Remove** to bring up the **Pick List Dialog**.



Button	Description
Add All	Click to add all items from the Pick list to the Chose list.
Add	Click to add selected item from the Pick list to the Chose list.
< Remove	Click to remove selected item from the Pick list.
Remove All	Click to remove all items from the Pick list.
OK	Click to save changes made to the chose list.
Cancel	Click to close without saving changes made to the chose list.

3. Highlight and then **Add>** the **Feed Forward Percent** parameter to the list.
4. Click **OK**.
5. Enter 100% for the value of Feed Forward Percent.
6. Move the axis at a fixed velocity in either always the positive or always the negative direction.
7. Observe the following error in the Servo View List. If the following error is not zero, the velocity scaling is incorrect.

8. Change the Motor RPM at Voltage in the Output Scaling of Axis Data.
For example, if you are jogging in the positive direction and the error is always negative, increase the RPMs and vice versa.
9. Make the servo setup function and download the ladder to the controller.
10. Again, observe the following error in the **Servo View List**.
11. Repeat the above steps until the following error is close to zero.
12. Finally, remake the servo setup function and perform a complete download of the ladder.

Entering Iterator Data for Servo Axis

Servo Axis Data - [Axis #1]

Scaling Data | **Iterator Data** | Position Loop Data

Iterator Data

Velocity Limit (AU/min)

Slow Velocity Filter (ms)

Fast Velocity Filter (ms)

Slow/Fast Velocity Threshold (AU/min)

Rollover Position (AU)

Ignore Limits?

Software Limit (Upper) (AU)

Software Limit (Lower) (AU)

Resumable E-Stop Allowed?

Linear Iterator

Acceleration Ramp (AU/min/sec)

Deceleration Ramp (AU/min/sec)

Controlled Stop Ramp (AU/min/sec)

S-Curve Iterator

Enable S-Curve

Move Accel/Decel

Max Acceleration (AU/min/sec)

Constant Jerk (AU/min/sec/sec)

Controlled Stop Decel

Max Acceleration (AU/min/sec)

Constant Jerk (AU/min/sec/sec)

OK Apply Cancel Help Export Import

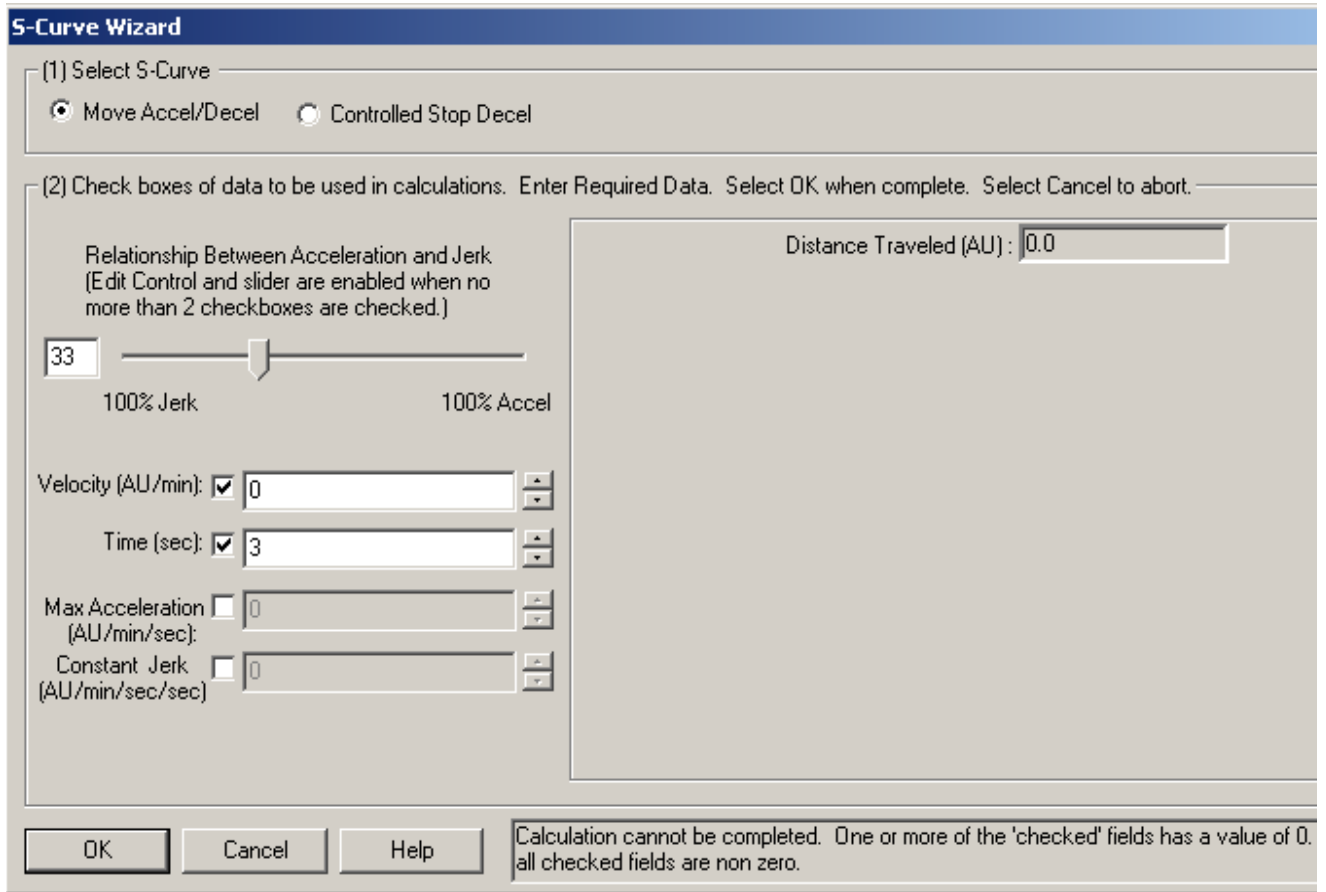
The following parameters comprise the Iterator Data for Servo, SERCOS, or Stepper Axis:

Iterator Data	Description
Velocity Limit (AU/min)	<p>Enter the maximum velocity the motor should ever be commanded to in axis units/min. Limits the maximum motor RPM. Does not apply to a slave axis.</p> <p style="text-align: center;">WARNING</p> <p>The position loop may cause the axis to travel faster than this speed if enough position, integral, and/or derivative error accumulates. Typical values are 100,000 to 10,000,000 AU/min (where 1 AU = 1 LU), not to exceed 4095 FU/update. Note: If 1 AU does not equal 1 LU, scale these values accordingly.</p>
Slow Velocity Filter	<p>Enter in milliseconds the filter it will take to smooth out a “step” change in velocity while the axis is moving at slow velocities.</p> <p>Note: Specifically, the value entered represents the milliseconds that the servo software takes to carry out 63.2% of the step change. Range = 0 - 10,000 ms</p>
Fast Velocity Filter (ms)	<p>Enter the milliseconds the filter it will take to smooth out a “step” change in velocity while the axis is moving at fast velocities.</p> <p>Note: Specifically, the value entered represents the milliseconds that the servo software takes to carry out 63.2% of the step change. Range = 0 - 10,000 ms</p>
Slow/Fast Velocity Threshold (AU/min)	<p>Enter the velocity in axis units/minute (AU/min) at which the slow or fast velocity filter should be applied.</p> <p>Typical value is 0, not to exceed 4095 FU/update.</p>
Rollover Position (AU)	<p>Enter the rollover position in axis units.</p> <p>Valid range is 0 to 536,870,911 FU. If the value is zero, rollover is off. If the value is non-zero, rollover is on.</p>
Ignore Limits?	<p>An Until Ref. here allows a machine reference to occur without exceeding the software limits designated earlier. This is useful if you exceed the software limits because of how far you have to travel to your reference switch or where your reference switch is located. After the machine reference is complete, the software limits are in effect. If Yes is selected, software limits cannot be set and .srv file cannot be saved to a format prior to V13.0. If No is selected, software limits are always in effect.</p>
Software Limit (Upper) (AU)	<p>Enter the software-imposed upper travel limit in axis units. Exceeding this limit will generate a C-stop condition.</p> <p>Typical values are 100 to 1,000,000 AU (where 1 AU = 1 LU), not to exceed 536,870,911 FU. Note: If 1 AU does not equal 1 LU, scale these values accordingly.</p>
Software Limit (Lower) (AU)	<p>Enter the software-imposed lower travel limit in axis units. Exceeding this limit will generate a C-stop condition.</p> <p>Typical values are 100 to 1,000,000 AU (where 1 AU = 1 LU), not to exceed 536,870,911 FU. Note: If 1 AU does not equal 1 LU, scale these values accordingly</p>

Resumable E-Stop Allowed?	<p>Options are Yes or No. Select Yes to allow Resumable E-Stop. Default value is No.</p> <p>If Yes is selected, the E_STOP function and the Excess Following Error E-Stop will execute a Resumable E-Stop. When a Resumable E-Stop occurs, the following happens:</p> <ol style="list-style-type: none">1. The servo loop is opened.2. Zero voltage is sent to the analog outputs.3. The moves in the active and next queues remain intact.4. The axis' Normal Interpolator remains running.5. The axis goes into Resume Mode. In Resume Mode, the axis will follow the Resume Interpolator. The Resume Interpolator will output zero velocity until the RESUME function is called. The RESUME function can only be called after the Resumable E-Stop has been reset and the servo loop has been closed. The axis remains in Resume Mode until the RESUME function brings it back on path or until a non-resumable E-Stop occurs and cancels Resume Mode. <p>If No is selected, the E_STOP function and the Excess Following Error E_Stop will execute a normal E_Stop (i.e. open the servo loop, zero voltage to the Analog outputs, and clear the active and next queues)</p> <p>The E_STOP function and the Excess Following Error E-Stop are the only types of E-Stops that are resumable. All other types of E_Stops will execute normally regardless of this selection.</p>
---------------------------	---

Linear Iterator	
Acceleration Ramp (AU/min/sec)	Enter the rate in axis units/minute/second at which the servo software will iterate the axis to reach a higher velocity command. Typical values are 10,000 to 10,000,000 AU/min/sec (where 1 AU = 1 LU), not to exceed 1023 FU/update/update. Note: If 1 AU does not equal 1 LU, scale these values accordingly.
Deceleration Ramp (AUmin/sec)	Enter the rate in axis units/minute/second at which the servo software will iterate the axis to reach a lower velocity command. Typical values are 10,000 to 10,000,000 AU/min/sec (where 1 AU = 1 LU), not to exceed 1023 FU/update/update. Note: If 1 AU does not equal 1 LU, scale these values accordingly.
Controlled Stop Ramp (AU/min/sec)	Enter the rate in axis units/minute/second at which the servo software will iterate the axis to reach a controlled stop. Typical values are 10,000 to 10,000,000 AU/min/sec (where 1 AU = 1 LU), not to exceed 1023 FU/update/update. Note: If 1 AU does not equal 1 LU, scale these values accordingly.
S-Curve Iterator	
Enable S-Curve	Check this box to enable the S-Curve Iterator. If enabled, S-Curve is the default iterator for this axis. If not enabled, it cannot be enabled from the ladder.
S-Curve Wizard	Selectable if Enable S-Curve box is checked. Select this button to display the S-Curve Wizard dialog that will assist in defining max acceleration and constant jerk for Move Accel/Decel and Controlled Stop Decel.
Move Accel/Decel	
Max Acceleration (AU/min/sec)	Enter the maximum acceleration for Move Accel/Decel.
Constant Jerk (AU/min/sec/sec)	Enter the constant jerk for Move Accel/Decel.
Controlled Stop Decel	
Max Acceleration (AU/min/sec)	Enter the maximum acceleration for Controlled Stop Decel.
Constant Jerk (AU/min/sec/sec)	Enter the constant jerk for Controlled Stop Decel.

Entering S-Curve Data for Servo Axis



(1) Select S-Curve

Move Accel/Decel Controlled Stop Decel

(2) Check boxes of data to be used in calculations. Enter Required Data. Select OK when complete. Select Cancel to abort.

Relationship Between Acceleration and Jerk
(Edit Control and slider are enabled when no more than 2 checkboxes are checked.)

33 100% Jerk 100% Accel

Distance Traveled (AU): 0.0

Velocity (AU/min): 0

Time (sec): 3

Max Acceleration (AU/min/sec): 0

Constant Jerk (AU/min/sec/sec): 0

OK Cancel Help

Calculation cannot be completed. One or more of the 'checked' fields has a value of 0. all checked fields are non zero.

Using this dialog, S-Curve information can be specified for both **Move Accel/Decel** and **Controlled Stop Decel**. Selecting **Move Accel/Decel** will display known information for **Move Accel/Decel**. Selecting **Controlled Stop Decel** will display known information for **Controlled Stop Decel**. When initially displayed:

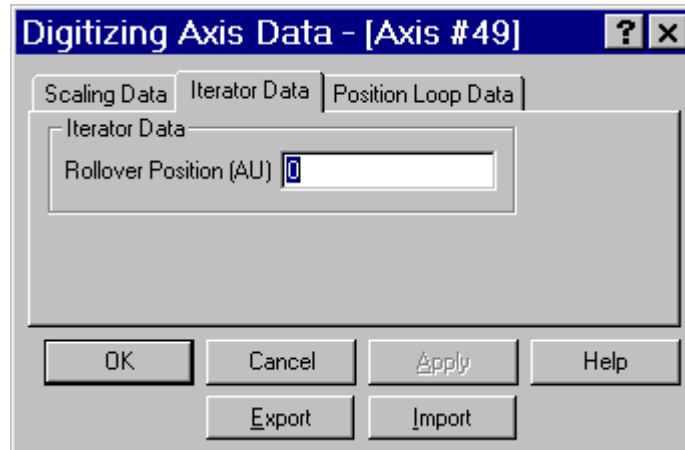
- **Move Accel/Decel** will be selected.
- If S-Curve information has previously been entered, existing values will be entered, and checkboxes checked accordingly. The slider will indicate the relationship between acceleration and jerk. The resulting S-Curve will be drawn with Distance Traveled displayed at the top.
- If S-Curve information has not been entered, Velocity will be checked and set to a value equal to that of Velocity Limit as specified in the iterator data. In addition Time will be checked and set to a value of 3 seconds. Constant Jerk and Max Acceleration will be calculated based on $\frac{1}{3} \frac{1}{3} \frac{1}{3}$ Velocity. The slider will be set to approximately $\frac{1}{3}$ from the left. And the resulting SCURVE will display Distance Traveled.

The checkboxes are used to indicate data that is entered. The values for fields where the checkbox is not checked will be calculated. At least two fields, but no more than three at the most can be checked in order for a calculation to take place and the graph to be updated. The spinner controls increment or decrement the value in the associated field by 1% of the value. They are enabled when the associated checkbox is checked.

The following parameters comprise the S-Curve data:

S-Curve Data	Description
Move Accel/Decel:	Select this button to display the S-Curve information for Move Accel/Decel.
Controlled Stop Decel:	Select this button to display the S-Curve information for Controlled Stop Decel. Note: This button will be disabled if S-Curve is being defined for a time axis.
Relationship Between Acceleration and Jerk	Enter % acceleration here or use the slider to express the relationship between acceleration and jerk. When the slider is positioned at the far left, the relationship is 0% acceleration, 100% jerk. If positioned at the far right, the relationship is 100% acceleration, 0% jerk. If positioned in the middle, the relationship is 50% acceleration, 50% jerk. Note: These controls will be enabled when no more than 2 checkboxes are checked.
Velocity (Au/min):	Check this box to use the velocity entered in the calculations. If the Velocity checkbox is checked, a velocity can be entered here and the spinner can be used to increase or decrease the value entered in the edit control. If the Velocity checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated velocity will be displayed here.
Time (sec):	Check this box to use the time entered in the calculations. If the Time checkbox is checked, a time can be entered here and the spinner control can be used to increase or decrease the value entered in the edit control. If the Time checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated time will be displayed here.
Max Acceleration (AU/min/sec):	Check this box to use the acceleration entered in the calculations. If the Max Acceleration checkbox is checked, an acceleration can be entered here and the spinner control can be used to increase or decrease the value entered in the edit control. If the Max Acceleration checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated Max Acceleration will be displayed here.
Constant Jerk (AU/min/sec/sec)	Check this box to use the jerk entered in the calculations. If the Constant Jerk checkbox is checked, Constant Jerk can be entered here and the spinner control can be used to increase or decrease the value entered in the edit control. If the Constant Jerk checkbox is not checked, the edit control (data entry field) will be grayed and disabled. The calculated Constant Jerk will be displayed here.
Distance Traveled (AU):	If an S-Curve is displayed, the distance traveled is displayed here.
OK:	Press to save data entered and return to the Iterator Data tab of the Axis Data dialog.
Cancel:	Press to close the dialog without saving data and return to the Iterator Data tab of the Axis Data dialog.
Help:	Press to display information related to the S-Curve Wizard dialog.

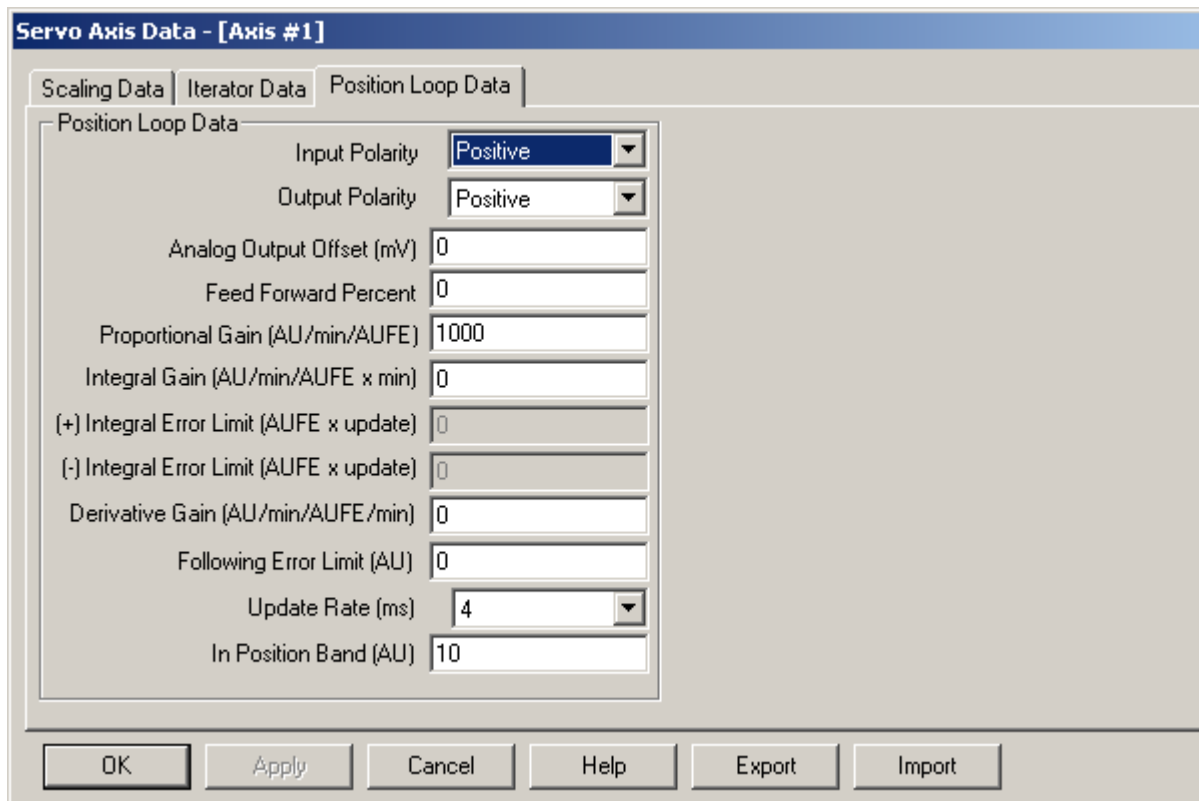
Entering Iterator Data for Digitizing Axis



Iterator Data	Description
Rollover Position (AU)	Enter the rollover position in axis units (AU) in this field. Valid range is 0 to 536,870,911 FU. If the value is zero, rollover position is off. If the value is non-zero, rollover position is on.

Entering Position Loop Data for Servo Axis

Position loop data provides the servo software with information on how the axis is set up for the position loop.



The following parameters comprise the Position Loop parameters:

Parameter	Description
Input Polarity	Provides a software method of changing the polarity of the position feedback device. If the input polarity is incorrect, the axis will either run away or move in the wrong direction. If this occurs, enter the opposite polarity here.
Output Polarity	Provides a software method of changing the polarity of the analog output. If the output polarity is incorrect, the axis will either run away or move in the wrong direction. If this occurs, enter the opposite polarity here.
Analog Output Offset (mV)	If it is not possible to get a zero volts reading from a voltmeter placed across the analog output channel for the axis, enter the amount of voltage in millivolts that allows you to reach a zero reading. Range: -10,000 to +10,000 mV
Feed Forward Percent	Enter a percentage (from 0 to 100%) that you want the position loop to compensate for the lag that occurs between the generation of the following error and the correction of that error by the PID calculations. Range: 0 to 100%
Proportional Gain (AU/min/AUFE)	Proportional gain calibrates corrective action proportional to the amount of following error. The value entered represents the axis unit per minute for each axis unit of following error. Typical values are 1,000 - 5,000 AU/min/AUFE
Integral Gain (AU/min/AUFE x min))	Integral gain determines corrective action proportional to the amount of following error summed over the time duration of the error at a zero velocity command. The longer the following error exists, the greater the integral error. The value entered represents the number of axis units per minute per axis unit of following error times minutes. Typical value is 0. If required, up to 32,000 AU/min/AUFE x min
(+) Plus Integral Error Limit (AUFE x update)	Tell the position loop to ignore any additional integral error beyond the value entered when the axis is moving in a positive direction. Typical value is 0. If required, from 100,000 to 500,000,000 AUFE x update (where 1 AU = 1 LU). Note: If 1 AU is not equal to 1 LU, scale these values accordingly.
(-) Minus Integral Error Limit (AUFE x update)	Tells the position loop to ignore any additional integral error beyond the value entered when the axis is moving in a negative direction. Typical value is 0. If required, from -500,000,000 to -100,000,000 AUFE x update (where 1 AU = 1 LU). Note: If 1 AU is not equal to 1 LU, scale these values accordingly.
Derivative Gain (AU/min/AUFE/min)	Derivative gain determines the corrective action proportional to the magnitude of change of the following error. The value entered represents the number of axis units per min for each axis unit of following error per minute. Typically, this is set to 0. Typical value is 0. If required, up to 500 AU/min/AUFE/min
Following Error Limit (AU)	Enter the amount of excess error in axis units to allow before an E-stop condition occurs. Typical values are 10 to 10,000 AU (where 1 AU = 1 LU), not to exceed 536,870,911 FU. Note: If 1 AU is not equal to 1 LU, scale these values accordingly.

Parameter	Description														
Update Rate (ms)	<p data-bbox="626 195 1422 342">Enter in milliseconds, (1, 2, 4, or 8) how often the servo update occurs. 4 ms is adequate for most applications. Lower values consume more CPU processing time. If too many axes have too low an update rate, the CPU may not have enough processing time available to run the application program.</p> <p data-bbox="626 369 1252 401">The iteration rate is eight times the servo update rate:</p> <table data-bbox="626 401 971 600"> <tr> <td>SUG (ms)</td> <td>Iteration Rate (ms)</td> </tr> <tr> <td>.25</td> <td>2</td> </tr> <tr> <td>.5</td> <td>4</td> </tr> <tr> <td>1</td> <td>8</td> </tr> <tr> <td>2</td> <td>16</td> </tr> <tr> <td>4</td> <td>32</td> </tr> <tr> <td>8</td> <td>64</td> </tr> </table> <p data-bbox="626 604 1409 663">For digitizing axes, the update rates are .25, .5, 1, 2, 4, 8, or 16 ms. A rate can be selected for each axis.</p> <p data-bbox="626 690 1422 779">Note: In master/slave moves, the iteration rate is the same as the servo update rate. The update rate for the master axis and the slave axis must be the same in any master/slave move.</p>	SUG (ms)	Iteration Rate (ms)	.25	2	.5	4	1	8	2	16	4	32	8	64
SUG (ms)	Iteration Rate (ms)														
.25	2														
.5	4														
1	8														
2	16														
4	32														
8	64														
In-Position Band (AU)	<p data-bbox="626 800 1422 947">Enter the distance in axis units that the axis must be on either side of its endpoint before the in position flag is set. This in-position flag can be used in your program. Typical values are 10 to 1,000 AU (where 1 AU = 1 LU), not to exceed 536,870,911 FU. Note: If 1 AU is not equal to 1 LU, scale these values accordingly.</p>														

Time Axis Setup Data Categories

There is only a single category of time axis setup data in the Servo Setup program that:

- specifies how data such as rollover, max acceleration and constant jerk will be handled by the move iterator,
- allows you to calculate default values for max acceleration and constant jerk.

Note: There must be at least one Servo or Digitizing axis defined if a Time Axis is defined.

Entering Iterator Data for Time Axis

Iterator Data	Description
Rollover Position (cnt)	Enter the rollover position in counts (cnt) in this field. Valid range is 0 to 536,870,911 cnt. If the value is zero, rollover is off. If the value is non-zero, rollover is on.
Enable S-Curve	Check this box to enable S-Curve for this axis. If enabled, S-Curve is the default iterator for this axis.
S-Curve Wizard	Selectable if Enable S-Curve box is checked. Select this button to display the S-Curve Wizard dialog that will assist in defining max acceleration and constant jerk for Move Accel/Decel.
Max Acceleration (cnt/min/sec)	Enter the maximum acceleration for Move Accel/Decel.
Constant Jerk (cnt/min/sec/sec)	Enter the constant jerk for Move Accel/Decel.
Calc Defaults	Select this button to initialize Max Acceleration to 30,000 cnt/min/sec and Constant Jerk to 30,000 cnt/min/sec/sec.

Entering S-Curve Data for Time Axis

(1) Select S-Curve

Move Accel/Decel Controlled Stop Decel

(2) Check boxes of data to be used in calculations. Enter Required Data. Select OK when complete. Select Cancel to abort.

Relationship Between Acceleration and Jerk
(Edit Control and slider are enabled when no more than 2 checkboxes are checked.)

33

100% Jerk 100% Accel

Velocity (cnt/min): 60000

Time (sec): 3

Max Acceleration (cnt/min/sec): 30075.1879699248

Constant Jerk (cnt/min/sec/sec): 29925.5601690794

Distance Traveled (cnt): 1500

6e+004

Velocity cnt/min

0

0 Time Sec

OK Cancel Help

Using this dialog, S-Curve information can be specified for **Move Accel/Decel**. **Controlled Stop Decel** is disabled and grayed because it is not valid for a time axis.

When initially displayed:

- **Move Accel/Decel** will be selected.
- If **Max Acceleration** and **Constant Jerk** are 0 when the S-Curve Wizard button is pressed, Max Acceleration and Constant Jerk will be calculated using a time of 3 seconds, a velocity of 60,000 cnt/min and a value of 33 for the relationship between acceleration and jerk.
- If S-Curve information has previously been entered, existing values will be entered, and checkboxes checked accordingly. The slider will indicate the relationship between acceleration and jerk. The resulting S-Curve will be drawn with Distance Traveled displayed at the top.

The checkboxes are used to indicate data that is entered. The values for fields where the checkbox is not checked will be calculated. At least two fields, but no more than three at most can be checked in order for a calculation to take place and the graph to be updated. The spinner controls increment or decrement the value in the associated field by 1% of the value. They are enabled when the associated checkbox is checked.

The following parameters comprise the S-Curve data for a Time Axis:

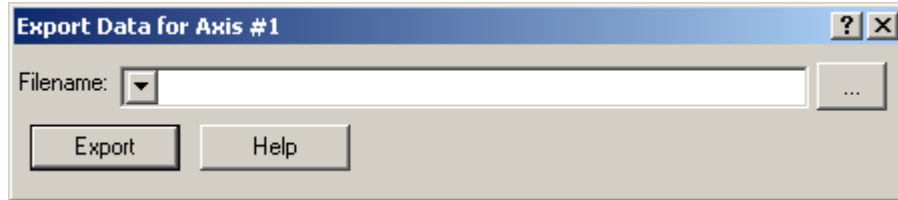
S-Curve Data	Description
Move Accel/Decel:	Select this button to display the S-Curve information for Move Accel/Decel.
Controlled Stop Decel:	This button will is disabled It is not valid for a time axis.
Relationship Between Acceleration and Jerk	Enter % acceleration here or use the slider to express the relationship between acceleration and jerk. When the slider is positioned at the far left, the relationship is 0% acceleration, 100% jerk. If positioned at the far right, the relationship is 100% acceleration, 0% jerk. If positioned in the middle, the relationship is 50% acceleration, 50% jerk. Note: These controls will be enabled when no more than 2 checkboxes are checked.
Velocity (cnt/min):	Check this box to use the velocity entered in the calculations. If the Velocity checkbox is checked, a velocity can be entered here and the spinner can be used to increase or decrease the value entered in the edit control. If the Velocity checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated velocity will be displayed here.
Time (sec):	Check this box to use the time entered in the calculations. If the Time checkbox is checked, a time can be entered here and the spinner control can be used to increase or decrease the value entered in the edit control. If the Time checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated time will be displayed here.
Max Acceleration cnt/min/sec):	Check this box to use the acceleration entered in the calculations. If the Max Acceleration checkbox is checked, an acceleration can be entered here and the spinner control can be used to increase or decrease the value entered in the edit control. If the Max Acceleration checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated Max Acceleration will be displayed here.
Constant Jerk (cnt/min/sec/sec)	Check this box to use the jerk entered in the calculations. If the Constant Jerk checkbox is checked, Constant Jerk can be entered here and the spinner control can be used to increase or decrease the value entered in the edit control. If the Constant Jerk checkbox is not checked, the adjacent edit control (data entry field) will be grayed and disabled. The calculated Constant Jerk will be displayed here.
Distance Traveled (cnt):	If an S-Curve is displayed, the distance traveled is displayed here.
OK:	Press to save data entered and return to the Iterator Data tab of the Axis Data dialog.
Cancel:	Press to close the dialog without saving data and return to the Iterator Data tab of the Axis Data dialog.
Help:	Press to display information related to the S-Curve Wizard dialog.

Importing / Exporting Axis Data

Axis data can be imported from or exported to a file which can be read by a spreadsheet or text editor. The data file will be in comma separated values (.csv) format.

Exporting Axis Data

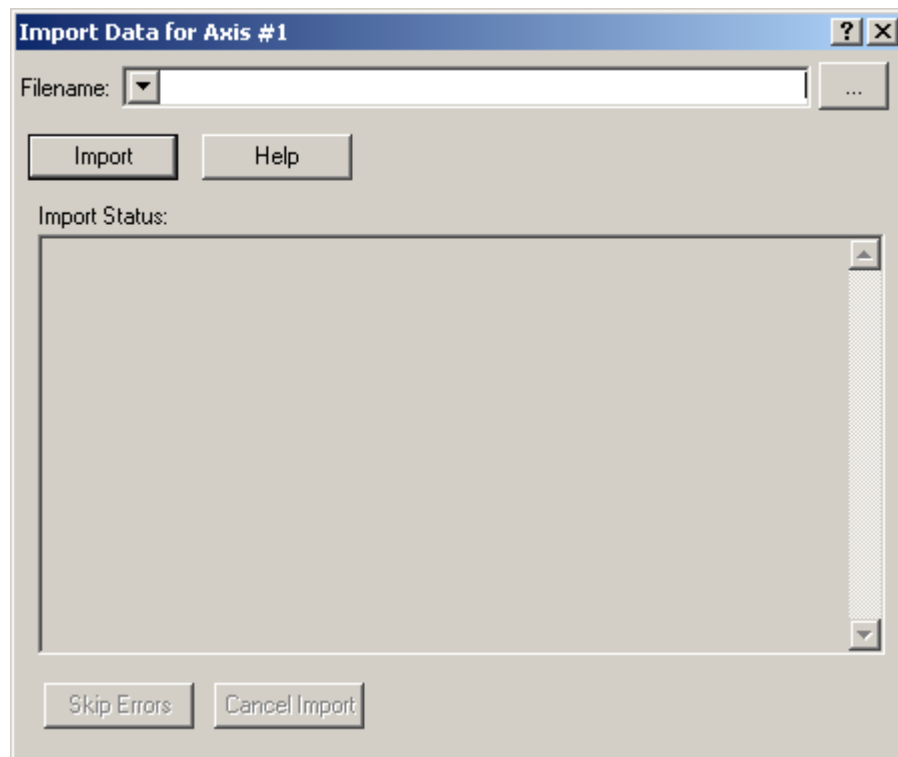
From the Axis Data dialog, click the **Export** button. To replace an existing file, select the filename from the dropdown most recently used list or use the browse (...) button.



To create a new file, that can be opened by Excel[®], enter a complete filename (include full path information) with a .csv extension. Click the **Export** button.

Importing Axis Data

From the Axis Data dialog, click the **Import** button. Select the filename from the dropdown most recently used list, use the browse (...) button, or type in the full path and filename. Then click the **Import** button.



Any error or warning messages will appear in the **Import Status:** box. The line number and type of error or warning will be listed. These errors or warnings could be caused by an invalid data field identifier, invalid value, or a field that is not valid for a particular type of axis. You can select to **Skip Errors**, which will ignore any invalid lines and import only valid data, or **Cancel Import**, which will close the dialog without importing any data. You can also exit this dialog by using the **X** in upper right corner, or the <ESC> key.

File Format for Servo Data File

You can create your own data file for importing using Excel or any text editor. However, there are some important rules to follow when developing this .csv file. You might want to consider exporting data first which will give you the full names of the identifiers with all the proper spacing, and the needed id and version information in your file. You can then edit the data elsewhere, and import it.

- The first line of the file must contain the following id and version: PiCPro Servo Data, V2.0 OR you can use PSD, V2.0. The file cannot be imported by PiCPro without this line. When exporting a file, PiCPro will add the version of PiCPro used to export the data to the end of this line.
- Each line is made up of 2 fields separated by a comma: identifier,value. The first specifies the data field, the second the value for that field. Any or all identifiers can be used. If an identifier isn't used, no change will be made to that axis data. If multiple entries for the same identifier are encountered, only the last one read will be used.
- Each line ends in a carriage return, line feed.
- To include a comment in the file, use // as a comment delimiter. Any information following this delimiter will be ignored.
- The data field identifiers can be entered using either the complete or abbreviated form and are not case sensitive.

Data Field Identifiers

Shown in the first column is the complete identifier. The second column shows the abbreviated identifier. The third column shows the valid range of values. The final column lists the axes type for which the data field is NOT valid. Only valid data fields can be imported.

Complete Data Field Identifier	Abbrev. Identifier	Value Range	NOT valid for Axes Type
Feedback Units	FU	1 – 4294967295	
Ladder Units	LU	1 – 4294967295	
Ladder Units/Axis Units	LU2AU	1, 10, 100, 1000, 100000, 1000000	
Commanded Voltage (mV)	CV	1 – 10000	Stepper, SERCOS, Digitizing, Time
Motor RPM at Voltage	RPM	1 – 10000	Stepper, SERCOS, Digitizing, Time
Counts/Motor Revolution	CPMR	1 – 100000	Stepper, SERCOS, Digitizing, Time
Velocity Limit (AU/min)	VL	0.0 – 4294967295.0	Digitizing, Time
Acceleration Ramp (AU/min/sec)	AR	0.0 – 3839941406.0	Digitizing, Time
Deceleration Ramp (AU/min/sec)	DR	0.0 – 3839941406.0	Digitizing, Time
Controlled Stop Ramp (AU/min/sec)	CSR	0.0 – 3839941406.0	Digitizing, Time
Slow Velocity Filter (ms)	SVF	0 – 10000	Digitizing, Time
Fast Velocity Filter (ms)	FVF	0 – 10000	Digitizing, Time
Slow/Fast Velocity Threshold (AU/min)	SFVT	0.0 – 4294967295.0	Digitizing, Time
Rollover Position (AU)	RP	-2147483648.0 – 2147483647.0	
Ignore Limits?	IL	Yes, No, Until Referenced, Y, N, U	Digitizing, Time
Software Limit (Upper) (AU)	SLU	-2147483648.0 – 2147483647.0	Digitizing, Time

Software Limit (Lower) (AU)	SLL	-2147483648.0 – 2147483647.0	Digitizing, Time
Input Polarity	IP	Positive, Negative, P, N	Stepper, SERCOS, Time
Output Polarity	OP	Positive, Negative P, N	Stepper, SERCOS, Digitizing, Time
Analog Output Offset	AOO	-10000 – 10000	Stepper, SERCOS, Digitizing, Time
Feed Forward Percent	FFP	0 – 100	Stepper, SERCOS, Digitizing, Time
Proportional Gain (AU/min/AUFE)	PG	0 – 20000	Stepper, SERCOS, Digitizing, Time
Integral Gain (AU/min/AUFE x min)	IG	0 – 32000	Stepper, SERCOS, Digitizing, Time
(+) Integral Error Limit (AUFE x update)	PIEL	0.0 – 2147483647.0	Stepper, SERCOS, Digitizing, Time
(-) Integral Error Limit	NIEL	-2147483648.0 – 0.0	Stepper, SERCOS, Digitizing, Time
Derivative Gain (AU/min/AUFE/min)	DG	0 – 1000	Stepper, SERCOS, Digitizing
Following Error Limit (AU)	FEL	0.0 – 4294967295.0	Stepper, Digitizing, Time
Update Rate (ms)	UR	16*, 8, 4, 2, 1, .5, .25	Time
In Position Band (AU)	IPB	0.0 – 4294967295.0	Stepper, Digitizing, Time
Move Accel/Decel Max Acceleration (AU/min/sec)	MACCEL	1 - 6.291455813e13	Digitizing
Move Accel/Decel Constant Jerk (AU/min/sec/sec)	MJERK	1 - 7.864319766e15	Digitizing

Controlled Stop Decel Max Acceleration (AU/min/sec)	CACCEL	1 - 6.291455813e13	Digitizing
Controlled Stop Decel Constant Jerk (AU/min/sec/sec)	CJERK	1 - 7.864319766e15	Digitizing
S-Curve Enabled?	SCRVEN	No, Yes, N, Y	Digitizing
Resumable E-Stop Allowed	REA	No, Yes, N, Y	Digitizing

***Note:** 16 is only valid for Digitizing Axes

Copying Axis Information

Once you have entered an axis in servo setup and entered all the axis data for that axis, you can use the copy/paste commands to add additional axes with the same information.

How to copy and paste an entire axis

1. Select the axis you want to copy. Choose the **C**opy command from the **E**dit menu or press <Ctrl + C> or right click and choose **C**opy.
2. To insert a new axis place the focus on the **EOF** line and choose the **P**aste command from the **E**dit menu or press <Ctrl + V> or right click and choose **P**aste. If there is already an axis with the same number as the one you are pasting, you will be advised that the axis will be pasted with the first available, valid axis number. You can confirm or cancel the procedure.
3. The **Axis Properties** dialog will appear, with the prompt in the **Axis Label:** box. You must give the axis a new label as there cannot be duplicates in the .srv file. You can also make any other changes to the axis properties or axis data that you wish.

How to copy axis data from one axis to another

From the servo setup screen you can also copy Position Loop, Iterator, and Scaling data for one axis into the axis data for another axis. This is a time-saver if the axis data parameters for both axes are similar. If some of the axis parameters for the second axis need to be changed, you can edit them after you have copied the axis data to the second axis.

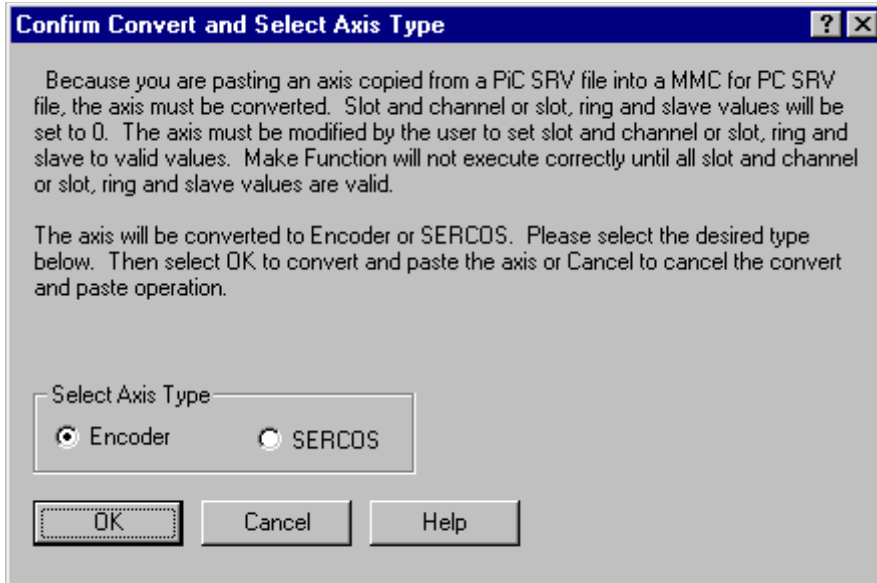
1. Highlight the axis with the data you want to copy, choose the **C**opy command from the **E**dit menu or press <Ctrl + C> or right click and choose **C**opy.
2. Highlight the axis you want to copy the axis data to and choose the **P**aste command from the **E**dit menu or press <Ctrl + V> or right click and choose **P**aste.
3. If you need to change some of the parameters for the second axis, choose **A**xis **D**ata from the **E**dit menu or press <Alt + Enter> to bring up the axis data box.

Note: When pasting axes, the servo view list and servo force list information will not be pasted. That information must be manually entered.

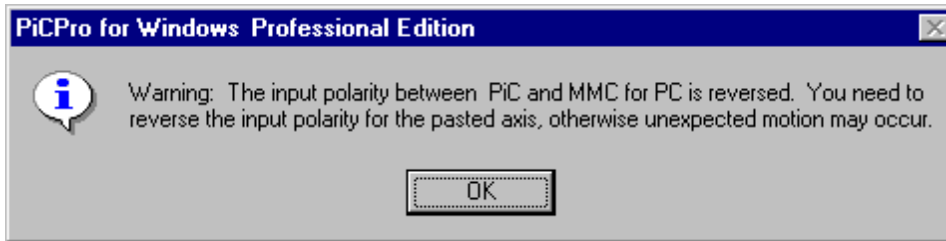
Pasting between .srv files with different CPU types

When copying from and pasting to files of different CPU types, you will be prompted that the slot, channel, or slot/ring/slave values will be set to zero. You must edit the axis to enter valid values.

If you are pasting into a standalone MMC or an MMC for PC file, you will also be asked to select the axis type, either D/A Encoder or SERCOS.



When pasting between files with different CPU types, you will receive a warning to check the input polarity. The input polarity of a PiC CPU is reversed from that of the standalone MMC or MMC for PC. You must edit the axis data to avoid unexpected motion when the ladder is run.



Saving a Servo Setup File

Save your file at regular intervals as you work on it.

To save a new file

1. Choose **F**ile|**S**ave.
2. In the **Save As** box that appears, choose a drive and folder where you want to save your setup file.
3. Enter a name in the **File name:** box. **Note:** It is strongly recommended that the filename use the DOS 8.3 naming convention of 8 characters, a period, and 3 characters.
4. Click **S**ave.

To save an existing file

1. Choose **F**ile | **S**ave.

OR

1. You can use the **F**ile | **S**ave **A**s command to change the name and/or location of a file. By giving the file a different name when you save it, you create a copy of the existing file while keeping the original intact.
2. Select the location for the file in the **Save in:** box.
3. Enter the new file name in the **File name:** box.
4. Click **S**ave.

To save a file in a format readable by a previous version of PiCPro

Servo Setup files that are created or modified in V13.0 or later cannot be opened directly by earlier versions of PiCPro. To save an SRV file for use with an earlier version of PiCPro for Windows do the following:

1. Choose **F**ile | **S**ave **A**s from the menu.
2. In the **Save as type:** list, select the desired type.
3. Click **S**ave. Any errors will be displayed in the Information Window.

Printing Axis Information

You can print the axis information (scaling, iterator, and position loop information) for one or all axes in your servo setup program.

To print axis information

1. Open the Servo Setup file (.srv) that you want to print. If you just want to print the axis information for one axis, select that axis.
2. Choose **F**ile | **P**rint from the menu or press <Ctrl + P>. The print dialog box appears and you can click **OK**. If you want to print the axis data for all axes in your setup file, choose **A**ll in the print dialog box and click **OK**.

Making a Servo Setup Function

After you have entered all the setup information for all the axes in your application, you will create a servo setup function to store all this information. This function will be stored in the servo library you designate and can then be called in your ladder program to initialize the setup data for your application. The axes setup information will then be sent to the control when you download your application.

The name of the function will be the same as the name of the .srv file name. It will appear in the list of functions in PiCPro.\

Note: Function names are limited to 8 characters.

To create a servo setup function

1. After all the setup information for all your axes has been entered, choose **C**ompile | **M**ake Function from the menu.
2. The **Save As** box will appear if you have not previously saved your setup file. Choose the location you want to save your .SRV file to. There is a default that is entered in the **File name** box beginning with Servo1.SRV. You can accept this name or enter your own.

Note: The second setup file you create will have the default name Servo2.SRV, the third will be Servo3.SRV and so on.

Note: It is strongly recommended that the filename use the DOS 8.3 naming convention of 8 characters, a period, and 3 characters.

3. The **Compile SRV** box appears. The **Registered ServoSetup/SercosSetup Libraries:** section will hold the names of any setup libraries you have created previously. You may choose from this list or create a new Servo Setup Library by entering one in the **Library Name:** box.

Note: If you are recompiling a function that was created earlier, you must place the function in the same library where it was originally located. You cannot select a new library location. The **Library Name:** box will be disabled (grayed out).

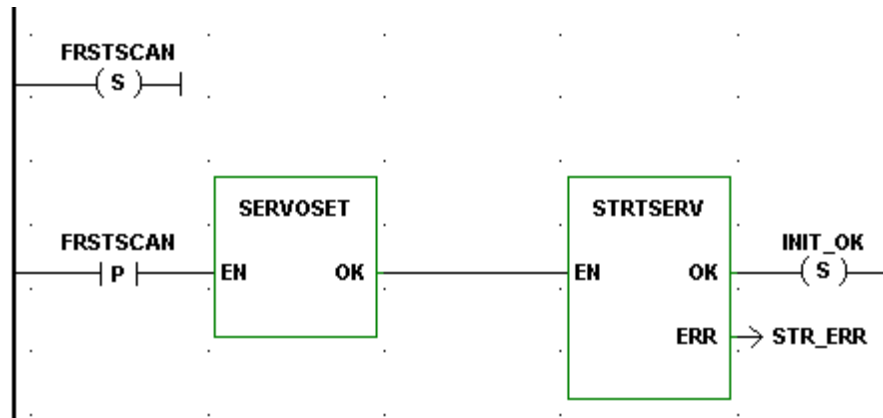
- Click **OK** to insert the servo setup function into the chosen library. If you are creating a new library a prompt will appear asking if you want to create a new library. If the directory path specified for the new library is not found in the current library path, a prompt will be displayed asking if this path should be added.

Choosing **Yes** will define the library path so PiCPro can locate the function. The library and setup function will immediately show up in PiCPro under **Ladder | Functions** and in the Functions list on the toolbar

- Do a full compile and download of the ladder using this function.

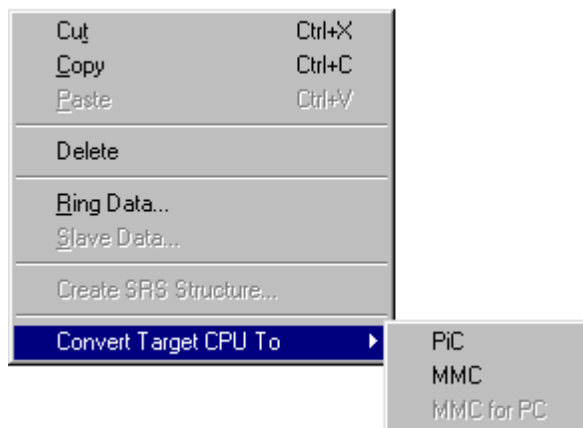
To initialize setup data in your ladder

The servo setup function you compile with the Servo Setup program must be incorporated into your ladder program. To initialize all the setup data you use the standard motion function STRTSERV with your setup function. Below is a network designed to do this. The example setup function is labeled SERVOSET.

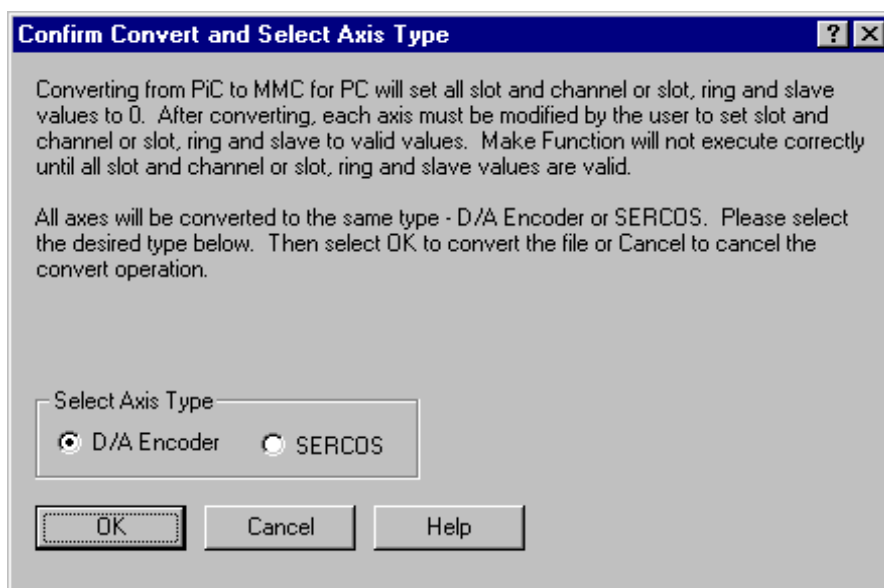


Converting a Servo Setup File CPU Type

1. Choose **Edit | Convert Target CPU To**.



2. A confirmation prompt will be displayed that indicates the types of changes that will be made to the servo setup information for this file. You will need to manually enter the slot, channel, and slot/ring/slave values.



Note: This function is not available in PiCPro Standalone MMC Edition.

Axis Tuning

Axis tuning allows you to make adjustments to offsets and gains for an axis through software rather than hardware. When the ladder containing your servo setup file is downloaded and scanning in the CPU, any servo axis defined in your servo setup file can be tuned using the Servo Force List (Write) and the Servo View List (Read) in Servo Setup. When you change a value through the Servo Force List, a prompt will appear when you close the force list asking if you want to update the axis data for the selected axis. If you answer Yes, the data will become a permanent part of your program.

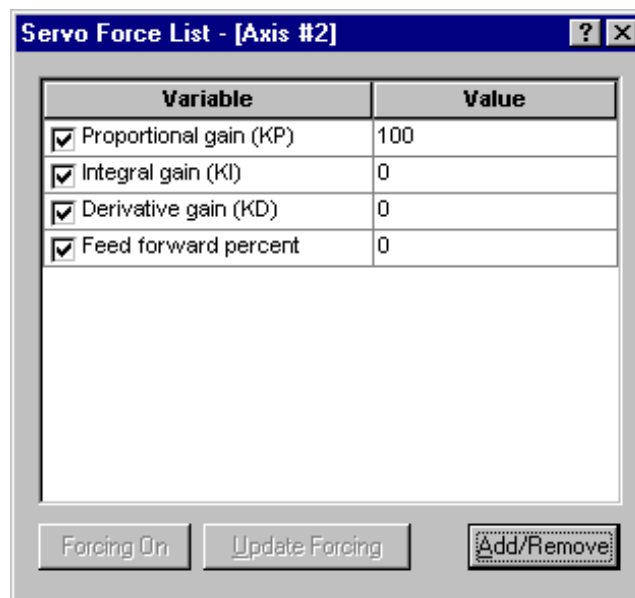
Note: In order to use Servo View and Servo Force, the latest version of the motion library must be installed. This occurs automatically if you choose Typical as an installation choice.

Axis Tuning using Forcing

You can write values into certain servo variables for an axis running in the CPU using the Servo Force List.

To force an axis servo variable in the Servo Force list

1. Open the ladder containing your servo function. Choose **View | Servo Function** from the menu or select the servo setup function, right click the mouse, and choose **View Servo Function** from the list.
2. Select the axis you want to force variable values on.
3. Choose **View | Servo Force List** from the menu bar or right click and choose **Servo Force List**. A Servo Force List can be opened for each servo axis you have defined in Servo Setup. A Servo Force List cannot be opened for a digitizing axis. The Axis number appears in the title bar of each list.
4. Use the **Add/Remove** button to edit the forcing list.
5. Click in the check box of the variables you want to force.
6. Enter a value for the variable in the **Value** column. A message will appear if the value is out of range for the variable.
7. Click **Update Forcing** to send the updated values to the CPU.

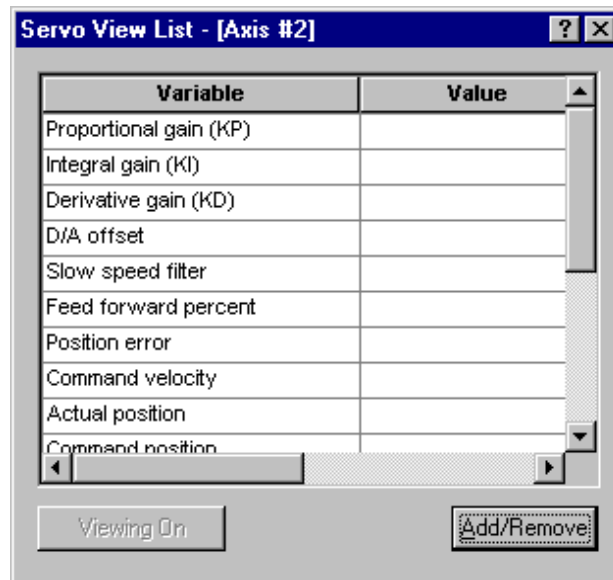


Axis Tuning using Viewing

You can read values from certain servo variables for an axis running in the CPU using the Servo View List.

To view an axis servo variable in the Servo View list

1. Open the ladder containing your servo function. Choose **View | Servo Function** from the menu or select the servo setup function, right click the mouse, and choose **View Servo Function** from the list.
2. Select the axis you want to view variable values on.
3. Choose **View | Servo View List** from the menu or right click and choose **Servo View List**. A Servo View List can be opened for each axis you have defined in Servo Setup. The Axis number appears in the title bar of each list.
4. Use the **Add/Remove** button to edit your view list.



Axis Tuning Variables

The following servo variables can be added to or removed from the **Servo Force List**:

- Proportional Gain
- Integral Gain
- Derivative Gain
- D/A Offset
- Slow Speed Filter
- Feed Forward Percent

Note: The Servo Force List variables can also be written to and read from your ladder using the TUNERead and TUNEWRITE functions.

In addition to the servo variables listed above, the **Servo View List** can display the following servo variables:

- Position Error
- Command Velocity
- Actual Position
- Command Position
- D/A Output Voltage
- Move Type
- Next Move
- Emergency Errors
- Controlled Stop Errors
- Programming Errors
- Loop State
- Resume Distance
- Resume Mode

For a digitizing axis, Actual Position and Emergency Errors are the only variables that can be displayed in the view list. The force list is not available for a digitizing axis, or a time axis.

NOTES

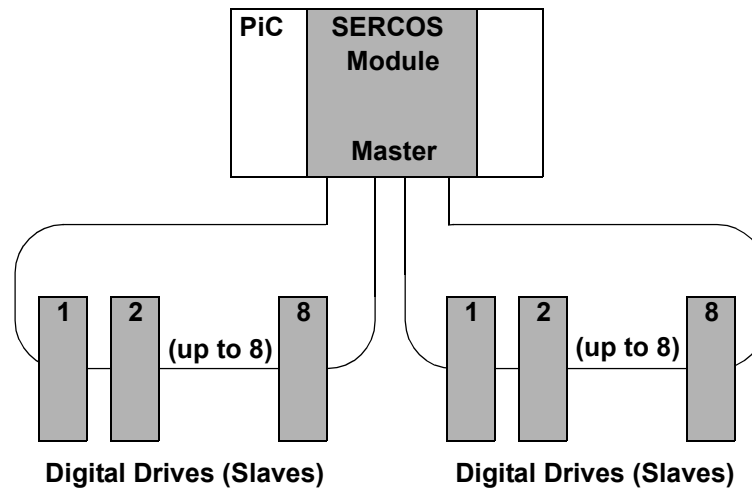
CHAPTER 5 PiCPro and SERCOS

SERCOS Setup Background Information

SERCOS is a Serial, Realtime, Communication System developed to interface with digital controls and drives. Fiber optic cables are used to transmit data serially with noise immunity and fast update times. SERCOS allows motion control in velocity, torque, or position modes.

Giddings & Lewis recommends that you acquire a copy of the NEMA SERCOS Specification for reference information, especially if you will be performing advanced control techniques. The SERCOS Specification is available through ANSI. To order, call ANSI at 212 642-4900 and ask for Manual IEC 61491. The web page for SERCOS North American user's group is www.sercos.com.

SERCOS allows you to create a digital instead of an analog motion control network. A typical PiC/SERCOS network is illustrated below. The network consists of up to eight digital drives connected to one master in a ring topology in which messages travel unidirectionally. The network can support two fiber optic rings from one SERCOS module.



Typically, the SERCOS drive accepts position commands from the PiC whereas an analog drive accepts velocity commands. In SERCOS, the position loop which compares the commanded position to the actual position and then applies a PID algorithm to the difference is performed in the drive rather than in the PiC.

SERCOS and the PiC work together to provide a mechanism to control the drive through communications. The following are all part of how communications is established between the PiC master and the drive slaves.

- Each drive identified with a unique number
- Five communication phases (0-4) that must be passed through before operation begins
- The AT and MDT telegrams
- The cyclic data containing IDN numbers
- The service channel containing data

The following are all part of how control is performed once communications is established.

