

# PiCPro™ for Windows®

## Software Manual

Part Number 108-31048-00

Version 10.2

---

**Giddings & Lewis**

Giddings & Lewis Controls, Measurement and Sensing

## 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.

108-31048-00

Version 2599  
©1998, 99 Giddings & Lewis, Inc.

IBM is a registered trademark of International Business Machines Corporation.  
Windows 95, 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 are trademarks of Giddings & Lewis, Inc.

# Table of Contents

## CHAPTER 1-Getting Started with PiCPro for Windows

---

Introduction to PiCPro for Windows .....	1-1
Customer Services .....	1-2
Principal technical support services .....	1-2
Bulletin Board System (BBS) .....	1-2
Hands-on training schools .....	1-2
Before Calling Technical Support .....	1-2
World Wide Web Site .....	1-2
The Computer Workstation .....	1-3
Hardware Connections .....	1-3
Installing PiCPro for Windows .....	1-3
The Work Area in PiC Pro for Windows .....	1-4
Window Elements .....	1-5
The Title Bar .....	1-5
The Menu Bar .....	1-5
Customizing PiCPro .....	1-6
Setting Options .....	1-6
User Preferences .....	1-6
Animation Color .....	1-7
Status Bar .....	1-8
Displaying/Hiding the Status Bar .....	1-9
Dialog Boxes .....	1-9
Customizing Toolbars .....	1-10
Toolbars .....	1-10
Moving Toolbars .....	1-11
Displaying/Hiding Toolbars .....	1-12
Organizing your PiCPro Files .....	1-13
Printing .....	1-14
Printing the Ladder .....	1-15
Printout Symbols .....	1-18
Troubleshooting the Print Option .....	1-19
Using Help .....	1-20
Using What's This? Help .....	1-20
Using On-line Help .....	1-20
Printing On-line Help .....	1-20

## CHAPTER 2-Working with PiCPro for Windows

---

A PiCPro Session .....	2-1
Starting PiCPro .....	2-1
Creating New Files .....	2-1
Opening an Existing Ladder .....	2-3
PiCPro Tools .....	2-8
Using the Pointer Tool .....	2-8
Zooming In and Out .....	2-8
Selecting and Deselecting Items .....	2-9
Focus .....	2-9
Selection .....	2-9
Dragging and Dropping .....	2-10
Deselecting .....	2-10
Entering a Network .....	2-11
Adding, Changing, or Deleting Networks .....	2-11
Copying and Deleting .....	2-12
Cutting an Item .....	2-12
Copying an Item .....	2-12
Pasting an Item .....	2-13
Deleting an Item .....	2-13
Finding and Replacing .....	2-14
Finding and Replacing .....	2-14
Filtering Find and Replace .....	2-15
Finding Duplicates .....	2-16
Saving, Closing, and Exiting .....	2-16
Saving Files .....	2-16
Saving Files with a Different Name .....	2-17
Closing Files .....	2-17
Exiting PiCPro .....	2-17
Working with a Split Screen .....	2-18
Hardware Declarations .....	2-19
Hardware Declarations Table - PiC CPU .....	2-19
Hardware Declarations Table - MMC CPU .....	2-20
Entering Hardware Declarations .....	2-21
Master Rack Declarations .....	2-22
Remote I/O Expansion Rack Declarations .....	2-25
Block I/O Declarations .....	2-27
Editing Hardware Declarations .....	2-29
Inserting and Deleting in the Hardware Declarations Table .....	2-29
Cutting and Copying Hardware Declarations .....	2-30
Pasting and Inserting Paste in Hardware Declarations .....	2-31
Printing Hardware Declarations .....	2-32
Closing and Saving Hardware Declarations .....	2-32
Software Declarations .....	2-33
Entering Software Declarations .....	2-33

Names and Long Names of Variables .....	2-34
Names .....	2-34
Long Names .....	2-34
I/O Points .....	2-35
MMC I/O Point Labels .....	2-35
Numbering .....	2-36
PiC CPU .....	2-36
Master Rack, PiC CPU .....	2-36
Master Rack MMC CPU .....	2-37
Expansion Rack I/O .....	2-37
Block Expansion Rack I/O .....	2-37
Blown Fuse Status .....	2-38
Fast Inputs .....	2-38
PiCPro CPU .....	2-38
MMC CPU .....	2-38
Initial Values .....	2-39
Working with Data Types .....	2-40
Bitwise .....	2-40
Numeric .....	2-42
String .....	2-43
Time .....	2-44
Time: duration .....	2-44
Function Block .....	2-45
Groups of data - Structures and Arrays .....	2-45
Arrays .....	2-45
Structures .....	2-45
Entering an Array in Software Declarations .....	2-46
Working with Attributes .....	2-47
Retentive Attribute .....	2-47
Global Attribute .....	2-47
Both Global and Retentive Attribute .....	2-47
External Attribute .....	2-47
In and Out Variable Attributes for UDFBs .....	2-48
Editing Software Declarations .....	2-48
Inserting/Deleting Software Declarations .....	2-48
Cutting, Copying, and Pasting Software Declarations .....	2-49
Searching in the Software Declarations Table .....	2-52
Find by Name Only .....	2-52
Find by Type .....	2-52
Find by I/O Point: .....	2-52
Using the Find Next Command .....	2-53
Purging Unused Variables from the Software Declarations Table .....	2-53
Working with Network Elements .....	2-54
Network Size .....	2-54
Elements of a Network .....	2-55
Contacts .....	2-55