- The value of the IDN in cyclic data
- The value of the data transferred over the service channel
- The mode defined by the status word
- The state of the control and status words

Note: Currently, The PiC/SERCOS and standalone MMC/SERCOS configuration supports up to eight slaves on the ring each slave controlling a single drive. MMC for PC/SERCOS supports up to thirty-two slaves on a ring. None of the configurations support a slave controlling a group of drives as described in the SERCOS Specification.

You define the SERCOS configuration for all the SERCOS axes using Sercos-Setup in PiCPro.

SERCOS Functions

The Motion library contains the following function/function blocks in the groups listed. Refer to the Function/Function Block Reference Guide for a complete description of the function/function blocks.

SCA_... function blocks have an AXIS input to identify the SERCOS axis that has been initialized in Servo setup.

SCS_... function/function blocks identify the SERCOS axis by slot, ring, and slave number.

SCR_... function/function blocks apply to one ring.

SC_... function applies to all rings.

DATA Group

SCA_CTRL	Writes control bits to the MDT for a servo axis
SCA_RCYC	Reads cyclic data from the AT for a servo axis
SCA_RECV	Receives data from the service channel of the AT for a servo axis
SCA_SEND	Sends data to the service channel of the MDT for a servo axis
SCA_STAT	Reports the status from the AT for a servo axis
SCA_WCYC	Writes cyclic data to the MDT for a servo axis

ERRORS Group

SCA_ERST	Resets E-errors and closes the loop on a servo SERCOS axis
----------	--

INIT Group

SCA_CLOS	Reads the current position and closes the loop on a servo SERCOS axis
----------	---

MOVE_SUP Group

SCA_PBIT	Initializes the SERCOS fast input (probe input)
----------	---

REF Group

SCA_ACKR	Acknowledges the reference cycle for a servo axis
SCA_REF	Starts the reference cycle on the servo SERCOS axis
SCA_RFIT	Initializes the reference cycle on the servo SERCOS axis

SERC_SLV Group

SCS_ACKR	Acknowledges the reference cycle for a non-servo axis
SCS_CTRL	Writes the control bits to the MDT
SCS_RECV	Receives data from the service channel of the AT for a non-servo axis
SCS_REF	Starts the reference cycle on the non-servo SERCOS axis
SCS_SEND	Sends data to the service channel of the MDT
SCS_STAT	Reports the status from the AT

SERC_SYS Group

SCR_CONT	Continues communication if it was halted after Phase 2
SCR_ERR	Reports ring errors
SCR_PHAS	Reports the highest phase number completed
SC_INIT	Copies the user data into the SERCOS module

Using Standard Motion Functions and Variables

There are certain things you should be aware of when using SERCOS feedback with the motion library of PiCPro.

READ_SV/WRITE_SV Functions

These READ_SV and WRITE_SV variables cannot be used when using SERCOS feedback on an axis.

Var #	Name
27	Backlash compensation
28	TTL feedback
46	Set user PID command
47	User PID command

All other READ_SV/WRITE_SV variables can be used with an axis using SERCOS feedback.

TUNERead/TUNEWRIT Functions

The filter variable is the only one that can be read and written by these functions when using a SERCOS axis. The remaining TUNERead/TUNEWRIT variables cannot be used with a SERCOS axis.

CLOSLOOP/OPENLOOP Functions

The CLOSLOOP function is not called on a SERCOS axis. The SCA_CLOS function block is used to close the loop on the SERCOS axis. The top three bits of the SERCOS control word are set on subsequent SERCOS updates. When the OPENLOOP function is called, the top three bits of the SERCOS control word are cleared on subsequent SERCOS updates.

E_RESET Function

The E_RESET function is not called on a SERCOS axis. The SCA_ERST is used to reset E-stop conditions on a SERCOS axis. The SCA_ERST function will read the current position of the drive prior to closing the position loop.

STRTSERV Function

The SERCOS ring must be into Phase 4 of its communication cycle before the servos can be initialized. Call STRTSERV after Phase 4 has started.

An additional error can appear at the ERR output with a SERCOS axis;

Error # 5	The axis update rate in PiCPro is different from the SERCOS ring update rate.
-----------	---

Using SERCOS Functions/Blocks with TASKS

Some of the SERCOS functions/function blocks can be used in TASK LDOs as well as in Main LDOs.

For use in Main LDOs only

SC_INIT
SCA_RECV
SCA_SEND
SCA_CLOS
SCA_ACKR
SCA_REF
SCA_ERST
SCS_SEND
SCS_RECV
SCS_ACKR
SCS_REF

For use in TASK or Main LDOs

SCR_CONT
SCR_ERR
SCR_PHAS
SCS_CTRL
SCS_STAT
SCA_CTRL
SCA_RCYC
SCA_WCYC

SERCOS Telegrams

SERCOS has a master/slave communications structure with the PiC as the master and the drives as the slaves. All the communication between the two is done with messages referred to as telegrams. A telegram is an ordered bit stream carrying data and timing information.

There are three standard telegrams used in SERCOS communication as described below:

1. **MST - Master Synchronization Telegram** - sent by the PiC to the drives to set the timing and synchronization for everything that happens on the network.
2. **AT - Amplifier Telegram** - sent by the drives to the PiC.
3. **MDT - Master Data Telegram** - sent by the PiC to the drives after it receives the last AT in a cycle.

The type of data transferred by these telegrams is determined by the telegram type you enter in SERCOS setup. IDN 15 contains the telegram types available. They are described below.

IDN 15 (Telegram Type)	Description
0	No data
1	Torque control
2	Velocity control, velocity feedback
3	Velocity control, position feedback
4*	Position control, position feedback
5	Position and velocity control, position and velocity feedback
6	Velocity control
7	User-defined

*If the slave will be controlled by Motion.Lib, 4 or 7 is the telegram type number that is entered in SERCOS setup and the primary mode (defined by IDN 32) must be set to the position mode. If telegram 7 is selected IDN 47 must be part of the MDT and IDN 51 must be part of the AT.

IDNs

SERCOS has a system of identification numbers (IDNs) used to access specific data. You can find a list of these IDNs in the SERCOS specification. There is a standardized set covering the following categories:

1. General parameters
2. Diagnostics
3. Lists of operation data
4. Internal and external signals
5. Manufacturer's specifications
6. Definitions of telegram contents
7. Drive operation modes
8. Standard operation data
9. Drive parameters

There is also a drive manufacturer-defined set of IDNs available.

Below is a list of IDNs that must be supported by any slave device you are using. They are listed in the order they are worked through as Phase 2 and 3 proceed. For complete descriptions of these IDNs, consult the SERCOS specification.

	IDN	
	READ	Description
Phase 2	00003	Shortest AT Transmission Starting Time (T1min)
	00004	Transmit/Receive Transmission Time (TATMT)
	00005	Minimum Feedback Processing Time (T4TMT)
	00088	Receive to Receive Recovery Time (TMTSY)
	00090	Command Value Proceeding Time (TMTSG)
	00096	Slave Arrangement 0
	WRITE	Description
	00001	Control Unit Cycle Time, Tncycd
	00002	Communication Cycle Time, Tscyc
	00006	AT Transmission Starting Time (T1)
	00007	Feedback Acquisition Capture
	00009	Position of Data Record in MDT
	00010	Length of Master Data Telegram
	00015	Telegram Type Parameter
	00032	Primary Operation Mode
	00089	MDT Transmission Starting Time
	R/W	Description
	00099	Reset Class 1 Diagnostic
	00127	CP3 Transition Check
Phase 3	00128	CP4 Transition Check

Working with Procedure Command Function IDNs

There are two types of IDNs used in SERCOS.

1. Parameter Data IDNs
These handle simple commands (i.e., IDN 100 where the value for gain can be read or written). You define the values for these IDNs.
2. Procedure Command Function (PCF) IDNs
These are commands that take a longer time to execute (i.e., IDN 262, load parameter data or homing cycle). With PCFs, the one word DATA values hold bit patterns that relay the progress of the function. The MDT has two bits and the AT has five bits. All PCFs use these bits the same way. This bit field provides the communication between the master and the slave.

Follow these steps to use a PCF IDN from your ladder.

1. Start the procedure by sending the PCF IDN with the data value of 3 using the SCS_SEND or SCA_SEND functions. This sets (bit 0 = 1) and enables (bit 1 = 1) the PCF. SIZE for PCF IDNs is one word.
2. In the SCS_RECV or SCA_RECV function blocks, enter a 1 as the ELEM member of the DATA structure.
3. Continually read the PCF IDN value using SCS_RECV or SCA_RECV function blocks. If the command was received by the drive, a status of 7 will be read. This means the drive acknowledges the command set (bit 0 = 1) and enabled (bit 1 = 1). If no error occurs and the procedure is complete, a status of 3 is returned. This means the drive acknowledges the command set (bit 0 = 1) enabled (bit 1 = 1), and executed (bit 2 = 0). This may take only a few SERCOS updates or several minutes depending on the PCF. If an error occurred during the procedure, status bit 3 will equal 1.
Note: There is a change bit (bit 5) in the status word which can be used to avoid continually reading the PCF IDN value as described above. See the SERCOS specification for more information on this change bit.
4. After the read value of the PCF IDN equals status of 3, the command set (bit 0 = 1), the enabled (bit 1 = 1), and executed (bit 2 = 0), or the read value has status bit 3 set (≥ 8), the PCF must be sent again using the SCS_SEND or SCA_SEND functions with data = 0. This cancels the procedure by sending the PCF IDN with bit 0 = 0.

This procedure and other advanced SERCOS features are described in the SERCOS specification. Refer to section 7.4.4 of the NEMA SERCOS specification for more detailed information on PCFs.

Note: PCFs may not be used in the SERCOS setup list.

You can also work with PCFs using the **Online | Execute Procedure CMD** within SERCOS setup.

SERCOS Phases

During initialization, the SERCOS system moves through five communication phases (0 through 4) before normal communication can begin. These are summarized below.

Phase #	Description
Phase 0	Closes the communication ring and allows initialization to proceed.
Phase 1	An identification phase where the PiC addresses each drive individually and waits for an acknowledgment.
Phase 2	Sets parameters for cyclic data transmission.
Phase 3	Sets parameters for non-cyclic or service channel data transmission.
Phase 4	The SERCOS ring can begin normal operation as defined by the application.

The movement through these five communication phases occurs in sequential order. In SERCOS setup, it is possible to decide to halt the movement after the completion of Phase 2 allowing you to send additional IDNs specific to your application which you did not enter in SERCOS setup and then resume moving through Phase 4. At the end of Phase 4, you have another opportunity to send IDNs.

Cyclic Operation

After initialization is complete, SERCOS transmits critical data such as command values, feedback values, etc. in synchronized, cyclic form. This data is updated cyclically in real time at the update rate set in the PiC. You define the cyclic data by choosing the **Define Cyclic Data** button within the **Insert Slave** box.

Service Channel

Non-critical data such as diagnostics, loop gains, etc. are transferred over a service channel. This data is transferred once each cycle and goes both ways. It may take several cycles to transfer data. The transfer is initiated by the PiC. Both operations within your ladder and within SERCOS setup make use of the service channel.

Comparing Cyclic Data and Service Channel Transfers

Cyclic Data

- Words in the telegram (AT/MDT) are devoted to each piece of cyclic data to be transferred.
- Data included in the cyclic data causes the telegram to be larger. Consequently, more communication time is used.
- Cyclic data is sent every time a communication cycle occurs on the ring.
- You define cyclic data in SERCOS setup and you cannot change this after the ring has completed phase 4.
Note: Once communications is established, you can access the cyclic data with the SCA_RCYC and SCA_WCYC functions.
- Cyclic data can only be sent after phase 4 is completed.
- Only the operation data of an IDN can be sent as cyclic data.

Service Channel Data

- One word in the telegram (AT/MDT) is shared by all service channel requests. The content of the word changes with each service channel request.
- Service channel requests have no effect on the length of the telegrams. One word in the telegram is devoted to the service channel whether there are any requests or not.
- Because only one word of the service channel data is sent each communication update, several updates are required to send or receive data over the service channel.
- You make service channel requests from the ladder or from SERCOS setup after communications is established. The requests are queued up on the SERCOS module and are completed in order.
Note: The ladder has priority over SERCOS setup.
- Service channel data can be sent after the completion of phases 2, 3, or 4.
- Any element (name, attribute, units, minimum value, maximum value, and operation data) of an IDN can be sent over the service channel.

Service Channel Background Information

For a complete description of the activity on the service channel, read the Non-Cyclic Data Exchange section of the SERCOS specification.

There is one word in the AT (from drive to PiC) and one word in the MDT (from PiC to drive) that is dedicated to service channel communication. One word of the service channel data is transmitted each way between the PiC and the drive during each update. Both the LDO and SERCOS setup use the service channel.

From the LDO, the following functions request information over the service channel:

SCA_RECV SCA_SEND SCA_REF SCA_CLOS
SCS_RECV SCS_SEND SCS_REF

From SERCOS setup, the following online commands request information over the service channel:

Execute	Receive IDN	Receive	Send IDN	Upload Drive
Procedure	CMD	Continuous		Information

Keep the following in mind:

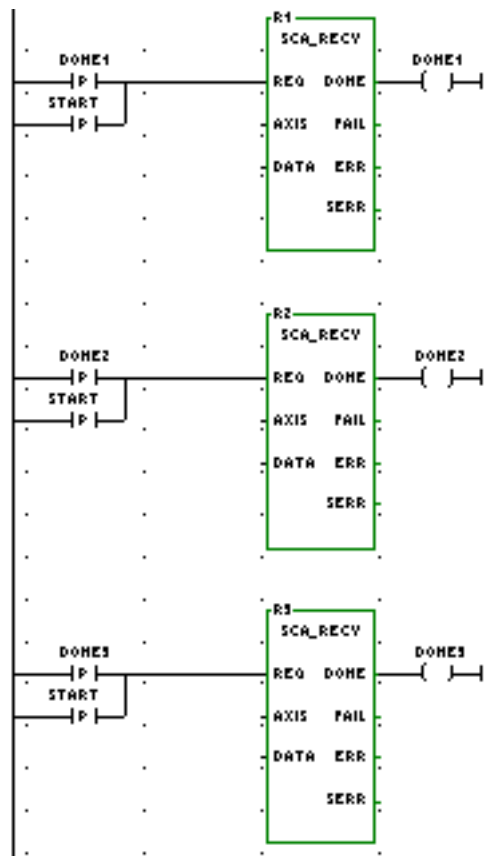
- The LDO requests are used for machine control.
- SERCOS setup requests are used for diagnosis.
- Only one request is processed on the ring at a time. It must be completed before the next one begins.
- The SERCOS module allows the LDO to queue up to 16 requests to the service channel.
- The LDO requests in the queue have priority over any requests coming from SERCOS setup.
- All LDO queue requests (1 to 16) will be processed before any requests from SERCOS setup.
- Resources are consumed in the transfer of information.
- The throughput of the service channel for a slave is also affected by the service channel activity of other slaves on the ring.

Examples of LDO Design to Access the Service Channel

There are many possible ways to design your LDO to access the service channel and base ladder logic on the results. In the examples presented here, partial logic is used to illustrate three approaches to monitoring the data of three IDNs in the LDO.

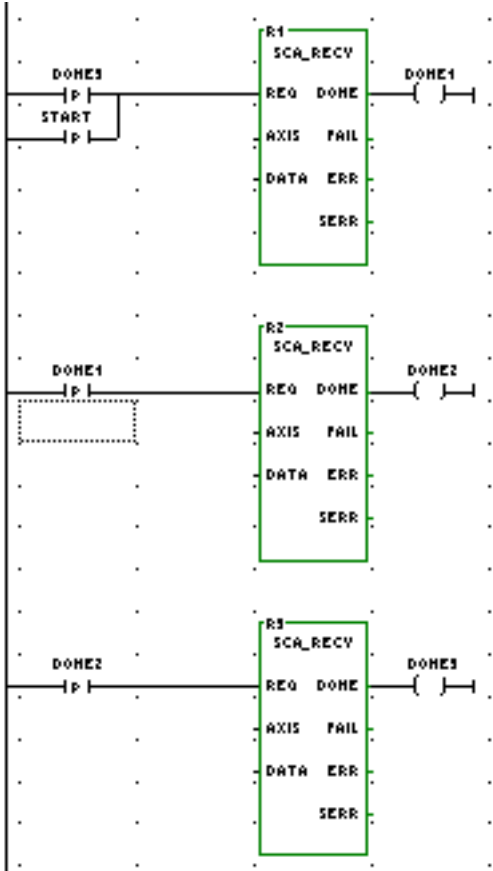
Example 1 Logic that gives priority to the LDO requests

In this example, you want to read IDN 1, read IDN 2, read IDN 3, then read IDN 1 when IDN 1 is complete, read IDN2 when IDN2 is complete, and read IDN3 when IDN3 is complete and repeat. This example uses three queues because all three are queued when START transitions. Any SERCOS setup requests will be processed after the three queues have finished.



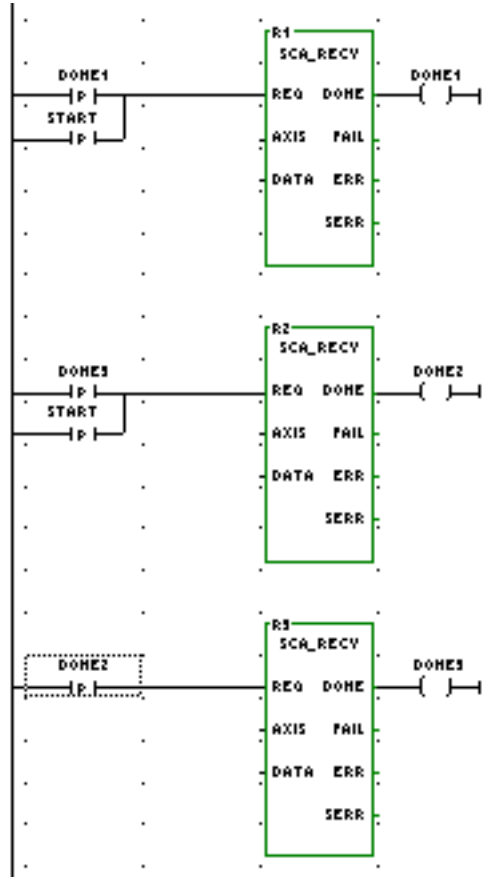
Example 2 Logic that allows SERCOS Setup commands to be read sooner

In this example, you want to read IDN1. Read IDN 2 when IDN 1 is complete. Read IDN 3 when IDN 2 is complete. Read IDN 1 when IDN 3 is complete and repeat. Using the logic shown below, only one queue is used because each request is queued after the previous one is complete. This allows requests from SERCOS setup to be processed after each function block.



Example 3 Logic that allows one IDN to be read more frequently.

In this example, you want to read IDN1 more frequently than IDN 2 or 3. Read IDN1. Read IDN 2 when IDN 1 is complete. Read IDN 1 again. Read IDN 3 when IDN 2 is complete and repeat. This example uses two queues: one for IDN1 and one shared by IDN 2 and IDN 3.



SERCOS Setup Overview

Two approaches to SERCOS startup are presented next. In the first situation, it is assumed that you are simply replacing your analog drives with SERCOS drives and not attempting to use the advanced features available. In the second situation, you are planning to use the advanced features available with SERCOS.

Replacing Your Analog System with SERCOS

When you are replacing your analog system with a SERCOS system, the analog output module and the feedback module are both replaced by the SERCOS communication module in the PiC. The fiber optic cable will replace the analog voltage wires and the feedback wires. The drive will connect to the feedback device and send that information back to the PiC over the SERCOS cable.

An overview of the procedure to follow if you simply want to use SERCOS without any advanced features is summarized here.

In SERCOS Setup:

1. Enter a ring for each SERCOS ring.
2. For each ring, set the update rate, the baud rate to the rate of the drive, and No pause after phase 2.
3. Enter a slave for each drive on each ring.
4. Leave the setup list empty.
5. Define the cyclic data to telegram 4 for all drives.
6. Define the operation mode for all slaves to operate in position control, with Following Distance, Cyclic and System Interface.
7. Make a SERCOS setup function.

In Servo Setup, define each slave axis for basic control:

1. Select SERCOS as the Input and Output Type for the axis.
2. Make other selections for the servo axes for your application.
3. Make a servo function.

In PiCPro, program a ladder that will initialize the SERCOS slaves, then the servo axes.

1. Call the SERCOS function you've made and then the SC_INIT function.
2. Call SCR_PHAS to read the phase and logic to detect phase 4.
3. When phase 4 is complete, call the servo function you made, then the STRT-SERV function.

You can now control the axes using the same functions you used for analog servos (with some exceptions including referencing and tuning).

Using SERCOS Advanced Features

There are features offered by a SERCOS system that go beyond simply replacing the analog signal with a digital interface. In addition to position command and feedback, other information can be exchanged between the drive and the control or the PC workstation. In order to use these advanced features, it is necessary to upload the IDN list that resides in the drive. This list holds all the information that can be read/written from/to the drive. The list also indicates the units and range for each IDN in the list. With this information, SERCOS setup can read or write this information, define custom cyclic data, or change modes.

The LDO functions can use the advanced features to read and write data as well. It is not required that the IDN list be uploaded from the drive in order to program the LDO. But the contents of the drive IDN list provide the feature description for the drive. Knowing the drive features available will aid you in your LDO design.

An overview of the procedure to follow if you want to use SERCOS with advanced features is summarized here.

In SERCOS Setup:

1. Enter a ring for each SERCOS ring.
2. For each ring, set the update rate, the baud rate to the rate of the drive, and No pause after phase 2.
3. Enter a slave for each drive on each ring.
4. Leave the setup list empty.
5. Define the cyclic data to telegram 4 for all drives.
6. Define the operation mode for all slaves to operate in position control, with Following Distance, Cyclic and System Interface.
7. Make a SERCOS setup function.

In Servo Setup, define each slave axis:

1. Select SERCOS as the Input and Output Type for the axis.
2. Make other selections for the servo axes for your application.
3. Make a servo function.

In PiCPro, program a ladder that will initialize the SERCOS slaves, then the servo axes.

1. Call the SERCOS function you've made and then the SC_INIT function.
2. Call SCR_PHAS to read the phase and logic to detect phase 4.

When phase 4 is complete, read the IDN list from the drive in SERCOS setup.

1. Select **Online | Upload Drive Information** from the menu.
2. Enter a unique file name and click on **Upload now**. The status bar indicates progress. This can take several minutes per drive. Typically, you only need to upload this information once for each drive type.

3. The drive file (.csv) is a comma-separated variable file that can be read by a text editor or spreadsheet.

Once the IDN list is read from the drive and SERCOS is initialized from the ladder, SERCOS setup can be used for the following:

- Run the drive via the battery box feature.
- Read the value of IDNs either once or continuously.
- Write the value of IDNs.
- Execute procedure command functions
- Customize the cyclic data.
- Operate the drive in different modes: position, velocity, or torque.
- Use secondary feedback. Secondary feedback can be read by customizing the telegram and using the SCA_RCYC function in your ladder. Motion.lib does not read secondary feedback.

Axis Positioning and Referencing in SERCOS

You can control axis position using either servo or non-servo SERCOS function/function blocks. In SERCOS setup, the telegram type must be Telegram 4 position mode. As defined by the SERCOS specification, with Telegram 4 IDN 47 is cyclic data sent from the PiC to the slave. IDN 51 is cyclic data sent from the slave to the PiC. When using a servo axis, SERCOS is selected as the feedback type in Servo Setup.

With a Servo SERCOS Axis

The servo setup data is made into the User setup function in servo setup. The STRTSERV function installs all the data. Both functions are put in your ladder.

To close a SERCOS axis loop, use the SCA_CLOS function. It will read IDN 47, update the servo data with the new position, send the value as commanded position, read rollover on position information, and set the control bits to cause the drive to close it.


To reference the axis, the SCA_RFIT function block must be called and completed prior to calling FAST_REF or LAD_REF.

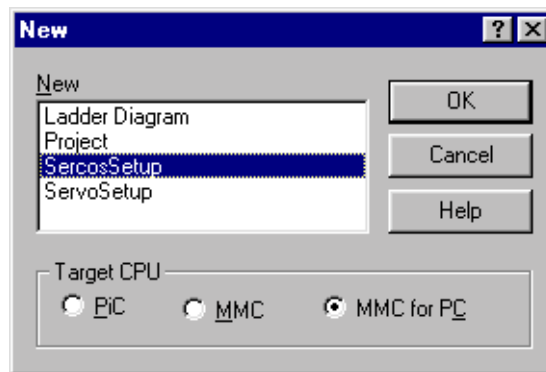
Another option is to reference the axis using the SERCOS drive's reference cycle via the SCA_REF function block. This function block reads the value for IDN 47 as the last step in the reference process and returns this value in the RSLT output. Use the SCA_ACKR function block to acknowledge that the reference cycle is complete. Control will return to the PiC after this function block is called.

Opening SERCOS Setup

With the SERCOS Setup program, you will create a .SRC file containing data for all the slaves in your SERCOS system. You convert this .SRC file to a function in order to send all the data to the CPU so that the SERCOS slaves can be initialized. There are several ways to access the SERCOS Setup program.

To create a new SERCOS setup file


1. Choose **File | New** from the menu or click on the  button on the standard toolbar.
2. Select **SercosSetup** from the box that appears



3. Select target CPU type (**PiC**, **MMC** or **MMC for PC**).
4. Choose **OK**.

Note: In the Standalone MMC Edition of PiCPro, Target CPU selection is not made. Target CPU is always standalone MMC.

To open an existing SERCOS setup file

1. Choose **File | Open** from the menu or click on the  button on the standard toolbar.
2. The **Open** dialog box appears. At the bottom, choose SERCOS Setup Files (*.src) from the drop down list in the **Files of type:** box. This brings up a list of all existing SERCOS setup files.
3. Highlight the file you want to open and click **Open** or press **<Enter>**.

Note: If you attempt to open a PiC or MMC for PC SERCOS setup file in the Standalone MMC Edition, you will be prompted to convert the file to a standalone MMC format. When opening an MMC for PC file in the standalone MMC Edition, and the baud rate of the ring is 8 or 16, the baud rate will be set to 4.

To open an existing SERCOS setup file from the setup function in a ladder

If you have already created a SERCOS setup function with your setup data and have included it in your ladder, you can access the SERCOS setup file from there.

1. Place focus on the setup function in your ladder
2. Right click your mouse. Choose **View SERCOS Function**.
or
3. Choose **View | SERCOS Function** from the menu and then select the desired setup function from the flyout list.

Note: In order to open a SERCOS setup file from within your ladder, the SERCOS file and the SERCOS function must have the same name. Any SERCOS setup file created with PiCPro for Windows automatically has the same name as the SERCOS setup function. However, any DOS SERCOS setup file may not have the same name as the SERCOS setup function. Change the name of the .src file to match the name of the SERCOS setup function if you want to be able to open the file from within the ladder.

To open an existing SERCOS setup file from Windows explorer

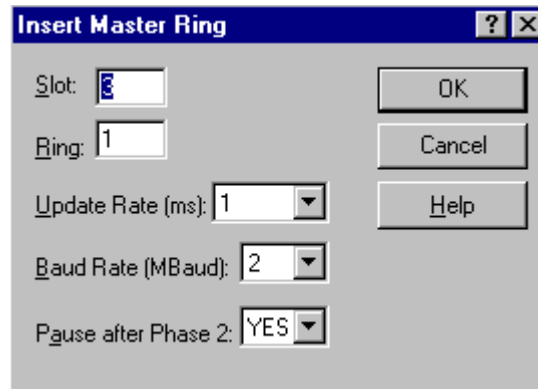
1. Open Windows explorer and find the .src file you want to open.
2. Double click on the .src file you want to open and the SERCOS Setup window will appear.

Inserting/Editing SERCOS Rings

A ring is added to SERCOS Setup by using the **I**nsert menu. Each time you insert a ring you must insert the slave axes you are connecting to that ring and any IDNs required.

Inserting/Editing a SERCOS ring

1. Choose **I**nsert | **R**ing from the menu. The **I**nsert Master Ring box appears.



2. Enter the slot number (3-13) identifying the slot in which your SERCOS module is installed in the PiC rack.
3. Enter the ring number (1-2).
Note: If a standalone MMC or an MMC for PC CPU is selected, only 1 ring can be entered. Slot and ring are limited to a value of 1.
4. The default update rate is 4 ms. Choose an **U**ppdate Rate from the drop down list (1, 2, 4, or 8) if you want to change this. The update rate for any ring cannot exceed the CPU update rate. The CPU update rate is the fastest of all servos initialized. If the SERCOS slave axis is a servo axis, the ring update rate must be the same as the axis update rate.
Note: If 1 ms is chosen as an update rate, only one ring with six slaves on that ring can be used regardless of CPU type. All other update rates allow two rings with eight slaves on each when used with a PiC CPU.
5. Choose the **B**aud Rate from the drop down list (2 or 4 for PiC and standalone MMC, and 2, 4, 8, or 16 for MMC for PC).
6. Choosing **Y**ES for **P**ause after Phase 2 allows you to send/receive additional IDNs via the ladder if required. Use the SCR_CONT function to continue on through phase 4.
7. Choose **O**K to accept your ring configuration.

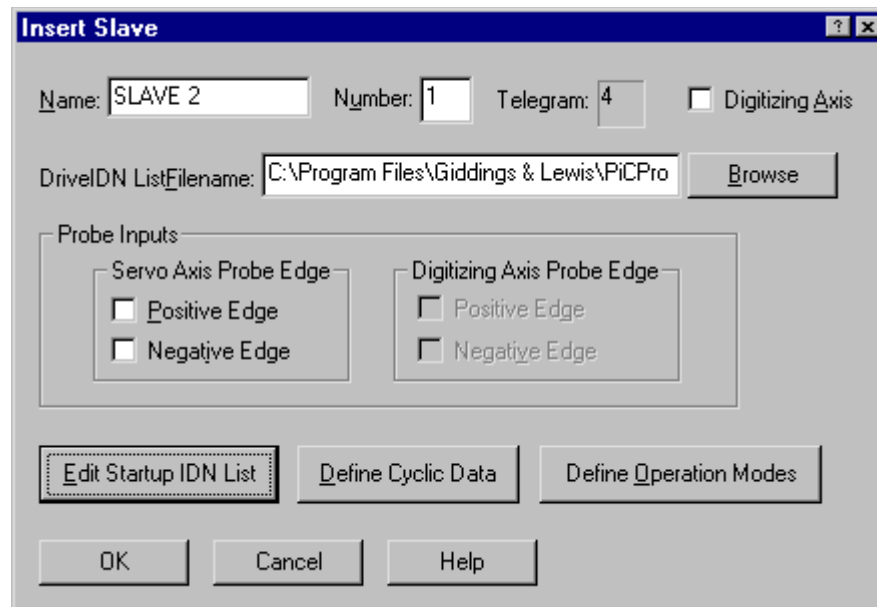
Once you have entered a ring in SERCOS setup, you can edit it by highlighting the line the ring is on and pressing <Enter> or going to the **E**dit menu and selecting **R**ing Data. This brings up the **E**dit Master Ring Data box.

To insert another ring, highlight an existing ring or the end of rings line and press <Insert>. The ring will be inserted above the highlighted line.

Inserting/Editing SERCOS Slaves

Inserting/Editing a SERCOS slave

After the ring has been entered, highlight the **End of Slaves** line in the Setup screen and press <Insert> or choose **Insert | Slave** from the menu. The **Insert Slave** box appears.



1. Enter the name you want to assign to the slave axis.
2. If the CPU is a standalone MMC or PiC, enter the slave number (1-8).
If the CPU is an MMC for PC, enter the slave number (1-32).

Note: The number entered here must match the address switch set on the slave.

With a PiC or standalone MMC CPU, the slaves on a ring must be numbered to match the quantity of slaves entered. For example, if you are entering three slave axes on this ring, number them in any order: 3, 1, 2; or 1, 3, 2 or 1, 2, 3. However, they cannot be numbered as 1, 2, 4.

If the CPU is an MMC for PC, enter the slave number (1-32) that matches the address switch set on the slave. Slaves do not have to be numbered to match the quantity of slaves entered. This is known as “Slave Gapping” and is allowed only with an MMC for PC CPU. If you later change the CPU type, you will need to renumber the slaves accordingly.

- The **Telegram**: number defaults to 4. If the slave will be controlled by motion.lib, then this is the telegram number you want. If you want to change to one of the numbers listed below, you can do so under **Define Cyclic Data**.

Telegram # Description

0	No data
1	Torque control
2	velocity control, velocity feedback
3	Velocity control, position feedback
4	Position control, position feedback
5	Position and velocity control, position and velocity feedback
6	Velocity control
7	User-defined

- Enter the name and location of the **DriveIDN ListFilename**: for this slave. Use the **Browse** button to locate the file if necessary.
- If this axis is a digitizing axis (external feedback will be used), check the **Digitizing Axis** box. The telegram changes to 7, **Digitizing Axis Probe Edge** checkboxes become active, and IDN 53 is added to the AT (if not already present).

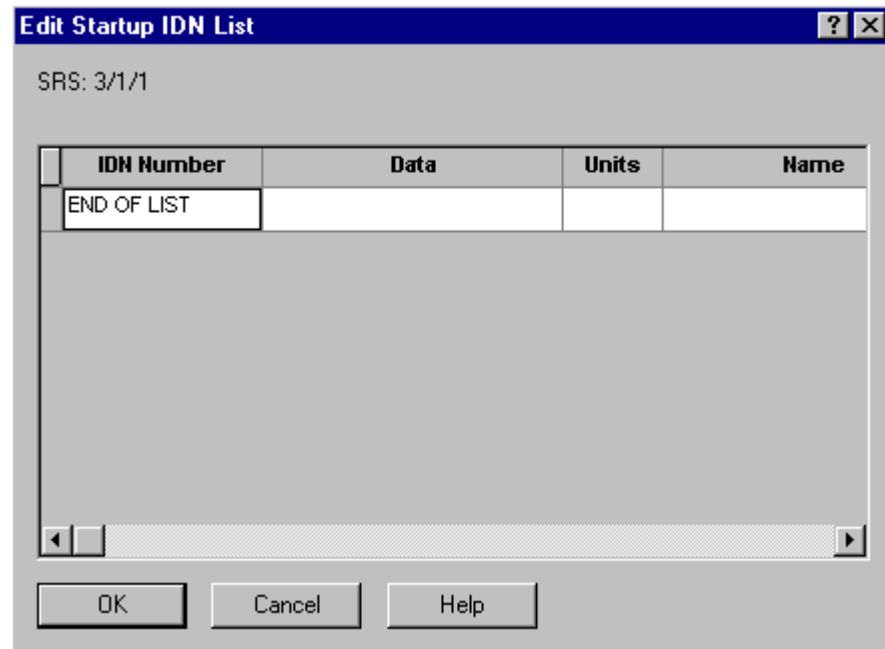
Note:A valid drive filename must be specified before you can select the **Digitizing Axis** box.

- In the **Probe Inputs** section, you can select whether the positive or negative edge of the probe input should be used for a servo or digitizing axis. If you use probe edges in your ladder, be sure to check them here in setup. The Telegram defaults to 7 when a probe input box is checked. The following IDNs will be added to the AT if they are not already present:
 - Servo Axis - 130 for positive edge, 131 for negative edge
 - Digitizing Axis - 132 for positive edge, 133 for negative edge
 - For either Servo or Digitizing Axis - 179 if any box is checked

Once you have inserted a slave in setup, you can edit it by highlighting the line it is on and pressing <Enter> or go to the **Edit** menu and select **Slave Data**. The **Edit Slave Data** box appears. The **Edit** dialog box looks just like the **Insert** dialog box.

Edit Startup IDN List

When you click on **Edit Startup IDN List**, the following dialog box appears.



If you are entering IDN numbers to be used at startup with this slave axis, double click on the **END OF LIST** entry in the IDN number column or press **<Insert>**. The **Insert IDN** box appears.

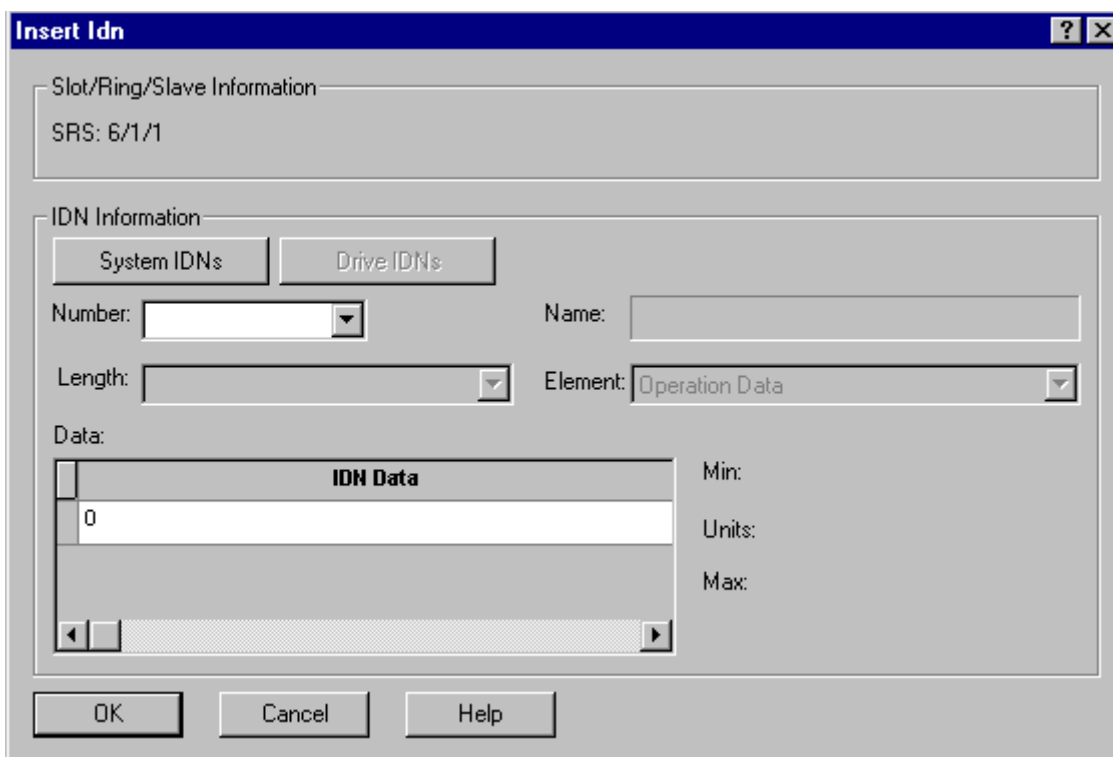
To edit an existing entry, double click on that entry in the list or press **<Enter>**. The **Insert IDN** box appears and you can make the necessary changes to the selected IDN.

If you press **<Insert>** on an existing entry, you can add a new IDN. It will be entered directly above the existing one.

You can also cut/copy/paste entries within this list using the **<Ctrl + X, Ctrl + C, and Ctrl + V >** keys.

Inserting/Editing IDNs

Enter the data for the IDN to be added to the startup list.



The **Slot/Ring/Slave Information** identifies the slave you are inserting an IDN for.

The **System IDNs** button brings up a list of system IDNs from the SERCOS specification. If you select an IDN from this list, it will appear in **Number:**.

The **Drive IDNs** button brings up a list of drive IDNs from the file you uploaded from the drive. If you select an IDN from this list, it will appear in **Number:**.

You can choose an IDN that is not in any list by entering its number from the range of 1 to 65535 (S1 to S32767 or P0 to P32767) in the **Number:** field.

The **Name:** box displays the IDN name for an IDN number that can be found in any list.

The **Length:** box indicates the size of the data (two or four byte, variable length one, two, or four byte data strings). This information is usually found in the system or drive IDN file. If not, refer to the SERCOS specification and enter the size listed there.

The **Element:** box indicates the SERCOS element to use.

Min: displays the minimum value allowed for the selected IDN.

Units: displays the units used with the selected IDN.

Max: displays the maximum value allowed for the selected IDN.

The **IDN Data** box allows you to enter the value within the minimum/maximum range given.

Clicking **OK** will place the data entered into the startup list. **Cancel** allows you to exit without saving any data.

Define Cyclic Data

From the **Edit Slave Data** box, click on **Define Cyclic Data**.

The **Slot/Ring/Slave Information** identifies the slave you are defining cyclic data for.

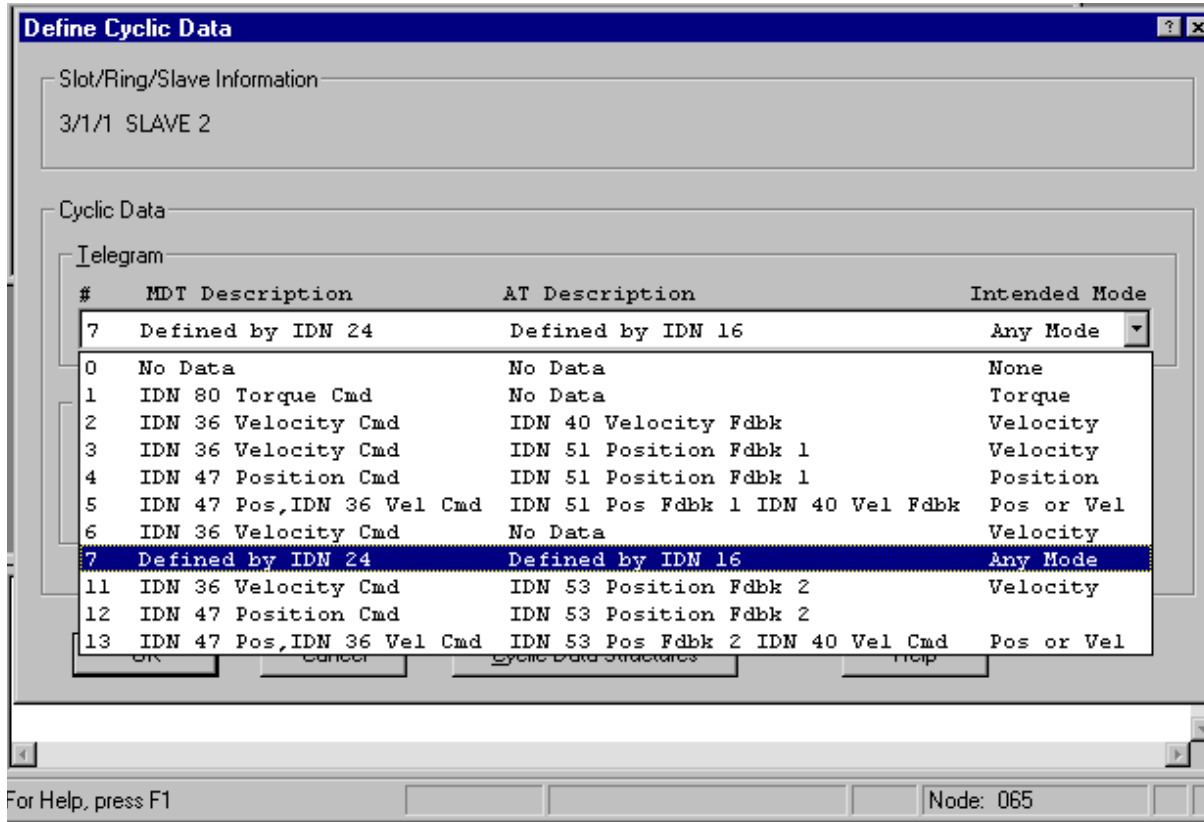
The **Telegram** area allows you to choose from the list of telegrams shown in a drop down list.

#	MDT Description	AT Description	Intended Mode
7	Defined by IDN 24	Defined by IDN 16	Any Mode

The telegram you choose is held in IDN 15. The **MDT:** and **AT:** telegram boxes hold the IDNs connected to the telegram you select.

Note: Once SERCOS is initialized, you cannot change this configuration. The operation mode must support these cyclic data requirements.

If you choose any **Telegram** number from the drop down list excluding number 7, the cyclic data will be defined for you. If you choose number 7, you must be sure that the drive IDN list has been uploaded from your drive and stored in a file. Then you define the cyclic data.



If you choose any telegram other than 7, you do not have to make any other choices.

If you choose telegram 3, 4, or 5, you are using feedback 1 on the drive.

If you choose telegram 11, 12, or 13, you are using feedback 2 on the drive.

If you choose telegram 7, you must also define a list of IDNs for sending and receiving cyclic data.

The **MDT** and **AT** buttons to the right of the **MDT:** and **AT:** description boxes are only active if telegram 7 is chosen. They include lists of IDNs from IDN 187 (AT) and IDN 188 (MDT). These come from the drive list that you uploaded from your drive. You click on an IDN in the list to add it to the description. You can click on a highlighted IDN in the list to remove it from the description.

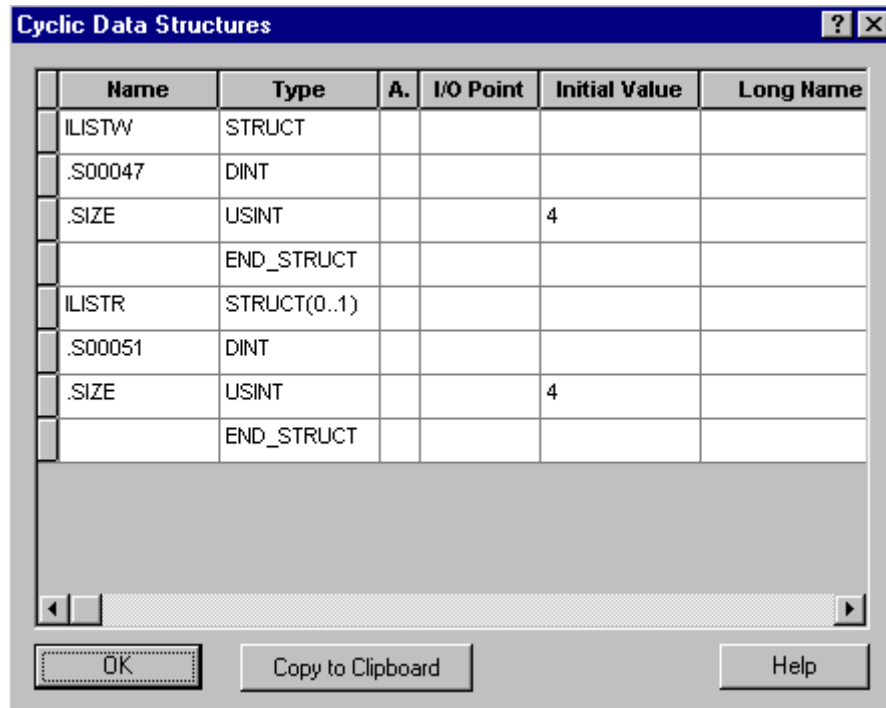
Note: When you are working with motion.lib, select either telegram 4 or telegram 7. If you choose 4, nothing further needs to be done. If you choose 7, IDN 47 (position read) must appear in the MDT description and IDN 51 for feedback 1 (IDN 53 for feedback 2) (position write) must appear in the AT description. You can then place additional IDNs in the description from the MDT/AT lists.

The list of IDNs for MDT become IDN 24 and for AT become IDN 16.

Cyclic Data Structures

From the **Define Cyclic Data** box, click on **Cyclic Data Structures**.

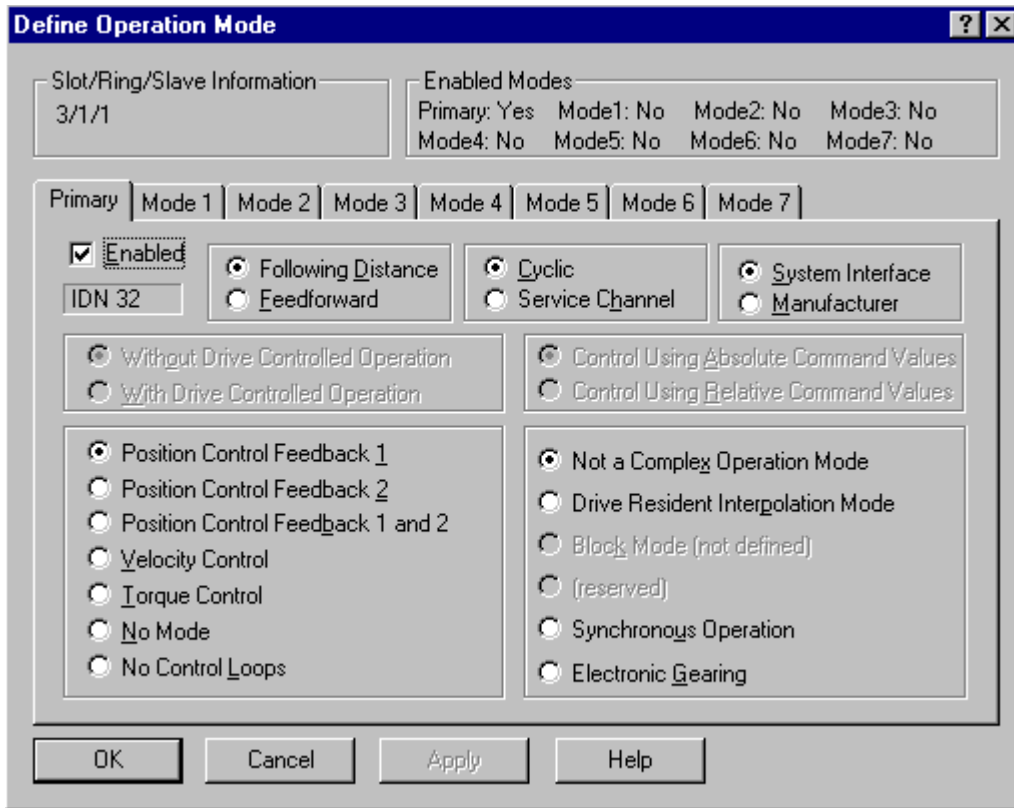
If you are using the SCA_RCYC and SCA_WCYC functions from motion.lib, you will need to include the structures in the **Cyclic Data Structures** box in your ladder. The **Copy to Clipboard** button allows you to transfer these structures to your software declarations table with the **Paste** command. The structure ILISTW is used with the SCA_WCYC function and the structure ILISTR is used with the SCA_RCYC function.



Define Operation Modes

From the **Edit Slave Data** box, click the **Define Operation Modes** button.

A SERCOS slave can operate in up to eight modes provided the drive you are using supports them. The modes are defined and enabled in the **Define Operation Mode** dialog box.



The slave is identified in the **Slot/Ring/Slave Information** area.

The enabled modes are shown in the **Enabled Mode** area.

When the **Define Operation Mode** box is opened, the **Primary** Mode tab is enabled with the defaults shown. You can change these defaults for the Primary Mode and you can enable any other mode and change the defaults connected to them. Simply click on the tab you want to define and enable.

The modes must support the cyclic data requirements that are configured using the **Define Cyclic Data** dialog.

Operation Mode Application Note

Typically, you will operate in the Primary Mode which represents position control. If your drive must operate in an additional mode such as torque, one or more of the Secondary modes is defined. The options you choose will depend on the features of the drive. If you choose something that is not supported by your drive, an error will be reported by the SCR_ERR function when the SC_INIT function block is called during the initialization of the drive. After you have made your SERCOS setup function, downloaded it to the PiC, and started the scan, your selections will be sent to the drive after phase 2. If an error occurs, it will appear in the SCR_ERR function and the initialization will stop. The slave number, the IDN, and the reason for the error are reported on the SLV, IDN, and SERR outputs respectively.

The option of **Cyclic** or **Service Channel** determines whether the command for this mode will be sent to the drive over the cyclic data or the service channel. Since torque control usually requires a fast update time, you would want to use cyclic data. Velocity control, on the other hand, can use the service channel. If cyclic is chosen, a custom telegram must be created. To do this, click on the **Define Cyclic Data** button in the **Edit Slave Data** dialog.

When you are using motion.lib to operate the drive, you use the position or Primary Mode. Position commands are sent to the drive. When operating in a Secondary Mode, the ladder has the responsibility of setting the mode and sending the appropriate commands.

The mode can be set by calling the SCA_CTRL function with the OPTN input set to 1. Prior to calling SCA_CTRL, the default is the primary mode. You can disconnect motion.lib from the drive by writing a 1 to WRITSV variable 48. Then you use the SCS_CTRL function to write the loop state, the cyclic data, and the operation mode to the drive.

The command is sent to the drive by the ladder. It is based on the mode selected and whether the transmission will be cyclic or service channel. If cyclic is chosen, the SCA_WCYC function is used to write the command from the ladder. It is recommended that this function be called from a task that is executed every servo update.

Copying SERCOS Data

Once you have entered a SERCOS configuration in setup, you can use the copy/cut/paste commands.

How to use the copy/cut/paste commands

1. Enter the SERCOS configuration.
2. Select the ring(s) or slave(s) you want to copy or cut. Choose the **C**opy or **C**ut command from the **E**dit menu or toolbar buttons, press <Ctrl + C> (copy) or <Ctrl + X> (cut), or right click and choose **C**opy or **C**ut.
Note: When you cut/copy a ring its slaves are also cut/copied.
3. Place the focus on the location you want to paste into and choose the **P**aste command from the **E**dit menu or toolbar buttons or press <Ctrl + V> or right click and choose **P**aste. The configuration will be pasted above the selected location.
Note: Pasting is limited by the valid number of rings and slaves allowed for the CPU type.
4. Double click on the slave. Make the necessary changes and choose **O**K.

Cut/Copy/Paste Limitations (between different SRC files)

When you cut/copy/paste between setup files that were created for different CPU types, you will receive confirmation messages about converting the slot, ring and slave values to zero. Each ring and slave will have to be edited to enter valid values.

If you are trying to paste more rings than allowed by the CPU type, you will receive an error message, and nothing will be pasted.

In some cases when pasting rings, more slaves will be pasted than allowed for the CPU type. They will have to be manually deleted.

If you are trying to paste more slaves than allowed by the CPU type, you will receive an error message, and nothing will be pasted.

When pasting a slave, the Receive Continuous and Send IDN entries for that slave will not be pasted. That information must be manually entered.

When pasting a ring from MMC for PC to PiC or standalone MCC, if the baud rate of the ring is 8 or 16, the baud rate will be set to 4.

<Ctrl + C> (copy) or <Ctrl + X> (cut), and <Ctrl + V> (paste) can also be used when editing the Startup IDN, Receive Continuous, and Send IDN lists.

Saving a SERCOS Setup File

It is a good idea to save your file at regular intervals as you work on it. Using the **S**ave command, you can save the file under its existing name. Using the **S**ave **A**s command, you can specify a new filename and/or a location where you want to store the file.

To save a new file

1. Choose **F**ile | **S**ave.
2. In the **S**ave **A**s box that appears, choose a drive and folder where you want to save your setup file.
3. Enter a name in the **F**ile **n**ame: box. **Note:** It is strongly recommended that the filename follow the DOS 8.3 naming convention of 8 characters, a period, and 3 characters.
4. Click **S**ave.

To save an existing file

1. Choose **F**ile | **S**ave.

OR

1. You can use the **F**ile | **S**ave **A**s command to change the name and/or location of a file. By giving the file a different name when you save it, you create a copy of the existing file while keeping the original intact.
2. Select the location for the file in the **S**ave **i**n: box.
3. Enter the new file name in the **F**ile **n**ame: box.
4. Click **S**ave.

To save in a format readable by a previous version of PiCPro

SERCOS Setup files created or modified in V13.0 cannot be read by previous versions of PiCPro. To save the file in a format that can be read by older versions of PiCPro choose **F**ile | **S**ave **A**s. In the **S**ave **a**s **t**ype: combo box select the desired PiCPro version. If invalid information for the specified format exists in the file, details will be shown in the Information Window to indicate the invalid ring(s) or slave(s). If the CPU type is MMC for PC, the file cannot be saved in a format readable by older versions of PiCPro.

5. In the **S**ave **a**s **t**ype: list, if the CPU type is:

Printing SERCOS Setup

You can print a highlighted selection or all the information in the SERCOS Setup file.

To print SERCOS data

1. Open the SERCOS Setup file (.src) that you want to print. If you just want to print the data for one ring, select that ring. If you want to print the data for a slave, select that slave.
2. Choose **F**ile | **P**rint from the menu, press <Ctrl + P>, or choose the print button from the toolbar. The print dialog box appears and you can click **OK**. If you want to print the data for everything in your setup file, choose **A**ll in the print dialog box and click **OK**.

Making a SERCOS Setup Function

After you have entered all the SERCOS setup information, you will create a SERCOS setup function to store all this information. This function will be stored in the SERCOS library you designate and can then be called in your ladder program to initialize the SERCOS setup data for your application. The SERCOS setup information will then be sent to the PiC when you download your application.

The name of the function will be the same as the name of the .src file name. It will appear in the list of functions in PiCPro.

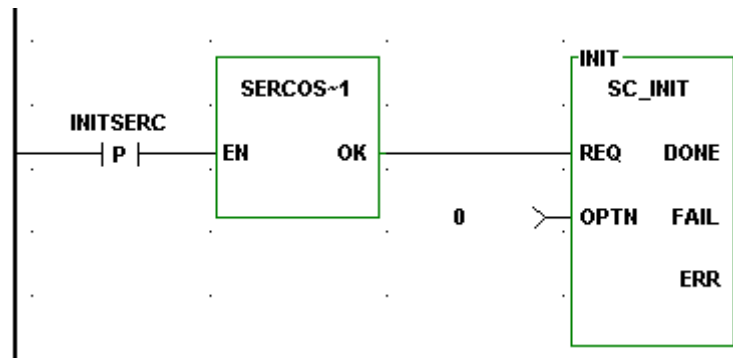
To create a SERCOS setup function

1. After all the setup information has been entered, choose **C**ompile | **M**ake **F**unction from the menu.
2. The **Save As** box will appear if you have not previously saved your setup file. Choose the location you want to save your .SRC file to. There is a default that is entered in the **F**ile **n**ame: box beginning with Sercos1.SRC. You can accept this name or enter your own.
Note: The second setup file you create will have the default name Sercos2.SRC, the third will be Sercos3.SRC and so on. It is strongly recommended that the filename be 8 characters, a period, and 3 characters.
3. The **Compile SRC** box appears. The **Registered ServoSetup/SercosSetup Libraries:** section will hold the names of any setup libraries you have created previously. You may choose from this list or create a new SERCOS Setup Library by entering one in the **L**ibrary **N**ame: box. **Note:** If you are recompiling a function that was created earlier, you must place the function in the same library where it was originally located. You cannot select a new library location. The **L**ibrary **N**ame: box will be disabled (grayed out).

4. Click **OK** to insert the SERCOS setup function into the chosen library. If you are creating a new library, a prompt will appear asking if you want to create a new library. If the location of the new library is not part of the PiCPro Library paths, a prompt will appear asking if the path should be added. Choosing **Yes** will define the library path so PiCPro can locate the function. The library and setup function will immediately show up in PiCPro under **Ladder | Functions** and in the Functions list on the toolbar.

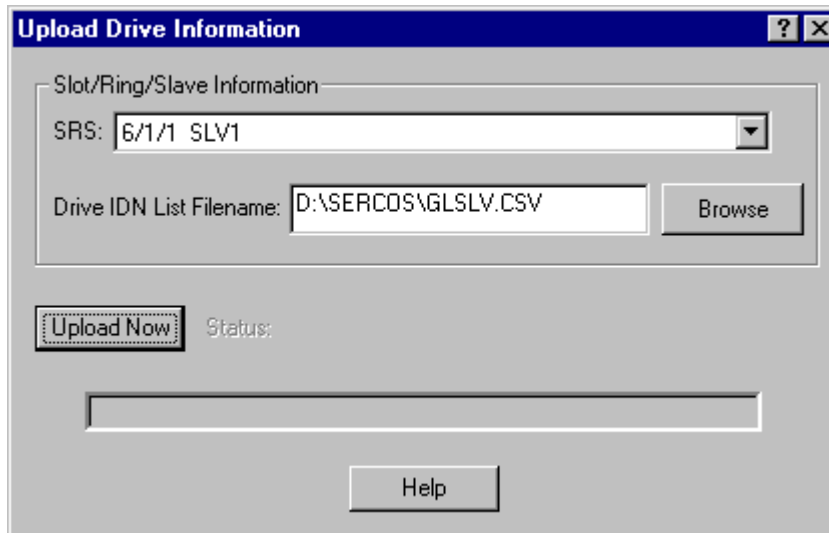
To initialize setup data in your ladder

The SERCOS setup function you compile with the SERCOS Setup program must be incorporated into your ladder program. To initialize all the setup data you use the standard motion SC_INIT function block with your setup function. Below is a network designed to do this. The example setup function is labeled SERCOS~1.



Upload Drive Information

If you want to use additional SERCOS features beyond replacing the analog signal with a digital interface, it will be necessary for you to upload the drive information from the drive. This information is in the form of a list of IDNs. The list will show all the information that can be read or written from the drive. It will indicate the units and the limits for each IDN in the list. Using this information, SERCOS setup can read or write information, define custom cyclic data, or change modes.



From the **Online** menu, select **Upload Drive Information**.

Available slave axes will be listed in the drop down list for **SRS:**. Choose the slave for which you want to upload the drive information.

The **Drive IDN List Filename:** box allows you to enter a path and filename or use the **Browse** button to select a folder and filename. It is recommended that the .csv extension be used for your file since it allows you to open the file in Excel. However, you can add any three letter extension you want to your filename.

The **Upload Now** button begins the upload process. The upload time can vary considerably based on the type of drive, the number of IDNs supported, and the type of PiC being used.

The **Status:** line will tell you if the upload is complete, if an error occurred and the process was halted, or if your PC is waiting for a response.

The status bar will show the progress of the upload.

Note: Upload can be done in phase 2 or 4.

Changing IDN Data and Uploading IDNs Note

Changing the data value of some IDNs can change related IDNs also. One example is the IDNs that control scaling. For example, IDN 76 position data scaling type determines the units of IDN 47 position command value (inches, millimeters, etc.). The attribute and units elements of IDN 47 change based on the value of IDN 76. If your drive allows you to write the data value of IDN 76, the drive should also make corresponding changes to the attribute and unit elements of IDN 47.

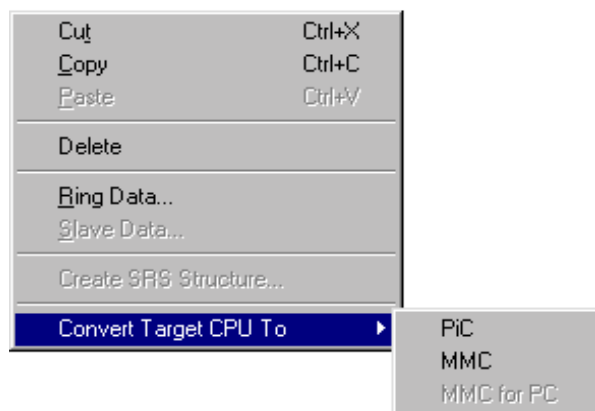
When you upload the IDN list from the drive, the units and attributes of each IDN are read and stored in the drive file. If, for example, changes are made to IDN 76 after the list is uploaded, IDN 47 attribute and unit elements will be different in the drive than in the drive file.

Always upload the drive IDNs after you make changes to IDNs that affect other IDNs. You can read the drive documentation to determine which IDNs affect the attribute and unit elements of other IDNs. Following is a list of common ones:

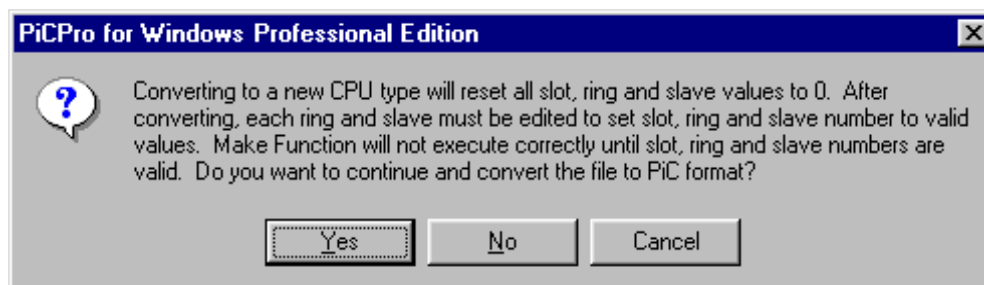
76	Position data scaling type
77	Linear position data scaling factor
78	Linear position data scaling exponent
79	Rotational position resolution
44	Velocity data scaling type
45	Velocity data scaling factor
46	Velocity data scaling exponent
160	Acceleration data scaling type
161	Acceleration data scaling factor
162	Acceleration data scaling exponent
86	Torque/force data scaling type
93	Torque/force data scaling factor
94	Torque/force data scaling exponent

Converting a SERCOS setup file's CPU Type

1. Choose **E**dit | **C**onvert Target CPU To.
2. Click on the flyout menu and select the CPU type you want.



A message box similar to the following will appear.



When you convert from one CPU type to another, all slot, ring, and slave values are set to zero. You will need to edit each one to assign the correct slot, ring and slave values. In cases where more rings or slaves are present than allowed for the chosen CPU type, you will need to delete them. The compile function will not allow you to have more rings or slaves than valid for the CPU type and all must have valid values.

Valid values are:

CPU Type	Number of Rings	Slaves per Ring	Slot Number	Ring Number	Slave Number
PiC	16	8	3-13	1-2	1-8
standalone MMC	1	8	1	1	1-8
MMC for PC	1	32	1	1	1-32

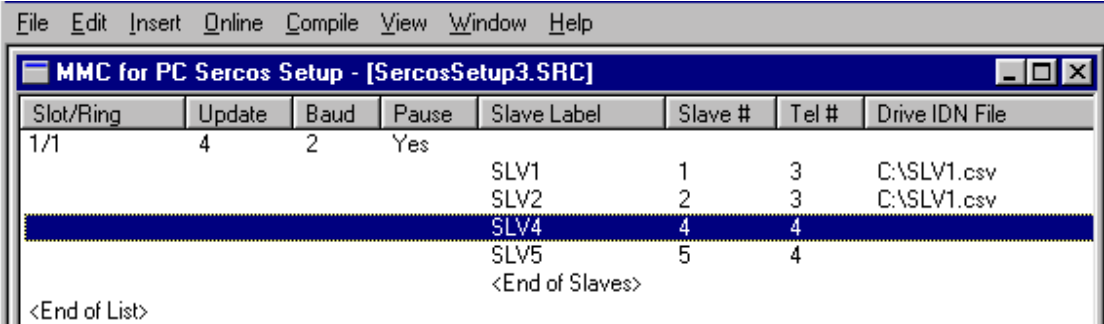
Note: Converting to a new CPU type will remove all entries from the Receive Continuous and Send IDN lists. The information will have to be manually reentered.

When converting from MMC for PC to PiC or standalone MMC, if the baud rate of the ring is 8 or 16, the baud rate will be set to 4.

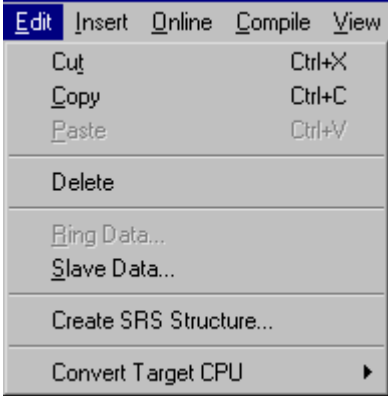
Copying an SRS structure to the clipboard

You can copy an SRS Structure to the clipboard for the slave that is currently selected. This structure contains slot, ring and slave information. The structure can be pasted into software declarations and used as the input to the SERCOS functions that need SRS. The procedure is as follows:

1. Open or create a SERCOS Setup file.
2. Select the Slave by clicking on the line containing the Slave.



3. From the **E**dit pull down menu, select **Create SRS Structure**.
The SRS Structure has now been copied to the clipboard and is available for pasting to software declarations.



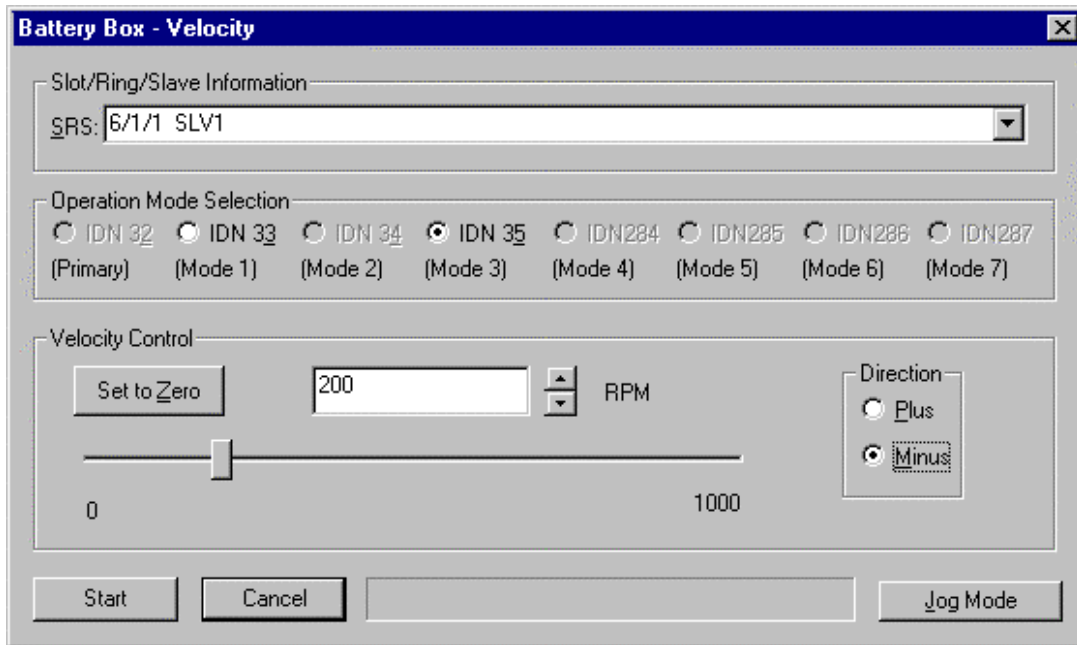
Note: If any slot, ring, or slave number was set to zero (because of a conversion of CPU type, or because of pasting between files for different CPU types) and was not manually changed to a valid number, an error message will be displayed, and the SRS Structure will not be pasted to the clipboard.

Battery Box

The battery box feature allows you to command velocity motion of a SERCOS slave from PiCPro.

IMPORTANT – It is the operator’s responsibility to prevent possible damage to machinery that is connected to the drive due to overspeed or direction errors.

From the **Online** menu, select **Batterybox**.

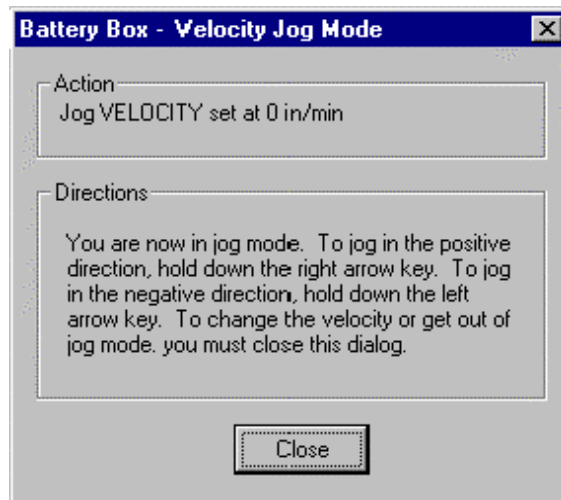


Using the Battery Box

The following procedure describes the steps to take to use the battery box to control the SERCOS drive.

1. In SERCOS setup, define the operation mode for the SERCOS slave as velocity over the service channel. It is recommended that you use one of the secondary modes for this since the primary mode is used by motion.lib. Click the mode tab, and click the **Enabled** box. Select the **Velocity Control** and the **Service Channel** radio buttons. Click **OK** when finished.
2. The Drive IDN S91 is the velocity limit for the battery box. If S91 is changed to enable the battery box (i.e. Send IDN), then the Drive IDN list file must be updated as well. If S91 is left at 0, the battery box velocity will be no greater than 0 preventing axis movement. **Note:** Reference IEC61491 for additional information on this IDN.
3. Ensure that all motion is stopped.
4. Call the OPENLOOP function to open the servo loop
Note: It is important to remember that there is a loop in the PiC and also a loop in the SERCOS drive that must be opened.

5. Call the CLSLOOP? function to ensure that both loops have been opened.
 6. Use the WRITE_SV function to set variable 48 to one. This prevents motion.lib in the PiC from controlling the drive. You can now use the battery box to move the axis.
- Note:** The drive loop is closed while the battery box is active. It does not open when you exit the battery box.
7. The Slot/Ring/Slave Information (**SRS:**) defaults to the selected slave in the SERCOS Setup List. If no slave was selected, all of the other controls in the dialog are disabled until you fill in the SRS.
 8. Select the operation mode.
 9. Type in the velocity or adjust the slider. You can also use the up and down buttons.
 10. Select either **Plus** or **Minus** for the direction.
 11. Click **Start**. Once started, the motion continues until you click on the **Stop** button.
OR
 12. Click **Jog Mode** for jogging the SERCOS drive and follow the directions shown in the dialog box.

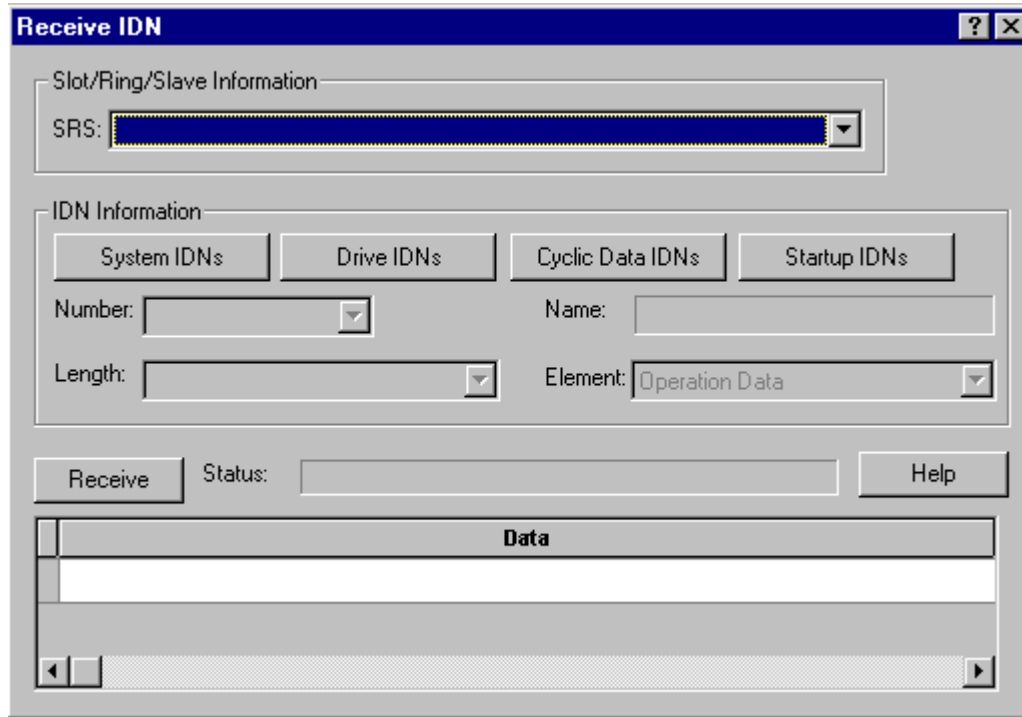


Note: With either method, there may be some delay between selecting an action and the drive motor reacting depending on the speed of the workstation.

13. When you are finished using the battery box, exit it.
 14. Use the WRITE_SV function to set variable 48 to zero. This gives control back to motion.lib in the PiC and both loops will be opened as in step 2.
 15. Ensure that all motion is stopped.
 16. Call the SCA_CLOS function to close the loop.
 17. Call the CLSLOOP? function to confirm that both loops are closed.
- You are now ready to resume control using motion.lib in the controller.

Receive IDNs

You can receive information on an IDN from the drive with this feature (**Online | Receive Idn...**). If you have a slave selected, the **Slot/Ring/Slave Information** will default to it. If there is no information in the **SRS:** box, you cannot proceed.



You can choose from four categories of IDNs: **System**, **Drive**, **CyclicData**, or **Startup IDNs**.

The **Number:**, **Name:**, **Length:**, and **Element:** boxes will be filled in when you select an IDN from one of the lists.

Or you can enter an IDN number (Range from 1 to 65535; or S1 - 32767 or P0 - 32767). If the number you entered can be found in one of the IDN lists, the data length will be displayed. If it cannot be found, you must enter the data length. Valid values will be listed in the drop down list.

The **Element:** box allows you to select what element of the IDN you want to receive. The default is the operation data. Other elements include the name, attribute, units, minimum value, and maximum value. Choose them from the drop down list.

After you have entered the IDN number, the **Receive** button is activated. Click it to read the requested IDN from the drive. The **Receive** button changes to **Abort**.

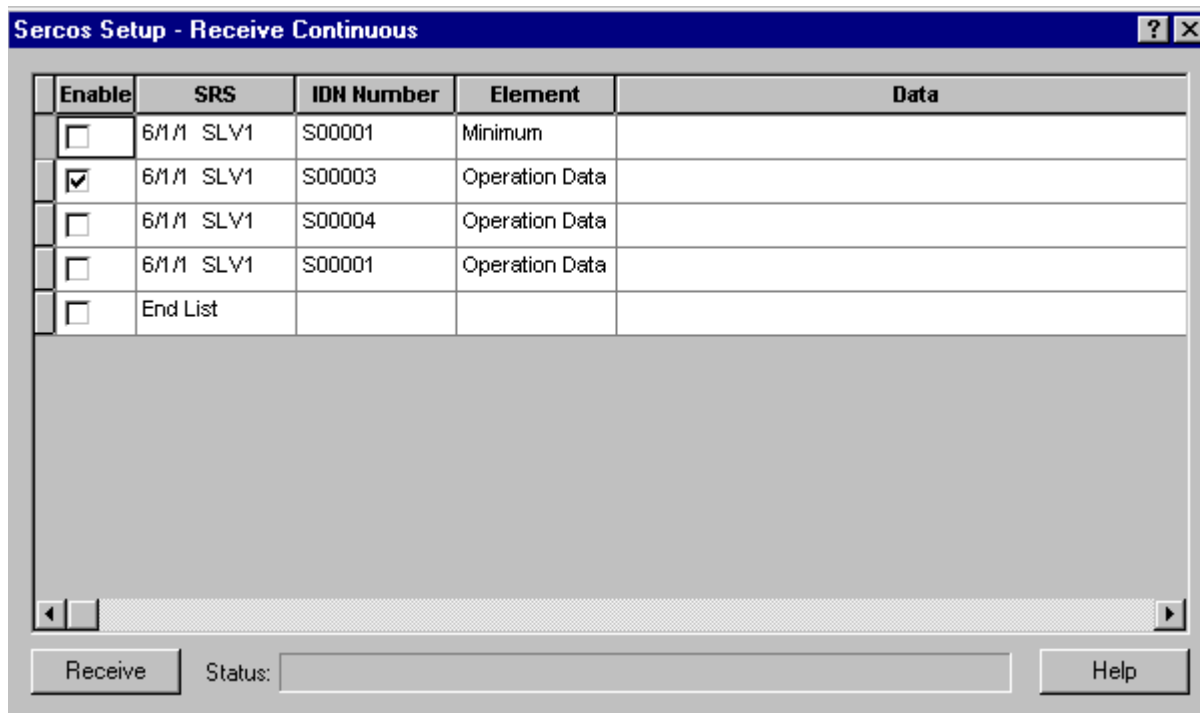
The **Status:** box reports on the progress of the receive command.

The **Data** box holds the data for the requested IDN. If the IDN is a string and is not completely displayed, double click on the Data entry to view the entire string. If the IDN requested is a variable length array, a grid will be displayed. Each array element is displayed in a separate row.

Receive Continuous IDN Data

You can choose to receive IDN data continuously using the **Online | Receive Continuous...** command. Add new IDNs to the list by double clicking or pressing the **<Insert>** key on the **End List** line. The **Insert IDN** dialog box appears.

Click in the **Enable** column to choose which IDNs you want to receive continuously.



With **Receive Continuous**, the **Element** column can include operation data, attribute, minimum value, and maximum value.

Note: Variable length data cannot be requested in **Receive Continuous**.
Use **Receive Idn**.

To delete an entry from the receive continuous list, click in the first column or select the entire row and press **<Delete>**. A warning message will appear. Click **OK** if you want to proceed with the deletion.

To cut, copy, and paste in the list use the **<Ctrl + X, Ctrl + C, and Ctrl + V >** keys.

When the **Receive** button is chosen, all the enabled IDNs will be received continuously. The button text changes to **Abort**. You can choose **Abort** to halt the receive continuous command. The **Status:** field reports on the status of the receive continuous command.

Inserting/Editing IDNs for Receive Continuous

To insert a new IDN in the Receive Continuous list, you can double click on the **End List** line or press the <Insert> key when the focus is on any line. The new IDN will be inserted above the line focus is on.

To edit an existing IDN in the Receive Continuous list, you can double click on the entry or press <Enter>. To enter a new IDN when the focus is on an existing entry, press <Insert>. The new IDN appears above the selected line.

The **Insert IDN** box shown below appears.

The screenshot shows the 'Insert IDN' dialog box. The title bar is 'Insert IDN' with a help icon and a close icon. The dialog is divided into two main sections. The first section, 'Slot/Ring/Slave Information', contains a dropdown menu labeled 'SRS:' with the value '6/1/1 SLV1'. The second section, 'IDN Information', contains four buttons: 'System IDNs', 'Drive IDNs', 'Cyclic Data IDNs', and 'Startup IDNs'. Below these buttons are several input fields: 'Number:' (a dropdown menu), 'Name:' (a text field), 'Length:' (a dropdown menu), 'Element:' (a dropdown menu showing 'Operation Data'), and 'Units:' (a text field). At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Help'.

The **SRS:** information must be entered in order to proceed.

You can choose an IDN from any of the four IDN buttons or enter one from the range 1 to 65535 or S1 to S32767 or P0 to P32767 in the **Number:** field.

If the number you entered is located in any of the IDN lists, a name will appear in the **Name:** field and a value will appear in the **Length:** field.

If the number is not found in any of the IDN lists, no name appears in the **Name:** field and you must enter the length in the **Length:** field.

The **Element:** field defaults to operation data. You can change this to attribute, minimum, maximum, or data from the drop down list.

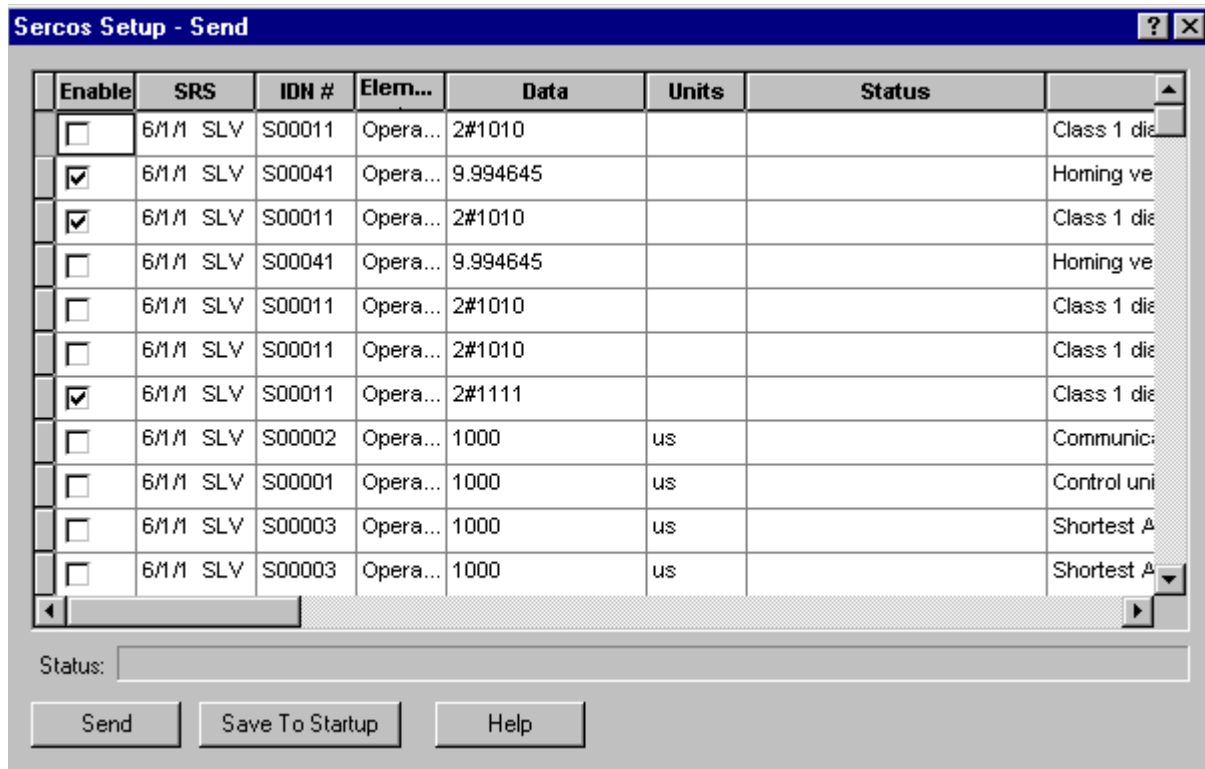
Units: displays the type of units for the selected IDN.

Choose **OK** to add the information entered to the Receive Continuous list or **Cancel** to ignore any changes and return to the **Receive Continuous** box.

Send IDNs

If you want to send IDNs, choose the **Send IDN** command from the **Online** menu or right click the mouse and choose **Send IDN**. Add new IDNs to the list by double clicking or pressing the **<Insert>** key on the **End List** line. The **Insert IDN** box appears and you enter the information for the IDN you want to add to the Send List.

Once your list is complete, you must check the box in the **Enable** column if you want to send that IDN when you select the **Send** button.



You can cut/copy/paste entries within this list using the **<Ctrl + X, Ctrl + C, and Ctrl + V >** keys.

When the **Send** button is selected, the text changes to **Abort** so that you can select it to stop the transfer at any time.

The **Status:** bar reports on the progress of the send command. Communication and drive errors that may occur are also reported here.

The **Save To Startup** button saves the IDNs you highlight to the appropriate startup list. You may want to do this after you know the data you have entered is working for your application.

Note: Only IDNs with the Operation Data element can be saved to the startup list.

Inserting/Editing IDNs for Send

To insert a new IDN in the Send list, you can double click on the **End List** line or press the <Insert> key when the focus is on any line. The new IDN will be inserted above the line focus is on. To edit an existing IDN in the Send list, you can double click on the entry or press <Enter>.

The **Insert IDN** box shown below appears.

The screenshot shows the 'Insert Idn' dialog box. The title bar is 'Insert Idn' with a question mark and a close button. The dialog is divided into two main sections. The top section is 'Slot/Ring/Slave Information' and contains a dropdown menu labeled 'SRS:' with the value '6/1/1 SLV1'. The bottom section is 'IDN Information' and contains four buttons: 'System IDNs', 'Drive IDNs', 'Cyclic Data IDNs', and 'Startup IDNs'. Below these buttons are four fields: 'Number:' (a dropdown menu), 'Name:' (a text field), 'Length:' (a dropdown menu), and 'Element:' (a dropdown menu showing 'Operation Data'). Below these fields is a 'Data:' section containing a text box labeled 'IDN Data' and three labels: 'Min:', 'Units:', and 'Max:'. At the bottom of the dialog are three buttons: 'OK', 'Cancel', and 'Help'.

The **SRS:** information must be entered in order to proceed.

You can choose an IDN from any of the four IDN buttons or enter one from the range 1 to 65535 or S1 to S32767 or P0 to P32767 in the **Number:** field.

If the number you entered is located in any of the IDN lists, a name will appear in the **Name:** field and a value will appear in the **Length:** field.

If the number is not found in any of the IDN lists, no name appears in the **Name:** field and you must enter the length in the **Length:** field.

The **Element:** field defaults to operation data and indicates the SERCOS element to use. You can change this to attribute, minimum, maximum, or data from the drop down list.

Min: displays the minimum value allowed for the selected IDN. **Units:** displays the units used with the selected IDN. **Max:** displays the maximum value allowed for the selected IDN.

The **IDN Data** box allows you to enter the value within the minimum/maximum range given.

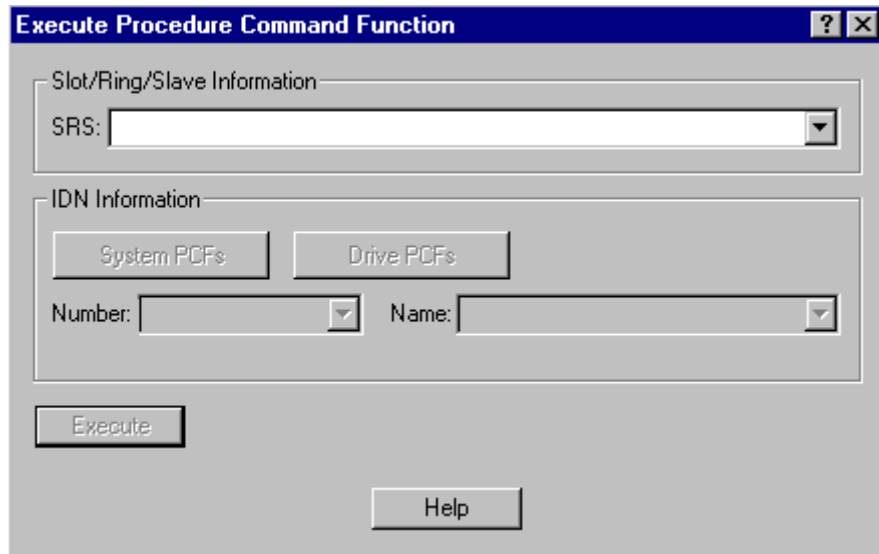
Clicking **OK** will place the data entered into the Send List. **Cancel** allows you to exit without saving any data.

Execute Procedure Command

The **Online | Execute Procedure CMD** Command Function (PCF) allows you to select and execute SERCOS procedure commands.

The **Slot/Ring/Slave Information** must be entered in order to proceed. If you have selected a slave in SERCOS setup, it will appear in the **SRS:** box. You can choose one from the drop down list.

The **System PCFs** and the **Drive PCFs** buttons hold lists of PCFs from the system file or the drive file respectively. Select a PCF (IDN) from a list or enter one. It will be displayed in the **Number:** box. The name appears in the **Name:** box.

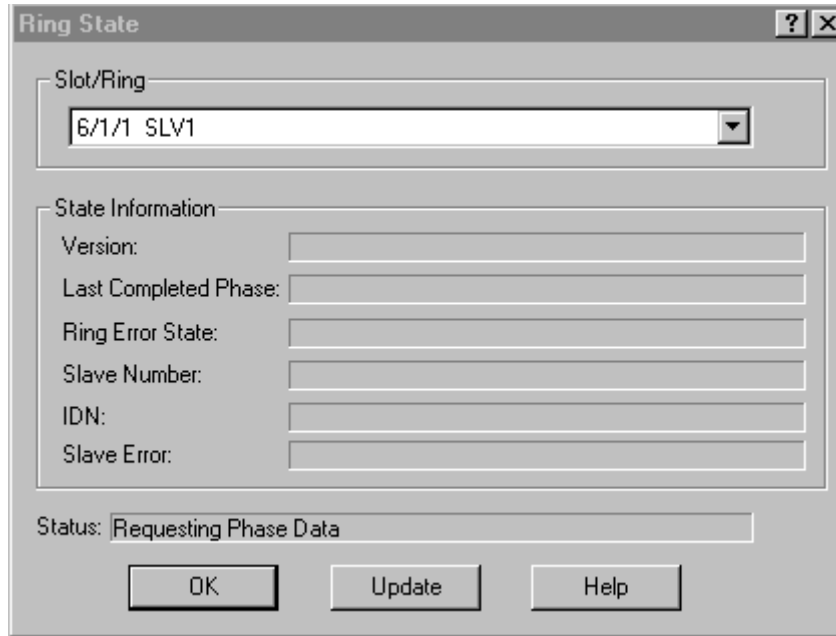


Once the PCF has been entered, the **Execute** button is active. Click it to send the PCF IDN to the drive and begin the procedure. The progress of the procedure will be noted to the right of the **Execute** button.

The **Execute** button converts to a **Cancel** button when progress begins.

Ring State

By using **Online | Ring State...** you can read the **State Information** shown below for a ring.



The **Version:** represents the version number of the firmware on the SERCOS module.

The **Last Completed Phase:** gives the phase number including whether the ladder is halted after phase 2.

The **Ring Error State:** displays one of the ring error numbers shown below if one occurs.

ERR#	Description	What to do/check
0	No error	
3	Axis is not initialized, is not a SERCOS axis, or the slot/ring/slave specification is incorrect.	<ul style="list-style-type: none"> ▪ SERCOS board in correct slot ▪ SR structure members correct
17	The SERCOS module did not receive an expected AT response. Cable could be disconnected.	<ul style="list-style-type: none"> ▪ Check connection
20	Phase 0 detected that the ring is not complete.	<ul style="list-style-type: none"> ▪ Check connection ▪ Ensure drive is turned on
65	Error occurred calculating when MDT should occur.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ One or more drives cannot accommodate required MDT

66	Error occurred calculating when drive data valid.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ One or more drives cannot accommodate command times
67	Error occurred calculating when feedback data valid.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ One or more drives cannot accommodate feedback capture times
68	Error occurred calculating total time required for communication cycle.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long ▪ Update rate too fast
69	Error occurred calculating cyclic data memory for SERCON processor.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
70	Error occurred calculating cyclic data memory for internal memory map.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
71	Error occurred calculating service channel memory map.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
74	CPU on SERCOS module has too many tasks during update.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
128	Slave error occurred. Read SERR output to identify error. The SLV output indicates the slave number.	<ul style="list-style-type: none"> ▪ SLV output contains slave number ▪ IDN output contains the IDN transfer that caused the error ▪ SERR output contains the drive generated error number ▪ Read Drive diagnostic IDN 95
136	Individual slave will not respond. The SLV output indicates the slave number.	<ul style="list-style-type: none"> ▪ Address switch on drive does not match slave number ▪ Baud rate switch on drive does not match rate in ring definition ▪ SLV output contains slave number that does not respond
144	Individual slave cannot carry out a Procedure Command Function. The SLV output indicates the slave number.	<ul style="list-style-type: none"> ▪ SLV output contains slave number ▪ IDN output contains the Procedure Command Function that caused the error ▪ For IDN = 127, read IDN 22 to read list of IDNs still required by the drive ▪ For IDN = 128, read IDN 23 to read list of IDNs still required by the drive ▪ Read Drive diagnostic IDN 95

The **Slave Number:** can be 1 to 8 when using a PiC or standalone MCC or 1-32 when using an MMC for PC CPU.

The **IDN:** is either a S (system) or P (product) IDN number.

The **Slave Error:** holds the SERR if one occurs.

Status: provides current information on the status of the ring.

Selecting the **Update** button updates the ring data in the box.

Slave Status/Control

In SERCOS, there is a status word from each slave to the control and there is a control word from the control to each slave. You can read the data in each word using the **Online | Slave Status/Control** command that brings up the box shown below.

The screenshot shows a software window titled "Status and Control". At the top is a dropdown menu labeled "Slot/Ring/Slave". Below this are two main sections: "Control" and "Status". Each section contains several input fields: "Drive" (three lines), "Operation Mode", "Realtime Bit 1", and "Value". At the bottom of the window are "Start" and "Help" buttons, and a "Status:" label followed by a text field.

The **Control** section displays the control word data continuously. The three lines for the **Drive:** tell whether it's on or off, enabled or not, halted or restarted. The **Operation Mode:** indicates which mode is in effect. The **Realtime Bits** indicate whether they are on or off. The word **Value:** is displayed in hex.

The **Status** section displays the status word data continuously. The **Drive:** line tells whether the drive is ready or not, the logic is ready, and the power is active or not. The **Error:** line reports on whether or not an error has occurred. The **Operation Mode:** indicates which mode is in effect. The **Realtime Bits** indicate whether they are on or off. The word **Value:** is displayed in hex.

IDN Lists

There are lists of IDNs for the system, the drive, the cyclic data, the startup, the system PCFs, and the drive PCFs. While working in SERCOS setup, there are several dialog boxes that allow you to access these lists of IDNs. You can also view the system and drive IDN lists from the **V**iew menu.

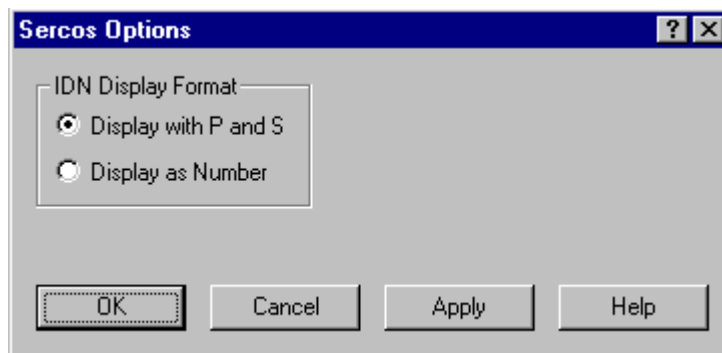
Options for IDN Display

There is an **O**ptions command found in the **V**iew menu in SERCOS setup. It allows you to choose how IDN numbers (1 to 65635) will be displayed. There are two groups of IDN numbers:

- The P group is specific to the drive manufacturer's product. They range from 32768 to 65635 or P0 to P32767.
- The S group is defined by the SERCOS specification. They range from 1 to 32767 or S1 to S32767.

You can choose to display IDNs with the P and S format (default) or with numbers only.

Note: Not all drive manufacturers support the P and S format.



When the **Display with P and S** option is selected, if you enter an IDN number without the letter prefix, it will be added for you when you tab off of the IDN field.

When the **Display as Number** option is selected, if you enter an IDN number with the letter prefix, it will be converted to the number only when you tab off the IDN field.

After selecting the IDN display format you want, choose **OK** to accept your format and close the dialog box. If you choose **Apply**, your format will be applied but the dialog box does not close.

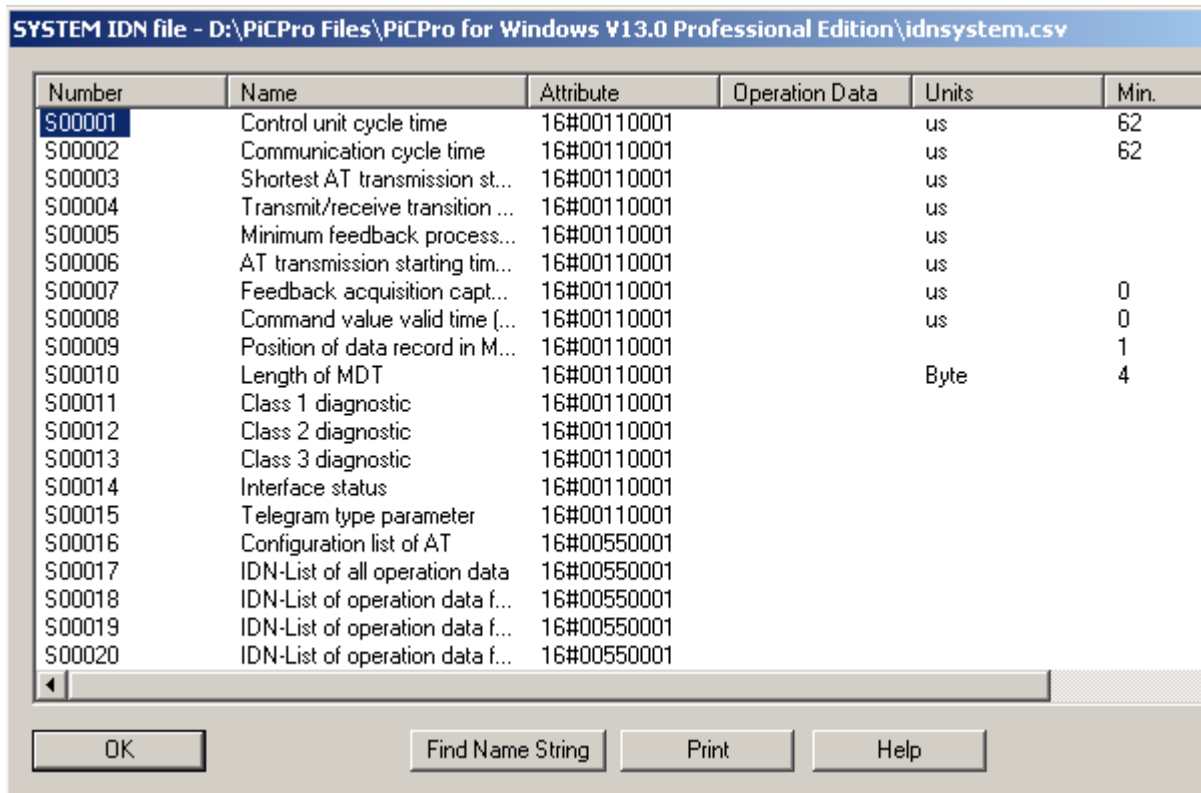
IDN Lists

The system IDN list appears when you click on the **System IDNs** button in a dialog box or choose **View | View System IDN** from the menu.

If under **View | Options** in SERCOS setup, **Display with P and S** is chosen, the System IDNs are identified with an S preceding the IDN number.

If **Display as Number** is chosen, then the system IDNs are numbered from 1 to 32767.

Clicking on the **Number** column sorts the list by number; clicking on the **Name** column sorts the list by name. Also, the columns will toggle between ascending and descending order.



SYSTEM IDN file - D:\PICPro Files\PICPro for Windows V13.0 Professional Edition\idnsystem.csv

Number	Name	Attribute	Operation Data	Units	Min.
S00001	Control unit cycle time	16#00110001		us	62
S00002	Communication cycle time	16#00110001		us	62
S00003	Shortest AT transmission st...	16#00110001		us	
S00004	Transmit/receive transition ...	16#00110001		us	
S00005	Minimum feedback process...	16#00110001		us	
S00006	AT transmission starting tim...	16#00110001		us	
S00007	Feedback acquisition capt...	16#00110001		us	0
S00008	Command value valid time (...)	16#00110001		us	0
S00009	Position of data record in M...	16#00110001			1
S00010	Length of MDT	16#00110001		Byte	4
S00011	Class 1 diagnostic	16#00110001			
S00012	Class 2 diagnostic	16#00110001			
S00013	Class 3 diagnostic	16#00110001			
S00014	Interface status	16#00110001			
S00015	Telegram type parameter	16#00110001			
S00016	Configuration list of AT	16#00550001			
S00017	IDN-List of all operation data	16#00550001			
S00018	IDN-List of operation data f...	16#00550001			
S00019	IDN-List of operation data f...	16#00550001			
S00020	IDN-List of operation data f...	16#00550001			

Buttons: OK, Find Name String, Print, Help

You can use the **Find Name String** button to bring up an **IDN Search** box.

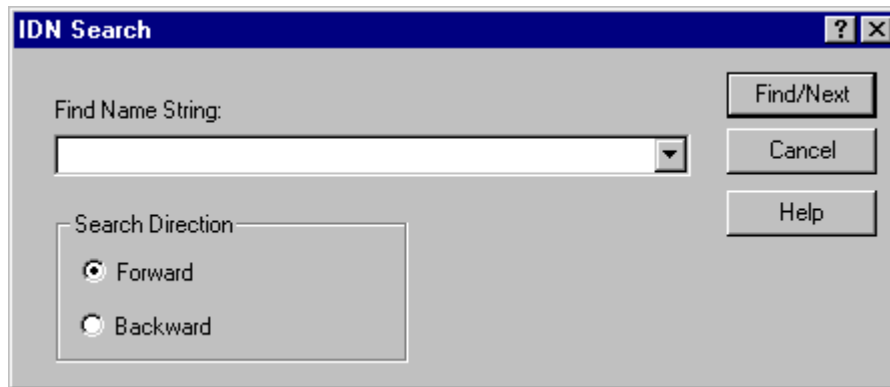
When you click on the **Drive IDNs** button in a dialog box, it displays a list of IDNs that you uploaded from the drive. You can sort by number or name, toggle the sort order, search by name, and print this drive IDN list.

When you click on the **Cyclic Data IDNs** button in a dialog box, it displays a list of IDNs that you entered as cyclic data.

The **Startup IDNs** button in a dialog box displays a list of IDNs you entered for startup.

Finding a Specific IDN in an IDN List

When viewing an IDN list, you can use the **Find Name String** dialog box to help you locate a specific IDN. If you know the name or part of the name of the IDN you want to search for, the **Find Name String** dialog locates the IDNs that match your search criteria quickly and easily.

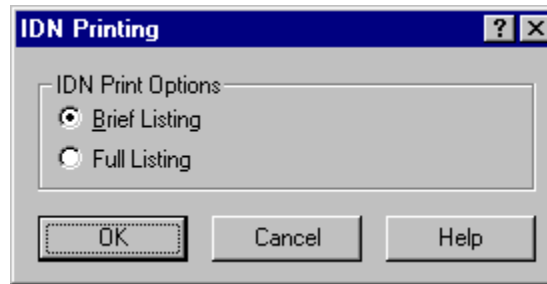


To find specific text in an IDN list, follow these steps:

1. Click the **Find Name String** button on the IDN list you are viewing. This will display the box above.
2. Type the desired text to search for in the field labeled **Find Name String**.
3. Select the direction you wish to search, **Forward** or **Backward**, from the currently highlighted IDN.
4. Click the **Find/Next** button to begin the search.
5. If a matching IDN is found, the IDN number is highlighted. To search again for another match, click the **Find/Next** button again.
6. The most recently searched for names will appear in the drop down list.

Printing IDNs from an IDN list

When viewing an IDN List, you can use the IDN printing dialog to print the IDN list.



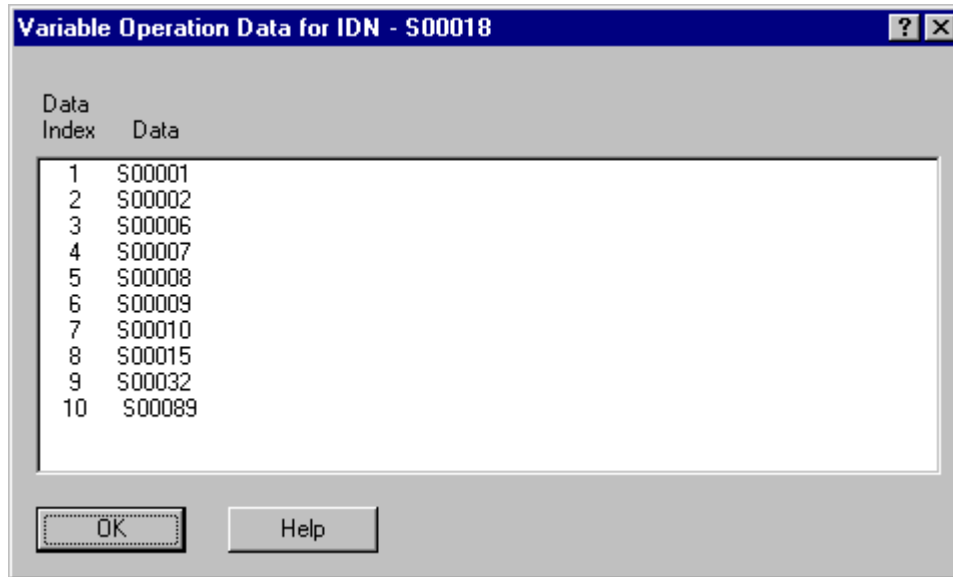
To print an IDN list, follow these steps:

1. Click the **Print** button on the IDN list you are viewing. This displays the IDN printing box shown above.
2. Select **Brief Listing** to print the IDN number, name, and data for all of the IDNs in the list.
3. Select **Full Listing** to print all the IDN elements for all the IDNs in the list.
4. Click **OK**. This will display the **Print** dialog for your default printer.

Viewing Variable Length Data from an IDN List

Some IDNs in IDN lists have variable length operation data. This is indicated by the word **Variable** appearing in the **Operation Data** column of the list. (S00018 is an example.)

To view this data you must use the **Variable Operation Data** dialog box shown below.



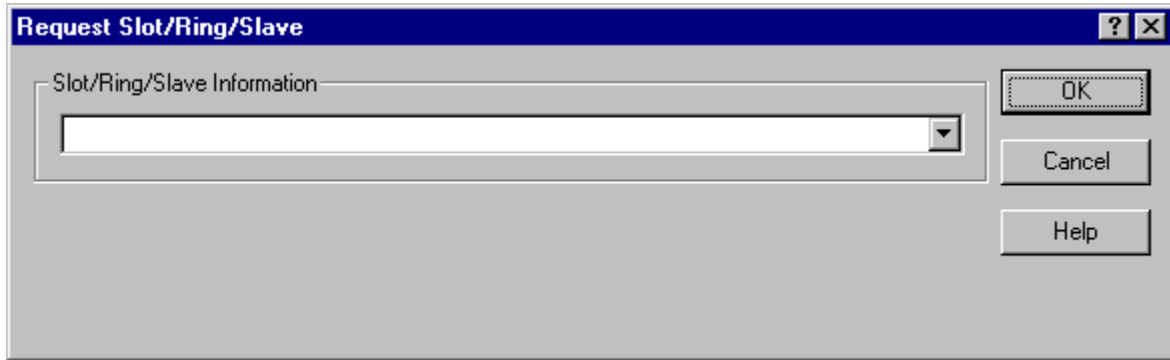
To view variable length operation data, follow these steps:

1. Right click on the IDN with variable data in the IDN list. This will display a popup menu. Select the **View Variable Data** item from this menu. This brings up the **Variable Operation Data** box shown above.
2. The variable length Operation Data is listed in this dialog under the heading **Data**.
3. You will notice a column to the left of the data labeled **Data Index**. This is just a column that numbers the data items in the displayed list. It is not a part of the Operation Data itself.
4. When finished, click **OK** to close the dialog box.

Specifying the SRS for Viewing the Drive IDN List

In order to view the Drive IDN List from the **V**iew menu on the main menu bar, you must indicate which drive you want.

Using the **Request Slot/Ring/Slave** dialog box, select the appropriate SRS which uniquely identifies the desired drive.



To specify the SRS, follow these steps:

1. From SERCOS Setup, select the **V**iew | **V**iew **D**rive **I**DN item from the menu.
2. The **Request Slot/Ring/Slave** box shown above appears. You specify which drive by selecting the appropriate SRS from the drop down list provided.
3. After choosing the SRS, click the **OK** button. The Drive IDN list for the specified drive will be displayed.

Troubleshooting SERCOS

Here are a few tips on troubleshooting while working in SERCOS.

Communication Problems

- Communication phase 4 must be completed before you can begin to control.
- The ERR output of the SC_INIT function block can provide information on what went wrong during communication.
- If SC_INIT executes correctly with no errors, you should check for ring errors next using the SCR_ERR function or within SERCOS setup using the **Online | Ring State** command. Information reported here indicates problems with moving through the communication phases.
- Check the phase number completed by using the SCR_PHAS function or within SERCOS setup using the **Online | Ring State** command.
- If the drive will not advance into phase 3 or 4, it is possible that there are IDNs that need to be defined. The IDNs that have not yet been defined are contained in IDN 22 and 23. This list can be read from the ladder using the SCA_RECV or SCS_RECV function or within SERCOS setup using the **Online | Receive Idn** command.

Control Problems

- If the loop will not close, you can read the status of the drive using the SCA_STAT or SCS_STAT function or the **Online | Slave Status/Control** command. The control word has 16 bits. Bits 1 and 2 indicate the state of power stage the drive is in. Bits 8, 9, and 10 indicate the operation mode. You may also want to read IDNs 11, 12, and 13.
- A loss of feedback error may occur if the ring has not reached phase 4.
- If the drive does not accept IDNs, the drive may not support the feature. Check the SERR output of a SERCOS function to learn more.
- Use SCS_ or SCA_RECV functions to read IDN 95. IDN 95 contains a text diagnostic message with basic error conditions reported from the drive.
- IDN 11, 12, and 13 are diagnostic IDNs that report common control problems. Consult your drive documentation for the bit definition of these IDNs.

CHAPTER 6 Working With Tasks and UDFBs

UDFBs

The User Defined Function Block (UDFB) feature allows you to convert the logic in a ladder module (.LDO) into an individual function block. The name you give to the LDO becomes the name of the UDFB. The UDFB is inserted into a library file (.LIB) and appears in the list of available functions. It can then be used in a network of any module you create.

Using the UDFB feature results in an application program that is better organized, more readable, and easier to maintain. Some ways UDFBs can be used include:

- UDFBs can segregate the logic for a section of an application program (i.e. diagnostics, fault logging, etc.).
- Custom functions with a higher level of functionality can be created (i.e. analog output with ramping).
- A single UDFB can replace in-line programming of repetitive machine functions (i.e. axis one fault control, axis two fault control, etc.).

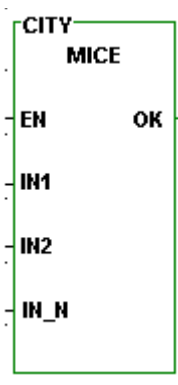
The UDFB follows the same conventions as the standard function blocks found in PiCPro. It must be declared in the software declarations table when it is used in the network of a module. You assign a name to it when it is declared.

Creating a UDFB

Briefly, you will do the following to create and use a UDFB.

1. Open a new ladder diagram and enter ladder logic which will accomplish what you want the UDFB to do.
2. Mark the variables that you want to appear as labels in the UDFB template as inputs or outputs in the software declarations table.
3. Compile the UDFB.
4. Open another ladder diagram in which you want to use the UDFB and insert the UDFB from the Function list into a network. It will have to be declared in the software declarations table and given a name (CITY in the example below.)

Ladder



Structured Text

```
<<INSTANCE NAME>>:MICE(EN:=<<BOOL>>,  
IN1:=<<BOOL>>, IN2:=<<BOOL>>, IN_N:=<<NUMERIC>>,  
OK => <<BOOL>>;
```

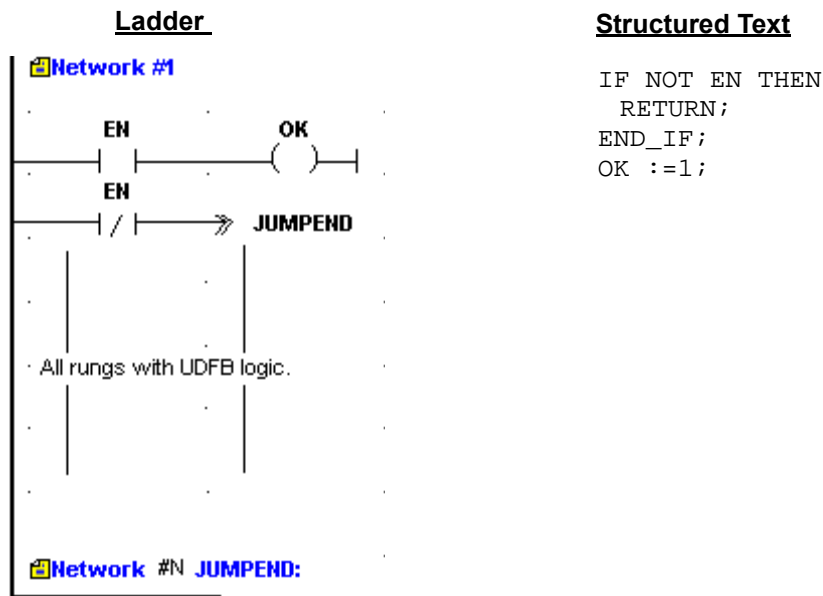
The EN input and the OK output are required. The remaining inputs and outputs are determined by you when creating the module.

Naming the UDFB

The name of the LDO module will become the name of the UDFB when you compile it. If you have not saved the module previously, you will be prompted to enter the name at compile time. If the name is more than eight characters, it will be truncated. Be sure to choose a name that does not already exist in another module you may want to convert to a UDFB or in a standard function/function block supplied by PiCPro.

Scanning UDFB Logic

The logic contained in the UDFB is scanned regardless of whether or not there is power flow into the EN input. Therefore, in the majority of cases, you will want to include the following lines of logic in your UDFB module.



This ensures that you will always have control of the EN and OK in any module you use the function block in. It causes the program to jump to the end of the UDFB logic if the EN is not set. This prevents the logic in the function block from being scanned unless the EN is set.

Note the wire in the empty network #N with the JUMPEND label. PiCPro does not allow empty networks. A horizontal wire was added so that the empty network error is not generated.

The only situation where you would not want to control the EN and OK in this manner is if you create a UDFB which would need to work when the EN is not set. The TOF (timer off) standard function block is an example. The function block continues to execute after the EN is reset.

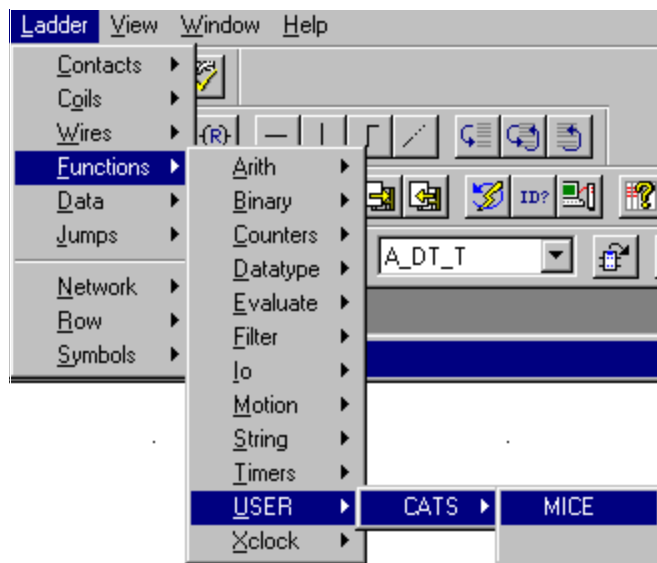
Contents of the UDFB LDO module

Only the logic you want handled by the UDFB should be included in the module. The size of any UDFB can be up to 64K. The size of your UDFB will be listed in the information window when you compile.

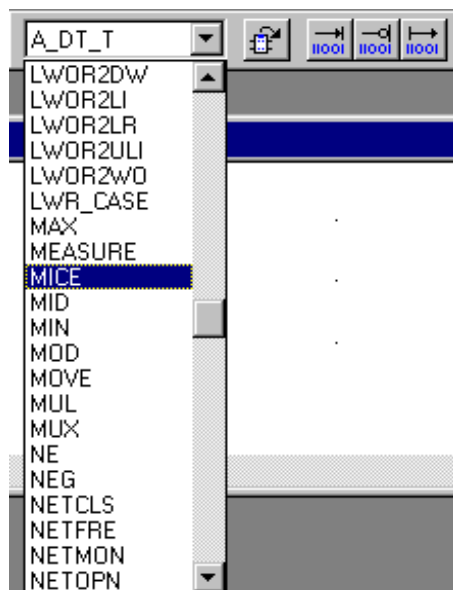
Size of UDFB Libraries

You may have one or multiple UDFB libraries. Because PiCPro copies the entire UDFB library whenever you use the **Compile | UDFB** command during program development, it is recommended that you keep the size of the UDFB library under 100K. This will improve the time it takes to build UDFBs.

The UDFB library will appear under the **Ladder | Functions | USER** menus. The individual UDFBs will appear in a flyout from the library. In the example below the function block named MICE is located in the library called CATS.



The UDFB will also appear in the list in the Function toolbar.



Data Handling in UDFBs

Data is passed to the UDFB from its inputs in one of two ways depending on its type.

- If the internal input is any variable other than a structure, string, or an array, then the value of the external input is passed to the UDFB. This value is acted upon internally in the UDFB leaving the external variable unchanged.
- If the internal input is a structure, string, or an array, then a pointer is passed to the UDFB. This pointer gives the location of the associated data in the calling program. The data is acted upon outside the UDFB.

IMPORTANT

Data is copied and pointers are passed into a UDFB only when there is power flow to the first input (EN). *String, array, or structure pointers within a UDFB must not be accessed until power flow to the EN has occurred.*

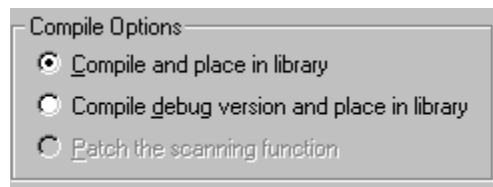
Compiling the UDFB

When you have entered all your logic for the UDFB in your ladder, marked all the Inputs and Outputs that you want to appear in the UDFB template, and saved the module, you can compile the UDFB and place it in a library. Use the **Compile | UDFB** command.

Note: If you are recompiling a function that was created earlier, you must place the function in the same library where it was originally located. You cannot select a new library location. The **UDFB Library Name:** box will be disabled (grayed out).

Editing a UDFB

You can edit the UDFB either on-line or off-line depending on the types of changes you are making. PicPro will determine this and make the appropriate options available in the **Compile Options** box.



Off-Line Editing

When you are doing off-line editing, the first two choices are available.

The **Compile and place in library** option changes the module in the library and requires a full download. If you add any function/blocks and/or declarations that require initializations, you must compile the UDFB and do a full download. The scan is stopped.

The **Compile debug version and place in library** option also changes the module in the library and requires a full download with the scan stopped. But it allows you to create a debug version of the UDFB. Creating a debug version adds additional data bits (40), data bytes (80), and function/jump links (20) to each instantiation of the UDFB for more extensive on-line changes.

Note: Minor changes that do not add things like new functions, declarations, or jump labels can be made on-line without creating a debug version.

On-Line Editing

When you are doing on-line editing, the third choice becomes available.

The **Patch the scanning function** option allows you to make an on-line edit change to this UDFB. This does not change the version of the module in the library, but does allow you to continue to animate and do on-line editing.

Viewing a UDFB

When you are working in the ladder in which you have entered your UDFB (called the parent ladder), you can view the UDFB ladder by choosing **View | UDFB/Task** or by right clicking on the function in the ladder and then selecting **View**.

Viewing the Parent Ladder

When you are working in the UDFB ladder, you can view the parent ladder by choosing **View | Parent** or by right clicking anywhere in the ladder and selecting **View | Parent Ladder**.

UDFB Software Declarations

Variables of any type with optional initial values may be declared for use in the UDFB module. However, variables cannot be external, retained, global, or discrete I/O.

Note: If you need to add discrete I/O in order to test the UDFB module, be sure to remove them from the module and the software declarations table before compiling the UDFB.


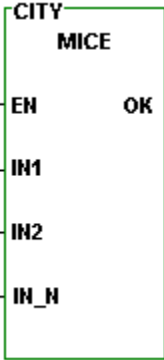
The variables which will become the inputs for the UDFB may be any data type except function blocks.

The variables which will become the outputs for the UDFB may be any data type except function blocks, structures, arrays, and strings.

Inputs that are structures, arrays, and strings all pass pointers to the UDFB and, therefore, cannot be used as outputs. It is possible, however, to design a function block in which the result of the function block being executed ends up in a structure, array, or string used as an input.

Naming the Variables

Up to the first four characters you assign to the UDFB input and output variables in the software declarations table will become the labels in the UDFB template.

Software Declarations Entries				Example UDFB Template	
					
Name	Type	Attribute			
EN	BOOL	I			
OK	BOOL	O			
IN1	INT	I			
IN2	INT	I			
IN_N	INT	I			
end list	void				

Order of UDFB Inputs and Outputs

The first input and the first output will always become the EN (enable) and the OK of the function block. Their data type is boolean. It is recommended that they appear before any other UDFB inputs and outputs in the software declarations table.

Additional UDFB inputs and outputs should be entered in the software declarations table in the order you want them to appear in the function block template. The inputs appear on the left side of the template and the outputs on the right.

Number of UDFB Inputs and Outputs

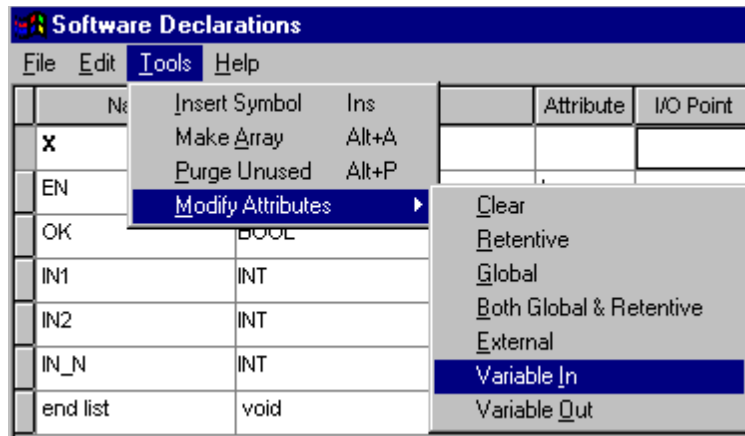
The maximum number of inputs or outputs for a UDFB is 64. It is recommended that you keep the number to a minimum (under 16). More can be declared if necessary, but transferring all the inputs and outputs to and from a function block does use scan time. There is also the constraint of the 255 element matrix to consider. If you have a large number of inputs, you may want to enter them as a structure using just one input.

Marking UDFB Inputs and Outputs

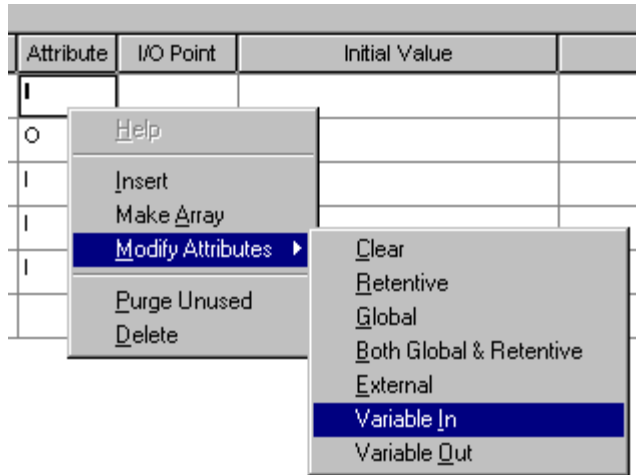
You must mark all the inputs and outputs for the function block in the software declarations table using the attribute tool. When you compile your UDFB ladder, these inputs and outputs will appear on the UDFB template in the order you have entered them. Remember that the EN input and the OK output must be entered in the software declarations table before any other UDFB inputs or outputs. Their data type is boolean.

To mark a variable as an input or output in the software declarations table:

1. Place the focus anywhere in the row of the software declarations table that holds the variable you want to mark.
2. From the **Tools** menu choose **Modify Attributes | Variable In** or **Variable Out**



or right click your mouse to bring up the following,



or with focus on a cell in the attribute column, type I or O.

All of these methods will enter the **I** or **O** symbol in the Attribute column.

Tasks

A task is a programming tool that allows you to create a module (.LDO) that will execute at periodic intervals or on the rising or falling edge of a boolean variable from within your main LDO. This provides the ability to schedule events from your main LDO.

WARNING

Use discretion when programming tasks in your LDO. You must have a thorough understanding of how they work and what effect they will have on your program to ensure that your program will execute properly.

Tasks are programmed similar to UDFBs. You convert the task LDO into a function block that is stored in a library. Whenever PiCPro is executed, the task will be available through this library under the Functions menu. It can then be used in a network of your main LDO.

Comparing UDFBs and Tasks

There are some differences between tasks and UDFBs.

- Tasks can access I/O modules located in the main rack of the control. UDFBs cannot.
- A task cannot be programmed within another task or within a UDFB. A UDFB can be nested within another UDFB.
- The on-line edit feature cannot be used in a network containing a task.
- Forcing is not allowed in a task.
- When you insert tasks into your main LDO, the function block template is always the same. You cannot program any inputs or outputs for a task. They are predefined.

Types of Tasks

There are three types of tasks available.

1. Servo Interrupt Tasks - triggered on the 1, 2, 4, 8, or 16 millisecond servo time tick.
2. Hardware Interrupt Tasks - triggered on a hardware event.
3. System Tick Tasks - triggered on a multiple of the 10 millisecond system tick task.

Note: Variables 44 through 48 are available to be used with servo tasks using the READ_SV and WRITE_SV functions. Refer to the Function/Function Block Reference Guide for information on these variables.

When multiple tasks are declared in the main LDO, the task that is declared first in the software declarations table is executed first.

When a task is triggered in the main LDO, the following hierarchy occurs.

- System tasks will interrupt the main LDO.
- Hardware tasks will interrupt system tasks and the main LDO.
- Servo tasks will interrupt hardware tasks, system tasks, and the main LDO.

Using the Task Template

Every task you create and subsequently use in your main LDO will use the template shown below. FILE will be replaced with the name of your task LDO file when you compile the task. You will provide a NAME when you declare the function block in the software declarations table of your main LDO.

TASK Template

NAME	
FILE	
EN	OK
SERV	FAIL
HDWR	ERR
SYST	

Description

Inputs:

EN (BOOL) - enables execution

SERV (TIME) - 1, 2, 4, 8, or 16ms constant for servo task. Enter in the T# format.

HDWR (BOOL) - I/O point used to trigger the hardware interrupt task must be programmed as Data In or Data Inverted which inverts a boolean input. Never program a wire or contact to this BOOL input.

SYST (TIME) - constant or variable based on 10ms system time tick. If a time is entered that is not a multiple of 10, it will be rounded up to the next 10ms increment. Time must be less than 10.92 minutes. Enter in the T# format.

Outputs:

OK (BOOL) - set if EN is on and the TASK is successfully installed. Does not indicate whether or not the task has run.

FAIL (BOOL) - set if ERR is not equal to zero.

ERR (INT) - Zero if no error, non-zero if an error.

Structured Text

```
<<INSTANCE NAME>>:FILE(EN:= <<BOOL>>,
SERV:= <<TIME>>, HDWR:= <<BOOL>>, SYST:= <<TIME>>,
OK => <<BOOL>>; FAIL => <<BOOL>>, ERR => <<INT>>;
```

The EN must be on for a task to run. The SERV, HDWR, and SYST inputs are mutually exclusive. Only one can be connected at a time depending on whether your task is a servo, hardware, or system interrupt task.

Errs at the ERR output of the task

The errs that can appear at the ERR output are:

ERR #	Description
1	Task installation failure
2	The time tick requested is larger than 10.92 minutes.
3	The servo task is faster than the servo interrupt time.

Using Functions from the I/O and Motion Libraries

There are some rules that apply to functions from the standard I/O and Motion libraries when working with tasks.

I/O Functions

Functions in these groups that access an I/O module must be called in the main LDO or within the same task.

ANLGIN
STEPPER

ANLGOUT

JKTHERM

READFDBK

RTDTEMP

I/O functions in these groups can be called in the main LDO only.

COMM

NETWORK

These I/O functions can be called in both the main LDO and the task LDO.

BAT_OK?
IPHOSTID
IPRECV

PID
IPIP2NAM
IPSEND

IPACCEPT
IPLISTEN
IPSOCK

IPCLOSE
IPNAM2IP
IPWRITE

IPCONN
IPREAD

Motion Functions

These motion functions can only be called in the main LDO.

STRTSERV
SC_START
SCA_REF
SCS_REF

CAM_OUT
SCA_SEND
SCA_ERST
SCA_PBIT

MEASURE
SCA_RECV
SCS_SEND
SCA_RFIT

REGIST
SCA_CLOS
SCS_RECV

SCURVE
SCA_ACKR
SCS_ACKR

These motion functions may not interrupt each other, i.e. if one function is accessing a queue, a second function is not allowed to interrupt the first function and access the same queue. If this occurs, the function in the interrupting task will not execute and the OK will not be set.

DISTANCE
RATIOCAM
RATIO_RL
IN_POS?
Q_AVAIL?
PART_CLR

POSITION
RATIOPRO
REP_END
NEWRATIO
Q_NUMBER
PART_REF

VEL_END
RATIOSLP
SYN_END
NEW_RATE
FAST_QUE

VEL_STRT
RATIOSYN
FAST_REF
ABRTALL
C_RESET

GR_END
RATIO_GR
LAD_REF
ABRTMOVE
E_RESET

These motion functions can be called in both the main LDO and any task LDO.

ACC_DEC
COORD2RL
E_STOP?
P_RESET
REF_END
R_PERCEN
SCA_RCYC

CAPTINIT
C_ERRORS
HOLD
RATIOSCL
TME_ERR?
SCR_CONT
SCS_STAT

CAPTSTAT
C_STOP
HOLD_END
READ_SV
TUNE_READ
SCR_ERR
SCA_WCYC

CLOSLOOP
E_ERRORS
OPENLOOP
READ_SV
TUNE_WRITE
SCR_PHASE
RESUME

CLSLOOP?
E_STOP
P_ERRORS
REF_DNE?
WRITE_SV
SCS_CTRL
RESMODE?

The STATUSSV motion function can only be called once because it does a read and then clear of all but one status flag. After an event occurs, only the first read of the status will recognize the corresponding flag.

Comparing Declaration Rules for Main and Task LDOs

There are rules for making hardware and software declarations in the main and task LDOs.

Hardware Declarations

Description	in Main LDO	in Task LDO
Hardware module used in task LDO	Yes	Yes
Hardware module used in main LDO	Yes	No
Hardware module used in both LDOs	Yes	Yes

Software Declarations

Description	in Main LDO	in Task LDO
Hardware input used in task LDO	No	Yes
Hardware input used in main LDO	Yes	No
Hardware input used in both LDOs	Yes	Yes
Hardware output used in task LDO	No	Yes
Hardware output used in main LDO	Yes	No
Hardware output used in both LDOs	Not allowed	
Variable used in task LDO	No	Yes
Variable used in main LDO	Yes	No
Variable used in both LDOs	Yes	Yes (Mark External*)

*Never mark with the External attribute in software declarations any direct I/O used in the LDO.

Hardware modules and I/O points may be used in multiple task LDOs. They must be declared in all task LDOs they are used in.

Creating a Task

Briefly, you will do the following to create and use a task.

1. Open a new ladder diagram and enter ladder logic which will accomplish what you want the task to do.
2. Mark as External any variable in the software declarations table of the task LDO that will be used in the task LDO and in the main LDO.
3. Compile the task.
4. Open another ladder diagram which will be the main LDO where you want to run the task(s). Insert the task from the Function list into a network. It will have to be declared in the software declarations table and given a name at that time.
5. Set up the type of interrupt (servo, hardware, or system) you want and when it will run.

Before you begin to create a task program or use tasks in your main LDO, keep in mind the following:

- Tasks can access I/O located in the main rack of the control. Declare any hardware module whose I/O you use in a task in both the main LDO and in the task LDO hardware declarations table. (In the main LDO, all hardware modules must be declared whether the main LDO uses them or not. In the task LDO, only the hardware modules used in the task LDO must be declared in the task LDO.
- Hardware outputs used in a task may not be used in the main LDO. Hardware inputs may be used in a task and in the main LDO. Hardware inputs and outputs used in one task may be used in other task(s). Declare them only in the software declarations table of the LDO(s) they will be used in.
- Inputs are strobed at the beginning of a task. Outputs are strobed at the end of a task.
Note: When the task is running, only the inputs and outputs in the task, not the inputs and outputs in the main LDO, will be strobed.
- Do not use timer functions in a task.
- Tasks can be animated but cannot be forced or patched.
- Tasks cannot be programmed within another task or within a UDFB.
- If software data information in a task needs to be shared with other tasks, declare it as External in the software declarations table of the task LDO. This data must also be declared in the software declarations table of the main LDO whether the main LDO uses it or not. In the main LDO, it is *never* marked as External or Global.
Note: Never assign the External attribute to direct I/O used in the LDO.
- To ensure that the data to be transferred between LDOs is from the same scan, interlock multiple externals, arrays, structures, and variables. This can be done with semaphore flags.

- Hardware interrupts can come from the first input in each group of eight on the digital input modules (numbers X.1, X.9, X.17, or X.25) or from the fast input on the encoder/resolver modules. The fast inputs for a PiC CPU must be numbered as X.1 for channel one, X.3 for channel two, X.5 for channel three, and X.7 for channel four. X represents the slot number for the module. The fast inputs for a standalone MMC must be numbered IFAUX.1 for channel one, IFAUX.2 for channel two, IFAUX.3 for channel three, and IFAUX.4 for channel four. The fast inputs for an MMC for PC must be numbered IFAUX#.1, IFAUX#.2, IFAUX#.3, and IFAUX#.4, where # represents the ASIU #.
- Servo interrupt tasks put the heaviest burden on system resources. Always try to conserve system resources by using the appropriate task, i.e., do not use a servo task when a system or hardware task can accomplish the same thing.

Naming the Task

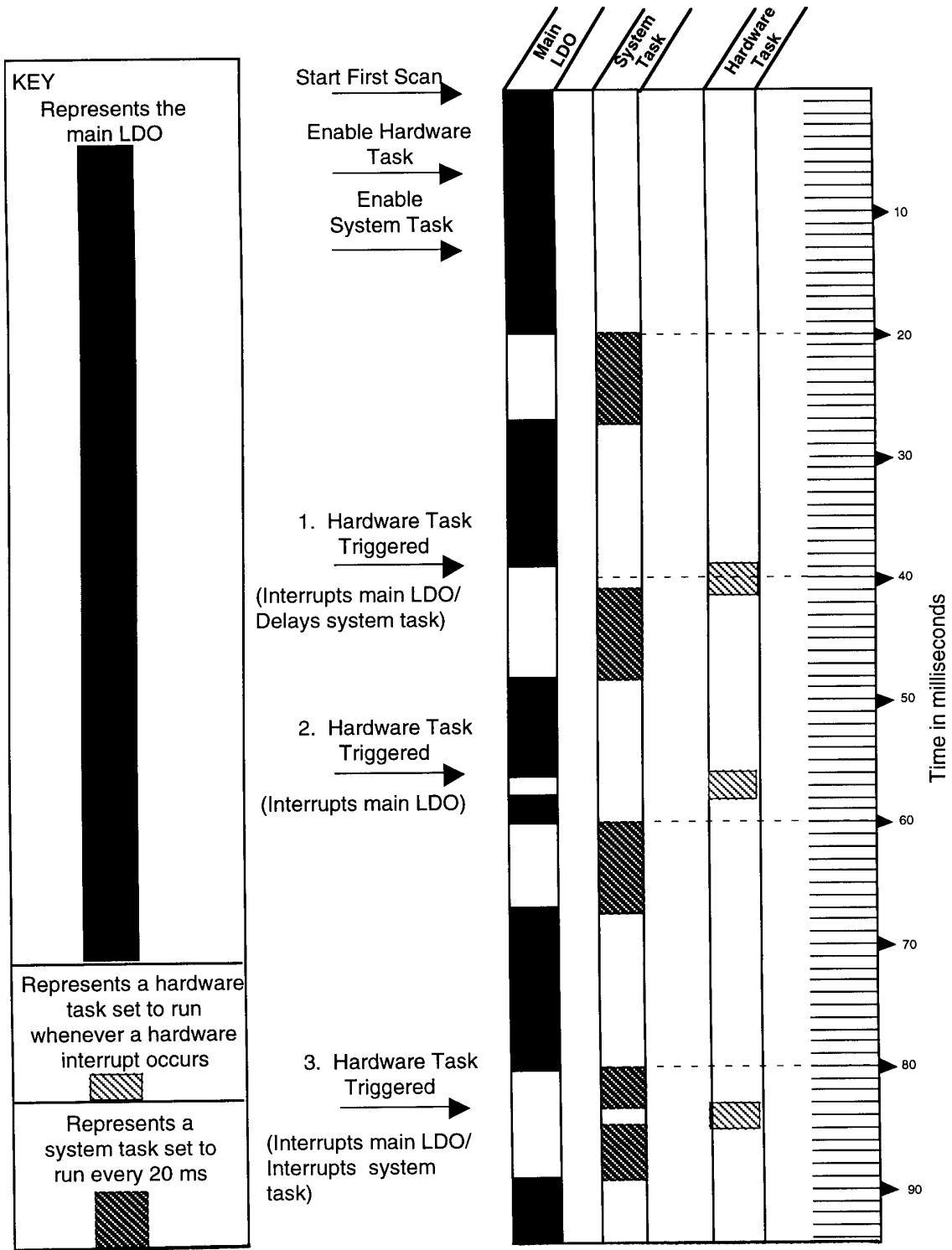
The name of the LDO module will become the name of the task when you compile it. If you have not saved the module previously, you will be prompted to enter the name at compile time. If the name is more than eight characters, it will be truncated. Be sure to choose a name that does not already exist in another module you may want to convert to a task or in a standard function/function block supplied by PiCPro.

Task Hierarchy

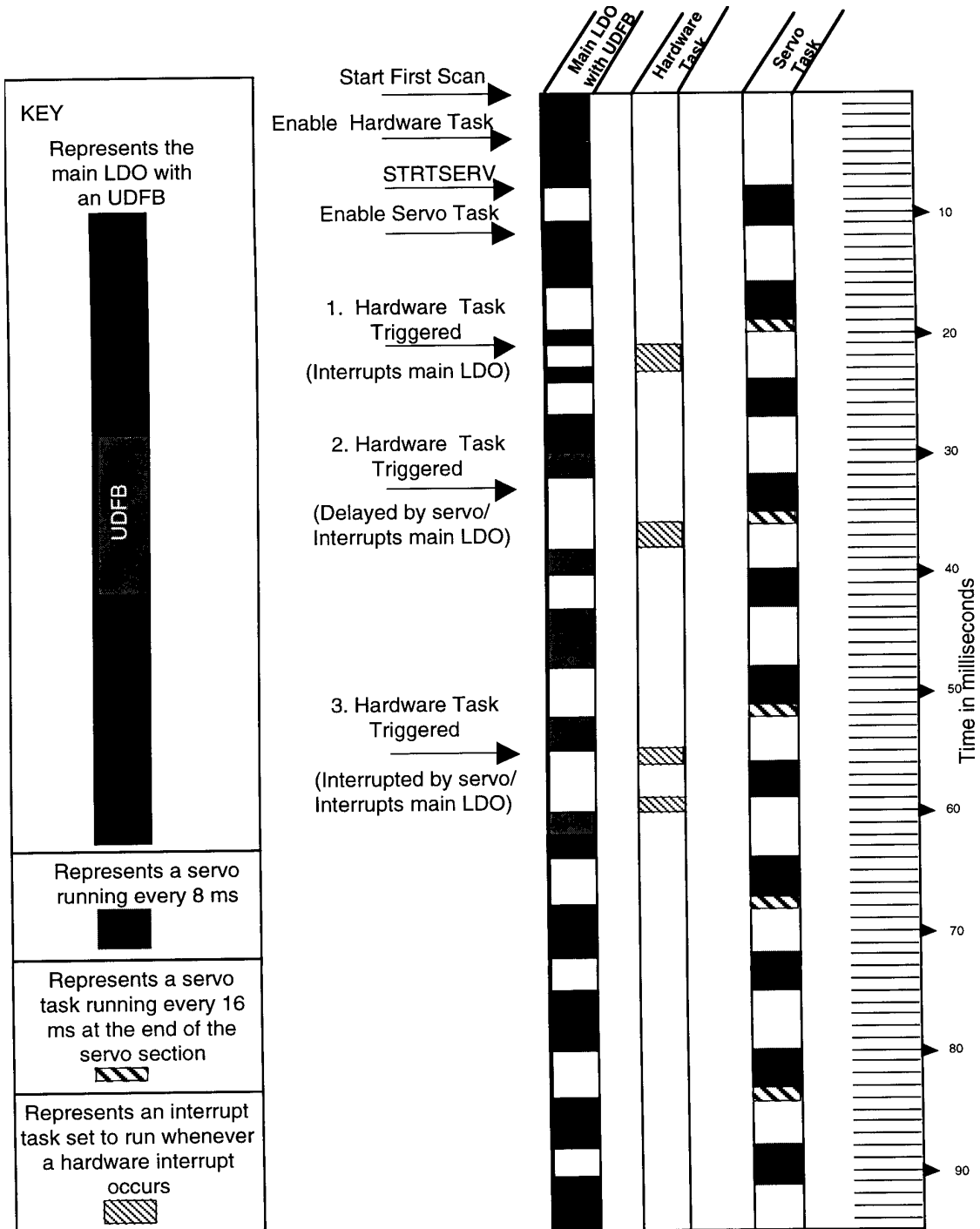
When tasks are triggered in the main LDO, system tasks will interrupt the main LDO; hardware tasks will interrupt system tasks and the main LDO; and servo tasks will interrupt hardware tasks, system tasks, and the main LDO.

If there are multiple tasks of one type in the main LDO, the task that is declared first in the software declarations table will execute first.

Priority Between the Main LDO, Hardware Tasks, and System Tasks



Priority Between the Main LDO, Hardware Tasks, and Servo Tasks



Interlocking Data in Tasks

Whenever data is exchanged between the main LDO and a task LDO, there is a possibility of inaccurate data being transferred. This is dependent on when a task interrupts the main LDO or interrupts a lower priority task accessing the same data.

In order to preserve the integrity of the data that is being read/written, you need to program an interlock between the main LDO and the task LDO. One method of interlocking data is to use semaphore flags. A semaphore flag refers to a way of communicating between the main LDO and a task LDO when a significant event has occurred.

Setting up Semaphore Flags

In setting up semaphore flags, the main LDO energizes a LOCK flag which prevents the task LDO from reading or writing data while data is being transferred to or from the main LDO. After the data transfer is complete, the main LDO de-energizes the LOCK flag and the task LDO is allowed to read or write the data.

There are two examples that follow. The first shows data being transferred from the task LDO to the main LDO. The second shows the data being transferred from the main LDO to the task LDO. In both examples the main LDO has control of the data even though tasks of higher priority are accessing it.

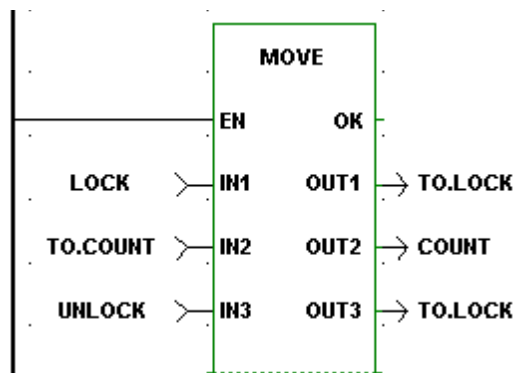
Semaphore Flag Example 1 - Data Transfer from Task LDO to Main LDO

The MOVE function with a task output TO data structure is used in both the main and task LDO. The TO structure acts as a buffer for the data transfer. The TO structure is marked External (only in the task LDO software declarations table, not the main LDO software declarations table) allowing that data to be used by both the main and task LDOs.

In the Main LDO

When the MOVE function in the main LDO is enabled, the following sequence of events occurs.

1. TO.LOCK is energized by moving LOCK into TO.LOCK.
2. TO.COUNT is moved into COUNT.
3. TO.LOCK is de-energized by moving UNLOCK into TO.LOCK.



The variables in the software declarations table of the main LDO are:

Variable	Definition
TO	Task output structure
.LOCK	Semaphore flag
.COUNT	Count variable to be written by the task LDO
COUNT	Main LDO count variable
LOCK	IN1 to MOVE function, set to 1
UNLOCK	IN3 to MOVE function, set to 0

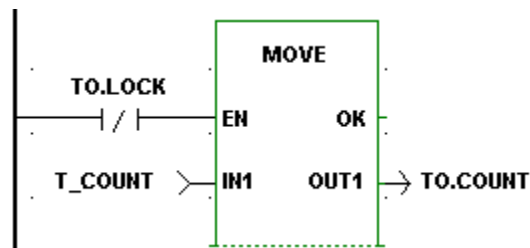
The declarations in the main LDO software declarations table are:

Software Declarations					
File Edit Tools Help					
Name	Type	A.	I/O Point	Initial Value	
LOCK	BOOL			1	
COUNT	DINT				
UNLOCK	BOOL			0	
TO	STRUCT				
.LOCK	BOOL				
.COUNT	DINT				
	END_STRUCT				
end list	void				

In the Task LDO

In the task LDO, the following sequence of events occurs.

1. If TO.LOCK is off, then move the data (Step 2).
If TO.LOCK is on, do not move the data.
2. T_COUNT is moved into TO.COUNT.



The variables in the software declarations table of the task LDO are:

Variable	Definition
TO	Task output structure
.LOCK	Semaphore flag
.COUNT	Count variable to write to the main LDO
T_COUNT	Task variable to be transferred into COUNT in main LDO

The declarations in the task LDO software declarations table are:

Software Declarations			
File Edit Tools Help			
	Name	Type	A.
	TO	STRUCT	E
	.LOCK	BOOL	
	.COUNT	DINT	
		END_STRUCT	
	T_COUNT	DINT	
	end list	void	

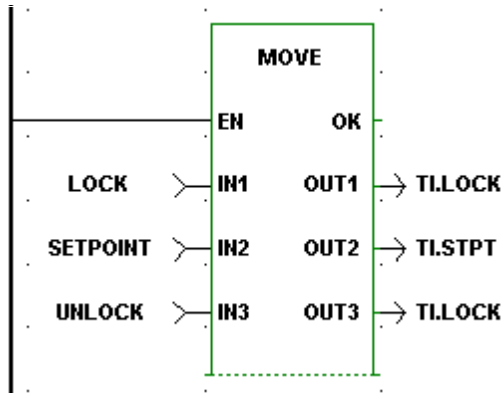
Semaphore Flag Example 2 - Data Transfer from Main LDO to Task LDO

The MOVE function with a task input TI data structure is used in both the main and task LDO. The TI structure acts as a buffer for the data transfer. The TI structure acts as a buffer for the data transfer. The TI structure is marked External (only in the task LDO software declarations table, not the main LDO software declarations table) allowing that data to be used by both the main and task LDOs.

In the Main LDO

When the MOVE function in the main LDO is enabled, the following sequence of events occurs.

1. TI.LOCK is energized by moving LOCK into TI.LOCK.
2. SETPOINT is moved into TI.STPT.
3. TI.LOCK is de-energized by moving UNLOCK into TI.LOCK.



The variables in the software declarations table of the main LDO are:

Variable	Definition
TI	Task input structure
.LOCK	Semaphore flag
.COUNT	Setpoint to be read by the task LDO
SETPOINT	Main LDO setpoint
LOCK	IN1 to MOVE function, set to 1
UNLOCK	IN3 to MOVE function, set to 0

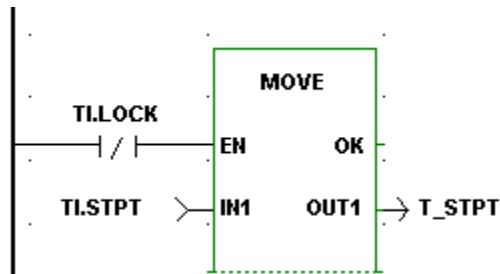
The declarations in the main LDO software declarations table are:

Software Declarations					
File Edit Tools Help					
Name	Type	A.	I/O Point	Initial Value	
TI	STRUCT				
.LOCK	BOOL				
.STPT	DINT				
	END_STRUCT				
SETPOINT	DINT				
LOCK	BOOL			1	
UNLOCK	BOOL			0	
end list	void				

In the Task LDO

In the task LDO, the following sequence of events occurs.

1. If TI.LOCK is off, then move the data (Step 2).
If TI.LOCK is on, do not move the data.
2. TI.STPT is moved into T_STPT.



The variables in the software declarations table of the main LDO are:

Variable	Definition
TI	Task input structure
.LOCK	Semaphore flag
.STPT	Setpoint to be read by the task LDO
T_STPT	Main LDO setpoint

The declarations in the main LDO software declarations table are:

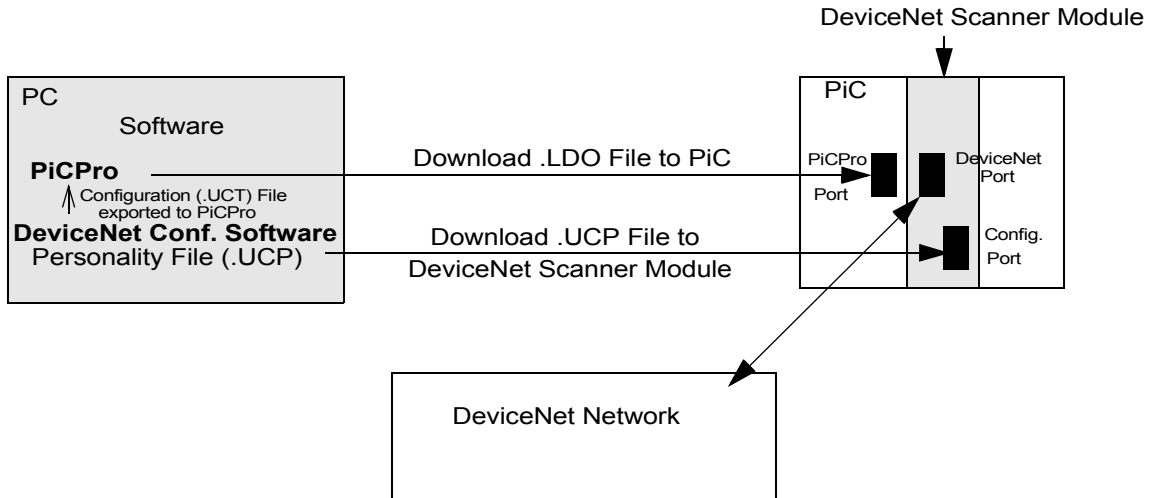
Software Declarations		
File Edit Tools Help		
Name	Type	A.
TI	STRUCT	E
.LOCK	BOOL	
.STPT	DINT	
	END_STRUCT	
T_STPT	DINT	
end list	void	

NOTES

CHAPTER 7 DeviceNet - Ethernet - TCP/IP

G&L DeviceNet Configuration Software

The diagram below illustrates a typical DeviceNet set up.



Overview of setting up a DeviceNet Network

1. Use the G&L DeviceNet Configuration Software to provide information about your DeviceNet system. This information includes a description of input and output data associated with each node, tags (variable names) for this data, and the address of each node. (See procedure that follows this section.)
2. Always use the “export” command of the G&L DeviceNet Configuration Software to generate two files in your PC:
 - a. current configuration file (.UCT) for PiCPro.
 - b. current personality file (.UCP) for the DeviceNet scanner module.

For PiC and standalone MMC:

3. Use the G&L DeviceNet Configuration Software to download the personality file to the DeviceNet scanner module. Be sure the PiCPro cable is connected to the Configuration Port.*
4. Design your ladder program using PiCPro (refer to the fieldbus function/function blocks in the Function/Function Block Reference Guide and the Theory of Operation section for your DeviceNet Scanner Module in the appropriate G&L Hardware Manual).

Typically DeviceNet communication is opened with an FB_OPN function block. After this is done FB_STA is used to continually monitor for possible error conditions and to update the online (ONLI) status flag. Once the network is online the FB_SND and FB_RCV functions are used to transfer variables to and from the memory image in the DeviceNet Scanner Module. The DeviceNet Scanner Module in turn transfers this memory image from and to actual I/O hardware associated with the tagged variables.

Once a DeviceNet network is properly online, only severe electrical noise or a hardware error will cause the network to go offline. Detection of a negative transition of the online status flag (from FB_STA) can be used as a warning that data from and to the DeviceNet network may no longer be correct. The value of each device status byte (described at the end of this chapter) can serve as a similar warning. In applications where DeviceNet data affects safety, these warnings should be used by the ladder for taking appropriate action.

5. Use PiCPro to download the ladder (.LDO) file to the PiC or standalone MMC. Be sure the PiCPro cable is connected to the PiCPro Port.*
6. Connect the DeviceNet cable to the DeviceNet Port on the DeviceNet scanner module.

***Note:** Before downloading the personality file, make sure the ladder scan in the CPU is stopped. The PiCPro cable from your PC is first used to download the personality file; this cable must be connected to the Configuration Port on the DeviceNet scanner module. You then need to connect this cable to the PiCPro Port on the CPU module in order to download the ladder.

For MMC for PC:

Use the PiCPro Download Hex feature to load the personality file via local or remote Ethernet. There are no cables or ports to connect directly. However, the ladder scan in the MMC for PC must be stopped before downloading.

Procedure for using the G&L DeviceNet Configuration Software

There are several ways to accomplish the following. Once you become familiar with the operations, you can use whichever method you prefer. **Note:** Additional information can be obtained by selecting **H**elp from the main menu or by using the **H**elp button in a given dialog box.

1. Select the G&L DeviceNet Configuration Software icon to get to the application window.
2. Select **F**ile | **N**ew to bring up your working screen. A window will appear with the heading GLDNCFG1. This is the default name for your configuration files. It should be renamed when saving the file to match the base name of your .LDO file.
3. Select the **N**etwork (Network Configuration) symbol and right click to bring up a menu. Select **P**roperties to bring up the **N**ode **P**roperties menu. This allows you to change the name and description of your DeviceNet system. It is recommended that you use the name of your configuration file. Select **O**K to exit this menu.
4. If the symbol for the G&L DeviceNet scanner module is not already displayed, right click to bring up another menu and select **N**ew. This will display the symbol for the scanner module. Right click on this symbol to bring up a menu and select **P**roperties. A menu with five tabs appears (only the first three are presently used). Enter appropriate information.

- **General TAB**
Enter a name for the scanner module or leave it as the default name Scanner1. Check “Assign network parameters” in order to program the following:
Always leave MAC ID at 0.
Select appropriate baud rate (125K bps, 250K bps, or 500K bps).
Leave the Scan Interval at the default setting of 0 for the fastest update rate possible for this scanner.
Select **Apply**.
- **I/O TAB**
(Not used for the G&L scanner module.)
- **Tag TAB**
You will come back to this tab after the other DeviceNet nodes are entered.
For now, select **OK**.

5. With the G&L Scanner symbol highlighted, right click to bring up a menu and select **N**ew. This brings up a **Select New Node** menu.
Select **Device** and select **OK** to enter a Device or node symbol.
Right click on this to bring up a menu and select **P**roperties.
A menu with three tabs appears.

- **Identity TAB**
Assign a “Name” and a “Description”.
Select a MAC ID for this slave node.
Other information is optional.
- **I/O TAB**
Select either “Polled” or “Strobed”.
If Polled is selected, enter the number of “Input” and/or “Output” bytes associated with the node. Leave Update Interval blank.
If Strobed is selected, enter the number of bytes input/booleans associated with the node. (Strobed outputs are not supported, COS and Cyclic are not supported.)
Select **Apply**.
- **Tag TAB**
Select **N**ew and then **E**dit. Another menu with two tabs appears.

Names Tab

Assign a Tag Name for an object (variable) in the DeviceNet node. Use the same name as used when defining the variable in your ladder with PiCPro.

Tag TAB

Select the appropriate I/O Type and select a Data Type to match the data type for that variable in your ladder.

Select **OK** to go back one level and select **OK** to get to the node symbol.

6. Repeat Step 5 for each DeviceNet slave node in your network.
7. After all nodes are entered, select the G&L Scanner symbol to highlight it.
Right click to bring up the **Node Properties** menu again.
Now select the **Tags** Tab.
Select **N**ew and then **E**dit. A **Tag Properties** menu appears allowing you to

assign a variable name for the status of each node you defined.

Under the **Tag** Tab, select **Device x Status** where x is the MAC ID of the desired node. Also, select **WORD** for Data Type. A description of the Device Status Word is shown at the end of this topic.

8. After assigning a device status variable for each node, go back two levels. Select the **G&L Scanner** symbol. Right click to bring up a menu. Select **Export** to bring up the **Save As** menu. Use the same base file name and the same directory as used for your ladder. When you Save, several files will be saved with the same base name but with different extensions. One of these files (.UCT) will be used by PiCPro when downloading your ladder. Another file (.UCP) will be used by this G&L DeviceNet Configuration Software when downloading the personality file to the DeviceNet scanner module.
9. For PiC and standalone MMC CPUs, after exiting the **Save As** menu, right click on the G&L Scanner symbol to bring up a menu. This time select **Download**. You must have the PiCPro cable connected between your PC and the configuration port on the DeviceNet scanner module. Select the **Download** button and wait for the operation to be completed.

Note: If the Download dialog indicates the Baud Rate as “Not Connected”, first check for proper cabling. Then, be sure the ladder scan in the CPU is stopped.

For the MMC for PC CPUs, use PiCPro Professional Edition, click **Online | Download Hex**. Enter the name of the personality file (.ucp) and select the port as MMC for PC. More information on this can be found in Chapter 3.

10. Return to the application screen and save your file before exiting.

Note: The online configuration option and EDS files are not supported in the G&L DeviceNet Configuration Software.

Device Status Word

A 16-bit device status word for each slave device can optionally be tagged. (Refer back to step 7 of the procedure for using the G&L DeviceNet Configuration Software.) The word has the following format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(Reserved)							Idle	STATUS							

STATUS -	Device status code (see next table)
IDLE -	Received idle, the device is configured to send one or more bytes of input data but is currently sending zero-length (idle) input messages

Device Status Code

Status	Description	Status	Description
00h	(Reserved)	0Dh	Invalid I/O connection 1 input size
01h	Device idle (not being scanned)	0Eh	Error reading I/O connection 1 input size
02h	Device being scanned	0Fh	Invalid I/O connection 1 output size
03h	Device timed-out	10h	Error reading I/O connection 1 output size
04h	UCMM connection error	11h	Invalid I/O connection 2 input size
05h	Master/Slave connection set is busy	12h	Error reading I/O connection 2 input size
06h	Error allocating Master/Slave connection set	13h	Invalid I/O connection 2 output size
07h	Invalid vendor id	14h	Error reading I/O connection 2 output size
08h	Error reading vendor id	15h	Error setting I/O connection 1 packet
09h	Invalid device type	16h	Error setting I/O connection 2 packet
0Ah	Error reading device type	17h	M/S connection set sync fault
0Bh	Invalid product code	18h	Error setting Production Inhibit Time
0Ch	Error reading product code	19h-FFh	(Reserved)

Ethernet - TCP/IP Configurator

This section covers the configuration of an Ethernet network for PiC and stand-alone MMC. This procedure is not needed when using an MMC for PC.

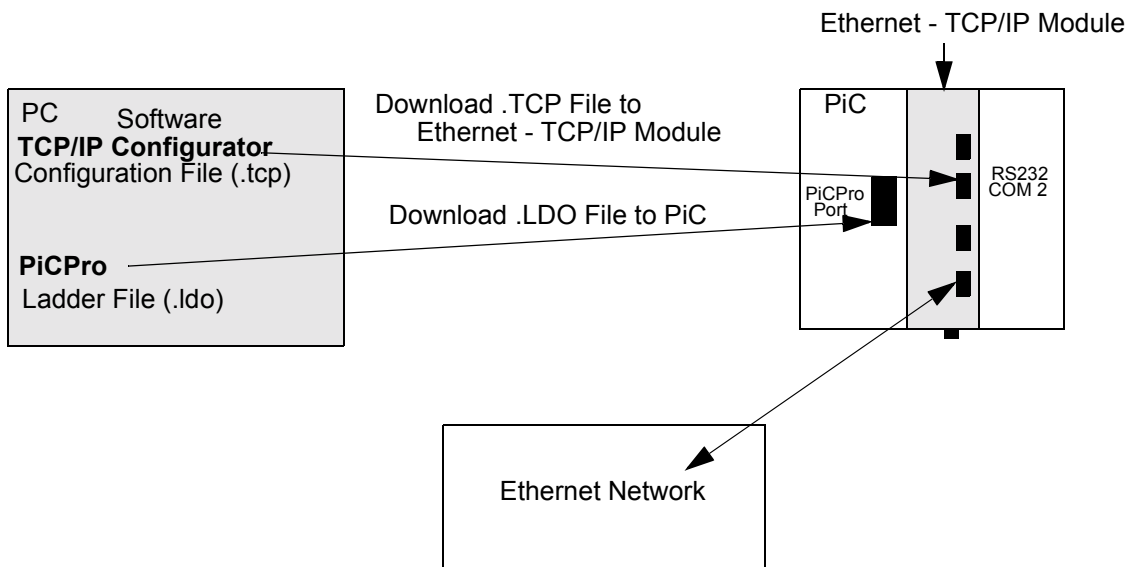
Overview of setting up an Ethernet Network

The following steps summarize what you need to do to set up an Ethernet network.

1. Use the Ethernet-TCP/IP Configurator within PiCPro to configure your system. (See procedure at the end of this section.)
2. Send the configuration information to the Ethernet - TCP/IP module using the PiCPro cable connected to the RS232 Com 2 port on the module.
3. Always cycle power after changing Ethernet settings.
4. Design your ladder program using PiCPro.
5. Download the .LDO file to the PiC using the PiCPro cable connected to the PiCPro Port.*
6. Make your connections to your ethernet - TCP/IP system.

*Note: The PiCPro cable is first used to download the configuration file by connecting it to the RS232 Com 2 port on the Ethernet - TCP/IP module. You then need to connect it to the PiCPro Port on the CPU module in order to download the ladder.

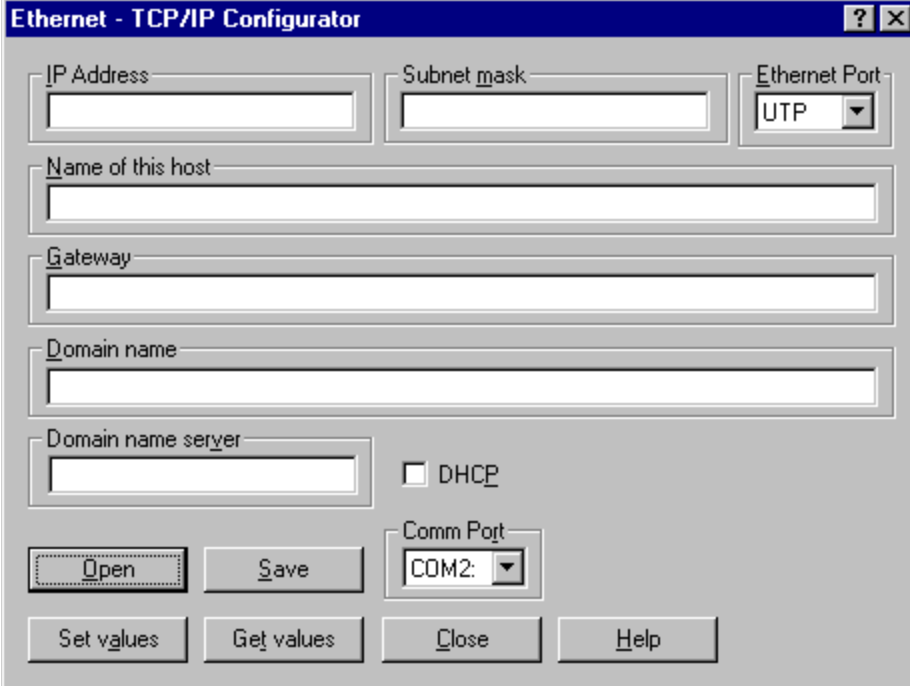
FIGURE L- 1. Block Diagram of Ethernet - TCP/IP Setup



Ethernet-TCP/IP Configuration Procedure

To set up an Ethernet – TCP/IP network, you will need to use the Ethernet – TCP/IP Configurator.

1. With PiCPro running on your PC, choose **Online | Configure TCP/IP** and the following dialog appears. If you do not have all the information needed for this dialog box, see your network administrator.



The screenshot shows the 'Ethernet - TCP/IP Configurator' dialog box. It features a title bar with a question mark and a close button. The main area contains several input fields: 'IP Address', 'Subnet mask', 'Ethernet Port' (a dropdown menu currently set to 'UTP'), 'Name of this host', 'Gateway', 'Domain name', and 'Domain name server'. There is also a checkbox labeled 'DHCP' which is currently unchecked. Below these fields are buttons for 'Open', 'Save', 'Set values', 'Get values', 'Close', and 'Help'. A 'Comm Port' dropdown menu is also visible, currently set to 'COM2'.

Animation of your program is halted when the configurator is chosen. All entries are blank except the Ethernet Port and Comm Port entries.

2. Enter the information in the following entry fields.

IP Address

Enter the IP address of the host in dotted decimal notation. **Note:** If DHCP is checked, you do not have to enter an address here.

Subnet Mask

Enter the Subnet mask in dotted decimal notation. **Note:** If DHCP is checked, you do not have to enter an address here.

Ethernet Port

There are three choices for the PiC900 Ethernet module: UTP (default), BNC, and AUI. There is one choice for the standalone MMC Ethernet module: UTP.

Name of this host

Enter the name of the PiC or standalone MMC host.

Gateway

Enter the gateway node by name if a name server is present. Otherwise, enter it by address in dotted decimal notation.

Domain name

Enter the domain name of this host.

Domain name server

Enter the address of the domain name server in dotted decimal notation.

Open

Choosing the **O**pen button allows you to open a previously saved .tcp file.

Save

Choosing the **S**ave button saves the TCP/IP configuration data to a .tcp file.

Comm Port

Enter the PC communications port you want to use from the drop down list. The list can hold up to nine ports depending on how many are available on your PC. Currently the baud rate is set at 9600.

DHCP

By checking the DHCP check box, the module asks for an IP address and other information. If this data is not available over the network, the information you entered will be used.

Set Values

Choosing the **S**et **v**alues button sends the entered data to the Ethernet-TCP/IP module via the PiCPro cable connected from the PC to the RS232 COM2 port on the module.

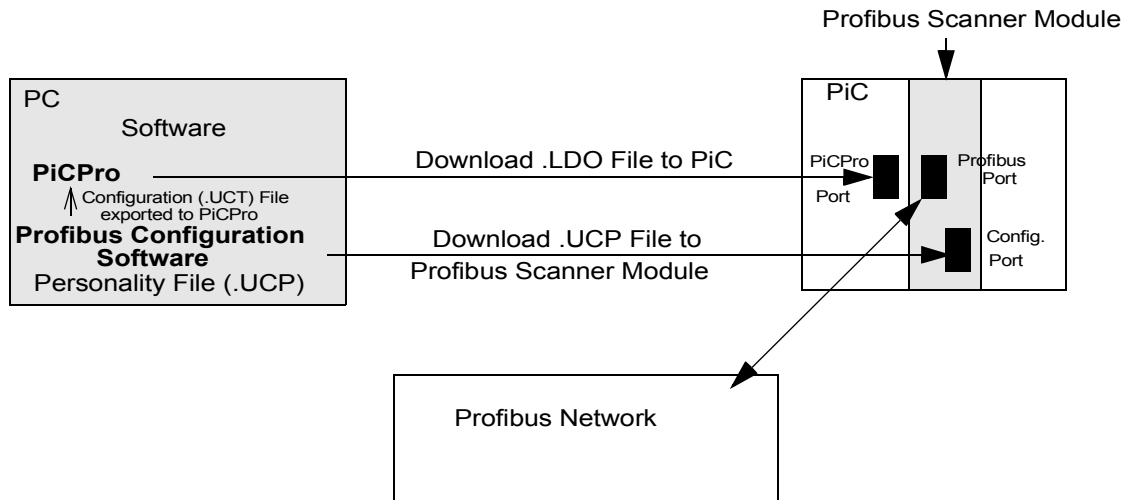
Get Values

Choosing the **G**et **v**alues button retrieves the data from the Ethernet-TCP/IP module via the PiCPro Cable connected from the PC to the RS232 COM2 port on the module. The data fills in the appropriate fields in the configurator dialog box.

3. After all the information is entered, choose **S**ave button to save your configuration data to a .tcp file.
4. Connect the PiCPro cable between the your PC and the COM2 port on the Ethernet - TCP/IP module.
5. Choose the **S**et **v**alues button to send the entered data to the Ethernet-TCP/IP module.
6. Cycle power.

CHAPTER 8 G&L Profibus Configuration Software

The diagram below illustrates a typical Profibus set up.



Overview of setting up a Profibus Network

1. Use the G&L Profibus Configuration Software to provide information about your Profibus system. This information includes a description of input and output data associated with each node, tags (variable names) for this data, and the address of each node. (See procedure that follows this section.)
2. Always use the **Export Tags** and **Export Binary** command (under the **File** menu) of the G&L Profibus Configuration Software to generate two files in your PC:
 - a. current configuration file (.UCT) for PiCPro.
 - b. current personality file (.UCP) for the Profibus scanner module.

For PiC and standalone MMC:

3. Use the G&L Profibus Configuration Software to download the personality file to the Profibus scanner module. Be sure the PiCPro cable is connected to the Configuration Port.*
4. Design your ladder program using PiCPro (refer to the fieldbus function/function blocks in the Function/Function Block Reference Guide and the Theory of Operation section for your Profibus Scanner Module in the appropriate G&L Hardware Manual).

Typically Profibus communication is opened with an FB_OPN function block. After this is done FB_STA is used to continually monitor for possible error conditions and to update the online (ONLI) status flag. Once the network is online the FB_SND and FB_RCV functions are used to transfer variables to and from the memory image in the Profibus Scanner Module. The Profibus Scanner Module in turn transfers this memory image from and to actual I/O hardware associated with the tagged variables.

Once a Profibus network is properly online, only severe electrical noise or a hardware error will cause the network to go offline. Detection of a negative transition of the online status flag (from FB_STA) can be used as a warning that data from and to the Profibus network may no longer be correct. The value of each device status byte (described at the end of this chapter) can serve as a similar warning. In applications where Profibus data affects safety, these warnings should be used by the ladder for taking appropriate action.

5. Use PiCPro to download the ladder (.LDO) file to the PiC or standalone MMC. Be sure the PiCPro cable is connected to the PiCPro Port.*
6. Connect the Profibus cable to the Profibus Port on the Profibus scanner module.

***Note:** Before downloading the personality file, make sure the ladder scan in the CPU is stopped. The PiCPro cable from your PC is first used to download the personality file; this cable must be connected to the Configuration Port on the Profibus scanner module. You then need to connect this cable to the PiCPro Port on the CPU module in order to download the ladder.

For MMC for PC:

Use the PiCPro Download Hex feature to load the personality file via local or remote Ethernet. There are no cables or ports to connect directly. However, the ladder scan in the MMC for PC must be stopped before downloading.

Procedure for using the G&L Profibus Configuration Software

There are several ways to accomplish the following. Once you become familiar with the operations, you can use the method you prefer. **Note:** additional information can be obtained by selecting **H**elp from the main menu or by using the **H**elp button in a given dialog box.

1. Select the G&L Profibus Configuration icon to get to the application window.
2. Select **F**ile | **N**ew to bring up your working screen. A window will appear with the heading New Network Configuration. Select **S**canner (to the left of G&L Profibus Sca...). The working screen will then appear with the default name GLPBCFG1. Another window will also appear prompting for scanner properties. Enter the appropriate information as follows:
 - **General TAB**
Enter a name for the scanner module or leave it as the default name Scanner. This scanner is a network master and the station address is typically left at 0.
 - **Parameters TAB**
Typically, leave Scan Cycle Times in Auto.
3. Select the **P**rofibus_ **D**P and double click to define overall network properties. A window with the heading of **N**etwork will appear with 3 tabs, as follows.
 - **General TAB**
Select the desired network baud rate. All nodes have to be capable of supporting the rate selected. The highest station address can be left at 126.

- **Timing TAB**
The timing parameters are defined in tbit units. Tbit equals 1/ baud rate. The default values are typically used.
 - **Parameters TAB**
Again, the default values for the parameters are typically used.
4. Add nodes under scanner by selecting **Slaves** in the left hand window. By drilling down, an icon and description for the desired node can be found. This feature is based upon information stored in .gsd files in a subdirectory of your choosing. You have to obtain the .gsd files from the manufacturer of the parts you wish to use. Most manufacturers have links to their sites from www.profibus.com. Once an icon is found and selected, drag and drop it under the scanner symbol. A window will appear with specific information about the node.
- **General TAB**
Assign a specific name if so desired and select a unique station address for this node in the network.
 - **Modules TAB**
If there are plug in modules associated with the node, add the description(s) for those modules.
 - **Address TAB**
Do not modify offsets. The default settings are required for the G&L applications.
 - **Tag TAB**
Select a tag name for each available variable
 - **Std. Prms TAB**
Some parts have additional information and do not require modification.
 - **Ext. Prms TAB**
Some parts have additional information and do not require modification.
 - **Diagnostics TAB**
Some parts have additional information and do not require modification.
- Continue to drag and drop icons from the .gsd files in the slave directory until all nodes have been defined for the network.
5. Select **File | Export Tags**. You will be prompted for a file name with a .uct extension. Then select **File | Export Binary**. You will be prompted for a file name with a .ucp extension.
Use the same base file name and same directory as that used for your ladder. One of these files (.uct) will be used by PiCPro when down loading your ladder. Another file (.ucp) will be used by G&L Profibus Configuration when down loading the personality file to the Profibus scanner module.

6. For PiC and standalone MMC CPUs, after exiting the **File** menu, right click on the G&L Scanner symbol to bring up a menu. This time select **Download**. You must have the PiCPro cable connected between your PC and the configuration port on the Profibus scanner module. Select the **Download** button and wait for the operation to be completed.
Note: If the Download dialog indicates the Com Port as “Not Connected”, first check for proper cabling. Then, be sure the ladder scan in the CPU is stopped.
7. For MMC for PC CPUs, use PiCPro Professional Edition, click **Online | Download Hex**. Enter the name of the personality file (.ucp) and select the port as MMC for PC. More information on this can be found in Chapter 2.
8. Return to the application screen and save your file before exiting.

Device Status

There is a status byte variable associated with each node (slave). It can be tagged with the G&L Profibus Configuration Software. When this byte is equal to zero, the node has not been detected properly by the G&L scanner module. When equal to 80h, the node is OK.

APPENDIX A - Quick Reference to In/Out Function Data

Function Input/Output Data Type Reference

This is a quick reference giving the data types of all the inputs and outputs of the functions and function blocks used in PiCPro. The input or output is on the left and the data type is on the right.

If an input/output is listed more than once, it has more than one data type depending on the function it comes from. In that case, the functions are listed below the input/output and data type. All the various IN inputs and OUT outputs are covered in tables that follow.

	CNTL..... UINT	DROPDINT
4mA0..... BOOL	COMNSTRUCT	DTST STRING
10ms..... BOOL	CONF.....STRUCT	DVSR.....same NUMERIC as DVND or DINT if DVND = TIME
100ms..... BOOL	COS REAL/LREAL	
A	CSTPBOOL	E
ACC.....LREAL	CNTL..... UINT	ELEMUINT
ACCL..... UDINT	CUBOOL	EN BOOL
ACT DINT	CV INT	ERR BOOL
PID	D	TME_ERR?, ANLG_OUT
ACT INT	DABL.....BOOL	ERR INT
IPREAD, IPRECV, IPWRITE, NETRCV, NETSND, READ, WRITE	DATA DINT	ASSIGN, CLOSE, CONFIG, COORD2RL, DELFIL, FB_STA, FRESPACE, IPACCEPT, IPCLOSE, IPCONN, IPHOSTID, IPLISTEN, IPNAM2IP, IPREAD, IPRECV, IPSEND, IPSOCK, IPWRITE, OI_SER, NETOPEN, NETRCV, NETSND, OPC_ENET, OPEN, READ, RENAME, RESUME, SCA_ACKR, SCA_CLOS, SCA_CTRL, SCA_ERST, SCA_PBIT, SCA_RCYC, SCA_RECV, SCA_REF, SCA_SEND, SCA_WCYC, SCR_ERR, SCS_ACKR, SCS_CTRL, SCS_RECV, SCS_REF, SCS_SEND, SEEK, STATUS, TUNEWRT, WRITE
ACTV.....WORD	DATA ARRAY	
ANGL..... REAL/LREAL	IO_CFG	
AXISUSINT	DATASTRUCT	
B	SCA_RECV, SCA_SEND, SCS_RECV, SCS_SEND	
BEG..... BOOL	DAY..... UINT	ERR SINT, UINT
BIPO..... BOOL	DBUF..... ARRAY, STRUCT, STRING	IO_CFG, SC_INIT
BKPR..... BOOL	DCNT..... INT	ERR USINT
BNDW UDINT	DECLUDINT	A_INCHIT, A_INCHRD, A_INMDIT, ARTCHIT, ARTDCHRD, ARTDMDIT, ATMPCHIT, ATMPCHRD, ATMPMDIT, STRTSERV, CAPTINIT, SCR_CONT, SERVOCLK
BTVL DINT	DEN..... INT	
BUFRARRAY, STRUCT, STRING	DERR..... INT	
C	DESTARRAY OF STRUCT	
CAM ARRAY OF STRUCT	DIDUSINT	
CD BOOL	DIFF(same as IN0)	
CFG.....STRING	DIM..... DINT	
CFGZ.....STRING	DIRSTRING	
CHANUSINT	DIST DINT	
CLRC..... UINT	DISTANCE, FAST_QUE	
CLSD..... BOOL	DISTUDINT	
CMD UINT	REGIST	ERRS..... WORD
CNFGSTRUCT	DVND NUMERIC or TIME	ESTP BOOL
CNTINT	DONE.....BOOL	ET TIME

F		MAST USINT	P	
FAHR.....	BOOL	MAX..... same as MIN	P.....	NUMERIC
FAIL.....	BOOL	MCND..... NUMERIC or TIME	DELETE, INSERT, MID, REPLACE	
FAST.....	USINT	MDST..... DINT	PHAS.....	USINT
FU.....	DINT	µsec..... UINT	PLUS.....	BOOL
FUNC.....	USINT	MIN..... any	PNT.....	USINT
G		MODE..... INT	PNUM.....	USINT
G.....	BOOL	MOVE..... STRUCT	PORT.....	STRING
H		MPLR..... same NUMERIC as MCND or DINT if MCND = TIME	OI_SER	
HILT.....	BOOL	MSTR..... DINT	PORT.....	UINT
HNDL.....	INT	N	IPCONN, IPSEND, OPC_ENET	
CLOSE, CONFIG, OPEN, READ, SEEK, STATUS, WRITE		N.....	POS.....	DINT
HNDL.....	UINT	NAME.....	POSN.....	ARRAY OF STRUCT
IPACCEPT, IPCLOSE, IPCONN, IPLISTEN, IPREAD, IPRECV, IPSEND, IPSOCK, IPWRITE		NAMZ.....	PRB.....	USINT
HOSZ.....	STRING	NODE.....	PRI.....	BOOL
I		NUM.....	PROD.....	same as MCND
I147.....	WORD	RATIOSCL	PROT.....	UINT
IDN.....	UINT	NUM.....	PT.....	TIME
IGNR.....	UDINT	NUM.....	PTR.....	ARRAY OF STRUCT
INDX.....	USINT	NUM2STR, STR2NUM	PV.....	INT
INPS.....	BOOL	NUM.....	Q	
INs..... (See Next Table)		REAL/LREAL	Q.....	BOOL
IPZ.....	STRING	EXP, LOG, LN	QAVL.....	BOOL
IST.....	STRUCT	NUM.....	QTY.....	DINT
J		IO_CFG, STR2USI, USIN2STR	QTY.....	USINT
JERK.....	LREAL	O	BIO_PERF, PLS	
K		OFF.....	QUE.....	USINT
K.....	NUMERIC	OFST.....	QUOT.....	same as DVND
L		OIER.....	R	
L.....	INT	OIFL.....	R.....	BOOL
LD.....	BOOL	OK.....	RACK.....	USINT
LEN.....	INT	ON.....	RATE.....	UDINT
LGTH.....	UDINT	ONCE.....	RATE.....	TIME
LOG.....	REAL/LREAL	ONLI.....	RSMD.....	BOOL
LOLT.....	BOOL	OPTN.....	RDNE.....	BOOL
LN.....	REAL/LREAL	SC_INIT, SCA_CTRL, SCS_CTRL	REAL.....	ARRAY OF STRUCT
LU.....	DINT	OPTN.....	REFD.....	DINT
M		FAST_REF, LAD_REF, RATIOCAM, RATIOSCL, RATIOSLP, RATIO_RL, SCA_REF, SCS_REF	REM.....	same as DVND
MAIN.....	STRUCT	ORG.....	REQ.....	BOOL
MAN.....	BOOL	OUTs..... (See Last Table)	RETR.....	BOOL
			REV.....	BOOL
			ROOT.....	same as SQR

RNGE.....USINT	T
RPER.....USINT	TAN..... REAL/LREAL
RPTP..... BOOL	TASK.....STRUCT
RSCD.....STRUCT	TBUF..... ARRAY, STRUCT, STRING
RSLT..... DINT	TCNT..... INT
RVAL..... BOOL	TIME..... UINT
S	TOLR.....UDINT
SDIR..... BOOL	TYPE.....USINT
SDST..... DINT	V
SEG1.....STRUCT	VALU..... INT
SERR..... UINT	VAR.....SINT
SET..... BOOL	VARS.....STRUCT
SID.....USINT	W
SIN..... REAL/LREAL	WEEK.....BOOL
SIZE..... DINT	
SIZE..... UINT	
SIZE.....USINT	
CAPTINIT	
SLOT.....USINT	
SLPE..... ARRAY OF STRUCT	
SLV..... UINT	
SPT..... DINT	
SQR..... UDINT, UINT, USINT, or constant	
SR.....STRUCT	
SRCE..... ARRAY OF STRUCT	
SRS.....STRUCT	
SSTR..... DINT	
STAT..... DWORD	
FB_STA	
STAT..... INT	
NETMON, SCA_REF, SCS_REF, STATUS	
STAT..... WORD	
SCA_STAT, SCS_STAT, STATUSSV, STEPSTAT	
STOP..... BOOL	
STR.....STRING	
STRC.....STRUCT	
STRT..... BOOL	
SUM..... same as IN1	

IN inputs

[Inputs for extensible functions are followed with (ext).]

Input	Data type	Functions
IN	any	SIZEOF
	BITWISE	NOT, ROL, ROR, SHL, SHR
	BOOL	TOF, TON, TP
	DATE	DATE2STR
	DATE_AND_TIME	CLOCK, DT2DATE, DT2STR, DT_2_TOD
	NUMERIC	ABS, NEG
	STRING	DELETE, LEFT, MID, RIGHT
	TIME	TIME2STR
	TIME_OF_DAY	TOD2STR
	same as MIN	LIMIT
IN0	NUMERIC or TIME	SUB
	any except STRUCT	SEL
IN0 (ext)	any except STRUCT	MUX
IN1	any except BOOL or STRUCT	NE
	BOOL	SCA_CTRL, SCS_CTRL
	DATE	D_TOD2DT, S_D_D
	DATE_AND_TIME	A_DT_T, S_DT_DT, S_DT_T
	STRING	FIND, INSERT, REPLACE
	TIME_OF_DAY	A_TOD_T, S_TOD_T, S_TOD_TO
	same as IN0	SEL, SUB
IN1 (ext)	any	MOVE
	any except BOOL or STRUCT	EQ, GE, GT, LE, LT, MAX, MIN
	BITWISE	AND, OR, XOR
	NUMERIC or TIME	ADD
	STRING	CONCAT
	same as IN0	MUX
IN2	BOOL	SCA_CTRL, SCS_CTRL
	DATE	S_D_D
	DATE_AND_TIME	S_DT_DT
	TIME	A_DT_T, A_TOD_T, S_DT_T, S_TOD_T
	TIME_OF_DAY	D_TOD2DT, S_TOD_TO
	same as IN1	NE
	STRING	FIND, INSERT, REPLACE
IN2 (ext)	same as IN1	ADD, AND, EQ, GE, GT, LE, MAX, MIN, OR, XOR
	STRING	CONCAT, REPLACE
IN3	BOOL	SCA_CTRL, SCS_CTRL

OUT outputs

Output	Data type	Functions
OUT	BOOL	CAM_OUT, EQ, GE, GT, LE, LT, NE, PLS
	DATE	DT2DATE
	DATE_AND_TIME	A_DT_T, CLOCK, D_TOD2DT, S_DT_T
	same as IN	ABS, NEG, NOT, ROL, ROR, SHL, SHR
	same as IN0	MUX, SEL
	same as IN1	AND, OR, XOR
	same as MIN	LIMIT
	NUMERIC	FIND
	TIME	S_D_D, S_DT_DT, S_TOD_TO
	TIME_OF_DAY	A_TOD_T, DT2TOD, S_TOD_T
OUT1	same as IN1	MAX, MIN, MOVE
OUT----OUT	STRING	CONCAT, DATE2STR, DELETE, DT2STR, INSERT, LEFT, MID, REPLACE, RIGHT, TIME2STR, TOD2STR,

NOTE

There are also INs/OUTs on all the data type conversion functions. In those functions, the IN is always the data type you are converting from and the OUT is always the data type you are converting to.

NOTES

APPENDIX B - Errors

Function Error Codes

The categories of function errors are:

- General function errors
- I/O block function block errors
- String function errors
- Stepper errors
- Start servo function errors
- Capture initialization function errors

General Function Errors

For all functions, the output variables will have unpredictable values and the output at OK, DONE, or Q will not be energized whenever the following occurs.

- An output variable does not have enough bits to hold the result.
- An output variable is an unsigned integer and the result is negative.
- The operation attempts to divide a number by zero.
- The input data is invalid.

I/O Function Block Error Codes

If an error occurs when the I/O functions execute, the following occurs.

- The output at DONE is not energized.
- The output at FAIL is energized.
- The output at ERR holds one of error numbers listed below.

Error #	Error Description
01	OPCODE_ERROR Invalid device open mode specified for function (e.g. 16#602)
02	handle_error The handle to a function is invalid. Possible reasons include: <ul style="list-style-type: none">▪ Using I/O functions with an unopened handle▪ Attempting to do a READ on a handle opened for WRITE ONLY▪ Attempting to do a WRITE to a handle opened for READ ONLY
03	already_open Reserved for Giddings & Lewis. Internal GLOS error
04	open_error OPEN mode (READ/WRITE/APPEND) specified is invalid.
05	device_empty Reserved for Giddings & Lewis

06	<p>read_error</p> <p>Depending on the device you are using, this error will be one of those listed below.</p> <ul style="list-style-type: none"> ▪ DISK driver detects a checksum error on the DISK. Media error on DISK reading disk FCB (File Control Block). ▪ A parity, overrun, or framing error exists in the data in the input buffer of the COM port.
07	<p>write_error</p> <p>Error writing to the device</p>
08	<p>write_protect</p> <p>Reserved for Giddings & Lewis</p>
09	<p>device_error</p> <p>Caused by any of the following:</p> <ul style="list-style-type: none"> ▪ The Seek origin is not A00, A01, or A02. ▪ Improper device name (Choices are PICPRO:, RAMDISK:, FMDSISK:, or USER:) ▪ Attempting to do a STATUS on a file ▪ Attempting to do a file operation on a non-DOS device ▪ Bad parity in the configuration string or a hardware error in configuring the port ▪ Attempting to do a second or subsequent operation before the first one was completed
10	<p>out_of_handles</p> <p>Too many files open. A maximum of 10 handles can be used. READ/WRITE or APPEND operations use two handles. READ ONLY or WRITE ONLY operations use one handle.</p>
11	<p>invalid_device</p> <p>Improper device name (Choices are PICPRO:, RAMDISK:, FMDSISK:, USER:, or any name you have assigned at the NAMZ input of a function block)</p>
12	<p>invalid_file_name</p> <p>Filename format is wrong.</p> <p>Filename is too large.</p>
13	<p>too_many_drivers</p> <p>Reserved for Giddings & Lewis. Internal GLOS error</p>
14	<p>too_many_connections</p> <p>Reserved for Giddings & Lewis. Internal GLOS error</p>
15	<p>read_write_error</p> <p>Error reading or writing a disk file</p>

16	device_error Reserved for Giddings & Lewis
17	file_not_open Attempting a READ, WRITE, or SEEK on a file that is not open.
18	invalid_access Caused by any of the following: <ul style="list-style-type: none"> ▪ Attempting to CLOSE a file that was not open ▪ SEEK has a problem seeking a new location. ▪ Attempting to write to a file opened for read only
19	file_not_in_dir Caused by one of the following: <ul style="list-style-type: none"> ▪ Filename not found in directory ▪ Volume name not found in directory
20	file_already_open Attempting to OPEN a file that is already open
21	write_protect Error occurs when attempting to delete a READ ONLY file from DISK.
22	MODE_ERROR Caused by one of the following: <ul style="list-style-type: none"> ▪ Invalid mode input to OPEN ▪ Invalid mode input to SEEK
23	OUT_OF_HANDLES A maximum of 10 handles are available in the PiC. This error means there are no more handles available (too many files open).
24	end_of_dir No more directory entries
25	function_locked Reserved for Giddings & Lewis. Internal GLOS error.
26	rename_new_exists Reserved for Giddings & Lewis
27	rename_old_open Reserved for Giddings & Lewis
28	rename_no_old Reserved for Giddings & Lewis
29	attrib_invalid Reserved for Giddings & Lewis. Internal GLOS error.
30	handle_too_large Reserved for Giddings & Lewis

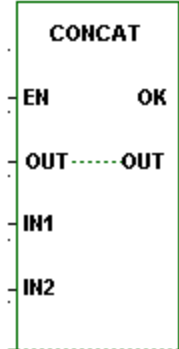
31	disk_error General error on disk drive
32	device_parity_error Reserved for Giddings & Lewis
33	end_of_devices Device driver for the device specified not found
34	Io_dev_aborted Reserved for Giddings & Lewis
35	duplicate_filename Reserved for Giddings & Lewis
36	no_memory Reserved for Giddings & Lewis
37	no_buffers No memory to get more file buffers
38	dir_not_empty Reserved for Giddings & Lewis
39	dir_not_found Reserved for Giddings & Lewis
40	out_of_fats No more entries available in the FAT (File Attribute Table). Reduce the number of files in this directory.
41	sdir_exists Reserved for Giddings & Lewis
42	filespec_too_long Reserved for Giddings & Lewis
43	no_default_device Reserved for Giddings & Lewis
44	parameter_error Occurs if the value entered at the CNT input of the READ function block is larger than the declared size of a string used as the BUFR input.

String Function Errors

If an error occurs when a string function executes the following occurs.

- The output at OK is not energized.
- The STRING variable output will be null (have a length of zero).

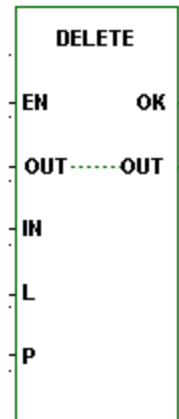
CONCAT Function



An error occurs if

The length of IN1 > the length of OUT
 The length of IN2 > the length of OUT
 The length of IN1 + the length of IN2 > the length of OUT
 IN2, IN3, IN4...IN17 = OUT

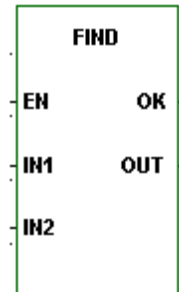
DELETE Function



An error occurs if

P = 0
 P > 255
 P > length of IN
 L > 255
 The length of IN - L > the length of OUT

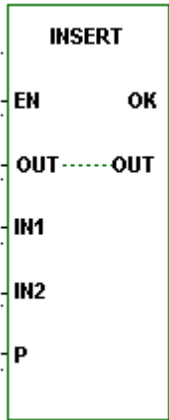
FIND Function



An error occurs if

The length of IN1 = 0
 The length of IN2 = 0
 The length of IN2 > the length of IN1

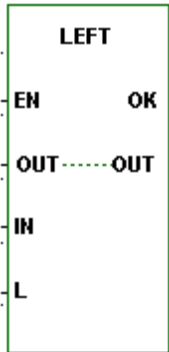
INSERT Function



An error occurs if

- P = 0
 - P > 255
 - P > length of IN
 - IN2 = OUT
- The length of IN1 + the length of IN2 > the length of OUT

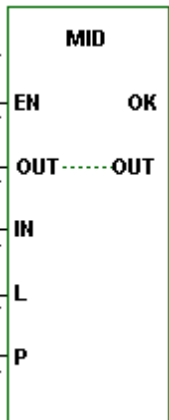
LEFT Function



An error occurs if

- L > 255
- L > the length of OUT

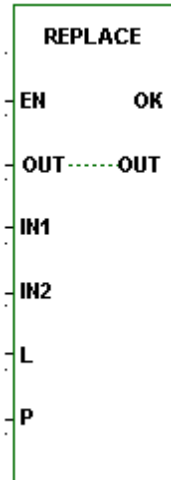
MID Function



An error occurs if

- P = 0
- P > 255
- P > length of IN
- L > 255
- L > the length of OUT

REPLACE Function

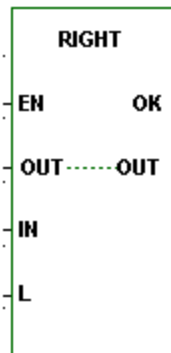


An error occurs if

- P = 0
- P > 255
- P > length of IN1
- L > 255
- IN1 = OUT
- IN2 = OUT

The length of IN1 + the length of IN2 > the length of OUT

RIGHT Function



An error occurs if

- L > OUT
- L > 255

Servo C-Stop, E-Stop, and Programming Error Codes

There are four types of errors that can occur when working with servo control. The first three apply to individual axes. The fourth, timing errors, is connected to the entire system.

1. C-stop (controlled-stop) errors
2. E-stop (emergency stop) errors
3. P (programming) errors
4. Timing errors

Servo C (controlled) - stop errors

When a C-stop (Controlled-stop) error occurs on an individual servo axis, the following happens:

- The axis remains in servo lock and the axis is brought to a controlled stop at the rate specified by the controlled stop ramp in servo setup.
- The active and next queues are cleared.
- The FAST_QUE mode is canceled when the C-stop is reset.

Bit Location (low byte)	Error Description	Hex* Value (decimal) (in LDO)
8	Part reference error Move was in progress when a part reference or a part clear function was called.	8080 (32896)
7	Part reference dimension error When the dimension for the part reference was converted to feedback units, it was too big to fit into 29 bits.	8040 (32832)
6	Distance or position move dimension error When the dimension for the move was converted to feedback units, it was too big to fit into 31 bits.	8020 (32800)
5	Feedrate error** When the feedrate for the move was converted to feedback units per servo update, it was too big to fit into 32 bits or it exceeded the velocity limit entered in setup. Note: This error can occur with feedrate override, new feedrate, position, distance, velocity, or machine reference moves.	8010 (32784)
4	Machine reference error When the dimension for the machine reference was converted to feedback units, it was too big to fit into 29 bits.	8008 (32776)
3	User-defined C-stop When this bit is set, a user-defined C-stop has occurred.	8004 (32772)
2	Negative software limit exceeded The command position exceeded the user-defined negative software end limit.	8002 (32770)
1	Positive software limit exceeded The command position exceeded the user defined positive software end limit.	8001 (32769)

*When more than one error occurs, the hex values are OR'd. For example, if 8001 and 8004 occur, the result is 8005 hex (32773 decimal).

**This error can occur with feedrate override, new feedrate, position, distance, velocity, or machine reference moves.

Servo E (emergency) - stop errors

When a E-stop (Emergency-stop) error occurs on an individual servo axis, the following happens:

- The system is out of servo lock.
- A zero voltage is sent to the analog outputs.
- The active and next queues are cleared.
- The FAST_QUE mode is canceled when the E-stop is reset.

EXCEPTION:

If a User-Set or Excess Error E-Stop occurs while Resumable E-Stop Allow is set (Servo Setup or WRITE_SV/READ_SV Variable 63), a Resumable E-Stop will occur and the following happens:

- The system is out of servo lock.
- A zero voltage is sent to the analog outputs.
- The moves in the active and next queues remain intact.
- The axis' Normal Interpolator remains running.
- The axis goes into Resume Mode. In Resume Mode, the axis will follow the Resume Interpolator. The Resume Interpolator will output zero velocity until the RESUME function is called. The RESUME function can be called after the Resumable E-Stop has been reset and the servo loop has been closed. The axis remains in Resume Mode until the RESUME function brings it back onto the Normal Interpolator's path or until a non-resumable E-Stop cancels Resume Mode.

Bit Location (low byte)	Error Description	Hex * Value (decimal)
8, 7	(Not Used)	in LDO-
6	SERCOS error Cyclic data synchronization error	8020 (32800)
5	SERCOS error SERCOS drive E-stop - Status word bits 15, 14, and 13 not equal to 1 1 0 respectively.	8010 (32784)
4	User-set An E-stop on a servo axis has occurred which was called in the ladder using the ESTOP function.	8008 (32776)
3	Overflow error A slave delta overflow during runtime has occurred. This problem is most likely to occur if you are moving at a high rate of speed and/or the slave distance is very large compared to the master distance. There are two conditions that can set this bit. 1 In FU, if the master moved position times the slave distance entered is greater than 31 bits. 2 In FU, if the master moved times the SDIS divided by the MDIS > 16 bits.	8004 (32772)

2	Excess error When an excess following error has occurred, the axis has exceeded the limit entered in the Servo setup program as the following error limit. This represents the maximum distance the commanded axis position can be from the actual axis position.	8002 (32770)
1	Loss of feedback A loss of feedback from the feedback device has occurred. Available for servo and digitizing axes.	8001 (32769)

* When more than one error occurs, the hex values are OR'd. For example, if 8001 and 8004 occur, the result is 8005 hex (32773 decimal).

Servo P (programming) - errors

P- (Programming) errors occur during master/slave moves or a FAST_QUE call. P-errors may prevent:

- The move from being placed in the queue (or if the move is in the queue, abort the move)
- or
- The OK on the function from being set

Bit Location (low byte)	Error Description	Hex* Value (decimal) in LDO
8	The FAST axis in the FAST_QUE function moved too far in the wrong direction The axis traveled more than 65,535 FU in the opposite direction of the value entered in DIST of the FAST_QUE function.	8080 (32896)
7	Profile number not found Data for a profile move is not valid.	8040 (32832)
6	Master axis not available This error can occur when using the FAST_QUE function or the functions for master/slave moves (RATIO_GR, RATIOSYN, or RATIO-PRO). The conditions that can set this bit include: 1 Master axis or fast axis not initialized 2 Interrupt rates different for axes 3 Axis at slave input is the same as axis at master input in master/slave moves	8020 (32800)
5, 4, 3, 2	Not Used	---
1	Master start position for lock on When the dimension for the lock position was converted to feedback units, it was too big to fit into 32 bits.	8001 (32769)

Bit Location (high byte)	Error Description	Hex* Value (decimal) in LDO
8	A programming error has occurred. There must be at least one other bit set in the word.	8000 (32768)
7, 6, 5	(Not Used)	---
4	Master axis beyond start point The master axis is beyond its starting point for a ratio synchronization (RATIOSYN) move.	8800 (34816)
3	Slave axis beyond start point The slave axis is beyond its starting point for a ratio synchronization (RATIOSYN) move.	8400 (33792)
2	Master distance not valid When the master distance is converted to feedback units, it is greater than 16 bits.	8200 (33280)
1	Slave distance not valid When the slave distance is converted to feedback units, it is greater than 16 bits.	8100 (33024)

*When more than one error occurs, the hex values are OR'd. For example, if 8100 and 8200 occur, the result is 8300 hex (33536 decimal).

Servo Timing Errors

All the servo calculations for one interrupt must be completed in the time frame selected by you in servo setup before the next interrupt begins. If they are not completed, a timing error occurs. The timing error is connected to the entire system. This error is monitored in the ladder program with the TME_ERR? function. If the boolean output at ERR is set, a timing error is occurring. Depending on the system, this can affect performance.

IMPORTANT

Always set an E-Stop on all axes when a timing error occurs.

SERCOS Error Codes

Ring Errors

The ring errors listed below appear at the ERR output of the SCR_ERR function and will appear on the Ring Error State line in the Ring State dialog box.

ERR#	Description	What to do/check
3	Axis is not initialized, is not a SERCOS axis, or the slot/ring/slave specification is incorrect.	<ul style="list-style-type: none"> ▪ SERCOS board in correct slot ▪ SR structure members correct
17	The SERCOS module did not receive an expected AT response. Cable could be disconnected.	<ul style="list-style-type: none"> ▪ Check connection
20	Phase 0 detected that the ring is not complete.	<ul style="list-style-type: none"> ▪ Check connection ▪ Ensure drive is turned on
65	Error occurred calculating when MDT should occur.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ One or more drives cannot accommodate required MDT
66	Error occurred calculating when drive data valid.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ One or more drives cannot accommodate command times
67	Error occurred calculating when feedback data valid.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ One or more drives cannot accommodate feedback capture times
68	Error occurred calculating total time required for communication cycle	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long ▪ Update rate too fast
69	Error occurred calculating cyclic data memory for SERCON processor.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
70	Error occurred calculating cyclic data memory for internal memory map.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
71	Error occurred calculating service channel memory map.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long
74	CPU on SERCOS module has too many tasks during update. Too many slaves on one ring.	<ul style="list-style-type: none"> ▪ Too many slaves on one ring ▪ Cyclic data on slaves too long

128	Slave error occurred. Read SERR output to identify error. The SLV output indicates the slave number.	<ul style="list-style-type: none"> ▪ SLV output contains slave number ▪ IDN output contains the IDN transfer that caused the error ▪ SERR output contains the drive generated error number ▪ Read Drive diagnostic IDN 95
136	Individual slave will not respond. The SLV output indicates the slave number.	<ul style="list-style-type: none"> ▪ Address switch on drive does not match slave number ▪ Baud rate switch on drive does not match rate in ring definition ▪ SLV output contains slave number that does not respond
144	Individual slave cannot carry out a Procedure Command Function. The SLV output indicates the slave number.	<ul style="list-style-type: none"> ▪ SLV output contains slave number ▪ IDN output contains the Procedure Command Function that caused the error ▪ For IDN = 127, read IDN 22 to read list of IDNs still required by the drive ▪ For IDN = 128, read IDN 23 to read list of IDNs still required by the drive ▪ Read Drive diagnostic IDN 95

Slave Errors

The slave errors listed below appear at the SERR output of certain slave SERCOS functions and will appear on the Slave Error line in the Ring State dialog box.

SERR #	Description
4097	This IDN does not exist.
4105	The data for this IDN may not be accessed.
8193	The name does not exist.
8194	The name transmission is too short.
8195	The name transmission is too long.
8196	The name may not be changed.
8197	The name is write-protected.
12290	The attribute transmission is too short.
12291	The attribute transmission is too long.
12292	The attribute is write-protected at this time.
16385	The units do not exist.
16386	The units transmission is too short.
16387	The units transmission is too long.
16388	The units may not be changed.
16389	The units are write-protected at this time.
20481	The minimum value does not exist.
20482	The minimum value transmission is too short.
20483	The minimum value transmission is too long.
20484	The minimum value may not be changed.
20485	The minimum value is write-protected.
24577	The maximum value does not exist.
24578	The maximum value transmission is too short.
24579	The maximum value transmission is too long.
24580	The maximum value may not be changed.
24581	The maximum value is write-protected.
28674	The data is too short.
28675	The data is too long.
28676	The data may not be changed.
28677	The data is write-protected at this time.
28678	The data is smaller than the minimum value.
28679	The data is larger than the maximum value.
28680	The bit pattern for this IDN is invalid.

TCP/IP Error Codes

The following errors can be reported out of the ERR output on the IPXXXX function blocks.

ERR	Description	ERR#	Description
0	No error	40	Destination address required
1	Not owner	41	Protocol wrong type for socket
2	No such file or directory	42	Protocol not available
3	No such process	43	Protocol not supported
4	Interrupted system call	44	Socket type not supported
5	I/O error	45	Operation not supported on socket
6	No such device or address	46	Protocol family not supported
7	Arg list too long	47	Address family not supported
8	Exec format error	48	Address already in use
9	Bad file number	49	Can't assign requested address
10	No children	50	Socket operation on non-socket
11	No more process	51	Network is unreachable
12	Not enough core	52	Network dropped connection on reset
13	Permission denied	53	Software caused connection abort
14	Bad address	54	Connection reset by peer
15	Directory not empty	55	No buffer space available
16	Mount device busy	56	Socket is already connected
17	File exists	57	Socket is not connected
18	Cross-device link	58	Can't send after socket shutdown
19	No such device	59	Too many references: can't splice
20	Not a directory	60	Connection timed out
21	Is a directory	61	Connection refused
22	Invalid argument	62	Network is down
23	File table overflow	63	Text file busy
24	Too many files open	64	Too many levels of symbolic links
25	Not a typewriter	65	No route to host
26	File name too long	66	Block device required
27	File too large	67	Host is down
28	No space left on device	68	Operation now in progress
29	Illegal seek	69	Operation already in progress
30	Read-only file system	70	Operation would block
31	Too many links	71	Function not implemented
32	Broken pipe	72	Operation cancelled
33	Resource deadlock avoided	1000	There is a non-zero terminated string that requires zero termination, or there is a zero length string.
34	No locks available	1001	There is a CNT input that is too large
35	Unsupported value	1002	The SLOT number requested does not contain an Ethernet board
36	Message size	1003	Either the firmware does not support TCP/IP or there is no Ethernet board in the rack
37	Argument too large	1004	The IPZ buffer is too small
38	Result is too large	1005	TCP/IP stack failure. Socket no longer valid.
10000 and up	Any error code above 10000 when using these functions blocks was generated by Microsoft. Please check their documentation.		

Compile Error Messages (Ladder)

If the compile process is unsuccessful, an error(s) will be reported in the information window. Typically, you must re-edit the ladder to correct the errors and recompile.

These errors can be the following types:

1. **Fatal** - indicates a severe problem that prevents the compile from being completed.
2. **Error** - indicates a program syntax error.
3. **Warning** - provides an informational message. A warning does not prevent the compile command from being completed, but it is recommended that the situation that caused the warning be corrected.

If a compile error does occur, you can double click on it in the information window and it will navigate you to the place in the ladder where the error occurred. You can get help on an error by single clicking on the error in the information window and pressing <F1>.

1000 BINARY: BUILD

A file open exception has occurred. Error opening file ____.
PiCPro has attempted to open the indicated file (usually a BIN or temporary file) and was unsuccessful.

Tip

Ask these questions:

- Is the disk drive full?
- Is the disk drive write protected?
- Is the TEMP/TMP environment variable pointing to a valid drive?

1002 BINARY: OUT OF MEMORY

Out of PiC application memory space.

The ladder code that is being compiled requires more memory than is available in the processor specified in the hardware declarations.

1003 BINARY: FUNCTION NOT FOUND

____ not found in libraries.

There is a function required by your application that cannot be found.

Tip

- Check to make sure your library paths are correct.

1004 BINARY: ARRAY MISMATCH

External data typing error, array mismatch, __ task __.

The indicated external variable declared in the indicated task was found in the main module, but one is declared as an array and the other is not. They cannot be different.

Tip

Either declare both as an array or not.

1005 BINARY: EXTERNAL UNDEFINED

External __ in task __ has no source in the main module.

A variable marked with the external attribute in the software declarations table in the task LDO was not declared in the main LDO as required.

Tip

Any variable used in a task that has been marked as external must also be declared in the main LDO whether or not the main LDO uses it. It must never be marked as external in the main LDO, only in the task LDO.

1006 BINARY: TYPE MISMATCH

External data typing error, _ in task _, _.

The indicated external variable in the indicated task was found in the main module, but the variable types are different as indicated. The types must match.

Tip

Be sure that the variable type of the external variable in a task has been declared in the main LDO with the same variable type.

1007 BINARY: OUT OF PATCHES

More than 100 patches. A scan stopped, full module download is required.

The PiC provides for 100 patches which have been used up.

Tip

- Do a full download to incorporate all the existing patches in the application.
- After a full download, the patch area is again available for another 100 patches.

As you work with on-line edit, check the resources available summary in the information window to see how many patches you have left.

1008 BINARY: TOO MANY PATCHES

Too many patch operations at once. A scan stopped, full module download is required.

There is a limit of 40 internal operations in any one patch download, depending on memory available. Every time a network is modified it is considered one patch, but it could include several internal operations. This error simply means you are trying to do too much at one time.

Tip

Do a full download and proceed.

1009 BUILDER: NAME CHANGE

Windows filename _ changed to _ in the PiC.

The PiC only supports the original DOS 8.3 filename format. The filename supplied does not comply with that format and has been modified as indicated for use in the PiC. This is an informational message.

Tip

To avoid this error always name your files using the DOS format of a maximum of 8 characters, followed by a period, then 3 characters.

1010 OLE: HARDWARE CHANGED

Hardware declarations have changed.

Changes to existing hardware declarations will prevent a patch download.

Tip

If you make changes to the hardware declarations table, you must perform a compile and download before attempting to patch the ladder.

1011 OLE: UDFB BIT MEMORY

Out of function block bit memory. A remake of this library function is required, along with a full compile and download.

There is at least one new BOOLEAN variable for which there is no room reserved in the PiC data memory. Either a debug version of the UDFB is not being used, or there have already been 40 additional bits used since the last remake and full download.

Tips

- Remake the UDFB to incorporate any new variables and download the entire application.

or

- Delete the additional variable(s) and all the places they are used.

Note: If you plan on adding variables you must have a debug version of the UDFB downloaded.

When working with on-line edit, check the resources available summary in the information window.

1012 OLE: UDFB BYTE MEMORY

Out of function block byte memory. A remake of this library function is required, along with a full compile and download.

There is at least one new variable for which there is no room reserved in the PiC data memory. Either a debug version of the UDFB is not being used or there have already been 80 additional bytes used since the last remake and full download.

Tips

- Remake the UDFB to incorporate any new variables and download the entire application.
- or

- Delete the additional variable(s) and all the places they are used.

Note: If you plan on adding variables you must have a debug version of the UDFB downloaded. When working with on-line edit, check the resources available summary in the information window.

1013 OLE: UDFB LINKS

Out of local label/function links.

There is at least one new function or network label for which there is no room reserved in this UDFB code memory. Either a debug version of the UDFB is not being used, or there have already been 20 additional network labels or functions used since the last remake and full download.

Tips

- Remake the UDFB to incorporate the new labels and download.
- or

- Delete the additional labels and functions.

Note: If you plan on adding variables you must have a debug version of the UDFB downloaded. When working with on-line edit, check the resources available summary in the information window.

1014 OLE: GLOBAL LINKS

Too many functions/labels. Out of global link table space.

PiCPro establishes a link table for labels and/or function/blocks added during on-line editing. There is a limit of 26 links. Every time you do a full download, this link area becomes available again.

Tip

Do a full download when you get this message and the link area will become available.

1015 OLE: LABEL UNDEFINED

Network label _ is undefined.

Network labels are required on Jump to Label and Jump to Subroutine commands. If you enter a label that you have not assigned to a network, you will get this error.

Tip

- Ensure that any network you want to jump to has a label assigned to it.
- If you get this error after entering a label with the jump command, make sure the label exists and/or check the spelling of the label to ensure it matches the label of the destination network.

1072 COMPILER: BUILD ERROR OBSOLETE

Displayed when the attempt is made to compile (bin, hex, task, UDFB) a ladder that specifies an obsolete CPU in Hardware Declarations.

Tip

- Edit Hardware Declarations and select proper CPU.

1073 APP: OUT OF RETAINED DATA MEMORY

Data memory has been exceeded. The limit is 24K.

Dependency List Error Messages

1016 DEPEND: FILE EXCEPTION

This error can occur when building a dependency list.

1018 DEPEND: FILE READ EXCEPTION

This error can occur when reading a file during the building of a dependency list.

1019 DEPEND: FILE WRITE EXCEPTION

This error can occur when writing a file during the building of a dependency list.

1020 DEPEND: SRVFILE NOT FOUND

The servo file could not be found.

1021 DEPEND: SCRFILE NOT FOUND

The SERCOS file could not be found.

Library Error Messages

1033 BINARY: PIC LIBRARY DIFFERENCE

Function/block LEVEL 1 version difference.

If you attempt to patch a UDFB network after completing a full download and then edit and recompile the UDFB, you will get this error.

Tips

You must perform a full download whenever you want to change a network containing a UDFB that has been edited and recompiled since the previous full download.

Communications Error Messages

Communications errors can occur whenever you are attempting to communicate with the PiC.

2000 COMM: NO CONNECTION

The workstation and the control are not communicating properly.

- Check physical connection to local PiC and/or network connections if you are connected to an ARCNET or TCIP/IP node.

2002 COMM: DIAGNOSTICS ERR

The control diagnostics indicates that a failure has occurred in either the master rack or an expansion rack. One of the hardware modules has failed to pass one of its diagnostic tests.

Record the test and error number(s) that appear for use in determining the cause of the failure.

It is not advisable to start your system before replacing/repairing the defective module.

2003 COMM: TIMEOUT

The control did not respond to a transmission from the workstation.

- Check the physical connection to the local PiC and/or the network connections if you are connected to an ARCNET or TCP/IP node.
- Check for a scan loss on the target PiC.

This error usually means that there is something in the PiC preventing the command from executing. Examples include:

- The key switch is turned off.
- No valid ladder is loaded in the control.
- No RAMDISK is present.

2005 COMM: OBSOLETE EPROM

The EPROMs in the PiC are not current for this version of PiCPro.

Contact Giddings & Lewis to receive updated EPROMs.

2006 COMM: NOT A BINARY FILE

Only binary files can be restored to the control. A precursory scan of the selected file indicates that it is not a binary file format.

2007 COMM: DOWNLOAD ABORTED

An error has occurred in the PiC during the downloading process. The PiC terminates the download.

2009 COMM: STOP THE SCAN TO RESTORE

The PiC is currently scanning a program. If you initiate restoring a module, you are required to stop the scan. The program in the control will be replaced with the restored module.

2010 COMM: EMPTY FILE

You have attempted to restore an empty file to the PiC. A file that has zero file length cannot be restored to the PiC.

2017 COMM: MESSAGE TOO LARGE

A message being sent to the PiC is larger than the allowable size. Contact Giddings & Lewis if you receive this error.

2025 COMM: NO FILE DOWNLOADED

You are attempting to backup a file in the PiC and there is no file there.

2026 COMM: BAD REGISTRY

The system registry path for communications port and baud rate variables could not be located.

Defaults of COM1 and 57600 will be used. The defaults will be written to the system registry.

2027 COMM: BAD REGISTRY PORT

The system registry variable for the communications port could not be located. COM1 will be used as the default. It will be written to the system registry.

2028 COMM: BAD REGISTRY BAUDRATE

The system registry variable for the baud rate could not be located. 57600 will be used as the default. It will be written to the system registry.

2029 COMM: CANNOT OPEN REGISTRY

The system registry could not be opened. Communications port and baud rate data cannot be saved for the next session. Your system registry may be corrupted. Consult your operating system manual for the procedure to restore a registry.

2034 COMM: NO ARCNET EPROM

The EPROMs currently installed in your PiC do not support ARCNET. Contact Giddings & Lewis.

2038 COMM: ERROR CONNECTING NODE

An error occurred when attempting to connect to a node.

- Check network connections.
- Ensure that the node number is correct.

2039 COMM: GET NODE

Get the ARCNET Node from the Dialog Box. Peer-to-peer communications using twisted pair wire can be set up between PiCs whose CPUs have network hardware.

Node Ids can range from 1 to 255. 65 is reserved for the PiC physically connected to the serial port of the PC.

2042 COMM: FILE PATH TOO LONG

The file path listed is too long for the PiC. The operation is canceled.

File paths on the RAM and FMS disks are limited to 127 characters.

2046 COMM: COPY ABORTED

There has been a problem transferring a file between the control and the PC. The PC received an abort message from the PiC. Copy file has been aborted.

2047 COMM: SEND ABORTED

There has been a problem transferring a file between the PiC and the PC. The PC received an abort message from the PiC. Send file has been aborted.

2048 COMM: FILE TOO BIG

Your disk is full and the operation cannot be completed.

Delete unused files to free up space or add more memory to your system.

2050 COMM: INVALID FILENAME

The filename is invalid. The PiC uses DOS 8.3 file naming convention and DOS character validation rules. Filenames can have from one to eight characters and require a three character extension. Each character must be from this list: A-Z a-z 0-9 \$%_@{}~`!#. See a DOS manual for additional requirements.

2053 COMM: UPDATE FMS

Copying files to the FMS disk requires a complete reformat of the disk. When this occurs, all existing files on the FMS disk are deleted and can no longer be retrieved or accessed.

2054 COMM: FORMAT ABORTED

The PiC did not respond to a transmission from the PC.

Check the physical connections to the local PiC and/or the network connections if connected to an ARCNET node. Check for a scan loss on the target PiC.

2055 COMM: ZERO LENGTH FILE

The selected file appears to be empty. Empty files cannot be copied to the RAM or FMS disks in the PiC.

2057 COMM: ERROR CREATING DIRECTORY

The specified directory path could not be created.

Ensure that the path is valid and that the disk is not full or write protected.

2058 COMM: NO SOURCE FILE ERROR

An entry in a dependency list file was missing the source file field at the specified line number in the file. Add a source path or remove the line from the file.

2059 COMM: INVALID COPY

The keys <Ctrl + C> initiate a Copy command. That command is invalid for the currently selected item.

2060 COMM: INVALID PASTE

The keys <Ctrl + V> initiate the Paste command. Either nothing has been previously copied or the command is invalid for the currently selected item.

2061 COMM: INVALID CUT

The keys <Ctrl + X> initiate the Cut command. That command is invalid for the currently selected item.

2062 COMM: INVALID DELETE

The <Delete> key initiates the Delete command. That command is invalid for the currently selected item.

2063 COMM: INVALID PASTE LIST

The keys <Ctrl + L> initiate a Paste List command. Either a list file has not been previously copied or the command is invalid for the currently selected item.

2064 COMM: INVALID RENAME

The keys <Ctrl + R> initiate the Rename command. That command is invalid for currently selected item.

2065 COMM: INVALID LIST

The selected dependency list file has an invalid format.

A dependency list file has the following format:

The # sign in the first position for comments or the source path; destination paths for entries.

2067 COMM: NO FLASHDISK

There is no FLASH disk available. Either the FMSDISK option has not been installed on your PiC CPU or it is defective.

2068 COMM: NO RAMDISK

There is no RAM disk available. Either the RAMDISK is not installed in your system or it is defective.

2072 COMM: FMS/RAM DISK

RAMDISK and FMSDISK are hardware options which make extra memory available to the PiC for use as a file system. These files are then accessible to the PiC whether the PC is connected or not. Refer to the COMM group of function blocks in the Reference Guide for more information.

2073 COMM: LIST FILE ERR

An entry in the dependency list file contains an invalid path.

Verify that the path is a valid path and that the file exists.

2076 COMM: TRANSFER IN PROGRESS

A new file transfer request has been received while a transfer is in progress. The I/O COMM OPEN function call in the scanning ladder is trying to open a file for background processing while a previous OPEN function currently has a file open.

Only one background file can be open at a time.

- Check your ladder logic.
- Close the open file.

2077 COMM: NO CONNECTION

The PC and the PiC are not communicating properly during an I/O COMM function call initiated by the scanning ladder.

- Check the physical connection to the local PiC.
- Check the network connections to an ARCNET node if applicable.

2078 COMM: BACKGROUND TIMEOUT

The PiC did not respond to a transmission from the PC initiated by an I/O COMM function call in the scanning ladder.

- Check the physical connection to the local PiC.
- Check the network connections to an ARCNET node if applicable.
- Check for scan loss on the target PiC.

2079 COMM: BACKGROUND RENAME

An I/O COMM RENAME function call in the scanning ladder failed.

- Check the NAMZ input variable to the function block for valid path names, disk full, or write protection conditions.

2080 COMM: BACKGROUND OPEN

An I/O COMM OPEN call in the scanning ladder failed.

- Check the MODE input to the function block for valid mode, disk full, or write protection conditions.

2081 COMM: BACKGROUND PATH

An I/O COMM OPEN call in the scanning ladder failed.

- Check the NAMZ input to the function block for a valid path name.

2082 COMM: BACKGROUND DELETE

An I/O COMM DELFIL call in the scanning ladder failed.

- Check the NAMZ input to the function block for a valid path or write protection conditions.

2083 COMM: BACKGROUND DIRECTORY

An I/O COMM OPEN write mode call in the scanning ladder failed.

- Check the MODE input to the function block for the correct mode, the NAMZ input for a valid path, disk full, or write protection conditions.

2084 COMM: BACKGROUND READ

An I/O COMM READ call in the scanning ladder failed. There is a problem with the PC operating system.

2085 COMM: BACKGROUND WRITE

An I/O COMM WRITE call in the scanning ladder failed. There is a problem with the PC operating system.

2086 COMM: ABANDON TRANSFER

An I/O COMM call in the scanning ladder to the PC is still being executed.

Terminating communication now will terminate the call before execution has completed.

2087 COMM: CHANNEL IN USE

An I/O COMM call in the scanning ladder to the PC is currently being executed. The communication channel required by the desired operation is in use by this operation. You must wait for the function block to complete execution or stop the scan.

2088 COMM: NO DESTINATION FILE

An entry in a dependency list file was missing the destination file field at the specified line number in the file. Add a destination path or remove the line from the file.

2089 COMM: COMMAND NOT EXECUTABLE

This error usually means that there is something in the PiC preventing the command from executing. Some possible causes include:

- Key switch is turned off.
- No valid ladder is loaded in the PiC.
- No RAMDISK is present.

2090 COMM: DATA NOT AVAILABLE

The requested data is not available. If you are trying to execute a disk operation, it could mean that you do not have memory installed for a RAMDISK/FLASHDISK in the PiC.

2091 COMM: DIRECTORY NOT FOUND

The specified directory cannot be found in the PiC.

2092 COMM: DATA NOT READY

The data is not ready in the PiC.

2093 COMM: DEVICE NOT READY

The specified device is not ready in the PiC.

2094 COMM: FILE IN USE

The specified file is currently in use.

2095 COMM: FILE NOT ASSIGNED

The specified file is not assigned.

2096 COMM: FILE NOT FOUND

The specified file was not found on the PiC's RAMDISK/FLASHDISK.

2097 COMM: FUNCTION NOT IMPLEMENTED

The specified function has not yet been implemented.

2098 COMM: FILE NOT OPEN

The specified file is not open in the PiC.

2099 COMM: FILE PROTECTION VIOLATION

The specified operation could not be performed on the file because it is protected.

2100 COMM: MESSAGE NOT AVAILABLE

The specified message is not available.

2101 COMM: NO DIRECTORY SPACE

There is no more room in the specified directory on the disk.

2102 COMM: PACKET NOT FOUND

Packet not found on the PiC.

2103 COMM: STORAGE MEDIA FULL

The storage media on the PiC RAMDISK/FLASHDISK is full.

2104 COMM: STATUS NOT AVAILABLE

The PiC status is not currently available.

2105 COMM: SYSTEM NOT READY

The PiC is not ready. Try again.

2106 COMM: INSUFFICIENT MEMORY SPACE

There is not enough memory in the PiC.

2107 COMM: REQUEST NOT RECOGNIZED

A request from the PC was not recognized by the PiC. The request was either invalid or was garbled in the transmission between the PC and the PiC.

2108 COMM: FIELD NOT RECOGNIZED

A message sent from the PC was not recognized by the PiC. Either an invalid message field was sent by the PC or the message from the PC to the PiC was garbled in transmission.

2109 COMM: MESSAGING ERROR

The control received an invalid message from the PC.

2110 COMM: UNSPECIFIED ERROR

This error should not occur in the normal execution of PiCPro. Contact Giddings & Lewis if this error occurs.

2113 COMM: NETWORK PATH LIMIT 70 CHARS

When copying files to RAMDISK over the network, the network source path, including the filename, cannot be greater than 70 characters.

2114 COMM: NETWORK FILENAME LIMIT 8 CHARS

When copying files to RAMDISK over the network, the source filename cannot be greater than 8 characters plus a 3-character extension.

Examples:

MY_FILE2.LDO ⇒ *valid*

MY_FILE22.LDO ⇒ *invalid because name portion of filename is greater than 8 chars (don't count the '.' when determining valid length)*

2115 COMM: NETWORK PATH LIMIT 12 CHARS

When copying files to RAMDISK over the network, the directories in the source path and the filename cannot each be greater than 12 characters including the slash. This means you are limited to 11 character directory names and 11 character filenames (where the name portion is 8 characters and the extension is 3).

Examples:

\\MY_DIRECT33\\MY_FILE2.LDO ⇒ *valid because length of directory name plus the preceding '\' is 12 (ok to be less than 12) and filename length plus the preceding '\' is 12 (ok to be less than 12)*

\\MY_DIRECTORY\\MY_FILE2.LDO ⇒ *invalid because length of directory name plus the preceding '\' is 13*

\\MY_DIRECT33\\MY_FILE22.LDO ⇒ *invalid because length of filename plus the preceding '\' is 13*

2116 COMM: NETWORK DEVICE LIMIT CHARACTERS

When copying files over the network to RAMDISK, the shared network device name cannot be greater than 6 characters (not including slashes).

Examples:

[\\SHARED\MY_FILE2.LDO](#)

⇒ *valid because length of shared directory name without the preceding '\\ is 6 (ok to be less than 6) and file-name length plus the preceding '\' is 12 (ok to be less than 12)*

[\\SHARED_DIR\MY_FILE2.LDO](#)

⇒ *invalid because length of shared directory name without the preceding '\\ is greater than 6 characters.*

Hardware Declarations Error Messages

The following errors or messages can be received when working with hardware declarations.

4000 HWD: REMOTE TO NONE

You have selected a master rack only hardware configuration. Your hardware is currently configured for remote I/O. All expansion rack information will be deleted and irretrievable.

4001 HWD: PiC9 BOARDS

Selecting a PiC9 CPU automatically puts a Servo Encoder declaration in slot 3 of the master rack and an In/Out 24V DC declaration in slot 4 of the master rack. If you currently have modules declared in these slots, they will be replaced.

4002 HWD: REMOTE TO BLOCK

You have selected a block I/O hardware configuration. Your hardware is currently configured for remote I/O. All expansion rack information will be deleted and irretrievable if you continue.

4003 HWD: BLOCK TO NONE

You have selected a master rack only hardware configuration. Your hardware is currently configured for remote I/O. All block I/O information will be deleted and irretrievable.

4004 HWD: BLOCK TO REMOTE

You have selected a remote I/O hardware configuration. Your hardware is currently configured for block I/O. All block I/O information will be deleted and irretrievable.

4006 HWD: DELETE BLOCK

When you delete a block I/O module, block I/O modules after the deleted module are shifted down and renumbered accordingly. If you want to remove the module without shifting, change the module to an Empty module.

4008 HWD: MAXIMUM RACKS EXCEEDED

The maximum number of expansion racks allowed exists. PiCPro currently supports a maximum of seven expansion racks.

4009 HWD: INSERT BLOCK WARNING

When you insert a block I/O module, all block I/O modules after the inserted module are shifted up and renumbered accordingly.

4010 HWD: MAXIMUM BLOCKS EXCEEDED

PiCPro currently supports 77 block I/O modules. You cannot execute an Insert After or Paste Insert After on block module 77.

4011 HWD: NO CLIPBOARD

The PC system clipboard could not be opened for the requested function (copy, cut). This indicates a problem with your PC. Make sure another application does not currently have the clipboard open.

4013 HWD: INVALID PASTE

The keys <Ctrl + V> initiate a Paste command. Either nothing has been previously copied or the command is invalid for the currently selected item.

4014 HWD: RESOLVER EXPANSION

There is an Input Resolver (2 or 4 ch) module on the clipboard. You cannot use either of these modules in an expansion rack.

4015 HWD: NOT BLOCK CPU

The CPU currently specified in the Master Rack does not support block I/O.

4016 HWD: NOT REMOTE CPU

The CPU currently specified in the Master Rack does not support remote I/O.

4017 HWD: LAST BLOCK NOT EMPTY

The 77th block module is not an Empty module. Performing any Insert function will remove this module.

4019 HWD: TOO MANY RACKS

This version of PiCPro supports only seven expansion racks.

The ladder you are loading currently has eight expansion racks.

The eighth expansion rack will not be included in hardware declarations. Saving a ladder in this condition will not save the eighth rack and render the hardware declarations for the eighth rack as irretrievable.

4021 HWD: INVALID COPY

The keys <Ctrl + C> initiate the Copy command. That command is invalid for the currently selected item.

4022 HWD: INVALID CUT

The keys <Ctrl + X> initiate the Cut command. That command is invalid for the currently selected item.

4023 HWD: INVALID PASTE/INSERT

The keys <Ctrl + A> initiate the Paste Insert After command. Either nothing has been previously copied or the command is invalid for the currently selected item.

4024 HWD: INVALID DELETE

The <Delete> key initiates the delete command. That command is invalid for the currently selected item.

4025 HWD: INVALID INSERT

The <Insert> key initiates the Insert After command. This command is invalid for the currently selected item.

4026 HWD: I/O CONFLICT

The current module in the specified slot of the master rack does not match the hardware declaration information which has been downloaded to the PiC.

4027 HWD: NO SOFT BIT

The binary file being downloaded to the PiC requires a CPU in the PiC which can handle software bit memory. The CPU currently declared in the hardware declarations is not capable of handling soft bit memory.

4029 HWD: CONFIRM SWITCH PIC TO STANDALONE MMC

When you switch from a PiC CPU to a standalone MMC CPU, this confirmation box will appear asking if you want to continue. If you do, slots 1 and 2 will hold modules for the chosen MMC configuration, slots 3 and 4 are emptied, and slots 5 to 13 and remote I/O are removed. You will want to check the I/O points in your software declarations table to ensure they are mapped to the correct locations. If they are not, make the necessary edits before you proceed.

4030 HWD: CONFIRM SWITCH STANDALONE MMC TO PIC

When you switch from an MMC CPU to a PiC CPU, a confirmation box will appear asking if you want to continue. If you do, slots 3 and 4 will be emptied, a CSM appears in slot 1, the chosen CPU appears in slot 2, and slots 5 to 13 are added to the master rack. The Remote I/O option is made available. You will want to check the I/O points in your software declarations table to ensure they are mapped to the correct locations. If they are not, make the necessary edits before you proceed.

4031 HWD: CONVERT INVALID MMC CPU FORMAT

If you attempt to open a ladder with a PiC CPU type in the MMC Edition of PiCPro, you can choose to convert the PiC CPU to an MMC CPU. However, all the I/O in the main and remote racks will be removed in the conversion to an MMC CPU. Software Declarations should be checked to ensure the I/O is mapped to the correct locations after the conversion. See Conversion References in this section.

4032 HWD: PASTE CONFIRM SWITCH MMC TO PIC

You are warned of the consequences of pasting the CPU type from one hardware declarations table to another with a different CPU type.

4033 HWD: PASTE CONFIRM SWITCH PIC TO MMC

You are warned of the consequences of pasting the CPU type from one hardware declarations table to another with a different CPU type.

4034 HWD: MMC EDITION DOWNLOAD TO STANDALONE MMC ONLY

When working with PiCPro Standalone MMC Edition, you can only download ladders to a standalone MMC control.

4035 HWD: DOWNLOAD FAILURE CPU TYPE MISMATCH

The CPU type you have selected in your ladder program does not match the CPU type in your control. Edit your hardware declarations so that the CPU type in your ladder matches the CPU type in your control.

4036 HWD: OUT OF DATA MEMORY

Out of Data Memory. Out of memory error. Your program requires more memory than is available in the Control CPU.

This is a compile time error of a main ladder. This can occur when the data memory required for retained variables (retentive) and declared hardware exceeds 64K.

This error can also occur on older style CPUs that have hardbit memory, when the number of retained BOOL's and declared hardware data memory exceeds 8K.

The only corrective action in either case is to reduce the amount of hardware or retained variables. Contact Giddings & Lewis for help in optimizing data memory usage.

4037 HWD: LADDER CONFIGURABLE I/O

You have chosen to enable Ladder Configurable I/O. This allows the ladder to continue scanning even if there has been a failure in the remote or block I/O connected to your system. I/O in the main rack will continue to work unless there has been a failure in the main I/O. If that occurs, all I/O in the system becomes non-operational. If the CPU is an analog MMC for PC CPU, ladder configurable I/O applies only to Block I/O. If there has been a failure in ASIU I/O, all I/O in the system becomes non-operational.

Warning:

Since this will allow the control to continue to scan even if there are I/O errors, the function block IO_CFG must be used to allow the ladder to react to I/O.

TIP

If you need information on IO_CFG, open Function Block Help by selecting **Help | Function/Function Block Help** from the main menu. Select the **Index** tab and then scroll down until you see IO_CFG. Double click on it. A description of this function block is then displayed on the right.

4038 HWD: CONVERT INVALID MMC FOR PC CPU

The ladder you are trying to open has an MMC for PC type CPU and you are using PiCPro Standalone MMC Edition software.

You are asked if you would like to convert the existing ladder to use a standalone MMC CPU instead.

Note: Master Rack and ASIU I/O will be removed because the standalone MMC CPU does not support this kind of I/O. Software Declarations should be checked to ensure the I/O is mapped to the correct locations after the conversion. See Conversion References in this section.

4039 HWD: LAST ASIU MODULE IS NOT EMPTY

ASIU 8 is not empty. An empty ASIU will be inserted below the ASIU with focus. All ASIUs below the one currently selected will be moved down, and what was ASIU 8 will be removed.

4040 HWD: INSERT ASIU WARNING

An empty ASIU will be inserted below the ASIU currently selected and all subsequent units will be shifted down.

4041 HWD: MAXIMUM ASIU MODULES EXIST

The maximum number of ASIU I/O slots already exist and therefore, no other modules can be inserted after ASIU 8.

4042 HWD: CONFIRM SWITCH MMC TO MMC FOR PC

When you switch from a standalone MMC to an MMC for PC CPU, this confirmation box will appear asking if you want to continue. If you do, slot 1 will hold the chosen MMC for PC module, slot 2 is emptied and slots 3 and 4 are removed.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct.

4043 HWD: CONFIRM SWITCH MMC FOR PC ANALOG CPU TO DIGITAL CPU

When you switch from an MMC for PC Analog Servo CPU to a Digital Servo CPU, all of the ASIUs will be removed. You are asked to confirm this action.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct.

4044 HWD: CONFIRM SWITCH PIC TO MMC FOR PC

When you switch from a PiC CPU to an MMC for PC CPU, slot 2 will be emptied and slots 3-13 will be removed. All remote I/O will also be removed. You are asked to confirm this action.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct.

4045 HWD: CONFIRM DELETE ASIU

When you delete an ASIU module, all of the subsequent module locations will be shifted. You are asked to confirm this action.

4046 HWD: PASTE NON-MMC FOR PC ANALOG SERVO

When you attempt to paste a non-MMC for PC Analog Servo master rack on an existing MMC for PC Analog Servo master rack, all ASIU I/O will be removed. You are asked to confirm this action.

4047 HWD: CONFIRM SWITCH MMC FOR PC ANALOG SERVO TO PIC

When you switch from an MMC for PC Analog Servo CPU to a PiC CPU, a CSM will be put into slot 1, the selected PiC CPU will be put in slot 2, and slots 3 – 13 will be added to the master rack. All ASIU I/O will be removed. You are asked to confirm this action.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct. See Conversion References in this section.

4048 HWD: CONFIRM SWITCH MMC FOR PC DIGITAL SERVO TO PIC

When you switch from an MMC for PC Analog Servo CPU to a PiC CPU, a CSM will be put into slot 1, the selected PiC CPU will be put in slot 2, and slots 3 – 13 will be added to the master rack. All ASIU I/O will be removed. You are asked to confirm this action.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct. See Conversion References in this section.

4049 HWD: CONFIRM SWITCH MMC FOR PC ANALOG SERVO TO STANDAL-ONE MMC

When you switch from an MMC for PC Analog Servo to a standalone MMC, the contents of slots 1 and 2 will be replaced, and slots 3 and 4 will be added to the master rack. All ASIU I/O will be removed. You are asked to confirm this action.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct.

4050 HWD: CONFIRM SWITCH MMC FOR PC DIGITAL SERVO TO STANDALONE MMC

When you switch from an MMC for PC Digital Servo to a standalone MMC, the contents of slots 1 and 2 will be replaced, and slots 3 and 4 will be added to the master rack. You are asked to confirm this action.

Note: The I/O points defined in Software Declarations should be checked to insure they are correct.

4058 HWD: TASK I/O ERROR

This message is displayed when the attempt is made to compile a task with Remote Rack or Block I/O modules declared.

4059 HWD: DOWNLOAD FAILURE CPU TYPE MISMATCH OBSOLETE CPU

- PiCTurbo2, part number 502-3814-00 with a 486DX processor is not compatible with PiCPro Version 13.0 or later. Use PiCPro Version 12.x or earlier.
- All other PiC CPU models with a 186 or 486SX processor, are not compatible with PiCPro Version 11.0 or later. Use PiCPro Version 10.x or earlier.

Ladder Error Messages

5000 COMPILE: INVALID TYPE

Data type mismatch. 'Data type' supplied. 'Data type' required.
The data type of the variable does not match the data type required.

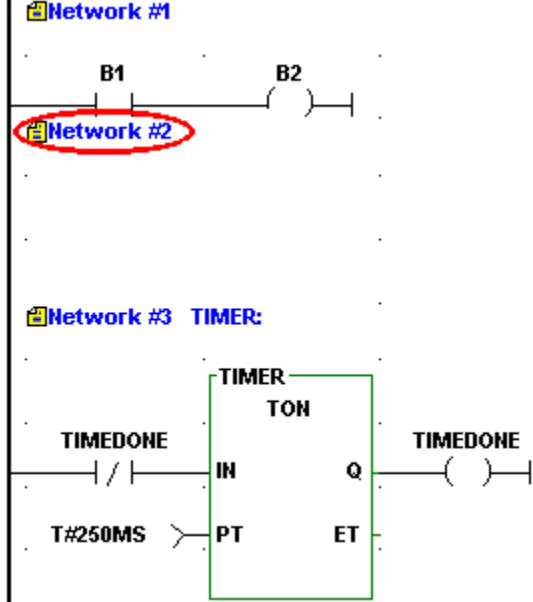
5001 COMPILE: DOUBLE DEFINED

'Label': has already been defined as a Label, cannot use again.
The specified label you have entered is already defined in another network. Each label must be unique.

5002 COMPILE: EMPTY NETWORK

Empty network not allowed.
Whenever an empty network is included in a ladder, an error message will appear. Delete the empty network if it is not needed or enter the necessary ladder logic.

Example of Error
Network 2 is empty.



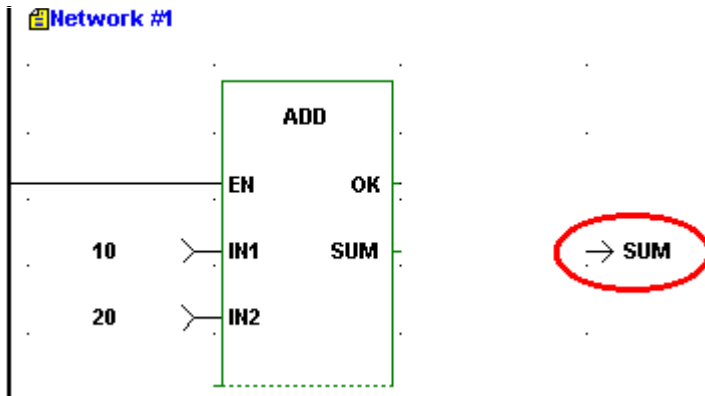
Note: If a network has a label assigned to it but no logic to execute (empty), the label will be ignored and may cause an undefined label error message to appear. If you need the network (i.e. when designing UDFBs), you can prevent this error by simply adding a horizontal wire in the empty network. The error will be avoided and you will be able to successfully compile your UDFB.

5003 COMPILE: CONNECTION ERROR

DATA IN, DATA OUT, or CONSTANT must be directly connected to a function/function block.

Example of Error

The **SUM** DATA OUT variable is not connected to the ADD function.



To correct this error do one of the following:

- Move the cell containing the DATA IN, DATA OUT, or CONSTANT to the function/function block input or output.
- Delete the cell containing the DATA IN, DATA OUT, or CONSTANT.

5004 COMPILE: FUNCTION ON LEFT RAIL

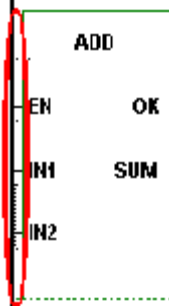
'Function': Function or Function Block cannot be directly connected to left rail.

You cannot connect a function/function block directly to the left power rail.

Example of Error

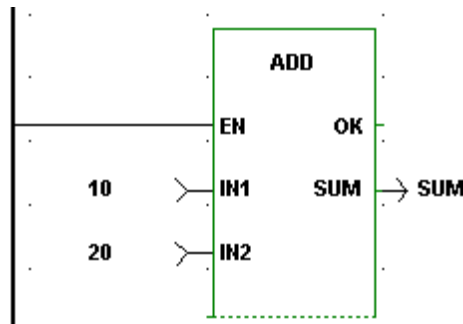
Incorrect

Function incorrectly placed at left power rail.



Correct

Function correctly positioned.



Function/function blocks can never be placed in the first column of your network. Reposition the function/function block to column two or greater and make the appropriate connections.

5005 COMPILE: DELETED NETWORK CONTAINS TASK

Deleted Network contains a task and cannot be patched. A scan stopped, full module download is required.

You cannot use the patch feature when you delete a network that contains a task. You must first stop the scan and perform a full download.

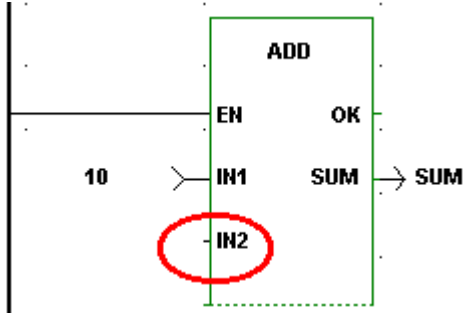
5006 COMPILE: NOT ENOUGH INPUTS

'Function Name': Function requires # inputs, only # supplied.

You have not supplied the correct number of required inputs for the function you are using.

Example of Error

An input at IN2 is required. Ensure that the correct number of inputs have been supplied to the function.



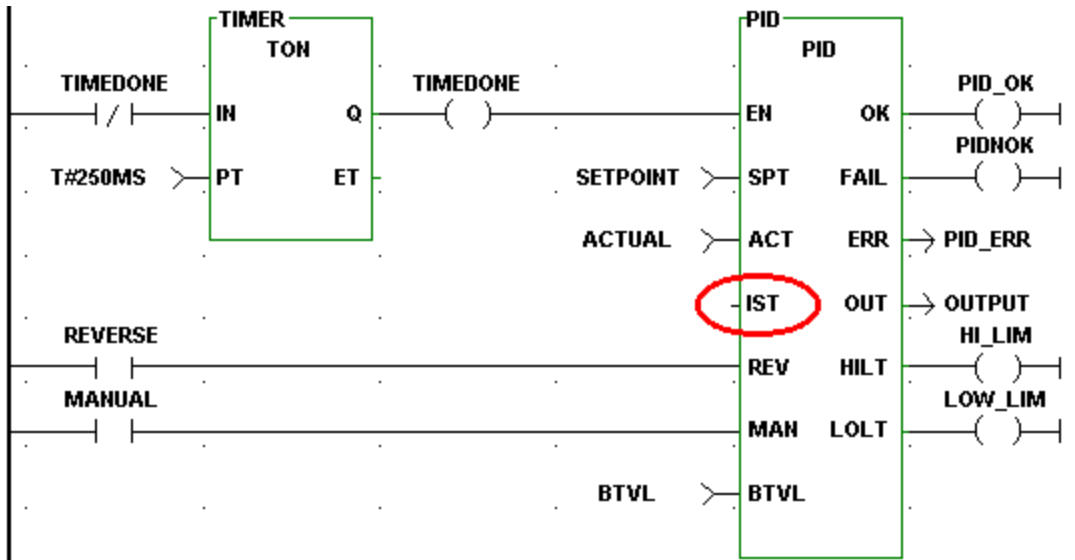
5007 COMPILE: INPUT REQUIRED

Input required

Some function blocks have inputs that are required.

Example of Error

The IST input on the PID function block requires an input and none was entered.



5008 COMPILE: NETWORK CONTAINS TASK

Modified network contains a task and cannot be patched. A scan stopped, full download is required.

You cannot use the patch feature when you modify a network that contains a task. You must stop the scan and perform a full download.

5009 COMPILE: FUNCTION CONNECTION ERROR

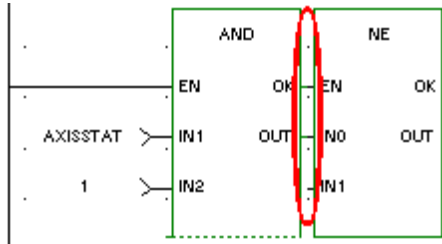
Connection Error. Function or Function Block must be connected to each other by wires. When connecting function/function blocks, you must leave at least one empty column between them and connect the outputs of the first to the inputs of the second with wires or contacts/coils.

Example of Error

Do not place a second function/function block directly next to an existing one. Leave a column between them and use wires or contacts/coils to make the required connections.

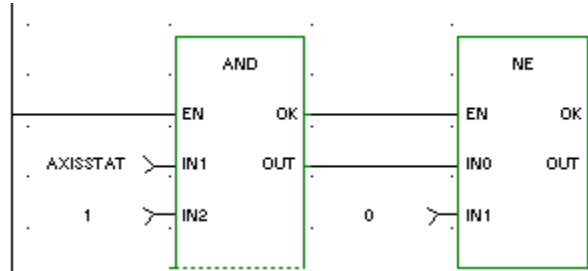
Incorrect

NE function incorrectly placed directly next to the AND function.



Correct

NE function correctly positioned and connected.



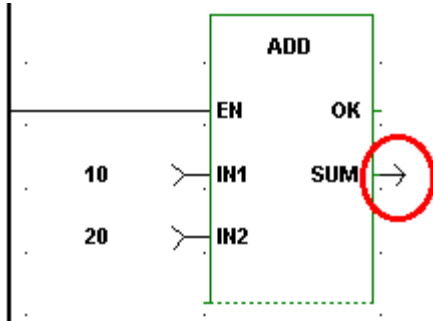
5010 COMPILE: VARIABLE REQUIRED

Variable name required.

DATA IN, DATA OUT, CONTACTS, and COILS all require variables or constants.

Example of Error

The name of the variable at the SUM output is missing.



You must enter a variable name declared with the appropriate data type to the DATA IN, DATA OUT, or the CONTACT/COIL location. The DATA IN may have a constant entered in place of a variable name. The data type for the contact/coil variable is always a boolean.

Note: Under the **View | Options** menu in User Preferences, you can choose to turn on Force Declarations. PiCPro will then prompt you to declare a variable each time you are required to enter one.

5011 COMPILE: DATAOUT CONNECTION ERROR

Connection Error. DATA OUT cannot be connected to first output.

DATA OUT cannot be connected to the first output on a function/function block. The first output can only be a contact/coil or a wire or left disconnected.

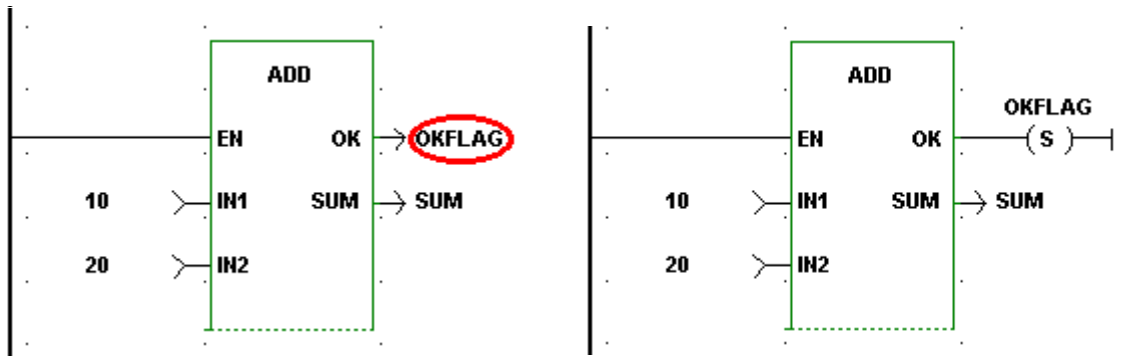
Example of Error

Incorrect

The DATA OUT variable OKFLAG is incorrect.

Correct

The set coil with variable name OKFLAG is an example of a correct connection to the first output of the ADD function.



A wire or a contact/coil can be connected to the first output (i.e. the OK) of a function/function block if desired. You may choose not to make any connection to the first output if you do not need to receive the output data.

5012 COMPILE: INVERT DATA TYPE WARNING

Data inversion not allowed on non-boolean data types. Input will not be inverted.

You can only invert boolean inputs on functions or function blocks. If you attempt to invert a non-boolean input this warning will appear and the input will be treated as a Data In, not as a Data Inverted.

- Change input to be a boolean constant, variable, or wire.
- Make the data input a Data In instead of Data Inverted.
- Consider changing the data type of the non-boolean variable to boolean.

5013 COMPILE: BOOLEAN REQUIRED

Incorrect data type specified, boolean required.

A contact or coil has been assigned a variable name that is some other data type than the boolean required.

- Enter a valid boolean variable.
- Declare variable as a boolean in the software declarations table.

5014 COMPILE: LABEL REQUIRED

Label required.

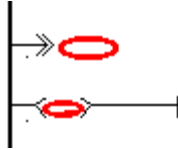
The label of the network you want to jump to with the Jump to Label or Jump to Subroutine commands must be entered with the jump command.

Example of Error

The Jump to Label and the Jump to Subroutine entries require a label of the network the jump is going to.

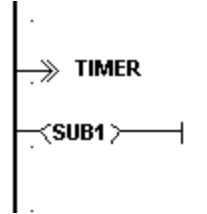
Incorrect

No labels entered for the Jump to Label or Jump to Subroutine commands.



Correct

Labels have been correctly entered with the Jump to Label and Jump to Subroutine commands.



- Ensure that you have assigned a label to the network you want to jump to.
or
- Remove the Jump to Label or Jump to Subroutine entry.

5015 COMPILE: LABEL UNDEFINED

Label is not defined.

Network labels are required on Jump to Label and Jump to Subroutine commands. If you enter a label that you have not assigned to a network, you will get this error.

- Ensure that any network you want to jump to has a label assigned to it.
- If you get this error after entering a label with the jump command, make sure the label exists and/or check the spelling of the label to ensure it matches the label of the destination network.

5016 COMPILE: NPX REQUIRED

Requires a CPU with an NPX processor.

A warning appears when a variable name or constant has been defined with a data type that requires a CPU with an NPX processor. The CPU currently declared in the Hardware Declarations table does not have an NPX processor.

- Declare a CPU with an NPX processor in the Hardware Declarations table.
or
- Change the data type from REAL, LREAL, LINT, etc. to a data type that does not require a NPX processor.

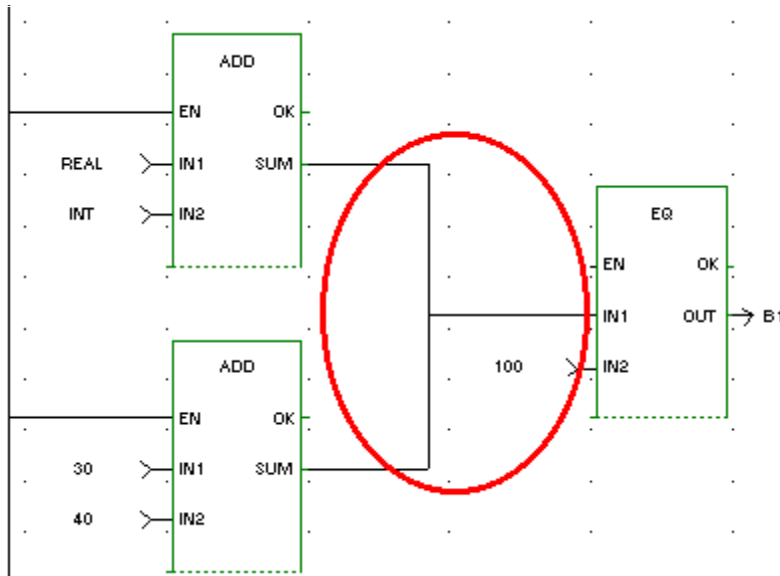
5017 COMPILE: INVALID BRANCH

Invalid branch.

An invalid branch has been attempted. Two or more wires/rungs may form a branch only if the data types being passed on the wire are boolean.

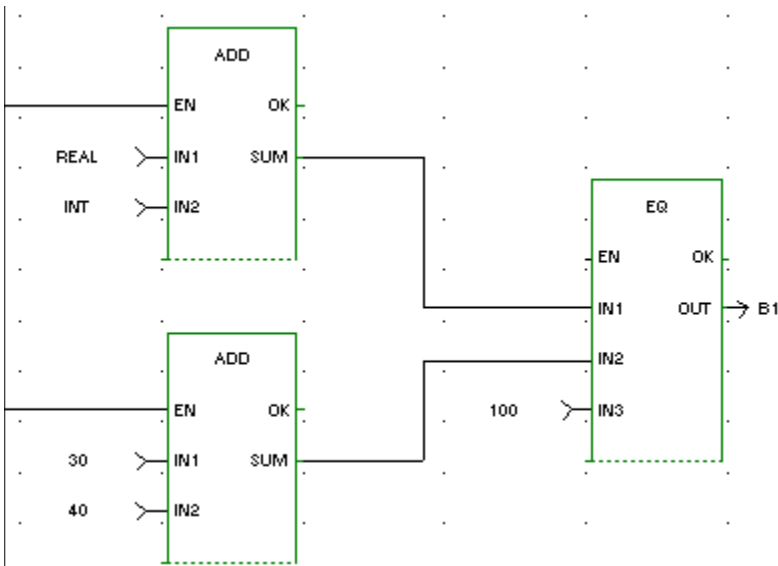
Example of Error Incorrect

An invalid branch is formed by the wires from the SUM outputs of the ADD functions going to the IN1 input of the EQ function. The SUM outputs are non-boolean data types and cannot be branched this way. **Note:** The OK outputs from the ADD functions could form a valid branch since they are both boolean data types.



Correct

The correct method of branching the above example is shown below.



It is valid to branch as shown in the incorrect example above if the two branch sources are both boolean data types.

5018 COMPILE: INTERNAL ERROR

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5019 COMPILE: INTERNAL LABEL NOT FOUND

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5020 COMPILE: INTERNAL EMPTY LIST

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5021 COMPILE: INTERNAL UNKNOWN ELEMENT

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5022 COMPILE: INTERNAL NO TEMPLATE

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5023 COMPILE: INTERNAL NO DESTINATION

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5024 COMPILE: INTERNAL INVALID OUTPUT COUNT

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5025 COMPILE: INTERNAL LDO ERROR

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5026 COMPILE: INTERNAL INVALID DATA TYPE

An internal error has occurred. Please contact Giddings and Lewis.

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings & Lewis.

5027 COMPILE: NO NETWORKS

Ladder does not contain any networks.

The ladder you are attempting to compile does not contain any networks.

5028 COMPILE: INVALID ARRAY INDEX

Array index must be either UINT or USINT.

The data type of the index for an array must be either a USINT or UINT.

- Use a variable or constant that is a USINT or UINT.
- Change the variable's data type to USINT or UINT.

5029 COMPILE: PARAMETER ERROR

Parameter error.

The parameter does not meet the necessary requirements for this function or function block.

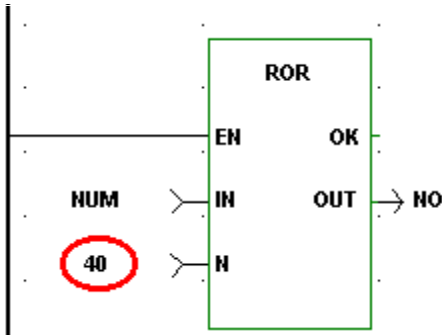
Review the documentation associated with the function/function block you are working with and correct the input.

5030 COMPILE: INVALID CONSTANT

Invalid constant. Constant value must be in the range '#' to '#'.

Example of Error

The constant 40 at the NUM input (declared as a WORD) of the ROR function is out of range for this input. The valid range is from 0 to 31.



Ensure that the value you enter as a constant is within the required range. The range is based on the number of bits specified for the data type.

Range if IN data type is less than 8 bytes (BYTE, WORD, or DWORD)
0 - 31

Range if IN data type = 8 bytes (LWORD)
0 - 63

5031 COMPILE: TASK IN FUNCTION

Tasks may not be called from within functions.

You have attempted to call a task from within a UDFB. Program a task within the main ladder only.

5032 COMPILE: TOO MANY TASK INPUTS

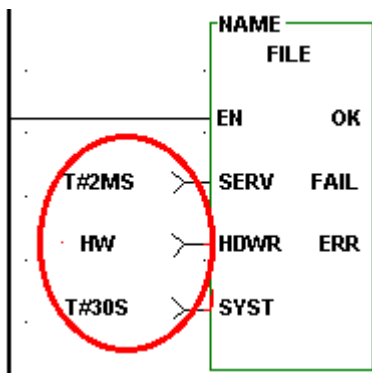
Too many task inputs specified. Task inputs SERV, HDWR, and SYST are mutually exclusive. Only one may be specified.

Task inputs SERV, HDWR, and SYST are mutually exclusive. You may specify only one of these inputs for a task.

Example of Error

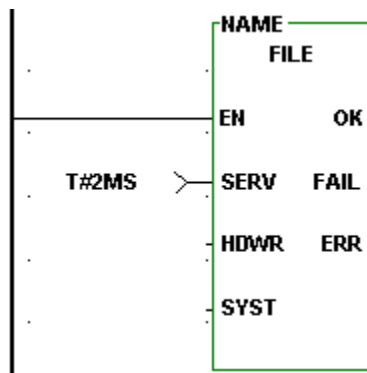
Incorrect

All three TASK inputs (SERV, HDWR, and SYST) have been entered.



Correct

Only one TASK input has been entered.



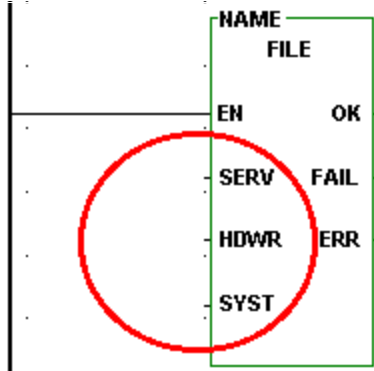
5033 COMPILE: TASK INPUT REQUIRED

An input is required at one of the task inputs.

You have not connected any input at any of the task inputs of SERV, HDWR, or SYST.

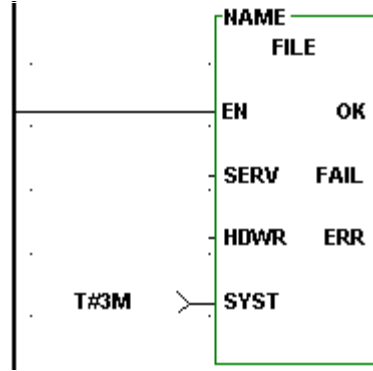
**Example of Error
Incorrect**

None of the three TASK inputs (SERV, HDWR, and SYST) has been entered.



Correct

One TASK input must be entered.



Connect one input on the task function block depending on whether your task is a servo, hardware, or system interrupt task.

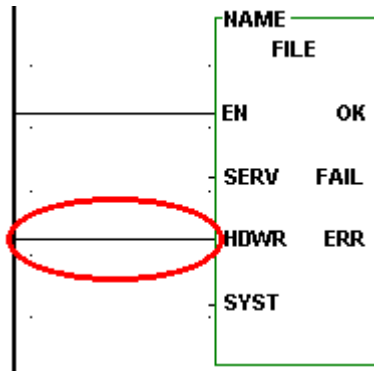
5034 COMPILE: DATA IN REQUIRED

Input must be DATA IN or DATA INVERTED.

When you want to use an I/O point to trigger a hardware interrupt task, the HDWR input of the task function block must be data in or data inverted. Never program a wire or contact for this input.

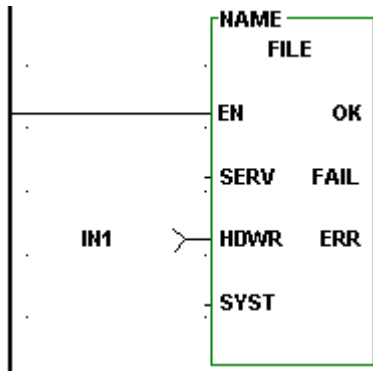
**Example of Error
Incorrect**

A wire has been connected to the HDWR input of the TASK function block.



Correct

Data In and the variable IN1 has been entered at the HDWR input of the TASK function block.



- Enter DATA IN or DATA INVERTED for the HDWR input of the task function block.
- or
- Convert the wire or contact to a DATA IN or DATA INVERTED.

5035 COMPILE: MAX IO EXCEEDED

The number of function block inputs or outputs exceed 64.
An internal error has occurred. Please contact Giddings and Lewis.

Tips

If you receive an internal error, take the following steps.

1. Write down the error number and the exact wording of the error message.
2. Note the version of software you are using.
3. Save your LDO file.
4. Send the above information, your LDO file and all related files to Giddings & Lewis.

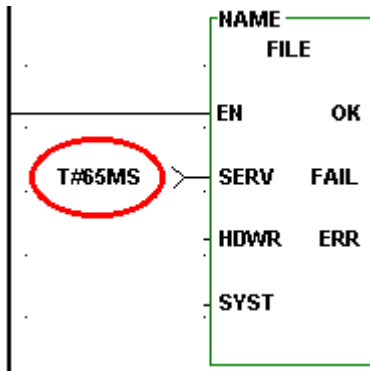
5036 COMPILE: BAD TASK SERV

Input must be one of the following constants: T#1MS, T#2MS, T#4MS, T#8MS or T#16MS.
When you want to trigger a servo interrupt task, the SERV input of the task function block must be one of these servo time tick constants: T#1MS, T#2MS, T#4MS, T#8MS or T#16MS.

Example of Error

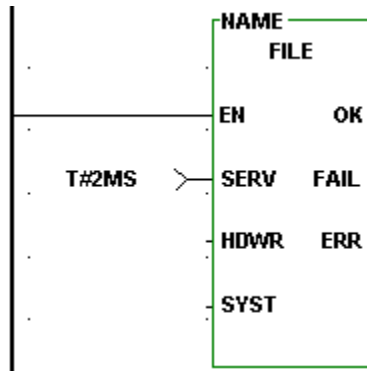
Incorrect

An invalid constant has been entered for the SERV input of the TASK function block.



Correct

One of the valid servo time ticks has been entered at the SERV input of the TASK function block.



5037 COMPILE: MAIN RACK ONLY

I/O point must be from main rack only.
Tasks can only access I/O modules located in the main rack. Enter only an I/O point from an I/O module in the main rack to trigger a hardware interrupt task.

5038 COMPILE: IO POINT REQUIRED

I/O point required.
When you want to use an I/O point to trigger a hardware interrupt task, you must enter the I/O point at the HDWR input of the task function block.

5039 COMPILE: INPUT REQUIRES BOOLEAN VARIABLE

Boolean variable required.
A constant was entered as an input and a boolean variable is required.

5101 COMPILE: INVALID USE OF KEYWORD

Keyword cannot be used as a variable name.

A keyword variable was found in a Structured Text statement. Keywords are not allowed as variable names and cannot be used as variable names in Structured Text statements. Change the name of this variable in software declarations.

Tip

Replace keyword with a variable that is not a keyword.

5102 COMPILE SYNTAX ERROR

Syntax Error.

The structured text syntax entered does not match the rules for structured text statements. Verify the statement or expression where the error occurred to make sure that it is syntactically correct.

Tip

Double click on the error message and examine the statement where the error has occurred. If in an expression make sure that:

- a “)” is not missing.
- an operator is not missing.

If in a structured text statement make user that an extra keyword or that a keyword was used out of context

Correct the syntax error and then recompile.

5103 COMPILE: INVALID ASSIGNMENT STATEMENT

Missing or Invalid assignment statement.

The assignment statement used to form a FOR loop is not syntactically correct.

Tip

Enter a correct assignment statement in the FOR loop.

5104 COMPILE: EXIT OUTSIDE LOOP

EXIT statement used outside of loop.

Exit statements can only be used within an iteration statement.

Tip

Delete the EXIT statement or move within an iteration statement.

5105 COMPILE: NO MATCHING WHILE

END_WHILE without a matching WHILE statement.

An END_WHILE statement was encountered that does not have a matching WHILE statement.

Tip

Delete the END_WHILE statement.

5106 COMPILE: NO MATCHING REPEAT

UNTIL or END_REPEAT without a matching REPEAT statement.

An UNTIL or END_REPEAT statement was encountered that does not have a matching REPEAT statement.

Tip

Delete the UNTIL or END_REPEAT statement.

5107 COMPILE: NO MATCHING FOR

END_FOR without a matching FOR statement.

An END_FOR statement was encountered that does not have a matching FOR statement.

Tip

Delete the END_FOR statement.

5108 COMPILE: NO MATCHING IF

ELSIF, ELSE, or END_IF without a matching IF statement.

An ELSIF, ELSE or END_IF statement was encountered that does not have a matching IF statement.

Tip

Delete the ELSIF, ELSE, or END_IF statement.

5109 COMPILE: UNEXPECTED KEYWORD

Unexpected Keyword found - TO, THEN, DO, NOT, OR, XOR, AND.

Keyword is used out of context.

Tip

Delete the keyword entered or fix the syntax error associated with this keyword.

5110 COMPILE: NESTING ERROR #1

Nesting error - Expecting UNTIL-END_REPEAT before END_WHILE.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Incorrect:

```
WHILE A DO
    REPEAT
END_WHILE;
    UNTIL B END_REPEAT;
```

Correct:

```
WHILE A DO
    REPEAT
        UNTIL B END_REPEAT;
END_WHILE;
```

Tip

Ensure that the REPEAT statement is entirely within the WHILE statement as the Correct example illustrates above.

5111 COMPILE: NESTING ERROR #2

Nesting error - Expecting END_FOR before END_WHILE.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Incorrect:

```
WHILE A DO
    FOR I=0 TO 5 DO
END_WHILE;
    END_FOR;
```

Correct:

```
WHILE A DO
    FOR I=0 TO 5 DO
        END_FOR;
END_WHILE;
```

Tip

Ensure that the FOR statement is entirely within the WHILE statement as the Correct example illustrates above.

5112 COMPILE: NESTING ERROR #3

Nesting error - Expecting ELSIF, ELSE, or END_IF before END_WHILE.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
WHILE A DO
    IF B THEN
        END_IF;
    END_WHILE;
```

Incorrect:

```
WHILE A DO
    IF B THEN
END_WHILE;
    END_IF;
```

Tip

Ensure that the IF statement is entirely within the WHILE statement as the Correct example illustrates above.

5113 COMPILE: NESTING ERROR #4

Nesting error - Expecting END_WHILE before UNTIL-END_REPEAT.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
REPEAT
    WHILE A DO
        END_WHILE;
UNTIL B END_REPEAT;
```

Incorrect:

```
REPEAT
    WHILE A DO
UNTIL B END_REPEAT;
    END_WHILE;
```

Tip

Ensure that the WHILE statement is entirely within the REPEAT statement as the Correct example illustrates above.

5114 COMPILE: NESTING ERROR #5

Nesting error - Expecting END_FOR before UNTIL-END_REPEAT.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
REPEAT
    FOR I := 0 TO 5 DO
        END_FOR;
UNTIL B END_REPEAT;
```

Incorrect:

```
REPEAT
    FOR I := 0 TO 5 DO
UNTIL B END_REPEAT;
    END_FOR;
```

Tip

Ensure that the FOR statement is entirely within the REPEAT statement as the Correct example illustrates above.

5115 COMPILE: NESTING ERROR #6

Nesting error - Expecting ELSIF, ELSE, or END_IF before UNTIL-END_REPEAT

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
REPEAT
    IF B THEN
        END_IF;
UNTIL B END_REPEAT;
```

Incorrect:

```
REPEAT
    IF B THEN
UNTIL B END_REPEAT;
    END_IF;
```

Tip

Ensure that the IF statement is entirely within the REPEAT statement as the Correct example illustrates above.

5116 COMPILE: NESTING ERROR #7

Nesting error - Expecting END_WHILE before END_FOR

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
FOR I := 0 TO 5 DO
    WHILE A DO
        END_WHILE;
END_FOR;
```

Incorrect:

```
FOR I := 0 TO 5 DO
    WHILE A DO
END_FOR;
    END_WHILE;
```

Tip

Ensure that the WHILE statement is entirely within the FOR statement as the Correct example illustrates above.

5117 COMPILE: NESTING ERROR #8

Nesting error - Expecting UNTIL-END_REPEAT before END_FOR.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
FOR I := 0 TO 5 DO
    REPEAT
        UNTIL B END_REPEAT;
END_FOR;
```

Incorrect:

```
FOR I := 0 TO 5 DO
    REPEAT
END_FOR;
    UNTIL B END_REPEAT;
```

Tip

Ensure that the REPEAT statement is entirely within the FOR statement as the Correct example illustrates above.

5118 COMPILE: NESTING ERROR #9

Nesting error - Expecting END_IF before END_FOR.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
FOR I := 0 TO 5 DO
    IF A THEN
        END_IF;
END_FOR;
```

Incorrect:

```
FOR I := 0 TO 5 DO
    IF A THEN
END_FOR;
    END_IF;
```

Tip

Ensure that the IF statement is entirely within the FOR statement as the Correct example illustrates above.

5119 COMPILE: NESTING ERROR #12

Nesting error - Expecting END_WHILE before END_IF.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
IF A THEN
    WHILE B DO
        END_WHILE;
END_IF;
```

Incorrect:

```
IF A THEN
    WHILE B DO
END_IF;
    END_WHILE;
```

Tip

Ensure that the WHILE statement is entirely within the IF statement as the Correct example illustrates above.

5120 COMPILE: NESTING ERROR #11

Nesting error - Expecting END_REPEAT before END_IF.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
IF A THEN
    REPEAT
        UNTIL B END_REPEAT;
END_IF;
```

Incorrect:

```
IF A THEN
    REPEAT
END_IF;
    UNTIL B END_REPEAT;
```

Tip

Ensure that the REPEAT statement is entirely within the IF statement as the Correct example illustrates above.

5121 COMPILE: NESTING ERROR #12

Nesting error - Expecting END_FOR before END_IF

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
IF A THEN
    FOR I := 0 TO 5 DO
        END_FOR;
END_IF;
```

Incorrect:

```
IF A THEN
    FOR I := 0 TO 5 DO
END_IF;
    END_FOR;
```

Tip

Ensure that the FOR statement is entirely within the IF statement as the Correct example illustrates above.

5122 COMPILE: MISSING MATCHING CASE

END_CASE without a matching CASE statement.

An END_CASE statement was encountered that does not have a matching CASE statement.

Tip

Delete the END_CASE statement.

5123 COMPILE: NESTING ERROR #13

Nesting error - Expecting END_CASE before END_WHILE.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
WHILE A DO
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
    END_CASE;
END_WHILE;
```

Incorrect:

```
WHILE A DO
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
    END_WHILE;
    END_CASE;
```

Tip

Ensure that the CASE statement is entirely within the WHILE statement as the Correct example illustrates above.

5124 COMPILE: NESTING ERROR #14

Nesting error - Expecting END_CASE before UNTIL-END-REPEAT.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
REPEAT
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
    END_CASE;
UNTIL A END_REPEAT;
```

Incorrect:

```
REPEAT
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
UNTIL A END_REPEAT;
    END_CASE;
```

Tip

Ensure that the CASE statement is entirely within the REPEAT statement as the Correct example illustrates above.

5125 COMPILE: NESTING ERROR #15

Nesting error - Expecting END_CASE before END_FOR.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
FOR I:=0 TO 5 DO
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
    END_CASE;
END_FOR;
```

Incorrect:

```
FOR I:=0 TO 5 DO
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
END_FOR;
    END_CASE;
```

Tip

Ensure that the CASE statement is entirely within the FOR statement as the Correct example illustrates above.

5126 COMPILE: NESTING ERROR #16

Nesting error - Expecting END_CASE before END_IF.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
IF A THEN
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
    END_CASE;
END_IF;
```

Incorrect:

```
IF A THEN
    CASE A OF
        1..10:
            SPEED := SPEED + 10;
    END_IF;
    END_CASE;
```

Tip

Ensure that the CASE statement is entirely within the IF statement as the Correct example illustrates above.

5127 COMPILE: NESTING ERROR #17**Nesting error - Expecting END_WHILE before END_CASE or CASE selector**

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:**Correct:**

```
CASE A OF
    1..10:
        WHILE B DO
            SPEED := SPEED + 10;
        END_WHILE;
    END_CASE;
```

Incorrect:

```
CASE A OF
    1..10:
        WHILE B DO
            SPEED := SPEED + 10;
        END_CASE;
    END_WHILE
```

Tip

Ensure that the WHILE statement is entirely within the CASE statement as the Correct example illustrates above.

5128 COMPILE: NESTING ERROR #18

Nesting error - Expecting END_REPEAT before END_CASE or CASE selector.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
CASE A OF
  1..10:
    REPEAT
      SPEED := SPEED + 10;
    UNTIL A END_REPEAT;
END_CASE;
```

Incorrect:

```
CASE A OF
  1..10:
    REPEAT
      SPEED := SPEED + 10;
    UNTIL A END_REPEAT;
END_CASE;
  UNTIL A END_REPEAT;
```

Tip

Ensure that the REPEAT statement is entirely within the CASE statement as the Correct example illustrates above.

5129 COMPILE: NESTING ERROR #19

Nesting error - Expecting END_FOR before END_CASE or CASE selector.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
CASE A OF
  1..10:
    FOR I:=0 TO 5
      SPEED := SPEED + 10;
    END_FOR;
END_CASE;
```

Incorrect:

```
CASE A OF
  1..10:
    FOR I:=0 TO 5
      SPEED := SPEED + 10;
END_CASE;
  END_FOR;
```

Tip

Ensure that the FOR statement is entirely within the CASE statement as the Correct example illustrates above.

5130 COMPILE: NESTING ERROR #20

Nesting error - Expecting END_IF before END_CASE or CASE selector.

Structured Text statements can be nested inside other structured statements. A statement that is nested within another statement must be entirely within the outer most statement. No overlapping of statements are allowed. See example below:

Example of Error:

Correct:

```
CASE A OF
  1..10:
    IF A THEN
      SPEED := SPEED + 10;
    END_IF;
END_CASE;
```

Incorrect:

```
CASE A OF
  1..10:
    IF A THEN
      SPEED := SPEED + 10;
END_CASE;
  END_IF;
```

Tip

Ensure that the IF statement is entirely within the CASE statement as the Correct example illustrates above.

5131 COMPILE: MISSING SELECTOR

Missing or expecting CASE selector.

A CASE selector must immediately follow the OF in the case statement.

Example of Error:

Correct:

```
CASE A OF
  1..10:
    SPEED := SPEED + 10;
END_CASE;
```

Incorrect:

```
CASE A OF
  SPEED := SPEED + 10;
END_CASE;
```

Tip

Add missing CASE selector.

5132 COMPILE: MISSING INTEGER

Missing Integer.

In the example below an integer number should appear after the “,” in the CASE Selector.

Example of Error:

Correct:

```
CASE A OF
  1,2:
    SPEED := SPEED + 10;
END_CASE;
```

Incorrect:

```
CASE A OF
  1,:
    SPEED := SPEED + 10;
END_CASE;
```

Tip

Either eliminate the extra comma or add integer number to the CASE selector.

5133 COMPILE: MISSING PARAMETER

Missing Function/Function Block Parameter.

Two commas right next to each other as illustrated in the example below is a syntax error and implies that a function/function block parameter is missing.

Example of Error:

Incorrect:

```
MAX(IN1:=A, ,IN2:=B);
```

Tip

Remove the extra comma or add another function/function block parameter between the commas.

5134 COMPILE: INVALID CONSTANT

Invalid Hex, Octal or Binary Constant.

An invalid character was used in a Hex, Octal, or a binary constant. See the manual for the proper syntax of these constants.

Example of Error:

16#G1AF is an invalid Hex constant because it contains the letter G.

Tip

- Verify that the hex number only contains the digits 0-9 and the letters A-F.
- Verify that the octal number only contains the digits 0-7.
- Verify that the binary number only contains the digits 0-1.

Make appropriate changes.

5135 COMPILE: EXPRESSION SYNTAX ERROR

Expression syntax error.

The structured text syntax entered does not match the rules for structured text statements. Verify the expression where the error occurred to make sure that it is syntactically correct.

Tip

Double click on the error message and examine the statement where the error has occurred. Ensure that:

- a “)” is not missing.
- an operator is not missing.

Correct the syntax error and then recompile.

5136 COMPILE: MISSING CLOSE COMMENT

Missing closing comment indicator “*”).

A comment start “(“ was encountered that does not have a matching comment end “*”).

Tip

Find the end of comment and add “*”).

5137 COMPILE: EXPECTING INTEGER

Expecting Integer.

Non-Integer constant value entered where an Integer constant value is expected. In the example below, B should be an integer constant. Only integer constants are allowed in CASE selectors.

Example of Error:

Incorrect:

```
CASE A OF
  1,B:
    SPEED := SPEED + 10;
END_CASE;
```

Tip

Enter only an Integer constant or delete the characters that caused the error.

5138 COMPILE: INVALID CASE EXPRESSION

Invalid CASE expression.

The structured text syntax for the entered CASE expression does not match the rules for structured text statements. Verify the expression where the error occurred to make sure that it is syntactically correct.

Tip

Double click on the error message and examine the statement where the error has occurred. Ensure that:

- a “)” is not missing.
- an operator is not missing.

5139 COMPILE: EXPECTING VARIABLE

Expecting variable or parameter name to the left of “:=” or “=>”.

Example of Error:

```
FOR := 0 TO 5
  ;
END_FOR;
```

In the above example the assignment variable is missing.

Tip

Add the missing variable or parameter name to the left of the “:=” or “=>”.

5140 COMPILE: MISSING FOR ASSIGNMENT

Missing assignment statement that initializes the iteration variable in a FOR loop.

As illustrated in the example below the assignment statement used to initialize the loop counter (LOOP vs. LOOP:=0) is missing or incomplete.

Example of Error:

Correct:

```
FOR LOOP:=0 TO 5
```

```
;
```

```
END_FOR;
```

Incorrect:

```
FOR LOOP TO 5
```

```
;
```

```
END_FOR;
```

Tip

Enter a complete assignment statement for the loop counter.

5141 COMPILE: FUNCTION BLOCK RESYNC

Syntax error in Function/Function block parameter.

The structured text syntax entered does not match the rules for structured text Function /Function Block parameters. Verify the statement or expression where the error occurred to make sure that it is syntactically correct.

Tip

Double click on the error message and examine the statement where the error has occurred. The offending characters will be highlighted.

- Ensure that “,” follows the Function/Function Block parameter.
- If error occurs right after an expression, verify the expression for accuracy.

Correct the syntax error and then recompile.

5142 COMPILE: INVALID ASSIGNMENT

Invalid expression assignment statement.

An internal error has occurred. Please contact Giddings and Lewis.

Tip

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of the software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings and Lewis.

5143 COMPILE: FUNCTION BLOCK PARAMETER ERROR

Expression must be assigned to a Function/Function Block input or output parameter name using ":= " or "=>".

The input or output parameter used in a function or function block is incomplete. The parameter name and ":= " or "=>" are missing.

Tip

Enter the complete function or function block parameter (i.e., IN0:=A+1 or OUT1=>B).

5144 COMPILE: STATEMENT SYNTAX ERROR

Statement syntax error.

The structured text syntax entered does not match the rules for structured text statements. Verify the statement or expression where the error occurred to make sure that it is syntactically correct.

Tip

Double click on the error message and examine the statement where the error has occurred. If in an expression make sure that:

- a ")" is not missing.
- an operator is not missing.

If in a structured text statement, that user makes, an extra keyword does not exist or that a keyword was not used out of context

Correct the syntax error and then recompile.

5145 COMPILE: MISSING EXPRESSION

Missing or incomplete expression.

A required expression is missing in an ST statement. In example #1 the assignment statement does not have an expression. In example #2 the MAX function is missing an expression after "IN1 :=".

Example of Error #1

```
A := ;
```

Example of Error #2

```
MAX(IN1:=, IN2:=A, OUT1=>B);
```

Tip

- Enter an expression.

5146 COMPILE: NEGATIVE SUBSCRIPT

Array subscripts cannot be negative.

Array subscripts cannot be negative and must be in the range 0 and 998, inclusive. Therefore, the negative sign is not permissible on an array subscript.

Tip

Ensure that the array subscript is between 0 and 998.

5147 COMPILE: NEGATIVE CONSTANT

Unary operator ("-") not allowed on this constant.

The unary operator “-” is not valid on constants that begin with 16# (Hex), 8# (Octal), 2# (Binary), TOD# (Time of Day), DT# (Date and Time), and T# (Time).

Tip

Enter a valid constant value by removing the unary operator “-”.

5148 COMPILE: POWER OPERATOR NOT SUPPORTED

The operator Raise to Power ("") is not currently supported.**

The operator Raise to Power (“**”) is not currently supported in PiCPro but may be supported in a future release of PiCPro.

Tip

Replace Raise to Power (“**”) operator with an expression that performs that same calculation .

5160 COMPILE: NO PARSER INTERFACE

INTERNAL ERROR: Failed to initialize/obtain parse engine interface.

An internal error has occurred. Please contact Giddings and Lewis.

Tip

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of the software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings and Lewis.

5161 COMPILE: NO GRAMMAR FILE

Cannot continue with compile. Missing internal system file "filename".

An internal error has occurred. Please contact Giddings and Lewis.

Tip

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of the software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings and Lewis.

5162 COMPILE: PARSER SYNCH ERROR

INTERNAL ERROR: Parser is out of sync with parse tree in routine "%s".

An internal error has occurred. Please contact Giddings and Lewis.

Tip

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of the software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings and Lewis.

5163 COMPILE: BY 0 NOT ALLOWED

Constant value of 0 not allowed after BY. Value produces infinite loop.

Specifying a constant value of 0 after the BY in a FOR loop is not allowed. Allowing this would cause an infinite loop that would scan loss the control.

Tip

Enter a non-zero constant value after the BY in the FOR loop.

5164 COMPILE: DUPLICATE SELECTOR VALUE OR RANGE

Duplicate selector value or selector range overlaps with a previous selector.

The selector values or range of values used in a CASE statement must be unique and cannot overlap.

Example of Error:

Correct:

```
CASE A OF
  2:
      SPEED := SPEED + 20;
  1, 3..10:
      SPEED := SPEED + 10;
END_CASE;
```

Incorrect:

```
CASE A OF
  2:
      SPEED := SPEED + 20;
  1..10:
      SPEED := SPEED + 10;
END_CASE;
```

Tip

Ensure that no selector values overlap and change accordingly.

5165 COMPILE: SELECTOR RANGE OUT OF ORDER

Selector range values should be from lowest to highest.

When specifying a selector range, the range must be lowest value first then highest value.

Example of Error:

Correct:

```
CASE A OF
    1..10:
        SPEED := SPEED + 10;
END_CASE;
```

Incorrect:

```
CASE A OF
    10..1:
        SPEED := SPEED + 10;
END_CASE;
```

Tip

Ensure that the range is lowest value first then highest value. Typically, all that needs to be done is to reverse the range values.

5166 COMPILE: FUNCTION MUST RETURN VALUE

Function must return a value to be used in an expression.

Functions that do not return a result, such as STEPCTL, cannot be called directly from an expression. . Functions that fall into this category must be invoked as independent standalone ST instructions that have similar syntax to a function block call.

Tip

Remove the function call from the expression and use function block syntax to make the function call. Results from the function call can then be used in the expression.

5167 COMPILE: TO MANY FUNCTION OUTPUTS

Functions that contain multiple output values cannot be used in an expression.

Functions such as MOVE and BY2TBOOL have multiple outputs. These functions cannot be called directly from any expression. Functions that fall into this category must be invoked as independent standalone ST instructions that have similar syntax to a function block call. In this case, function output parameters are typically specified.

Tip

Remove the function call from the expression and use function block syntax to make the function call. Results from the function call can then be used in the expression.

5168 COMPILE: FUNCTION UNDEFINED

Function, Function Block, or Function Block title is not defined.

The specified Function, Function Block, instance name, or title is not defined.

Tip

- Check the spelling of the Function, Function Block, instance name, or title.
- Ensure that the specified Function or Function Block is in a library (.lib) that is defined in the PiCPro Libraries path.

5169 COMPILE: FUNCTION BLOCK TITLE MISMATCH

":Function Block Title" syntax cannot be used with Functions. Ignoring.....

The syntax "Function Block instance name:Function Block title" is only valid with Function Blocks. Specifying this syntax with a Function is an error.

Tip

Remove ":Function Block Title" from the specified function call.

5170 COMPILE: FUNCTION BLOCK IN EXPRESSION

Function Blocks cannot be used with an expression.

Only Functions can be called from an expression. Function Blocks cannot be called from an expression. Function Block calls are independent standalone statements in ST. The results from a Function Block call can be used in any expression.

Tip

Remove the Function Block call from the expression and make the call to the Function Block a standalone ST statement. If necessary, use results from the Function Block call in an expression.

5171 COMPILE: FUNCTION PARAMETER ALREADY SPECIFIED

Input/Output parameter was already specified.

A Function or Function block parameter input or output can only be specified once per Function or Function Block call. The specified parameter name was already specified for this Function or Function Block.

Tip

Remove either of the duplicated parameter names.

5172 COMPILE: FUNCTION I/O PARAMETER NOT DEFINED

Input/Output parameter is not defined for this Function or Function Block.

The specified input or output parameter name is not defined for this function. Verify this name against the definition.

Tip

- Check the spelling of the input or output name.
- Replace the specified input or output name with a valid input or output name.

5173 COMPILE: FUNCTION OUTPUT NOT ALLOWED

Cannot specify an output parameter on a function used within an expression.

When a Function is specified in an expression, output parameters cannot be used. The results from the function call are directly used in the expression.

Tip

Remove all output parameters from this Function call.

5174 COMPILE: NOT DEFINED AS FUNCTION BLOCK

Not defined as a Function Block.

The specified variable is not defined as a function block in software declarations.

Tip

- Check the spelling of the variable name.
- Replace specified variable with one that is defined as a function block.

5175 COMPILE: BAD FUNCTION BLOCK TITLE

Function Block Title entered does not match definition. Ignoring.....

The specified function block title does not match the function block title defined in software declarations for the specified function block instance name.

Tip

- Change Function Block title so it matches definition in software declarations.
- Ensure that the right function block instance name is specified.

5176 COMPILE: VARIABLE REQUIRES SUBSCRIPT

Variable is defined as an array and requires an array subscript.

The specified variable is defined as an array and requires an array subscript.

Tip

- Add array subscript to variable.
- Verify spelling of variable name.

5177 COMPILE: STRUCTURE UNDEFINED

Variable is not defined or data type of defined variable is not a structure.

The variable is not defined as a structure in software declarations or is not defined at all.

Tip

- Check the spelling of the variable name.
- Define the specified variable as a structure.
- Replace specified variable with one that is defined as a structure.

5178 COMPILE: VARIABLE UNDEFINED

Variable is not defined.

The specified variable is not defined in software declarations.

Tip

- Check the spelling of the variable name.
- Define the specified variable.
- Replace specified variable with one that is defined.

5179 COMPILE: VARIABLE NOT ARRAY

Variable is not defined as an array.

The specified variable is used as an array but is not defined as an array in software declarations.

Tip

- Check the spelling of the variable name.
- Remove the array subscript
- Define the specified variable as an array.
- Replace specified variable with a defined array name.

5180 COMPILE: STRUCTURE ELEMENT UNDEFINED

Structure Element is not defined.

The specified structure element name is not defined in software declarations.

Tip

- Check the spelling of the element name.
- Define the specified element name.
- Change the specified element name to one that is already defined.

5181 COMPILE: SUBSCRIPT OUT OF RANGE

Constant value for array subscript is out of range. Range values are 0 to 998.

Array constants must be between 0 and 998.

Tip

Ensure that the constant value specified for the array subscript is between 0 and 998.

5182 COMPILE: REQUIRED INPUT NOT SPECIFIED

"Input Parameter Name" - Required input not specified.

There are certain input parameters in a function or function block that are required and are not optional. These parameters must be specified when calling this function or function block.

Tip

Enter the required function or function block parameter.

5183 COMPILE: MISUSED RELATIONAL OPERATOR

A logical operator (OR, XOR, or AND) must be used to separate expressions containing different relational operators.

Expressions such as:

$A < B < C < D$

Is legitimate in our implementation of ST, while the expression:

$A < B < C > D$

Is invalid in our implementation of ST. To correct the second expression, it should be rewritten as:

$A < B < C \text{ AND } C > D$

The basic rule of thumb is all operands in a relational expression must use the same relational operator. If more than one relational operator is required to form the expression then a logical expression must be used.

Tip

Separate into two or more relational expressions and separate with a logical operator.

5184 COMPILE: BAD CONSTANT

Invalid constant, value too large or improperly formed.

The constant entered is either syntactically incorrect or is too large. For more information about constant syntax rules, see the manual.

Tip

Ensure that the constant entered is syntactically correct and within the range of the data type.

5185 COMPILE: CANNOT INTERMIX OPERATORS

Cannot intermix logical operators with arithmetic operators. Resultant data types do not match.

Example of Error:

$D := A + 5 \text{ AND } B + C;$

If A and B are defined as integers, then this statement does not make logical sense and results in this error.

Tip

Ensure that logical operators and arithmetical operators do not get intermixed in the same expression.

5186 COMPILE: INTERNAL PARSER ERROR

INTERNAL ERROR: In first Pass of Structure Text. Report error code - "%d"

An internal error has occurred. Please contact Giddings and Lewis.

Tip

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of the software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings and Lewis.

5187 COMPILE: MISSING TERM

Missing “term”

In parsing an ST network the expected next term was not present.

Example

WHILE DO

Note in the example above that the expression in the WHILE is missing and the END_WHILE statement is missing. This example will generate 2 errors: Missing “expression” and Missing “END_WHILE”.

Tip

Review the syntax of the statement that has the error and then add in the missing term.

5188 COMPILE: BAD ERROR NUMBER

INTERNAL ERROR: Error number out of range: %s.

An internal error has occurred. Please contact Giddings and Lewis.

Tip

If you receive an internal error, take the following steps.

- 1 Write down the error number and the exact wording of the error message.
- 2 Note the version of the software you are using.
- 3 Save your LDO file.
- 4 Send the above information, your LDO file and all related files to Giddings and Lewis.

5189 COMPILE: NETWORK CODE SIZE EXCEEDED

Exceeded network code size limit. Limit is 4095, Actual is %s.

The binary code generated for any given network within a ladder is limited to 4095 bytes. If you receive this error the offending network must be split into 2 or more networks.

Tip

Split network into 2 or more networks.

5190 COMPILE: VARIABLE NAME TOO LONG

Variable names are limited to 63 characters.

A variable name as defined in software declarations is limited to 63 characters. The variable name entered contains more than 63 characters. See the manual for the syntax rules of a variable name.

Tip

Reduce the size of the variable name entered to 63 characters or less.

5191 COMPILE: FUNCTION NAME TOO LONG

Function name or Function Block title is limited to 8 characters.

Function names and Function Block titles are limited to 8 characters. The name entered contains more than 8 characters.

Tip

Reduce the size of the Function name or Function Block title to 8 characters or less.

5192 COMPILE: INVALID FUNCTION BLOCK NAME

Function Block instance name contains an invalid character.

Function Block instance names can only consist of characters that are alphanumeric or contain an “_” (underscore). Characters not part of this set cause this error. For more information about Function Block instance name syntax rules see the manual.

Tip

Delete the invalid characters and verify that the Function Block instance name is not misspelled.

5193 COMPILE: UNDEFINED FUNCTION BLOCK INSTANCE NAME

Function Block variable/instance name is not defined.

The specified Function Block instance name is not defined in software declarations.

Tip

Verify the Function Block instance name is not misspelled. If misspelled, correct the spelling; otherwise, define the Function Block instance name in software declarations.

5194 COMPILE: FUNCTION INPUT NOT ALLOWED

Function input parameter "EN" is not allowed in an ST network function call.

When invoking a Function in an ST network, the input parameter enable (i.e., EN) should not be specified. Functions in an ST network always execute unless there is conditional code around the Function call.

Example:

```
IF INPUT1 THEN
    B := MAX(IN1:=C, IN2:=D);
END_IF;
```

Tip

Delete this parameter from the Function call and if necessary, add a conditional statement around the Function call to conditionally execute the function.

5195 COMPILE: COUNTDOWN FOR NOT ALLOWED

Countdown FOR loop is not allowed when data type is unsigned.

Countdown FOR loops are not permissible when the data type specified for the counter variable is USINT, UINT, UDINT, or ULINT.

Tip

- Change the data type used for the counter in the FOR loop to be SINT, INT, DINT, or LINT.
- Change the conditions of the FOR loop to count up versus counting down.
- Change the constant after BY to be positive.
- Consider using a WHILE loop instead of a FOR loop. Most FOR loops can be easily converted into a WHILE loop.

7002: FUNCTION REQUIRES NPX

Function `_` requires numeric coprocessor

A warning appears when a variable name or constant has been defined with a data type that requires a CPU with an NPX processor to process and the CPU declared in the Hardware Declarations table does not have an NPX processor.

- Declare a CPU with an NPX processor in the Hardware Declarations table.
- Change the data type from REAL, LREAL, LINT, etc. to a data type that does not require a NPX processor.

7003: LIBRARY NOT FOUND

`_` was not found.

When PiCPro scanned the libraries, it found a library which it cannot now open to retrieve functions for compiling. Possibly the library has been deleted or renamed with Windows Explorer while PiCPro was running. The library has to be found for the compile to be successful.

- Check the library paths.
- Go to the libraries dialog and click OK to rescan the libraries.
- Do not make any changes to the library paths or directories from outside of PiCPro when PiCPro is running.

7004: TASK IO CONFLICT

Task `_` contains an I/O board conflict.

The I/O used by this task cannot be different than the I/O specified by the main module. Compare the hardware declarations for this task module and for the main module and ensure that the I/O declared is the same.

9001 COMPILE: NPX REQUIRED

Incompatible Variable `_//_`. Use of `_` type requires a CPU with an NPX processor.

A warning appears when a variable name or constant has been defined with a data type that requires a CPU with an NPX processor to process and the CPU declared in the Hardware Declarations table does not have an NPX processor.

- Declare a CPU with an NPX processor in the Hardware Declarations table.
- Change the data type from REAL, LREAL, LINT, etc. to a data type that does not require a NPX processor.

9002 COMPILE: NEW STRUCTURE MEMBER

'`_. _`' is a new structure member that cannot be properly assigned. A scan stopped, full module download is required.

If you add a new member to an existing structure, you must perform a full download with the scan stopped in order for PiCPro to recognize the new member.

9003 COMPILE: MODIFIED STRUCTURE MEMBER

'`_. _`' is an existing structure member that has been modified. A scan stopped, full module download is required.

If you modify an existing structure member, you must perform a full download with the scan stopped in order for PiCPro to recognize the modification.

9004 COMPILE: FUNCTION MODIFIED

'`_`' is an existing function block instantiation that has been modified. A scan stopped, full module download is required.

If you modify a function block, you must perform a full download with the scan stopped in order for PiCPro to recognize the modification.

9005 COMPILE: INVALID ATTRIBUTE

Invalid symbol attribute in '`_`'.

Ensure that the attributes you assign to any variable in the software declarations table is valid.

9006 COMPILE: SYMBOL MODIFIES

'___' is an existing variable that has been modified. A scan stopped, full module download is required.

If you modify an existing variable, you must perform a full download with the scan stopped in order for PiCPro to recognize the modification.

9007 COMPILE: NEW RETAINED

'_____' is a new RETAINED variable that cannot be properly assigned. A scan stopped, full module download is required.

If you enter a new variable with the retained attribute, you must perform a full download with the scan stopped in order for PiCPro to recognize the new retained variable.

9008 COMPILE: FUNCTION INITIALIZATION REQUIRED

'_____' calls for initialization. A remake of this library function is required, along with a full module download.

If you make changes that require initialization i.e. change initial values, add function/blocks, declarations, (Note: strings always require initialization), you must recompile the function block and perform a full module download.

Whenever a change is made that requires initialization, you must:

- Recompile the function block.
- Download the module.

9009 COMPILE: NEW/MOVED INPUT/OUTPUT

A new/moved input/output '_____' cannot be added in a patch. A remake of this library is required, along with a full download.

You have attempted to patch the module after adding a new or moving an existing input or output. This requires that you recompile the module and perform a full download. Choose **Compile | Bin File | Compile & Download** from the main menu.

9010 COMPILE: I/O NOT AN INPUT

'_____' calls for an input point '_____' in rack '_____' slot '_____' which is not configured.

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is compiled. PiCPro will detect an error if you attempt to define an input point at an output location. Ensure that discrete I/O points declared in software match the hardware declarations.

9011 COMPILE: I/O NOT AN OUTPUT

'_____' calls for an output point '_____' in rack '_____' slot '_____' which is not configured.

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is compiled. PiCPro will detect an error if you attempt to define an output point at an input location. Ensure that discrete I/O points declared in software match the hardware declarations.

9012 COMPILE: CANNOT FIND FUNCTION

Unable to locate function block '_____' in the function/block libraries.

The function block cannot be found in the function/block libraries. Compile the function block and designate the library it should be stored in.

9013 COMPILE: INSUFFICIENT DATA MEMORY

Out of data memory. Out of memory error. Your program requires more memory than is available in the Control CPU.

You have reached the memory limits on your current system.

9014 COMPILE: INPUT NOT ALLOWED

'_____', input not allowed in TASK.

The input you have entered is not allowed in a TASK. Do not mark any variables in the software declarations table with the Variable In attribute in a task LDO.

9015 COMPILE: OUTPUT NOT ALLOWED

'_____', output not allowed in task.

The output you have entered is not allowed in a task. Do not mark any variables in the software declarations table with the Variable Out attribute in a task LDO.

9016 COMPILE: FIRST IN IS NOT A BOOLEAN

'_____', the first input, is not a BOOLEAN.

UDFBs require that the first input be a boolean. Enter the BOOL data type in the software declarations table for the first input to your UDFB.

9017 COMPILE: FIRST OUTPUT IS NOT A BOOLEAN

'_____', the first output, is not a BOOLEAN.

UDFBs require that the first output be a boolean. Enter the BOOL data type in the software declarations table for the first output to your UDFB.

9018 COMPILE: INVALID FUNCTION INPUT

'_____ ' is an invalid function INPUT.

Inputs to UDFB can be any data type except function blocks.

9019 COMPILE: INVALID FUNCTION OUTPUT

'_____ ' is an invalid function output.

Outputs to UDFBs can be any data type except function blocks, structures, arrays, and strings.

9020 COMPILE: EXTERNAL INITIAL VALUE IGNORED

Initial value for EXTERNAL '_____ ' is ignored.

An initial value was entered for a variable with an EXTERNAL attribute. PiCPro ignores that value. Since UDFBs cannot have any variables marked EXTERNAL, do not apply this attribute to any UDFB variable, nor enter an initial value for it.

9022 COMPILE: NO BOOLEAN INPUTS

No UDFB Variable In attributes.

The first input to the UDFB must be a boolean and it must be assigned the variable In attribute in the software declarations table.

When creating an UDFB, ensure in the software declarations table that the first input:

- Is a boolean.
- Is assigned the Variable In attribute.

9023 COMPILE: TOO MANY INPUTS

Too many UDFB Variable In attributes.

The total number of inputs and outputs for any UDFB is 64. You have exceeded that number.

It is recommended that you keep the number of inputs and outputs to the UDFB to a minimum (under 16). More can be declared if necessary, but transferring all the inputs and outputs to and from a function block does use scan time. There is also the constraint of the 255 element matrix to consider. If you have a large number of inputs, you may want to enter them as a structure using just one input.

9025 SWD: INSERT NEW SYMBOL

If you insert a new symbol into your ladder that has not previously been declared in the software declarations table, a message will ask if you want to add the new symbol now. If you choose Yes, the software declarations table appears and you can insert the new symbol. If you choose No, you will not be able to add the undeclared symbol to your ladder until it has been declared.

9026 SWD: AUTO INSERT

This symbol cannot be inserted into a structure.

When you have entered a variable name in your ladder and are prompted to enter it in the software declarations table, you cannot insert the new variable in software declarations if the focus is on a structure member or on END_STRUCT. Move the focus anywhere else in the table or to End List and press <Insert>.

9027 SWD: DIRECT I/O

Direct I/O points cannot be assigned in the software declarations table in the following situations:

- If the variable is a member of a structure.
- If the variable has an initial value.

9028 SWD: INITIAL VALUE

Initial values cannot be assigned in the software declarations table in the following situation:

- If the variable has an I/O point assigned to it.

9029 SWD: VAR IN

You have attempted to assign the Variable In attribute to a symbol that cannot accept it. The Variable In attribute cannot be assigned to the following:

- To the member of a structure (Attributes can only be defined for the entire structure)
- To a variable with the function block type
- To a variable with an I/O point assigned

9030 SWD: VAR OUT

You have attempted to assign the Variable Out attribute to a symbol that cannot accept it. The Variable Out attribute cannot be assigned to the following:

- To the member of a structure (Attributes can only be defined for the entire structure.)
- To a variable with the function block, structure, string, or array type
- To a variable with an I/O point assigned

9031 SWD: FUNCTION

You have attempted to assign the Function data type to a symbol that cannot accept it. The Function data type cannot be assigned to the following:

- To the member of a structure

9032 SWD: STRUCTURE

You have attempted to assign the Structure data type to a symbol that cannot accept it.

9033 SWD: ARRAY

You have attempted to assign the Array data type to a symbol that cannot accept it.

9034 SWD: TYPE CHANGE

If you attempt to change the data type of a symbol with an initial value to an incompatible data type, an error message will appear.

9035 SWD: RETENTIVE

You have attempted to assign the Retentive attribute to a symbol that cannot accept it. The Retentive attribute cannot be assigned to the following:

- To a variable with an I/O point assigned

9036 SWD: NAME FORMAT

You have either left a variable unnamed or assigned a duplicate name. Every variable entered in your ladder must have a unique name assigned to it in the software declarations table.

9037 SWD: DIRECT I/O FORMAT

Direct I/O must be entered in the software declaration table as boolean data type. The address format must follow the conventions for master rack, expansion rack, block I/O, or ASIU I/O as listed below.

Master Rack, PiC CPU

The master or CPU rack is #0. Expansion racks are numbered 1 - 7, where #1 is the rack connected to the master, #2 is the rack connected to #1, etc. Slots are numbered left to right when facing the PiC rack. Slot 1 and slot 2 are reserved for the CSM/CPU module. On an expansion rack, slot 2 is reserved for the I/O driver module.

Enter four to six characters.

1 st	I or O	Input or Output
2 nd	0 - 1	First digit of module slot number* (can omit if 0)
3 rd	0 - 9	Second digit of the module slot number*
4 th	. (point)	Used as a separator
5 th	0 - 3	First digit of channel number** (can omit if 0)
6 th	0 - 9	Second digit of channel number**

* Valid slot numbers are 3 - 13.

**Valid channel numbers are 1 - 64.

Example: If the input is in the master rack at slot 4, channel 3, enter: I4.3

Expansion Rack I/O

Note: Expansion Rack I/O is only available if PiC CPU is chosen. Expansion Rack I/O is not available for any MMC CPU.

Enter six to eight characters.

1 st	I or O	Input or Output
2 nd	1 - 7	Expansion rack number
3 rd	. (point)	Used as a separator
4 th	0 - 1	First digit of module slot number* (can omit if 0)
5 th	0 - 9	Second digit of the module slot number*
6 th	. (point)	Used as a separator
7 th	0 - 3	First digit of channel number** (can omit if 0*)
8 th	0 - 9	Second digit of channel number**

*Valid slot numbers are 3 - 13. **Valid channel numbers are 1 - 64.

Example: If the output is from expansion rack #7 at slot 12, channel 10, enter: O7.12.10

Master Rack Standalone MMC CPU

1 st	I or O	Input or Output
2 nd	GEN, AUX, A1, A2, A3, A4, FAUX	Connector/Type
3 rd	. (point)	Used as a separator
4 th	0 - 4	First digit of channel number
5 th	0 - 6	Second digit channel number

ASIU I/O for MMC for PC CPU(Two Options)

1 st	I or O	Input or Output
2 nd	GEN, AUX, FAUX	Connector/Type
3 rd	1 - 8	ASIU number
4 th	. (point)	Used as a separator
5 th	0 - 4	First digit of channel number
6 th	0 - 6	Second digit channel number

1 st	I or O	Input or Output
2 nd	1 - 8	ASIU number
3 rd	A1, A2, A3, A4	Connector/Type
4 th	. (point)	Used as a separator
5 th	1 - 2	First digit of channel number

Block Expansion Rack I/O

Note: Block Expansion Rack I/O is only available for certain CPUs.
Enter five to seven characters.

1 st	B	Block
2 nd	I or O	Input or Output
3 rd	0 - 7	First digit of module number* (can omit if 0)
4 th	0 - 9	Second digit of the module number*
5 th	. (point)	Used as a separator
6 th	0 - 6	First digit of point number** (can omit if 0)
7 th	0 - 9	Second digit of point number**

*Valid block module numbers = 1 - 77.

**Valid point numbers = 1 - 64.

Example: If the input is in block I/O module 33, point 5, enter: BI33.5

Blown Fuse Status

The status of up to four fuses on an AC Output, DC Output, or a combination I/O module can be made available to your ladder program. You declare the fuses as inputs in the software declarations table. On modules having both inputs and outputs, the points are numbered sequentially (starting at 1 for inputs and starting at 1 for outputs) in the software declarations table as shown in the two examples below.

The 24 V DC Output 16 point module with four fuses			The 24V I/O 16/8 source module with two fuses		
Name	Type	I/O Point	Name	Type	I/O Point
OUT1	BOOL	O4.1	IN1	BOOL	I5.1
OUT2	BOOL	O4.2	IN2	BOOL	I5.2
.
.
OUT16	BOOL	O4.16	IN16	BOOL	I5.16
FB1	BOOL	I4.1	OUT1	BOOL	O5.1
FB2	BOOL	I4.2	OUT2	BOOL	O5.2
FB3	BOOL	I4.3	.	.	.
FB4	BOOL	I4.4	OUT8	BOOL	O5.8
			FB1	BOOL	I5.17
			FB2	BOOL	I5.18

Short Circuit Detection

The status of the short circuit detection feature of the general DC outputs for the standalone MMC, the MMC for PC ASIU, and the block I/O output modules can be made available to the ladder diagram. There is one circuit for each group of outputs. The module's hardware description defines how many output points are in a common electrical group. You declare the circuit as an input in Software Declarations.

	<u>16 point DC out</u>	<u>Mixed (8 in/ 8 DC out)</u>
Block I/O	BI#.1 or 2	BI#.9
Standalone MMC	ISGEN.1 or 2	not applicable
MMC for PC	ISGEN#.1 or 2	not applicable

For example: If the short circuit input is from a 24VDC input and output block I/O, module number 33, there is only one group of outputs, the detection input is 9, enter: BI33.9

If the short circuit input is from the GEN outputs for a standalone MMC, there are two groups of outputs (1 to 8 and 9 to 16), there can be two detection inputs: ISGEN.1 and ISGEN.2. The detection input from an ASIU would also include the ASIU number.

Fast Inputs

PiC CPU

The fast inputs available on the encoder, resolver, and servo encoder hardware modules can be declared as inputs:

	For Channel 1	For Channel 2	For Channel 3	For Channel 4
	X.1	X.3	X.5	X.7
Standalone MMC CPU				

The fast inputs available on the standalone MMC can be declared as inputs:

IFAUX.1	Channel 1	(AXIS 1)
IFAUX.2	Channel 2	(AXIS 2)
IFAUX.3	Channel 3	(AXIS 3)
IFAUX.4	Channel 4	(AXIS 4)
IFAUX.49	Channel 5	(AXIS 49)

MMC For PC CPU

The fast inputs available on the MMC for PC can be declared as inputs (:where # is the ASIU number 1-8)

IFAUX#.1	Channel 1	(AXIS 1)
IFAUX#.2	Channel 2	(AXIS 2)
IFAUX#.3	Channel 3	(AXIS 3)
IFAUX#.4	Channel 4	(AXIS 4)
IFAUX#.49	Channel 5	(AXIS 49)

9038 SWD: ARRAY FORMAT

Only variables in the software declarations table that are not yet referenced in the ladder can be made into arrays. The size of an array must be between 2 and 999 elements. Function block data type cannot be made into an array.

9039 SWD: COMPLEX NAME FORMAT

If the format of a complex name is incorrect, an error message appears.

9042 SWD: FIND NAME FORMAT

When using the Find/Find Next command in the software declarations table, the following applies:

- When searching by Name, you can enter the structure or member name, but not an element of an array or an array of structures name.
- When Whole Name match is selected, an entry is required in the Name box.

9044 SWD: INITIAL VALUE LIMITS

You have exceeded the limits on initial values in the software declarations table. The limits are:

Data Type	Minimum Value	Maximum Value
BOOL	0	1
BYTE	0	255
DATE	D#1988-1-1	D#2051-12-31
DATE_AND_TIME	DT#1988-1-1-00:00:00	DT#2051-12-31-23:59:59
DINT	-2,147,483,648	2,147,483,647
DWORD	0	4,294,967,295
FUNCTION BLOCK	N/A	N/A
INT	-32,768	32,767
LINT	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
LREAL	*	*
LWORD	0	18,446,744,073,709,551,615
REAL	*	*
SINT	-128	127
STRING	0	255 ASCII characters
STRUCT	N/A	N/A
TIME	0	T#49d17h2m47s294ms T#1193h2m47s294ms T#71582m47s294ms T#4294967s294ms T#4294967294ms
TIME_OF_DAY	TOD#00:00:00	TOD#23:59:59
UINT	0	65,535
UDINT	0	4,294,967,295
ULINT	0	9,223,372,036,854,775,807
USINT	0	255
WORD	0	65,535

*Validation done for invalid characters.

9045 SWD: STRING FORMAT

You cannot change the length of a string in the software declarations table to a value that is shorter than the length of any initial values entered.

9046 SWD: SAVE CHANGES

All your changes to the software declarations table will be lost if you do not save before exiting.

9047 SWD: FORCE LIST ENTRY LIMITATIONS

Entries in the Force List must be explicit. For example, to enter an element of an array, enter *name* (3), not *name* (index).

9048 SWD: STRING LENGTHS EXTENDED

If you edit the initial values for an array of strings and any of the strings are now longer than the declared string length, these string lengths will be automatically extended.

9049 SWD: STRING LENGTH EXTENDED

If you edit the initial values for a string and the string is now longer than the declared string length, the string length will be automatically extended.

9051 COMPILE: OI LIBRARY NOT AVAILABLE

The ASFB file for storing PiC Operator Interface information could not be located in the library directories.

If you are using the Operator Interface feature, you must have the Operator Interface ASFBs installed. They are in the *opinter.lib* supplied by Giddings & Lewis.

Check the following:

- The Operator Interface ASFBs have been installed on your PC.
- The path to the *opinter.lib* containing the ASFBs is defined.

9052 COMPILE: INVALID STRING SPECIFIED

The string you have specified is invalid.

9053 COMPILE: UDFB DISCRETE IO

Discrete I/O cannot be declared in UDFB.

You cannot declare discrete I/O points in the software declarations table of a UDFB.

Even though you cannot use discrete I/O in the finished UDFB, you may need to add discrete I/O in order to test your UDFB. If you do this, be sure to remove all discrete I/O before you compile the UDFB.

9054 COMPILE: NO BOOLEAN OUTPUTS

No UDFB Variable Out attributes.

The first output to the UDFB must be a boolean and it must be assigned the Variable Out attribute in the software declarations table.

When creating a UDFB, ensure in the software declarations table that the first output:

- Is a boolean.
- Is assigned the Variable Out attribute.

9055 COMPILE: TOO MANY OUTPUTS

Too many UDFB Variable Out attributes.

The total number of inputs and outputs for any UDFB is 64. You have exceeded that number.

It is recommended that you keep the number of inputs and outputs to the UDFB to a minimum (under 16). More can be declared if necessary, but transferring all the inputs and outputs to and from a function block does use scan time. There is also the constraint of the 255 element matrix to consider. If you have a large number of inputs, you may want to enter them as a structure using just one input.

9056 COMPILE: BLOCK I/O NOT INPUT

' _____ ' calls for an input point ' _____ ' in block ' _____ ' which is not configured.

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is compiled. PiCPro will detect an error if you attempt to define an input point at an output location. Ensure that discrete I/O points declared in software match the hardware declarations.

9057 COMPILE: BLOCK I/O NOT OUTPUT

' _____ ' calls for an output point ' _____ ' in block ' _____ ' which is not configured.

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is compiled. PiCPro will detect an error if you attempt to define an output point at an input location. Ensure that discrete I/O points declared in software match the hardware declarations.

9058 COMPILE: INVALID UDFB ATTRIBUTE

Invalid symbol attribute in ' _____ '.

There is an invalid symbol attribute in the software declarations table for the UDFB.

Ensure that the attributes you assign to any variable in the software declarations table is valid. For variables that will be inputs or outputs to the UDFB, attributes may not be external, retained, global, or discrete I/O.

9059 SWD: MODIFY NAME USED IN LADDER

When you attempt to modify the name of a symbol used in your ladder, this warning/confirmation message appears. Changing the name of a symbol means that every occurrence of the name in your ladder will be changed.

9060 SWD:DELETE

You can delete any selected item from the software declarations table that is not used in your ladder. If you want to delete a structure, you must be sure to select the entire structure.

9061 SWD: STRUCTURE ATTRIBUTES

You have attempted to add an attribute to a member of a structure.

9062 SWD: PASTE IN STRUCTURE

You cannot insert a function block, structure, or a symbol with a direct I/O point into a structure.

9063 SWD: SYMBOL EXISTS TYPE INVALID

You cannot name a ladder element with the name of an existing symbol whose data type is invalid for this ladder element.

9064 SWD: INVALID ARRAY COUNT EDIT

You cannot enter a name in your ladder which is an array without entering the array index. Conversely, you cannot enter a name with an array index if the array does not exist.

9065 SWD: INVALID CONSTANT EXPRESSION

The constant you entered is invalid.

9066 SWD: INVALID CONSTANT TYPE

The constant you entered is valid but the data type of the constant is not the data type required.

9067 SWD: FUNCTION BLOCKS UNAVAILABLE

The function block menu cannot be displayed from within the software declarations table.

9082 SWD: TOO BIG FOR SEGMENT

STRUCT or STRING too big, must be less than 64K.

A structure or string variable declared in Software Declarations exceeds the maximum size allowed (64K bytes).

Double clicking on the error in the information window will reposition you to the declaration in software declarations that is causing the problem. Edit the structure or string variable in Software Declarations and recompile.

9083 SWD: INSUFF DATA MEMORY

Out of Data Memory. A scan stop and a full compile and download is required.

This error occurred while patching in new software declarations variables and extended data memory is being used for this ladder.

There are a total of 3 data segments. They are only used as needed. Patching of declarations can only occur until the current segment is full. If the current segment becomes full when you patch in new declarations, you'll get this message. Then you must do a full compile and download of the ladder. If a full compile and download does not fix the problem, all three segments are full and you must optimize your usage of data memory by consulting with Giddings & Lewis.

9086 COMPILE: SYMBOL MODIFIED

“Variable name” cannot be inserted. It is not a valid variable name.

Variable names are limited to 63 characters where the first character is an alpha character and the remaining characters are alphanumeric or underscores.

9090 COMPILE: DUPLICATE INPUTS NOT ALLOWED

***Input:* The first four characters must be different from other inputs.**

When compiling a UDFB, the input names are truncated to four characters. The truncated input must be different from the other truncated inputs. Double click on the error in the information window to display that variable in software declarations. Rename the variable to make the first four characters of the variable name different from the other inputs.

9091 COMPILE: DUPLICATE OUTPUTS NOT ALLOWED

***Output:* The first four characters must be different from other outputs.**

When compiling a UDFB, the output names are truncated to four characters. The truncated output must be different from the other truncated outputs. Double click on the error in the information window to display that variable in software declarations. Rename the variable to make the first four characters of the variable name different from the other outputs.

Fieldbus Error Messages

9071 FIELDBUS: LIBRARY NOT AVAILABLE

Check that you have indicated the correct directory for the libraries. Select **F**ile from the main menu and then select **PiCPro L**ibraries.

9072 FIELDBUS: UNRECOGNIZED FORMAT

Your ladder's UCT file does not have the correct format. The most likely cause is manual editing of the UCT file. Re-create this file using the G&L DeviceNet™ Configurator.

9073 FIELDBUS: UNDEFINED TAG

The tag name specified in the error message cannot be found in the software declarations for the associated ladder. The tag name may have been spelled wrong when it was entered in the G&L DeviceNet™ Configurator. If so, run the Configurator and correct the problem. Or, the variable really does need to be added to your ladder's software declarations. If this is the case, go to software declarations and add a variable with the same name and type as declared in the Configurator.

9074 FIELDBUS: INVALID LOCATION

The tag name specified in the error message has an invalid IRAM location configured for it. The most likely cause of this error is manual editing of the UCT file. Re-create the file using the G&L DeviceNet™ Configurator.

9076 FIELDBUS: INVALID MASK

The tag name specified in the error message has an invalid bit mask associated with it. The valid values are 0 through 7. The most likely cause of this error is manual deletion of the UCT file. Re-create the file using the G&L DeviceNet™ Configurator.

9077 FIELDBUS: INVALID TYPE

The tag name specified in the error message has a data type that does not match the corresponding variable's type in the ladder's software declarations. Either run the G&L DeviceNet™ Configurator and change the data type for this tag name to match the type for the same variable in software declarations. Or edit the software declarations for your ladder and change the variable's type.

9078 FIELDBUS: INVALID SIZE

The tag name specified in the error message has an invalid size associated with it. The most likely cause of this error is manual editing of the UCT file. Re-create the file using the G&L DeviceNet™ Configurator.

9079 FIELDBUS: INVALID UPDATE

The tag name specified in the error message has an invalid update type associated with it. The currently supported values are associated with "polled output", "polled input", and "strobed input". The most likely cause of this error is manual editing of the UCT file. Re-create the file using the G&L DeviceNet™ Configurator.

9080 FIELDBUS: INVALID FORMAT

The specified line in your UCT is invalid. The most likely cause of this error is manual editing of the UCT file. Re-create the file using the G&L DeviceNet™ Configurator.

9081 FIELDBUS: INVALID SLOT

The specified line in your UCT has an invalid slot number associated with it. The most likely cause of this error is incorrect manual editing of the UCT file. Make sure the slot number is within the valid range of 3 through 13.

Servo Setup Error Messages

11001 SERVO: INVALID AXIS CUT COPY

The axis data you were attempting to cut or copy has been corrupted. Close the program without saving any changes and try again.

11002 SERVO: INVALID AXIS PASTE

The axis data you are attempting to paste has been corrupted. Close the program without saving any changes and try again.

11003 SERVO: INVALID AXIS LABEL

You must enter a unique axis label for each axis you are entering before proceeding. This label can be up to eight characters in length.

11004 SERVO: TUNE NO PARENT

When you attempt to view or force variables within servo setup, the servo setup file must be opened from within the parent ladder by choosing Servo function from the View menu. If you open the .SRV file using the Open command, viewing and forcing will be disabled.

11005 SERVO: TUNE NO MAIN

If you attempt to activate servo viewing and forcing when the path to the main ladder has not been defined, an error will occur.

11007 SERVO: NO MORE DATA

This is an internal software condition. Please note error number and consult factory.

11008 SERVO: AXIS INFO UNREADABLE

The axis data has been corrupted.

11019 SERVO: NO AXIS DEFINED

No axes have been defined for this servo setup function. Insert one or more axes into the servo setup program.

11020 SERVO: SOFTWARE UPPER LIMIT CALCULATION ERROR

Overflow calculating the Software Upper Limit. Check your inputs for Axis #__.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the software upper limit. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{software upper limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of -536870912 to 536870911 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11021 SERVO: SOFTWARE LOWER LIMIT CALCULATION ERROR

Overflow calculating the Software Lower Limit. Check your inputs for Axis #__.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the software lower limit. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{software lower limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of -536870912 to 536870911 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11022 SERVO: FOLLOWING ERROR LIMIT CALCULATION ERROR

Overflow calculating Excess Error Limit. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the excess error limit. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{excess error limit} * \text{FU}}{\text{LU}} = \text{N (where N must be within range of 0 to 536870911 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11023 SERVO: IN POSITION BAND CALCULATION ERROR

Overflow calculating the In Position Band. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the in position band. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{in position band} * \text{FU}}{\text{LU}} = \text{N (where N must be within range of 0 to 536870911 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11024 SERVO: ROLLOVER POSITION CALCULATION ERROR

Overflow calculating the Rollover Position. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the rollover position. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{rollover position} * \text{FU}}{\text{LU}} = \text{N (where N must be within range of 0 to 536870911 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11025 SERVO: PLUS INTEGRAL LIMIT CALCULATION ERROR

Overflow calculating Plus Integral Error Limit. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the plus integral error limit. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{plus integral error limit} * \text{FU}}{\text{LU}} = \text{N (where N must be within range of 0 to 536870911 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11026 SERVO: MINUS INTEGRAL LIMIT CALCULATION ERROR

Overflow calculating Minus Integral Error Limit. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the minus integral error limit. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{minus integral error limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of -536870912 to 0 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11027 SERVO: VELOCITY LIMIT CALCULATION ERROR

Overflow calculating Velocity Limit. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the velocity limit (entered in LU/min). The software uses the formula shown to convert this information into feedback units per iteration and checks that the result is within the acceptable range.

$$\frac{\text{velocity limit} * \text{FU} * 8 * \text{update rate}}{\text{LU}} = N \text{ (where N must be within range of 0 to 32767 FU/iteration)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11028 SERVO: INTEGRAL GAIN CALCULATION ERROR

Overflow calculating Integral Gain. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the integral gain. The value entered in setup for integral gain represents the ladder units per minute per ladder unit of following error (FE) times minutes. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{integral gain value} * 3253 * \text{update rate}}{\text{FU/min-volt}} = N \text{ (where N must be within range of 0 to 32767)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. A typical value for integral gain entered in setup is zero. If required, up to 32,000 LU/min/LUFE * min. can be entered.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11029 SERVO: PROPORTIONAL GAIN CALCULATION ERROR

Overflow calculating Proportional Gain. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU) and the proportional gain. The value entered in setup for proportional gain represents the ladder units per minute for each ladder unit of following error. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{proportional gain value} * 762601}{\text{FU/min-volt}} = N \text{ (where N must be within range of 0 to 32767 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. Typical values for proportional gain entered in setup are from 1,000 to 5,000 LU/min/LUFE.

11030 SERVO: DERIVATIVE GAIN CALCULATION ERROR

Overflow calculating Derivative Gain. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the derivative gain. The value entered in setup for derivative gain represents the ladder units per minute for each ladder unit of following error per minute. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{derivative gain value} * 178734}{\text{FU/min-volt} * \text{update rate}} = N \text{ (where N must be within range of 0 to 32767 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. A typical value for derivative gain entered in setup is zero. If required, up to 500 AU/min/AUFE/min can be entered.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11031 SERVO: FEED FORWARD CALCULATION ERROR

Overflow calculating Feed Forward Factor. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the feed forward percent. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{feed forward percent} * 457560436}{\text{FU/min-volt} * \text{update rate}} = N \text{ (where N must be within range of 0 to 524272)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11032 SERVO: CSTOP RAMP CALCULATION ERROR

Overflow calculating Controlled Stop Ramp. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the controlled stop ramp in ladder units/minute/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{cstop ramp} * \text{FU} * \text{update rate} * \text{update rate}}{\text{LU} * 937500} = N \text{ (where N must be within a range shown below)}$$

If SERCOS, then N must be within the range of 1 to 536870911.0 FU.

If Encoder, Resolver, or TTL, then N must be within the range of 1 to 67108864.0 FU.

If Analog Output or Stepper, then N must be within the range of 1 to 262144.0 FU.

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. Typical values entered in setup are 10,000 to 10,000,000 LU/min/sec, not to exceed 1023 FU/update/update.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11033 SERVO: ACCEL RAMP CALCULATION ERROR

Overflow calculating Acceleration Ramp. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the acceleration ramp in ladder units/minute/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{acceleration ramp} * \text{FU} * \text{update rate} * \text{update rate}}{\text{LU} * 937500} = N \text{ (where N must be within a range shown below)}$$

If SERCOS, then N must be within the range of 1 to 536870911.0 FU.

If Encoder, Resolver, or TTL, then N must be within the range of 1 to 67108864.0 FU.

If Analog Output or Stepper, then N must be within the range of 1 to 262144.0 FU.

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. Typical values entered in setup are 10,000 to 10,000,000 LU/min/sec not to exceed 1023 FU/update/update.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11034 SERVO: DECEL RAMP CALCULATION ERROR

Overflow calculating Deceleration Ramp. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate and the deceleration ramp in ladder units/minute/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{deceleration ramp} * \text{FU} * \text{update rate} * \text{update rate}}{\text{LU} * 937500} = \text{N (where N must be within a range shown below)}$$

If SERCOS, then N must be within the range of 1 to 536870911.0 FU.

If Encoder, Resolver, or TTL, then N must be within the range of 1 to 67108864.0 FU.

If Analog Output or Stepper, then N must be within the range of 1 to 262144.0 FU.

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. Typical values entered in setup are 10,000 to 10,000,000 LU/min/sec, not to exceed 1023 FU/update/update.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11035 SERVO: BUILDER FEED OVERRIDE

This is an internal error. Please make a note of the error number and consult factory.

11036 SERVO: SLOW FILTER CALCULATION ERROR

Overflow calculating Slow Velocity Filter. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, the slow filter, and the slow velocity filter in milliseconds. When the slow filter is non-zero, the software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$65535 * (1 - e^{-Y}) = \text{N (where N must be within range of 0 to 65535)}$$

and where:

$$-Y = -(\text{update rate/slow filter})$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11037 SERVO: FAST FILTER CALCULATION ERROR

Overflow calculating Fast Velocity Filter. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, the fast filter, and the fast velocity filter in milliseconds. When the fast filter is non-zero, the software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$65535 * (1 - e^{-W}) = \text{N (where N must be within range of 0 to 65535)}$$

and where:

$$-W = -(\text{update rate/fast filter})$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11038 SERVO: FILTER VELOCITY THRESHOLD CALCULATION ERROR

Overflow calculating Slow/Fast Velocity Threshold. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU) and ladder units (LU), the update rate, and the slow/fast velocity threshold in ladder units/minute. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{velocity threshold} * \text{FU} * \text{update rate}}{\text{LU} * 60000} = N \text{ (where N must be within range of 1 to 65535 FU)}$$

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range. Typical value entered in setup is zero, not to exceed 4095.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11039 SERVO: FILTER LAG CALCULATION ERROR

Overflow calculating Velocity Filter Lag Error. Check your inputs for Axis # __.

In servo setup you have entered scaling data for feedback units (FU), and ladder units (LU), the update rate, the velocity threshold, and the fast velocity filter in milliseconds. When the fast filter is non-zero, the software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$(V * \text{FU} * \text{update rate}) / (\text{LU} * 65535 * (1 - e^{-Y})) = N \text{ (where N must be within range of 0 to 65535)}$$

and where:

V = velocity threshold

-Y = -(update rate/slow filter)

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value, i.e. for a 2 ms update rate, use "2".

11040 SERVO: UPDATE SCALING CALCULATION ERROR

Overflow calculating Servo Update Rate. Check your inputs for Axis # __.

The selected update rate entered in milliseconds is out of range.

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

4ms is adequate for most applications. Lower values consume more CPU processing time. If too many axes have too low an update rate, the CPU may not have enough processing time available to run the user program.

11041 SERVO: D/A OFFSET CALCULATION ERROR

Overflow calculating Analog Output Offset. Check your inputs for Axis # __.

The D/A offset is out of range.

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

11042 SERVO: BUILDER NOT FOUND

If the filename is invalid when a function is made, this message appears. Please make a note of the error number and consult the factory.

11043 SERVO: BUILDER NEW FILE

If the filename is not filled in when a function is made, this message is displayed. Please make a note of the error number and consult the factory.

11044 SERVO: UPDATE DIFFERENCES CALCULATION ERROR

Invalid ratio between slowest/fastest update rates with Axis # __ and Axis # __.

The ratio between the updates of the fastest and slowest is greater than 16.

Adjust the update rate in the position loop data of axis data so that the result of the conversion calculation will fall within the acceptable range.

11102 SERVO: OVERFLOW CONSTANT JERK ERROR

Overflow calculating Constant Jerk for Move Accel/Decel.

There was an overflow calculating Constant Jerk for Move Accel/Decel.

In Servo Setup you have entered scaling data for feedback units (FU), ladder units (LU) and Ladder Units to Axis Units (LU2AU), the update rate, and the Constant Jerk for Move Accel/Decel in ladder units/minute/second/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{Constant Jerk} * \text{FU} * \text{update rate} * \text{update rate} * \text{update rate} * \text{LU2AU}}{\text{LU} * 60000 * 1000 * 1000} = N \quad (\text{where } N > 0 \text{ and } < 67108863.0)$$

Tip

Adjust the axis data information so that the results of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value (e.g. for a 2 ms update rate, use 2).

11103 SERVO: OVERFLOW MAX ACCELERATION

Overflow calculating Max Acceleration for Move Accel/Decel.

There was an overflow calculating Max Acceleration for Move Accel/Decel.

In Servo Setup you have entered scaling data for feedback units (FU) ladder units (LU) and Ladder Units to Axis Units (LU2AU), the update rate, and the Max Acceleration for Move Accel/Decel in ladder units/minute/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{Max Acceleration} * \text{FU} * \text{update rate} * \text{update rate} * \text{LU2AU}}{\text{LU} * 60000 * 1000} = N \quad (\text{where } N > 0 \text{ and } < 67108863.0)$$

Tip

Adjust the axis data information so that the results of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value (e.g. for a 2 ms update rate, use 2).

11104 SERVO: OVERFLOW CALCULATING CONSTANT JERK FOR CONTROLLED STOP DECEL

Overflow calculating Constant Jerk for Controlled Stop Decel.

There was an overflow calculating Constant Jerk for Controlled Stop Decel.

In Servo Setup you have entered scaling data for feedback units (FU), ladder units (LU) and Ladder Units to Axis Units (LU2AU), the update rate, and the Constant Jerk for Controlled Stop Decel in ladder units/minute/second/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{Constant Jerk} * \text{FU} * \text{update rate} * \text{update rate} * \text{update rate} * \text{LU2AU}}{\text{LU} * 60000 * 1000 * 1000} = N \quad (\text{where } N > 0 \text{ and } < 67108863.0)$$

Tip

Adjust the axis data information so that the results of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value (e.g. for a 2 ms update rate, use 2).

11105 SERVO: OVERFLOW CALCULATING MAX ACCELERATION FOR CONTROLLED STOP DECEL.

Overflow calculating Constant Jerk for Controlled Stop Decel.

There was an overflow calculating Max Acceleration for Controlled Stop Decel.

In Servo Setup you have entered scaling data for feedback units (FU) ladder units (LU) and Ladder Units to Axis Units (LU2AU), the update rate, and the Max Acceleration for Controlled Stop Decel in ladder units/minute/second. The software uses the formula shown to convert this information into feedback units and checks that the result is within the acceptable range.

$$\frac{\text{Max Acceleration} * \text{FU} * \text{update rate} * \text{update rate} * \text{LU2AU}}{\text{LU} * 60000 * 1000 * 1000} = N \quad (\text{where } N > 0 \text{ and } < 67108863.0)$$

Tip

Adjust the axis data information so that the results of the conversion calculation will fall within the acceptable range.

Note: When using a formula in any of the Servo error calculations calling for an update rate entry, use the number of milliseconds for an update rate value (e.g. for a 2 ms update rate, use 2).

11800 AXIS: CUT

Check that you want to cut the selected axis.

11801 AXIS: DELETE

Check that you want to delete the selected axis.

11802 AXIS: MAXIMUM ALLOWED

You have exceeded the maximum number of axes and cannot proceed with pasting your selection.

11803 AXIS: PASTE DUPLICATE ERROR

Axis # already exists. The axis will be inserted as axis #_. You can choose to continue or cancel.

11804 AXIS: PASTE INCOMPATIBLE TYPE

When you paste an axis over another axis, both axes must be the same type: servo or digitizing.

11805 AXIS: PASTE DIFFERENT I/O

The axis data you are pasting is from an axis with an input type of _ and on output type of _. You can choose to continue or cancel.

11806 AXIS: PASTE DIFFERENT INPUT TYPE

The axis data you are pasting is from an axis with an input type of _. You can choose to continue or cancel.

11807 FORCE: UPDATE CONFIG DATA WITH FORCE VALUES

Forcing values have changed. You can choose to update the axis configuration data with the current forcing values or cancel.

11808 FORCE: ENTRY ERR PGAIN

The value entered in the Force List data for proportional gain is out of the acceptable range.

11809 FORCE: ENTRY ERR IGAIN

The value entered in the Force List data for integral gain is out of the acceptable range.

11810 FORCE: ENTRY ERR DGAIN

The value entered in the Force List data for derivative gain is out of the acceptable range.

11811 FORCE: ENTRY ERR FEEDFWD

The value entered in the Force List data for feed forward percentage is out of the acceptable range.

11812 FORCE: ENTRY ERR SFILTER

The value entered in the Force List data for the slow speed filter is out of the acceptable range.

11813 FORCE: ENTRY ERR DAOFF

The value entered in the Force List data for the D/A offset is out of the acceptable range.

11850 SERVO: MMC AXES NOT SAME TYPE

This error message is displayed when attempting to compile a standalone MMC or MMC for PC SRV file which contains more than one axis type. All axes must be either D/A Encoder or SERCOS axes, but not both.

11858 SERVO: SAVE AS FAILED

This error message is displayed when using Save As with a PiC or standalone MMC SRV file to an earlier version of PiCPro.

Invalid means:

- Too many axes are present
- Invalid axis numbers are specified
- Axis type combination is wrong
- A slot, channel, ring or slave number is zero
- Ladder Units or Feedback Units exceed 65535

11859 SERVO: INVALID PIC AXIS

This error message is displayed when using Save As to save a PiC file in a previous PiCPro version (10.0 through 11.0) which contains one or more axes with a slot, channel, ring, or slave value of zero. Also displayed when attempting to compile a PiC SRV file where any of the axes has a 0 value for slot, channel, ring, or slave.

11860 SERVO: INVALID MMC DIGIT AXIS

This error is displayed when using Save As with a standalone MMC SRV file which contains a digitizing axis with a slot, channel, ring, or slave value of zero to a previous version of PiCPro (10.2 through 11.0). It is also displayed when compiling a standalone MMC SRV file where any of the axes has a 0 value for slot, channel, ring, or slave.

11861 SERVO: INVALID MMC FOR PC DIGIT AXIS

This error message is displayed when saving or compiling an MMC for PC SRV file which contains a digitizing axis with a slot, channel, ring, or slave value of zero.

11862 SERVO: INVALID MMC SERVO AXIS

This error is displayed when using Save As with a standalone MMC SRV file which contains a Servo axis with a slot, channel, ring, or slave value of zero to a previous version of PiCPro (10.2 through 11.0). It is also displayed when compiling a standalone MMC SRV file where any of the axes has a 0 value for slot, channel, ring, or slave.

11863 SERVO: INVALID MMC FOR PC SERVO AXIS

This error message is displayed when saving or compiling an MMC for PC SRV file which contains a Servo axis with a slot, channel, ring, or slave value of zero.

11864 SERVO: MMC TOO MANY AXES

This error message is displayed when using Save As with a standalone MMC SRV file which contains more axes than allowed in previous versions of PiCPro (10.2 through 11.0). It is also displayed in the Standalone MMC Edition when compiling a standalone MMC SRV file which contains more axes than are allowed.

11866 SERVO: INVALID MMC DIGIT AXIS NUMBER

This error message is displayed when using Save As with a standalone MMC SRV file which contains a digitizing axis with an invalid axis number or axis type combination to a previous version of PiCPro (10.2 through 11.0). It is also displayed in the Standalone MMC Edition when compiling a standalone MMC SRV file which contains a digitizing axis with an invalid axis number or axis type combination.

11867 SERVO: INVALID MMC SERVO AXIS NUMBER

This error message is displayed when using Save As with a standalone MMC SRV file which contains a servo axis with an invalid axis number or axis type combination to a previous version of PiCPro (10.2 through 11.0). It is also displayed in the Standalone MMC Edition when compiling a standalone MMC SRV file which contains a digitizing axis with an invalid axis number or axis type combination.

11868 SERVO: INVALID AXIS FU OR LU EXCEEDS 65535

This error message is displayed when using Save As to save an SRV file into a previous version format (10.2 through 11.0) when the file contains an axis whose feedback units or ladder units exceed 65535.

11869 SERVO: UNABLE TO ANIMATE

This error message is displayed when downloading a ladder which does not contain the STRTSRV function, viewing the servo setup function in the ladder, displaying a servo view or force list, and selecting "Viewing On" or "Forcing On".

11870 SERVO: INVALID AXIS FU/LU RATIO EXCEEDS 65535

This error message is displayed when compiling a servo setup function which contains an axis whose Feedback Units to Ladder Units ratio is larger than 65535.

11885 SERVO: INVALID MMC EDITION SERVO AXIS

This error message is displayed in the Standalone MMC Edition when using Save As with a standalone MMC SRV file which contains a Servo axis with a slot, channel, ring, or slave value of zero to a previous version of PiCPro (10.2 through 11.0). It is also displayed when compiling a standalone MMC SRV file where any of the axes has a 0 value for slot, channel, ring, or slave.

11886 SERVO: INVALID MMC EDITION DIGIT AXIS

This error message is displayed in the Standalone MMC Edition when using Save As with a standalone MMC SRV file which contains a digitizing axis with a slot, channel, ring, or slave value of zero to a previous version of PiCPro (10.2 through 11.0). It is also displayed when compiling a standalone MMC SRV file where any of the axes has a 0 value for slot, channel, ring, or slave.

SERCOS Error Messages

12002 SERCOS: IDN VALUE FORMAT

An invalid value has been specified. The acceptable ranges are listed below.

Range

Two byte value = -32768 to 32767

Four byte value = -2,147,483,648 to 2,147,483,647

12003 SERCOS: CUT DELETE

A confirmation prompt asking if you are sure you want to cut or delete this ring or slave appears. You may choose to proceed or cancel.

12004 SERCOS: FATAL ERROR CREATING TEMP FILE

The slave data could not be edited because a temporary file could not be created.

12005 SERCOS: INVALID SELECTION

You have selected more than eight slaves to be cut/copied. The limit is eight since that is the maximum that can be pasted on a ring.

You may select rings or slaves; you cannot select both. If both are selected, an error will occur.

12006 SERCOS: RING INFO UNREADABLE

The ring data is corrupted.

12007 SERCOS: SLAVE INFO UNREADABLE

The slave data is corrupted.

12008 SERCOS: NO RING

No rings have been defined. Define a ring in SERCOS setup and proceed.

12009 SERCOS: NO SLAVE

No slaves have been defined in Slot/Ring ___. Define slaves in SERCOS setup.

12010 SERCOS: DUPLICATE

There is a duplicate slot definition at _____. Slot definitions cannot be duplicated.

12011 SERCOS: DUPLICATE SLAVE

There are duplicate slave numbers entered. Each slave must have a unique number.

12012 SERCOS: NOT SEQUENTIAL

The slave numbers must be sequential.

12013 SERCOS: CLEAR DATA VALUE

A confirmation prompt asking if you are sure you want to clear the selected data. You may choose to proceed or cancel.

12014 SERCOS: INTERNAL ERROR

An internal error of flag byte out of range has occurred. Report this error to Giddings & Lewis.

12015 SERCOS: NOT INITIALIZED

SERCOS is not initialized in your ladder. The SC_INIT function block was not successfully called in your ladder.

12016 SERCOS: SERVER QUEUE IS FULL

The server queue is full. Too many requests have been made. The requested information exceeds the available buffer space in the PiC.

12017 SERCOS: SERCOS BOARD DOES NOT SUPPORT ANIMATION

The SERCOS board you have does not support animation. Contact Giddings & Lewis for new SERCOS firmware.

12018 SERCOS: ERR 3

The axis is not a SERCOS axis, is not initialized, or slot/ring/slave specification is incorrect.

- Check that your SERCOS module is in the correct slot.
- Check that the SC_INIT function block in your ladder was called successfully to initialize SERCOS.

12019 SERCOS: DRIVE ERROR

This is a drive error. Refer to your drive documentation for information on it.

12020 SERCOS: UNDEFINED INTERNAL ERROR

An undefined error has been detected. Write down the error message and the error number and contact Giddings & Lewis.

12021 SERCOS: NO ATTRIBUTE IN FILE

The attribute is missing for <IDN> in <IDN filename>.

All IDNs must have an attribute. If an attribute is missing from your drive IDN file, contact your drive manufacturer and then contact Giddings & Lewis. We may be able to help you with a work-around. If an attribute is missing from your system IDN file, contact Giddings & Lewis.

12027 SERCOS: INVALID RING SPECIFIED %s. SLOT AND RING CANNOT BE 0

A ring in your file has a slot or ring value set to zero. You must specify a valid slot or ring value before you can save this file in a previous version format (10.2 through 11.0) or compile the function.

12029 SERCOS: INVALID SLAVE SPECIFIED %s. SLAVE NUMBER CANNOT BE 0

A slave in your file has slave number of zero. You must specify a valid slave number before you can save this file in a previous version format (10.2 through 11.0) or compile the function.

12030 SERCOS: TOO MANY RINGS HAVE BEEN SPECIFIED

The file has too many rings specified for the CPU type. The ring(s) must be deleted, or the CPU type changed before the file can be saved in a previous version format (10.2 through 11.0) or the function compiled.

12031 SERCOS: TOO MANY SLAVES HAVE BEEN SPECIFIED

The file has too many slaves specified for the CPU type. The slave(s) must be deleted, or the CPU type changed before the file can be saved in a previous version format (10.2 through 11.0) or the function compiled.

APPENDIX C - PiCPro Reference Card - Errors/Variables

Reference Card

#	STRTSERV func errs
0	No error
1	Bad user function data
2	Not enough low memory
3	Feedback module(s) not found
4	Analog module(s) not found

Word output from STATUSV function		
Characteristic	Binary value	Hex
Move started	00000000 0000000(1)	0001
Fast input occurred	00000000 000000(1)0	0002
Fast input on	00000000 00000(1)00	0004
Good mark detected	00000000 0000(1)000	0008
Bad mark detected	00000000 000(1)0000	0010
DIST + TOLR exceeded	00000000 00(1)00000	0020
Fast input rising	00000000 0(1)000000	0040

E-stop Error	Bit Location (low byte)								Hex Value (in LDO)
	8	7	6	5	4	3	2	1	
SERCOS synchronization error			E						8020 32800
SERCOS drive E-stop				E					8010 32784
User-set					E				8008 32776
Overflow error						E			8004 32772
Excess error							E		8002 32770
Loss of feedback								E	8001 32769

C-stop Error	Bit Location (low byte)								Hex Value (in LDO)
	8	7	6	5	4	3	2	1	
Part reference error	E								8080 32896
Part reference dimension error		E							8040 32832
Distance or position move dimension error			E						8020 32800
Feedrate error				E					8010 32784
Machine reference error					E				8008 32776
User-defined C-stop						E			8004 32772
Negative software limit exceeded							E		8002 32770
Positive software limit exceeded								E	8001 32769

Programming Error	Bit Location (high byte)								Hex Value (in LDO)
	8	7	6	5	4	3	2	1	
Set whenever a P error occurs.	X								8000 32768
Master axis beyond start point					E				8800 34816
Slave axis beyond start point						E			8400 33792
Master distance not valid							E		8200 33280
Slave distance not valid								E	8100 33024

Programming Error	Bit Location (low byte)								Hex Value (in LDO)
	8	7	6	5	4	3	2	1	
The FAST axis in the FAST_QUE function moved too far in wrong direction	E								8080 32896
Profile number not found		E							8040 32832
Master axis not available			E						8020 32800
Master start position for lock on								E	8001 32769

Variables used with the READ_SV (**Read** column) and WRITE_SV (**Write** column) functions. These variables are used with servo (**S**), time (**T**), and/or digitizing (**D**) axes.

V#	Variable Description	Read	Write	V#	Variable Description	Read	Write
1	Actual position	S, T, D	T	29	Reference switch position	S	
2	Move Type 11 pos 18 ratiopro 12 dist 20 ratiosyn/gr 14 vel st 22 ratiocam 16 lad ref 23 ratioslp or fast ref 24 ratio real	S		30	Filter time constant	S	S
3	Command position	S, D		31	Filter error limit	S	S
4	Position error	S		32	Velocity compensation flag	S	S
5	Slow velocity filter error	S		33	Filter lag	S	
6	Command velocity	S, T	T	34	Position change over several interrupts	S, D	S, D
7	Position change	S, D		35	Part reference offset	S, D	
8	Feedback last	S, D		36	Software upper limit	S	S
9	Fast input position	S, D		37	Software lower limit	S	S
10	Reg/ref position change	S, D		38	Commanded position (before slow velocity filter)	S, D	
11	Consecutive bad marks	S, D	S, D	39	Following error limit	S	S
12	Rollover on position	S, T, D	S, T, D	40	In-position band	S	S
13	Slave offset incremental	S	S	41	Current segment number	S	
14	Master offset incremental	S	S	42	Slave distance into segment	S	
15	Slave offset absolute	S	S	43	Master distance into segment	S	
16	Master offset absolute	S	S	44	Set user iteration command	S	S
17	Slave offset filter		S	45	User iteration command	S	S
18	Master offset filter		S	46	Set user PID command	S	S
19	Fast input direction		S, D	47	User PID command	S	S
20	Fast input distance	S, D		48	Disable servo software	S	S
21	Reversal not allowed	S	S	49	(Reserved)		
22	Fast input position (software)	S, D		50	Override endlimit check	S	S
23	Position (software) of axis 1 with fast input on axis 2	S, D	S, D	51	SERCOS command position	S	
24	Registration switch	S, D	S, D	55	Queued move type 11 position, 12 distance, 14 velocity start, 16 fast/ladder reference, 18 ratiopro, 20ratiosyn/gear, 22 ratiocam, 23 ratioslp, 24 ratioreal	S	
25	Fast queuing	S	S	58	SERCOS Modulo Value	S	
26	Synchronized slave start	S, D, T	S, D, T	59	Command Position Based Master/Slave	S	S
27	Backlash compensation	S	S	60	Servo Axis S-Curve interpolation	S	S
28	TTL feedback	S, D	S, D	61	SERCOS Velocity Compensation	S	S
				62	Velocity Compensation Filter	S	S
				63	Resumable E-Stop Allow	S	
				64	Resume Distance	S	

SERCOS Errors

The errors listed below can appear at the ERR output of certain SERCOS functions/function blocks.

ERR #	Description
0	No error
1	IDN queue was busy when called.
2	Quantity specified in the .AVAIL structure member is not large enough for received data.
3	Axis is not initialized, is not a SERCOS axis, or the slot/ring/slave specification is incorrect.
4	Invalid data in DATA input structure
5	Error reset function could not be completed.
6	SERCOS ring 1 busy
7	SERCOS ring 2 busy
8	SERCOS ring 1 configuration size error
9	SERCOS ring 2 configuration size error
10	Function block enabled while already in process
11	Bit 3 or bit 8 set in the procedure command acknowledgment (data status) Either operation data invalid or procedure command error
12	Not enough pool memory available
13	Change bit in status word was zero after reference complete.
14	The IDN queue was cleared during an IDN transfer, typically caused by calling the SC_INIT function while an IDN is being read or written.
15	SERCOS module is unavailable for IDN transfer because the phase-to-phase transition in progress is between phase 2 and phase 4.
16	Slave response timed out
17	The SERCOS module did not receive an expected AT response. SERCOS cable may be disconnected.
18	Number of SERCOS slots equals zero.
19	The SERCOS module did not receive an expected MDT response. SERCOS cable may be disconnected.
20	Phase 0 detected that the ring is not complete. The optic cable could be open or drive turned off.
21	The SERCOS module firmware is outdated for the features requested from a newer version of the motion library.
22	The SERCOS module firmware is a newer version and the motion library is outdated and unable to interface.
23	The version of PiCPro used to create the SERCOS setup data is outdated for the features requested from the library or the SERCOS module firmware.
24	The version of PiCPro used to create the SERCOS setup data is a newer version and the library is unable to interface.
25	A two-ring SERCOS module was specified in SERCOS setup but the module is a one-ring SERCOS module.
26	Invalid PRB input on the SCA_PBIT or SCA_RFIT function blocks or invalid OPTN input on the SCA_RFIT function block.
27	The SERCOS setup data was configured for a different CPU (PiC, MMC, or MMC for PC).
28	The SERCOS ring is not currently halted in phase 2. SERCOS Setup may not have specified "Pause after Phase 2".
29	The axis is in Resume Mode or Resumable E-Stop Allow (READ_SV/WRITE_SV Variable 63) is set
30	The drive status word (bit 13=1) indicates an error.
31	An E-stop condition exists for this axis in the PiC900.
32	Incorrect phase number, contact Giddings & Lewis.
33	Incorrect address error, contact Giddings & Lewis.
34	Incorrect AT number error, contact Giddings & Lewis.
35	Variable 48 is set to 1 and you attempt to close the loop
36	OPTN input is invalid.
37	The quantity specified in the .AVAIL structure member is not large enough for the received data. The actual size of the received data is returned in the .ACTUAL structure member. This error is reported by the motion library software
38	Open loop was requested while SCA_CLOS was in progress.
48	Service channel not ready when attempt to send/receive non-cyclic data
49	No data to send or receive
50	The value of the .SIZE member of the TASK input structure does not match the byte count in the SERCOS module.
51	The value of the .SIZE member of the MAIN input structure does not match the byte count in the SERCOS module.
65	Error occurred calculating when MDT should occur.
66	Error occurred calculating when drive data valid.
67	Error occurred calculating when feedback data valid.
68	Error occurred calculating total time required for communication cycle.
69	Error occurred calculating cyclic data memory for SERCON processor.

70	Error occurred calculating cyclic data memory for internal memory map.
71	Error occurred calculating service channel memory map.
72	Incorrect ring error, contact Giddings & Lewis.
73	Incorrect AT count error, contact Giddings & Lewis.
74	CPU on SERCOS module has too many tasks during update.
128	Slave error occurred. Read SERR output to identify error. The SLV output indicates the slave number.
136	Slave will not respond in phase 1. The SLV output indicates the slave number.
144	Procedure command error - The slave number can be viewed at the SLV output and the IDN number at the IDN output.
152	CRC error. The bit pattern received by the SERCOS receiver is corrupted.

The errors listed below can appear at the SERR output of certain SERCOS functions/function blocks.

SERR #	Description	SERR #	Description
4097	This IDN does not exist.	20482	The minimum value transmission is too short.
4105	The data for this IDN may not be accessed.	20483	The minimum value transmission is too long.
8193	The name does not exist.	20484	The minimum value may not be changed.
8194	The name transmission is too short.	20485	The minimum value is write-protected.
8195	The name transmission is too long.	24577	The maximum value does not exist.
8196	The name may not be changed.	24578	The maximum value transmission is too short.
8197	The name is write-protected.	24579	The maximum value transmission is too long.
12290	The attribute transmission is too short.	24580	The maximum value may not be changed.
12291	The attribute transmission is too long.	24581	The maximum value is write-protected.
12292	The attribute is write-protected at this time.	28674	The data is too short.
16385	The units do not exist.	28675	The data is too long.
16386	The units transmission is too short.	28676	The data may not be changed.
16387	The units transmission is too long.	28677	The data is write-protected at this time.
16388	The units may not be changed.	28678	The data is smaller than the minimum value.
16389	The units are write-protected at this time.	28679	The data is larger than the maximum value.
20481	The minimum value does not exist.	28680	The bit pattern for this IDN is invalid.

Data Capture

Axis variables that can be captured on a servo interrupt basis with the CAPTINIT function.

VAR	Description	Type	VAR	Description	Type
1	Actual Position	DINT	7	Position change	INT
2	Fast Input	BYTE	8	Feedback position	DINT
3	Commanded position	DINT	9	Prefilter commanded	DINT
4	Position error	DINT	10	Prefilter command change	INT
5	Slow velocity filter error	INT	11	Remaining master offset	DINT
6	Command change	INT	12	Remaining slave offset	DINT

Error numbers/descriptions for CAPTINIT function ERR output are listed below.

0	No error
1	The CAPTSTAT function has not stopped capturing data from a previous data capture initialization.
2	An axis number in the structure is invalid.
3	The limit of eight variables in the array of structures has been exceeded.
4	Parameter number in the structure is out of range.
5	The CAPINIT function was called before the STRTSERV function was called.

APPENDIX D - Stepper Reference Card

Reference Card

#	Profile Commands	Range
1	Distance move	±2,147,352,575 steps
2	Position move	±2,147,352,575 steps
3	Velocity move	±1,000,000 steps/sec
4	Set maximum velocity	1-1,000,000 steps/sec
5	Set acc/dec rate	1-16,777,215 steps/sec/sec
6	Set reference	±2,147,352,575 steps
7	Pause	N/A

#	Control Words
1	Enable profile
2	Pause profile
3	Continue profile
4	E-stop
5	C-stop
6	Step/direction mode (default)
7	CW/CCW mode

Word output from STEPSTAT function			
Characteristic	Binary value	Decimal	Hex
Profile enabled	00000000 0000000x(1)	1	0001
Profile paused	00000000 000000x(1)0	2	0002
At velocity	00000000 00000x(1)00	4	0004
Que empty	00000000 0000x(1)000	8	0008
Que full	00000000 000x(1)0000	16	0010
Control word not processed	00000000 00x(1)00000	32	0020

Errors displayed in the .ERROR member of stepper structure	
Error	#
No error	0
Invalid rack number or remote rack not available	1
Invalid slot number	2
Module not found at rack and slot location or not enough channels on module	3
Invalid command number	4
Invalid data for the command	5
Invalid control number	6
Stepper function called before STEPINIT function called	7

NOTES

APPENDIX E - Diagnostic LED Error Codes

Error Codes

While the control is running, the DIAG LED on the CPU module (status software with MMC for PC) will flash a three digit code signal if there is an error. For example, if there is a long pause-flash-pause-flash-flash-pause-flash-flash-flash-long pause, the code is 123. The errors are described below

Code	Error	Description
122	No math coprocessor	Attempted to perform floating point operation with no math coprocessor installed on the CPU.
123	Scan too long	A ladder scan loss has occurred because the CPU takes more than 200 ms to scan the application program. Whenever the scan light is out, the discrete outputs go to the OFF state and the analog outputs are zeroed.
124	Excessive overhead	The system overhead update time is excessive.
125	Insufficient memory	There is insufficient memory on the CPU to run the current program.
126	No hardware bit memory	There is no bit memory installed on the CPU and the program requires it.
127	No software bit memory	There is no bit memory capability via software and the program requires it.
222	Driver error	No driver support on the CPU for the I/O module. Update your system EPROMs.
22_	Master rack error	The I/O modules in the master rack do not match what was declared in the hardware master declaration table. The number of flashes in the third digit identifies the slot number that is in error.
231*	No daughter board	There is no communications daughter board installed on the CPU when attempting to do expansion I/O communications.
232*	Communications error	A failure has occurred in remote I/O communications.
233*	Number of racks error	The number of expansion racks in the system does not match the number of expansion racks declared in the expansion hardware declaration table.
24_	Master rack error	The I/O modules in the MMC for PC master rack do not match what was declared in the hardware master declaration table. The number in the third digit identifies the slot number that is in error.
25_	ASIU Error	The ASIU modules found do not match what was declared in the hardware master declaration table. One or more ASIU modules may have the same address switch setting. The number in the third digit indicates the ASIU address.
3__*	Expansion rack error	The I/O modules in the expansion rack(s) or the block I/O modules do not match what was declared in the expansion hardware declaration table. For rack expansion: The number of flashes in the second digit indicates the remote rack (1 through 8). The number of flashes in the third digit indicates the slot number. For block I/O modules: The number of flashes in the second and third digits indicates the block I/O module (01 through 77). The second digit will flash a 1 - 7, 10 for 0. The third digit will flash a 1 - 9, 10 for 0. For example, if the second digit flashes 3 times and the third digit flashes 10 times, the module is 30 .
260	ASIU Data Overrun Error	The specified number of axes at the specified update rate exceeds the throughput capacity of CPU to ASIU communications.

* Errors connected with I/O expansion. Refer to the I/O Driver Module write-up in the PiC900 Hardware Manual for more information.

NOTES

APPENDIX F - IBM ASCII Chart

ASCII Chart

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
00	0x00	NUL	32	0x20	SPC	64	0x40	@	96	0x60	`
01	0x01	☐	33	0x21	!	65	0x41	A	97	0x61	a
02	0x02	☐	34	0x22	"	66	0x42	B	98	0x62	b
03	0x03	©	35	0x23	#	67	0x43	C	99	0x63	c
04	0x04	®	36	0x24	\$	68	0x44	D	100	0x64	d
05	0x05	β	37	0x25	%	69	0x45	E	101	0x65	e
06	0x06	™	38	0x26	&	70	0x46	F	102	0x66	f
07	0x07	BEL	39	0x27	'	71	0x47	G	103	0x67	g
08	0x08	BS	40	0x28	(72	0x48	H	104	0x68	h
09	0x09	HT	41	0x29)	73	0x49	I	105	0x69	i
10	0x0A	LF	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	VT	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	FF	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	CR	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	♪	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	j	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	~	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	ÿ	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	þ	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	!	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	¶	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	§	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	-	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	þ	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	¡	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	Ø	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	Æ	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	¨	59	0x3B	;	91	0x5B	[123	0x7B	{
28	0x1C	¿	60	0x3C	<	92	0x5C	\	124	0x7C	:
29	0x1D	◀	61	0x3D	=	93	0x5D]	125	0x7D	}
30	0x1E	s	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	t	63	0x3F	?	95	0x5F	_	127	0x7F	>

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	0x80	Ç	160	0xA0	à	192	0xC0	ı	224	0xE0	‡
129	0x81	ü	161	0xA1	í	193	0xC1	ì	225	0xE1	·
130	0x82	é	162	0xA2	ó	194	0xC2	ı̇	226	0xE2	,
131	0x83	â	163	0xA3	ú	195	0xC3	Đ	227	0xE3	„
132	0x84	ä	164	0xA4	ñ	196	0xC4	f	228	0xE4	‰
133	0x85	à	165	0xA5	Ñ	197	0xC5	Ý	229	0xE5	Â
134	0x86	â	166	0xA6	ª	198	0xC6	ý	230	0xE6	Ê
135	0x87	ç	167	0xA7	º	199	0xC7	«	231	0xE7	Á
136	0x88	ê	168	0xA8	ı̇	200	0xC8	»	232	0xE8	Ë
137	0x89	ë	169	0xA9	©	201	0xC9	...	233	0xE9	È
138	0x8A	è	170	0xAA	™	202	0xCA	þ	234	0xEA	¾
139	0x8B	ï	171	0xAB	´	203	0xCB	À	235	0xEB	Ï
140	0x8C	î	172	0xAC	¨	204	0xCC	Ã	236	0xEC	×
141	0x8D	ì	173	0xAD	ı̇	205	0xCD	Õ	237	0xED	ý
142	0x8E	Ä	174	0xAE	«	206	0xCE	Œ	238	0xEE	Œ
143	0x8F	Å	175	0xAF	»	207	0xCF	œ	239	0xEF	«
144	0x90	É	176	0xB0	×	208	0xD0	–	240	0xF0	½
145	0x91	Æ	177	0xB1	±	209	0xD1	—	241	0xF1	±
146	0x92	Æ	178	0xB2	ð	210	0xD2	“	242	0xF2	Š
147	0x93	ô	179	0xB3	Š	211	0xD3	”	243	0xF3	ð
148	0x94	ö	180	0xB4	¥	212	0xD4	‘	244	0xF4	Û
149	0x95	ó	181	0xB5	μ	213	0xD5	’	245	0xF5	€
150	0x96	û	182	0xB6	¹	214	0xD6	÷	246	0xF6	³
151	0x97	ù	183	0xB7	²	215	0xD7	þ	247	0xF7	Ý
152	0x98	ÿ	184	0xB8	³	216	0xD8	ÿ	248	0xF8	º
153	0x99	Ö	185	0xB9	¼	217	0xD9	ÿ̇	249	0xF9	•
154	0x9A	Û	186	0xBA	½	218	0xDA	/	250	0xFA	Ž
155	0x9B	ç	187	0xBB	ª	219	0xDB	□	251	0xFB	Đ
156	0x9C	£	188	0xBC	º	220	0xDC	<	252	0xFC	,
157	0x9D	¥	189	0xBD	¾	221	0xDD	>	253	0xFD	”
158	0x9E	û	190	0xBE	æ	222	0xDE	?	254	0xFE	ž
159	0x9F	ü	191	0xBF	ø	223	0xDF	?	255	0xFF	

APPENDIX G - Time Axes

Using a Time Axis

The time axis feature allows a servo axis to be slaved to time instead of a physical master position transducer. All the master/slave functions can be used with a time axis.

If time axes are going to be used they must be defined in Servo Setup. There are four axis numbers reserved for a time based master; 25, 26, 27, and 28. This is the number used to identify the master time axis on the input to a master/slave function i.e. `RATIO_GR`.

The `S_CURVE` or `ACC_DEC` functions or the command velocity variable 6 can be used to move a time axis. The time axis can be manipulated with variables 1, 12, and 26. Use the `WRITE_SV` and `READ_SV` functions to work with these variables.

Referencing a Time Axis

Actual Position (Variable 1)

The actual position variable allows you to read the position of the time axis or change the current position by writing a value with the `WRITE_SV` function. Range: +2,147,483,647 to -2,147,836,648 counts.

Controlling Time Axis Velocity

You can use either the `S-CURVE` or `ACC_DEC` functions or the command velocity variable 6 to move a time axis.

`S_CURVE` and `ACC_DEC` Functions

When using the `S_CURVE` or `ACC_DEC` functions with a time axis, you can use the distance, position, or velocity moves to move the axis. The `S_CURVE` or `ACC_DEC` functions must be called first when using these moves. See the `S_CURVE` and `ACC_DEC` descriptions in the Function/Function Block Reference Guide.

Command Velocity (Variable 6)

If you are not using the `S_CURVE` or `ACC_DEC` functions, the command velocity variable can be used to define how fast the time axis will travel. It is programmed in counts per second. When the `WRITE_SV` function is called with variable 6, the time axis will step to the programmed velocity.

For example, if a value of 1000 is programmed as the number of counts per second for the velocity, then the time axis would move one count in one millisecond. If the master distance (MDST) was set at 1000 and the slave distance (SDST) was set at 2000 in the `RATIO_GR` function, it would take the slave axis one second to move 2000 units.

By entering a zero, the time axis is stopped. This provides the ability to synchronize multiple slave axes. You call all the moves you want to synchronize and then write a non-zero value to variable 6. All the axes will begin motion at the same time.

NOTE: In order for all slave axes to start at the same time, the master start position of any master/slave move with a MSTR input would have to have the same value (or zero) at its MSTR input. If the option to ignore master start is selected, the slave axes will start when the master axis begins to move.

An alternative method for synchronizing slave starts is to use variable 26.

Range: +/-2,000,000 counts/sec.

Rollover on Position with a Time Axis

The rollover on position variable allows you to select where the time axis will reset to zero. The variable is entered in ladder units.

Note: Without rollover on position, when 2,147,483,647 is reached, the next number will be -2,147,483,648. The count continues to zero and back up to 2,147,483,647, etc.

Range: 1 to 536,870,912 counts (Entering a zero turns rollover on position off.)

Synchronizing Slave Axes with a Time Axis

The synchronized slave start variable allows you to tell the time axis which of its slave axes must be queued up before any of them begin their move. Each slave axis you want to synchronize is identified by setting a bit in a DINT using the lower 16 bits where the LSB = axis 1 and the MSB = axis 16. When the last set axis has been queued, all the slave axes will begin their move on the next interrupt.

The WRITE_SV function with variable 26 must be called before the move. It can be called again when you want to identify a different set of synchronized slave axes. Change the bits only after the slave axes identified in the first WRITE_SV function have started to move.

Writing a zero to variable 26 clears all identified axes. The READ_SV function can be used to read the number of slave axes being synchronized.

APPENDIX H - Stepper Axis Module Notes

Introduction

Stepper motors can be controlled by:

- The stepper motor control module (SMCM) using the PiCPro stepper functions.

or

- The stepper axis module (SAM) using servo setup and the move types available in the PiCPro motion library. It can be a master or a slave in the application.

This appendix covers the stepper axis module. Any move type from the motion library can be used to perform motion control with the stepper except those move types requiring a fast input. There is no feedback from the stepper axis module.

Servo setup is used to set up the stepper axis module and create a start servo function. Once all the setup data has been entered, compile the servo function. This function will be stored in the servo library and can then be called in your ladder program to initialize the setup data for your application.

Notes on using Motion Library Functions and Variables with the Stepper

This section summarizes things you should be aware of when using the stepper and the motion library of PiCPro.

READ_SV/WRITE_SV Functions

These READ_SV and WRITE_SV variables cannot be used when using the stepper on an axis.

Var #	Name
4	Position error
9	Fast input position (Hardware)
10	Registration/referencing position change
11	Consecutive bad marks
19	Fast input direction
20	Fast input distance
24	Registration switch
27	Backlash compensation
28	TTL feedback
29	Reference switch position
46	Set user PID command
47	User PID command
48	Disable servo software

All other READ_SV and WRITE_SV variables can be used with a stepper.

NOTE: Feedback units are stepper units. Ladder units may still be used.

TUNERead/TUNEWRIT Functions

The filter variable is the only one that can be read and written by these functions when using a stepper axis. The remaining TUNERead/TUNEWRIT variables cannot be used with a stepper axis.

CLOSLOOP, OPENLOOP, REGIST, and MEASURE Functions

These functions cannot be used on a stepper axis.

Reference-Related Functions

The reference-related functions in PiCPro on the left below cannot be used with a stepper axis. The functions on the right can be used with a stepper axis.

Not Available with Stepper	Available with Stepper
FAST_REF	PART_REF
LAD_REF	PART_CLP
REF_DNE	
REF_END	

STRTSERV Function

Call STRTSERV to initialize the stepper axis and begin the stepper motion.

ERRORS

There is no loss of feedback or excess error. If an E-stop error occurs, the command to the stepper will be zeroed.

APPENDIX I - Toolbar Buttons

Below is a list of the toolbars available in PiCPro.











- Standard
- Basic Online Operations
- Advanced Operations
- Ladder
- View Navigator
- Functions
- Compiler
- Structured Text Tools
- Insert New Network

Toolbars and tool buttons are shown below.

Standard Toolbar













The tools available on the standard toolbar are listed below.

	New Document
	Open Document
	Save Document
	Cut
	Copy
	Paste
	Print
	About
	Help
	Options






Basic Online Operations Toolbar



-  Stop the Scan
-  Run One Scan
-  Hot Restart (Red Arrow)
-  Warm Restart (Yellow Arrow)
-  Cold Restart (Blue Arrow)
-  Backup User Program
-  Backup Ramdisk
-  Restore User Program
-  Reset Power
-  PiC Status



















Advanced Operations Toolbar



-  Toggle Animation
-  Toggle Forcing
-  Group Enable
-  Update Force Values
-  Abort last patch








Ladder Toolbar



	Pointer
	Normally Open Contact
	Normally Closed Contact
	Normally Open Positive Transition Contact
	Normally Closed Positive Transition Contact
	Normally Open Negative Transition Contact
	Normally Closed Negative Transition Contact
	Energized Coil
	De-energized Coil
	Set Coil
	Reset Coil
	Horizontal Wire
	Vertical Wire
	Combination Wire
	Point to Point Wire
	Jump to Label
	Jump to Subroutine
	Return from Jump

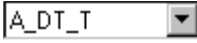




View Navigator Toolbar



-  Display Software Declarations
-  Displays Hardware Declarations
-  Displays View List
-  Update Forcing List
-  Toggles Long Name Display
-  Zoom Display Out
-  Zoom Display In






Functions Toolbar



-  Function/Function Block List
-  Place the Selected Function
-  Data In
-  Data In Inverted
-  Data Out










Compiler Toolbar



-  Compile Bin File
-  Compile and Download Bin File
-  Dump a Hex-86 File
-  Build a Task
-  Build a UDFB



Structured Text Tools Toolbar



	IF-THEN
	IF-THEN-ELSE
	ELSEIF-THEN-ELSE
	CASE
	FOR-DO
	WHILE-DO
	REPEAT-UNTIL
	Insert Variable
	Check ST Syntax

Insert New Network Toolbar



	Insert Ladder Network
	Insert Structured Text Network

NOTES

APPENDIX J - Module Filenames

The following tables list the files that are created during the programming process.

Extensions	Type of File	Created/Updated Upon:
LDO Ladder Diagram Object	Ladder or network logic file	Save
LBK Ladder BacKup	Backup copy of the LDO file - is the version previous to the version created with the last Save	Save
REM REMArks	The documentation or comments for all networks in a module	Save
RBK Remarks BacKup	Backup copy of the remarks file	Save
FRC FoRCing	List of variables that can be forced and their values	Save, if forced variables are designated
RTD Real Time Display	List of variable in the View Variables List	Edit view list
LST LiST	Print file containing LDO, REM, cross-reference, formatted for printing	Print
HEX HEXadecimal	Hex representation of the LDO file	Hex compile
BIN BINary	Compiled (PiC language) LDO file - cannot be converted back into uncompiled state for animating, etc.	Download
SRV SeRVo setup	Setup data for all servo axes used in application	Save
SVT Servo View/Tune	Viewing and tuning data for the servo axes in application	Save
SRC SeRCos setup	SERCOS setup data for SERCOS axes used in application	Save
SCT SerCos View/Tune	Viewing and tuning data for the SERCOS axes used in application	Save
LIB LiBrary	User-defined functions/function blocks to be used in LDO	Make function
BAK library BacKup	Backup copy of LIB file - the version created with the last save	Save
DPL DePendency List	List of all the files the currently loaded module depends upon	Build dependency list

PRJ PRoJect	Contains a project generated from Project Manager	New project
G&L Giddings & Lewis	Compresses the project file created in Project Manager	New project
OID Operator Interface Definition	Used to support the operator interface or Ethernet module	Compile when “Construct Data File” is checked in Settings
MAP MAP	Produces a readable symbol map file	Compile when “Generate Symbol Map” is checked in Settings
PPR PicPro Restore	Used to restore program to the control	“Save to File” is chosen in the PiC Restore dialog
HTM HTML	HTML Help file you create to define your UDFBs	Edit Function Block Help
UCT Universal Communication Text	Used by PicPro when downloading your LDO	Save from the G&L DeviceNet Configuration Tool
UCP Universal Communication Personality	Used by the configuration tool when downloading the personality file to the DeviceNet scanner module.	Save from the G&L DeviceNet Configuration Tool

APPENDIX K - Service Pack Installation

Service Pack Installation

The Wise™Update installation update program allows you to locate PiCPro Service Packs at a specified Giddings & Lewis CMS Internet website location. This program is automatically installed when PiCPro is installed.

Note: Connection to the Internet is required to use this program.

This update program can be used on either a case by case manual basis or be set to work automatically.

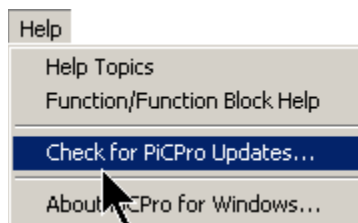
Service Pack Update Installation - Manual Method

The update program can be started manually in one of three ways:

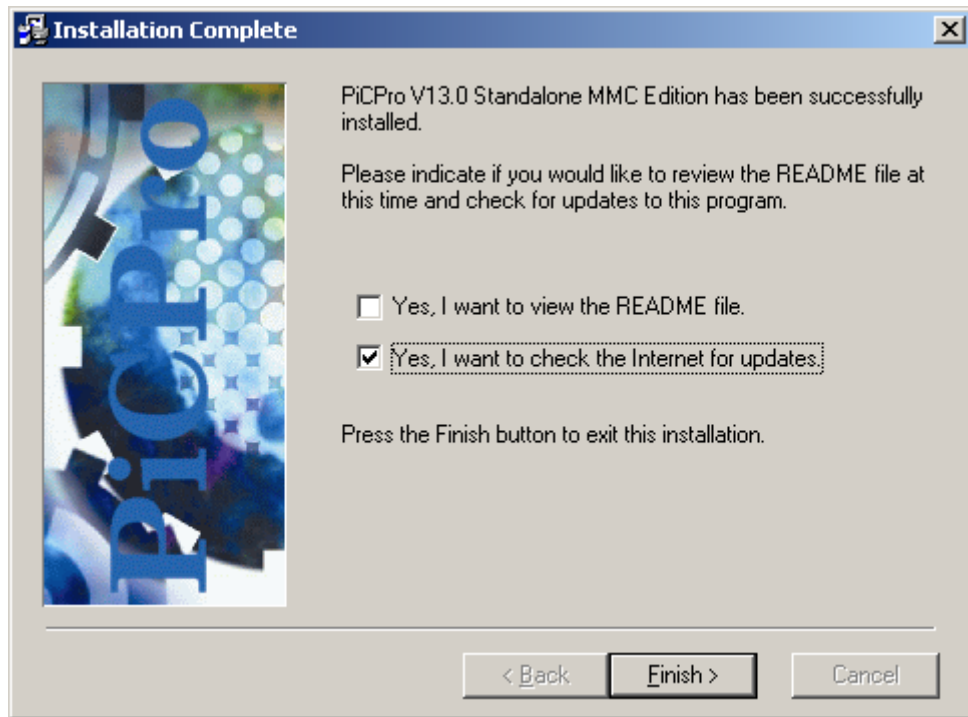
- From the Windows Start menu select:
Start | Programs | PiCPro for Windows | Check for PiCPro Vxx.x Updates.



- From within PiCPro, using **Help | Check for Updates**. If the updated program is started from the PiCPro **Help** menu, it can be run at the same time as PiCPro.



- Or after PiCPro installation is complete by using a check box.

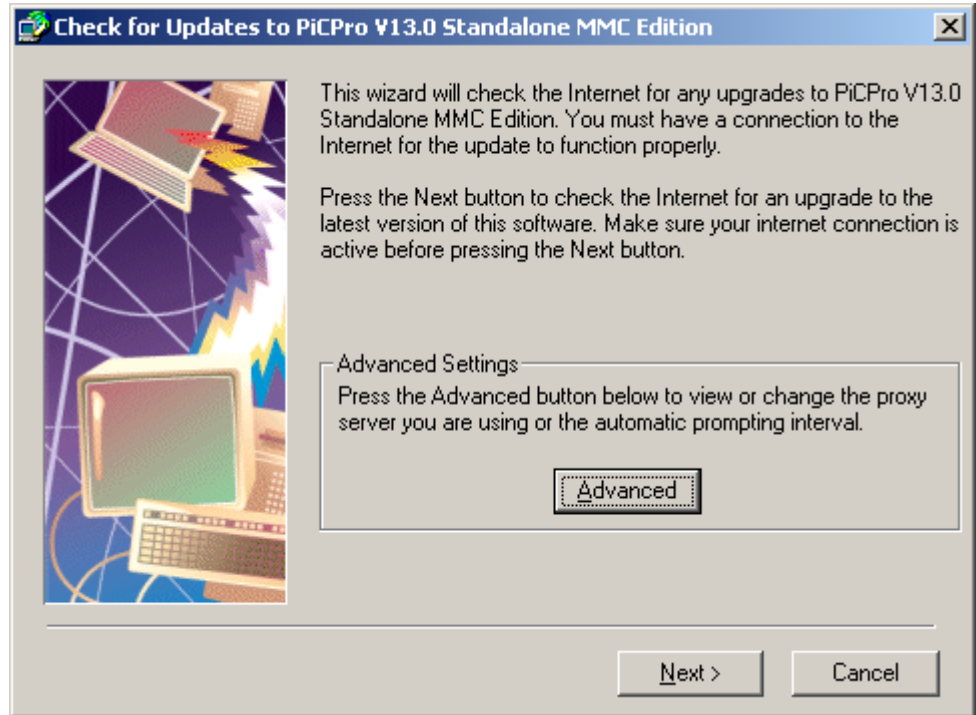


Check the box for **Yes, I want to check the Internet for updates** and Click **Finish**.

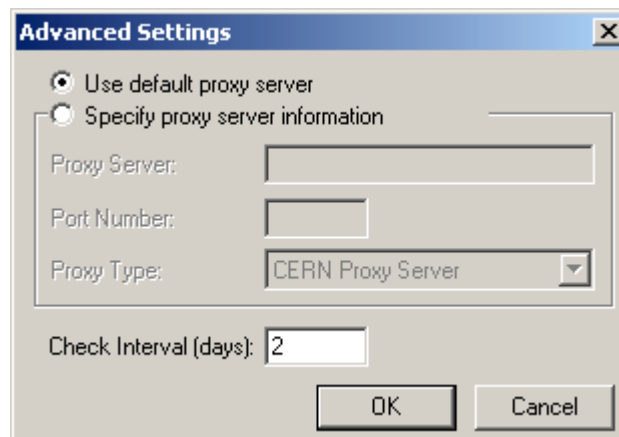
Note: If the update program is downloaded, it will not be installed until PiCPro is closed.

The update program can now be downloaded as follows:

1. The following window will be displayed:



Clicking on the Advance button will bring up a dialog that allows you to change the Internet access server and the time period between update checks.



2. Make the desired changes in the Advanced Settings dialog and click **OK**.

3. Click **N**ext and the update program will check for any PiCPro updates. If you already have the latest PiCPro Service Pack, a message box will be displayed stating the following:

You are currently running the latest version of PiCPro V13.0 "Edition" Please check for updates again in the future. Click OK.

If a new PiCPro Service Pack exists, a dialog will be displayed stating the following:

Your are now ready to download the update to PiCPro Vxx.xx Edition.

If PiCPro is opened, PiCPro will automatically be closed before the download starts.

Press the Next button to begin the download. After the update is downloaded you will be prompted to run the installation to update your software to the newest version.

4. After the download is complete, a dialog will be displayed stating the following:

The update for PiCPro Vxx.xx Edition has been successfully downloaded.

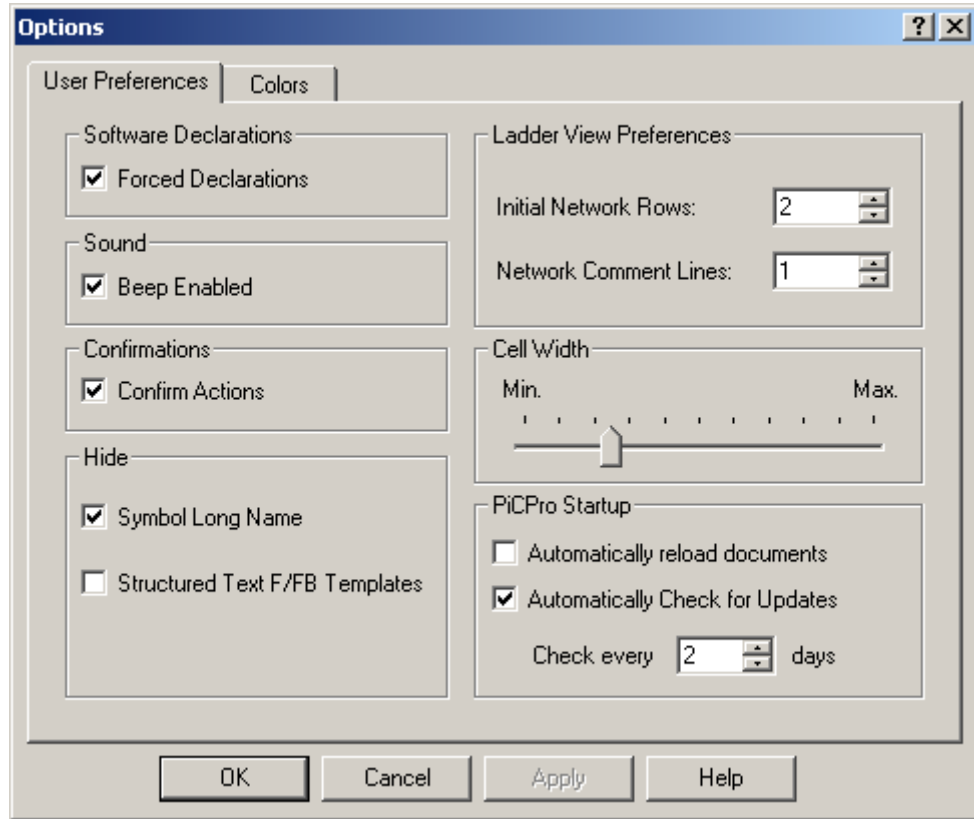
5. Press the **F**inish button to execute the update installation.

Service Pack Installation - Automatic Mode

When run in automatic mode, the update program is added to the start of PiCPro and is executed every time PiCPro is started.

The update program is set up for automatic mode by doing the following:

1. From the PiCPro menu, choose **View | Options**.



2. Check the box for **Automatically Check for Updates**.
3. Specify the time period between updated checks in the **Check every** ___ **days** field.
4. Click **OK**.
5. Follow steps 1 through 5 from the previous section titled “Service Pack Update Installation - Manual Method.

NOTES

APPLICATION NOTE

Application Note 1

Reading and Writing STRINGS from a Structure

The following applies only to Reading and Writing STRINGS that are members of a Structure *using just the structure name* as the **Memory Area (BUFR)** input to the READ and WRITE function blocks in PiCPro.

Any variable of type 'STRING' has an actual size that is 2 more than its 'Declared' length. These two bytes are in the beginning of the STRING and are normally hidden from the user. The first byte contains the maximum or 'Declared' length of the STRING, and the second byte contains the actual length of the STRING.

This means that the structure size will be increased by two bytes for every STRING element in the structure.

Examples:

```
1. MY_STRUC    STRUCT
   .ONE        DINT
   .NAME       STRING[10]
   .TWO        DINT
               END_STRUCT
```

Example 1 has a size of 20 bytes (4 bytes for .ONE, 12 bytes for the .NAME element and 4 bytes for .TWO).

```
2. MY_STR2    STRUCT
   .NAMES      STRING[10](0..3)
               END_STRUCT
```

Example 2 has a size of 48 bytes (12 bytes for each STRING of 10, and it is an array of 4 STRINGS).

So if you Write a structure that contains a 'STRING' type member, **it is important** to write out the complete structure (i.e. if you write out MY_STR2 make sure you write out 48 bytes and not 40 bytes).

Similarly if you Read into a structure that contains a STRING type member, **it is important** to read in the complete structure (i.e. if you read in MY_STR2 make sure you read in 48 bytes and not 40 bytes).

This does not apply if you Read from/Write to a STRING type variable itself. In the example above, it would not apply if you read from/write to MY_STRUC.NAME or MY_STR2.NAMES(3). In this case, PiCPro automatically updates the two hidden length bytes as the STRING is read in.

Some additional examples:

1. Consider a ladder with a structure declared as:

```
MY_STR2  STRUCT
.NAMES   STRING[10](0..3)
END_STRUCT
```

To write out all four STRINGS in the structure completely, you would use,

—| |—

To read all four STRINGS into the structure completely, you would use,

—| |—

Note that this also means that the device you are reading the STRINGS from in this manner MUST store them in the following format:

1 BYTE 'Declared' Length in PiCPro (*In this example 10*)

1 BYTE Actual Length of the STRING (actual number of characters in this STRING)

(The actual length must be less than or equal to the declared length)

BYTES of the STRING itself. (*In this example 10 Bytes. Even if the actual number of characters in the STRING is less than the Declared Length, **the STRING must be padded up to the declared length.** You can use any character you choose to pad the STRING.*)

If you are reading from/writing to the STRING itself, **you do not** need to account for the two hidden bytes as in the examples below. PiCPro will automatically update these bytes when it knows the input to BUFR is a 'STRING'.

2. Consider a STRING declared as:

```
NAMES    STRING[10](0..3)
```

To write out just one STRING completely, you would use,

—| |—

To read in the NAME written above, you would use,

—| |—

NOTES

INDEX

A

- aborting a patch 3-125
- advanced operations toolbar I-2
- analog input setup 4-11
- analog output offset 4-30
- animating the ladder 3-112
- application note
 - reading strings from a structure AN-3
- arrays 3-55
 - entering 3-56
- ASCII
 - chart F-1
- ASIU I/O numbering B-76
- AT - Amplifier Telegram 5-6
- attribute 3-57
 - external 3-58
 - global and retentive 3-58
 - retentive 3-57
 - variable in/out 3-58
- auto fill axes 4-16
- axis data 4-18
- axis positioning, SERCOS 5-17
- axis properties 4-18
- axis tuning 4-45
 - using forcing 4-45
 - using viewing 4-46
 - variables 4-47

B

- backup 3-140
- backup application 3-103
- basic online operations toolbar I-2
- battery box 5-39
- Bin file 3-104
- block expansion rack I/O
 - numbering 3-48, B-76
- blown fuse status 3-48, B-76
- build a dependency list 3-130

C

- calculate defaults 4-21
- calling Technical Support 1-3
- changing memory configurations 1-20
- Clear application memory 3-139

- closing files 3-24
- codes
 - diagnostic E-1
- coil 3-73
- cold restart 3-122
- command velocity (variable 6) G-1
- comment export 3-132
- comment import 3-131
- comments 3-68
- communications
 - baud rate 3-132
 - extend timeouts 3-133
 - infrared drivers 1-4
 - IP address 3-132
 - port 3-132
 - serial 3-132
 - settings 3-132
 - TCP/IP 3-132
- communications error messages B-20
- comparing to cyclic data 5-9
- comparing to service channel 5-9
- comparing UDFBs and Tasks 6-9
- compile error messages
 - ladder B-16
- compiler toolbar I-4, I-5
- compiling
 - Bin file 3-104
 - Hex file 3-106
 - ladder 3-103
 - SERCOS setup function 5-32
 - servo setup function 4-42
 - Task 3-107
 - UDFB 3-108, 6-4
- computer workstation 1-4
- configuration
 - DeviceNet software 7-2
 - Profibus software 8-1
- connecting hardware 1-4
- constants 3-69
- contacts 3-71
- controlling a time axis velocity G-1
- converting
 - servo setup CPU 4-44
- copying
 - a Project 2-29
 - an item 3-13

- axis information 4-39
 - between SRC files 5-30
 - hardware declarations 3-37
 - SERCOS data 5-30
 - software declarations 3-60
 - SRS structure 5-38
- copying and deleting 3-11
- creating
 - a New Project 2-18
 - new files 3-1
 - SERCOS setup file 5-18
 - servo setup file 4-2
 - servo setup function 4-42
 - Task 6-13
 - UDFB 6-1
- C-stop error codes B-8
- customer service 1-3
- customizing PiCPro 1-12
- cutting an item 3-12
- cyclic data structures 5-27
- cyclic operation 5-9

D

- D/A output setup 4-9
- data handling in UDFBs 6-4
- data type
 - arrays 3-55
 - functions 3-55
 - numeric 3-52
 - structures 3-55
 - time 3-54
- data types 3-50
 - inputs/outputs A-1
- define cyclic data 5-25
- define operation mode 5-28
- Delete 3-11
- dependency list 3-130
 - errors B-19
- derivative gain 4-30
- determining scaling information 4-22
- device status (Profibus) 8-4
- device status code 7-5
- device status word 7-4
- DeviceNet configuration software 7-1
 - procedure 7-2
- DeviceNet error messages B-82

- diagnostic error codes E-1
- dialog boxes 1-11
- displaying/hiding toolbars 1-16
- Download a hex file 3-135
- dragging and dropping 3-15
- duplicate 3-22

E

- editing
 - axis data 4-18
 - axis properties 4-18
 - function block help 1-21
 - hardware declarations 3-36
 - IDNs 5-24
 - IDNs for send 5-45
 - servo setup 4-18
 - software declarations 3-59
 - startup IDN list 5-23
 - UDFB 6-4
- encoder input setup 4-10
- entering
 - forcing variables 3-115
 - iterator data 4-23, 4-27, 4-33
 - iterator data for time axis 4-29, 4-32
 - network 3-8
 - position loop data 4-29
 - scaling data 4-20
- entering software declarations 3-40
- error codes B-1
 - C-stop error codes B-8
 - diagnostic E-1
 - E-stop error codes B-9
 - function codes B-1
 - LED codes E-1
 - programming error codes B-10
 - servo timing error function B-11
 - stepper H-2
- error messages
 - communications B-20
 - compile ladder B-16
 - dependency list B-19
 - DeviceNet B-82
 - fieldbus B-82
 - hardware B-27
 - ladder B-32
 - library B-19

- SERCOS B-95
- servo B-83
- E-stop error codes B-9
- execute procedure command 5-46
- exiting 3-24
- expansion rack I/O
 - numbering 3-47, B-75
- export comments 3-132
- extended data memory 3-110
 - working with 3-111
- external attribute 3-58
- F**
- fast inputs 3-49
- faults, diagnostic E-1
- feed forward percent 4-30
- fieldbus error messages B-82
- files
 - created types J-1
 - extensions J-1
 - organizing 1-17
- filter search 3-21
- find
 - IDNs 5-52
- finding and replacing 3-18
- finding duplicates 3-22
- focus 3-6
- following error limit 4-30
- forcing
 - entering variables 3-115
 - variables 3-114
- forcing/grouping 3-116
- Function Block Help 1-21
 - edit 1-21
- function/function blocks 3-55, 3-75
 - data 3-75
- functions
 - error codes B-1
 - inputs
 - data types A-1
 - outputs
 - data types A-1
 - SERCOS 5-3
 - SERCOS setup 5-32
 - servo setup 4-42
 - with Stepper Axis Module H-1

- functions toolbar I-4
- G**
- G&L scanner symbol 7-4
- G&L TCP/IP configuration software
 - procedure 7-7
- global and retentive attribute 3-58
- grouping 3-116
- H**
- hardware connections 1-4
- hardware declarations 3-26
 - closing 3-39
 - copying 3-37
 - cutting 3-37
 - editing 3-36
 - pasting 3-38
 - printing 3-39, 3-128
 - saving 3-39
- hardware error messages B-27
- help 1-20
 - creating Function Block Help 1-21
 - editing Function Block help 1-21
 - Function Block 1-21
 - Whats This 1-20
- help files 1-11
- Hex file 3-106
- I**
- I/O point 3-43
- I/O Points, numbering 3-46
- IDNs 5-7
 - insert for receive continuous 5-43
 - insert for send 5-45
 - insert startup 5-24
 - parameter data 5-8
 - PCF 5-8
 - print 5-53
 - receive 5-41
 - receive continuous 5-43
 - search 5-52
 - send 5-44
- import comments 3-131
- information window 1-7
- infrared communication drivers 1-4
- initial values 3-50

- prefixes 3-50
- initialize
 - servo setup data 4-43
- in-position band 4-31
- input polarity 4-30
- input scaling 4-20
- inserting
 - IDNs 5-24
 - IDNs for send 5-45
 - networks 3-8
 - software declarations 3-59
- installation
 - program directory 1-17
- integral error limit 4-30
- integral gain 4-30
- interlocking Task data 6-17
- introduction to PiCPro 1-1
- iterator data 4-23, 4-27, 4-33
- iterator data for time axis 4-29, 4-32

J

- jump command 3-76

L

- label 3-68
- ladder error messages
 - errors B-32
- ladder toolbar I-3
- LED error codes E-1
- libraries 1-18
- library
 - error messages B-19
- long names 3-41

M

- master rack I/O
 - numbering B-75
- MDT - Master Data Telegram 5-6
- memory configurations 1-20
- memory, extended data 3-110
- menu bar 1-11
- MMC for PC
 - SERCOS 5-2
- MST - Master Synchronization Telegram 5-6

N

- naming

- Task 6-14
- UDFB 6-2
 - variables 3-41
- NEMA SERCOS Specification 5-1
- network
 - coil 3-73
 - comments 3-68
 - contacts 3-71
 - elements 3-71
 - function/function blocks 3-75
 - jump command 3-76
 - label 3-68
 - wires 3-74
- numeric 3-52

O

- off-line editing (UDFB) 6-4
- on-line editing 3-123
- on-line editing (UDFB) 6-5
- online Help system
 - printing 1-20
- opening
 - a Project 2-21, 2-24, 2-25
 - an existing ladder 3-2
 - SERCOS setup file 5-18
 - servo setup file 4-3
- operation mode
 - defining 5-28
- operator interface 3-110
- options
 - setting 1-12
- options for IDN display 5-50
- organizing your PiCPro files 1-17
- output
 - polarity 4-30
 - scaling 4-20
- Overview
 - Using Projects 2-1

P

- parameter data IDNs 5-8
- pasting an item 3-13
- patching the ladder 3-123
- PCF
 - IDNs 5-8
- phases

- SERCOS 5-9
- PiC
 - and SERCOS 5-2
 - restore 1-20
- PiCPro
 - closing files 3-24
 - copy command 3-11
 - creating files 3-1
 - customizing 1-12
 - define 1-1
 - deleting command 3-11
 - entering networks 3-8
 - exiting 3-24
 - filtering search 3-21
 - finding and replacing 3-18
 - finding duplicates 3-22
 - getting started 1-1
 - libraries 1-18
 - opening a file 3-2
 - organizing files 1-17
 - printing 1-20
 - programs 1-1
 - reference card C-1
 - saving 3-23
 - scanning 3-120
 - searching in 3-21
 - selecting items 3-6
 - splitting the screen 3-25
 - starting 1-22
 - tools 3-4
 - work area 1-6
 - working in 3-1
- pointer tool 3-4
- position loop data 4-29
- printing 1-20
 - comments 3-128
 - cross reference 3-127
 - force list 3-129
 - hardware declarations 3-128
 - IDNs 5-53
 - information window 3-129
 - ladder 3-126
 - long names 3-128
 - online Help system 1-20
 - projects 2-29
 - SERCOS 5-32
 - servo setup axis information 4-42
 - view list 3-129
- procedure
 - DeviceNet configuration software 7-2
 - G&L TCP/IP configuration software 7-7
 - Profibus configuration software 8-2
- procedure command 5-46
- Procedure Command Function 5-8
- Profibus configuration software 8-1
 - procedure 8-2
- program directory 1-17
- programming error codes B-10
- Project 2-21, 2-24, 2-25
 - copy 2-29
 - DOS compressed project file 2-25
 - DOS project file 2-24
 - include sources 2-5
 - launch 2-28
 - new 2-18, 2-19
 - open 2-21, 2-22, 2-24
 - open compressed 2-22
 - open from control 2-22
 - print 2-29
 - save as 2-29
 - uncompress 2-22, 2-23
 - version mismatch 2-28
- project
 - paths 2-30
- Projects 2-1
- projects
 - commands 2-1
 - managing 2-1
- proportional gain 4-30
- purging unused variables 3-65

R

- racks
 - numbering 3-46
- receive continuous IDNs
 - SERCOS
 - IDNs
 - receive continuous 5-42
 - receive IDNs 5-41
- recommended workstation 1-4
- reference card
 - PiCPro C-1

- referencing a time axis G-1
- requirements 1-4
- resolver input setup 4-11
- restore 3-103, 3-141
- restore PiC 1-20
- retentive attribute 3-57
- ring errors 5-47
- ring state 5-47
- rollover on position with a time axis G-2
- running one scan 3-121

S

- saving
 - files 3-23
 - hardware declarations 3-39
 - prior version PiCPro 4-41
 - SERCOS setup file 5-31
 - servo setup 4-41
- scaling information 4-22
- scanner symbol 7-4
- scanning 3-120
 - cold restart 3-122
 - hot restart 3-121
 - stopping 3-121
 - warm restart 3-122
- search IDNs 5-52
- searching 3-21
 - software declarations 3-64
- selected cells 3-7
- selecting items 3-6
- semaphore flags 6-17
- SERCOS
 - advanced features 5-16
 - and MMC for PC 5-2
 - and PiC 5-2
 - and standalone MMC 5-2
 - AT 5-6
 - axis positioning 5-17
 - battery box 5-39
 - changing IDN data and uploading IDNs 5-35
 - copying between SRC files 5-30
 - copying data 5-30
 - creating setup file 5-18
 - cyclic data structures 5-27
 - cyclic operation 5-9

- define cyclic data 5-25
- define operation modes 5-28
- error messages B-95
- execute procedure command 5-46
- functions 5-3
- IDNs 5-7
 - display options 5-50
 - find 5-52
 - insert for send 5-45
 - parameter data 5-8
 - PCF 5-8
 - printing 5-53
 - receive 5-41
 - receive continuous 5-43
 - send 5-44
 - startup list 5-23, 5-24
 - upload drive file 5-42
- IDNs, changing data 5-35
- IDNs, uploading 5-35
- initialize setup data in your ladder 5-33
- inserting rings 5-20
- inserting slaves 5-21
- MDT 5-6
- MST 5-6
- numbering slaves 5-21
- operation mode
 - application note 5-29
- options for IDN display 5-50
- phases 5-9
- printing 5-32
- receive continuous IDNs 5-42
- receive IDNs 5-41
- referencing axes 5-17
- replacing analog system 5-15
- ring errors 5-47
- ring state 5-47
- rings,inserting 5-20
- saving setup file 5-31
- send IDNs 5-44
- service channel 5-9
 - accessing 5-12
 - background information 5-11
- servo axis 4-13
- setup 5-1, 5-18
- setup function 5-32
- slave status/control

- slave status/control 5-49
 - slaves, inserting 5-21
 - specifying the SRS 5-55
 - SRS structure
 - copy to clipboard 5-38
 - standard Motion Functions and Variables 5-4
 - startup IDN List 5-23
 - telegrams 5-6
 - troubleshooting 5-56
 - upload drive information 5-42
 - using Functions/Blocks with TASKS 5-5
 - variable length data 5-54
- service channel 5-9
 - accessing 5-12
 - background information 5-11
- servo
 - calculate defaults 4-21
 - error messages B-83
 - timing error function B-11
- servo setup 4-1
 - analog input 4-11
 - analog output offset 4-30
 - auto fill axes 4-16
 - creating
 - file 4-2
 - function 4-42
 - D/A output 4-9
 - data categories 4-19
 - derivative gain 4-30
 - editing 4-18
 - encoder input 4-10
 - entering
 - iterator data 4-23, 4-27, 4-33
 - iterator data for time axis 4-29, 4-32
 - entering scaling data 4-20
 - feed forward percent 4-30
 - following error limit 4-30
 - initializing 4-43
 - in-position band 4-31
 - input polarity 4-30
 - input scaling 4-20
 - integral error limit 4-30
 - integral gain 4-30
 - iterator data 4-19
 - open existing file 4-3
 - output polarity 4-30
 - output scaling 4-20
 - overview 4-1
 - position loop data 4-19, 4-29
 - printing 4-42
 - proportional gain 4-30
 - resolver setup 4-11
 - saving 4-41
 - scaling data 4-19
 - scaling information 4-22
 - SERCOS setup 4-13
 - TTL input 4-12
 - update rate 4-31
- setting user preferences 1-12
- settings command 3-109
- size of UDFB Libraries 6-3
- slave gapping 5-21
- software declarations 3-40
 - arrays 3-56
 - attributes 3-57
 - copying 3-60
 - cutting 3-60
 - editing 3-59
 - entering 3-40
 - fast inputs 3-49
 - I/O point 3-43
 - initial values 3-50
 - long names 3-41
 - pasting 3-60
 - purging unused variables 3-65
 - searching 3-64
- specification
 - NEMA SERCOS 5-1
- split screen 3-25
- SRS structure
 - copying 5-38
- standalone MMC
 - SERCOS 5-2
- standard toolbar I-1
- starting PiCPro 1-22
- startup IDN list 5-23
- status
 - blown fuse 3-48, B-76
- status bar 1-8
 - displaying/hiding 1-10
- stepper

- axis module 2-1, H-1
- error handling H-2
- reference card D-1
- using functions with H-1
- stopping the scan 3-121
- STRING
 - read/write from STRUCT AN-3
- structured text constructs toolbar I-5
- structures 3-55
- support services 1-3
- T**
- Task 6-1
 - comparing to UDFB 6-9
 - compiling 3-107
 - creating 6-13
 - hardware declarations 6-12
 - interlocking data 6-17
 - naming 6-14
 - semaphore flags 6-17
 - software declarations 6-12
 - template 6-10
 - types 6-9
 - using Functions 6-11
- technical support 1-3
- telegrams
 - SERCOS 5-6
- time 3-54
- time axis G-1
 - rollover on position G-2
 - setup data categories 4-27
 - synchronizing slave axes with G-2
- timeouts 3-133
- title bar 1-10
- toolbar I-1
 - advanced operations I-2
 - basic online operations I-2
 - compiler I-4, I-5
 - customizing 1-15
 - displaying/hiding 1-16
 - docking 1-16
 - functions I-4
 - ladder I-3
 - moving 1-15
 - standard I-1
 - structured text constructs I-5

- view navigator I-4
- troubleshooting
 - SERCOS 5-56
- TTL input setup 4-12
- U**
- UDFB 6-1
 - comparing to Task 6-9
 - compiling 3-108, 6-4
 - creating 6-1
 - data handling 6-4
 - edit 6-4
 - inputs and outputs
 - marking 6-7
 - number of 6-7
 - order of 6-6
 - library size 6-3
 - naming 6-2
 - software declarations 6-6
 - view 6-5
 - parent ladder 6-5
- Uncompressing a Project 2-22
- update rate 4-31
- upload drive information 5-42
- user preferences 1-12
- User Programs
 - backup 3-140
 - restore 3-141
- using a time axis G-1
- using projects 2-1

- V**
- variable in/out attribute 3-58
- variable length data
 - viewing 5-54
- variables 3-41, 3-50, 3-69
 - forcing 3-114
 - names and long names 3-41
- view navigator toolbar I-4
- viewing
 - enabled variables 3-118
 - parent ladder 6-5
 - UDFB 6-5

- W**
- web site 1-3

Window elements 1-10
wires 3-74
work area
 PiCPro 1-6
working with extended memory 3-111

working with split screen 3-25
workstation 1-4
Z
zooming 3-4

NOTES