Coils .....	2-56
Wires .....	2-57
Functions/Function Blocks .....	2-58
Data .....	2-58
Jumps .....	2-59
Labels .....	2-60
Long Names .....	2-60
Comments .....	2-61
Variables and Constants .....	2-62
Compiling and Downloading .....	2-63
Compiling a Bin File .....	2-64
Compile and Download .....	2-64
Compile Only .....	2-65
Compiling and Downloading a Hex File .....	2-65
Compiling a Task .....	2-67
Compiling a UDFB .....	2-68
Settings .....	2-69
Animating the Ladder .....	2-70
Forcing Variables .....	2-72
Entering Variables in the Force List .....	2-73
Turn Forcing/Grouping On/Off .....	2-74
Updating the PiC after Force List is edited .....	2-75
Viewing Enabled Variables .....	2-76
Turning Animation On in the View List .....	2-77
Running the Ladder .....	2-78
Connecting the PC to the PiC Controller .....	2-78
Controlling the Scan .....	2-78
Stopping the Scan .....	2-79
Running One Scan .....	2-80
Doing a Hot Restart .....	2-80
Doing a Warm Restart .....	2-80
Doing a Cold Restart .....	2-81
On-line Editing .....	2-81
Backing Up and Restoring User Programs .....	2-85
Backing Up User Programs .....	2-85
Restoring User Programs .....	2-86
Building a Dependency List .....	2-87
Communications .....	2-88
Setting Node IDs .....	2-89
Connecting the PC to a PiC in the Network .....	2-90

## CHAPTER 3-Servo Setup and Tuning

---

Servo Setup .....	3-1
Opening Servo Setup .....	3-2
To create a new servo setup file for a PiC CPU .....	3-2

To open an existing servo setup file for a PiC CPU .....	3-2
To convert an open existing servo setup file for a PiC CPU to a servo setup for an MMC CPU .....	3-3
To create a new servo setup file for an MMC CPU .....	3-4
To open an existing servo setup file for an MMC CPU .....	3-5
To convert an open existing servo setup file from MMC CPU to a servo setup file for a PiC CPU .....	3-6
To create a new servo setup file using PiCPro for Windows MMC Edition .....	3-7
To open an existing servo setup file using Windows for PiCPro MMC Edition .....	3-8
To open an existing servo setup file from the setup function in your ladder .....	3-9
To open an existing servo setup file from Windows explorer .....	3-10
Copying Axis Data .....	3-10
How to use the copy/paste commands .....	3-10
How to copy axis data from one axis to another .....	3-11
Editing a Servo Setup File .....	3-13
Editing Axis Properties .....	3-13
Editing Axis Data .....	3-13
Saving a Servo Setup File .....	3-13
To save a new file .....	3-13
To save an existing file .....	3-13
To save a file under a different name or in a different location .....	3-14
To save a Servo Setup File in a format that can be read by earlier versions of PiCPro for Windows (prior to V10.2) .....	3-14
Printing Axis Data .....	3-14
To print axis data .....	3-14
Making a Servo Setup Function .....	3-14
To create a servo setup function .....	3-14
To initialize setup data in your ladder .....	3-15
Inserting an Axis in Servo Setup .....	3-15
Servo Axis Data .....	3-16
To insert a servo axis .....	3-17
To insert default axis using Auto Fill .....	3-19
D/A Output Setup .....	3-20
To configure the D/A output using PiCPro with PiC Target CPU .....	3-20
To configure the D/A output using PiCPro with MMC Target CPU .....	3-20
Encoder Input Setup .....	3-21
To configure the encoder input using PiCPro with PiC Target CPU .....	3-21
To configure the encoder input using PiCPro with MMC Target CPU ..	3-22
Resolver Input Setup (Only for PiC CPU) .....	3-22
Analog Input Setup (Only for PiC CPU) .....	3-22
TTL Input Setup (Only for PiC CPU) .....	3-24
SERCOS Setup (Only for PiC CPU) .....	3-25
Stepper Setup (PiC CPU only) .....	3-26

Entering servo setup data .....	3-26
Entering servo setup data for PiC CPU .....	3-26
Entering servo setup data for MMC CPU .....	3-29
Servo Axis Setup Data Categories for PiC and MMC CPU .....	3-32
Entering Scaling Data .....	3-33
Determining Scaling Information .....	3-34
Entering Iterator Data .....	3-35
Entering Position Loop Data .....	3-37
Axis Tuning .....	3-40
Axis Tuning using Forcing .....	3-40
To force an axis servo variable in the Servo Force list .....	3-40
Axis Tuning using Viewing .....	3-41
To view an axis servo variable in the Servo View list .....	3-41
Axis Tuning Variables .....	3-41

## **CHAPTER 4-PiC and SERCOS**

---

SERCOS Setup Background Information .....	4-1
SERCOS Functions .....	4-3
SERCOS Telegrams .....	4-4
IDNs .....	4-5
Working with Procedure Command Function IDNs .....	4-6
SERCOS Phases .....	4-7
Cyclic Operation .....	4-7
Service Channel .....	4-7
Comparing Cyclic Data and Service Channel Transfers .....	4-8
Service Channel Application Note .....	4-9
Background Information .....	4-9
Examples of LDO Design to Access the Service Channel .....	4-10
Example 1 Logic that gives priority to the LDO requests .....	4-10
Example 2 Logic that allows SERCOS Setup commands to be read sooner .....	4-11
Example 3 Logic that allows one IDN to be read more frequently. ....	4-12
Using Standard Motion Functions and Variables .....	4-13
Using SERCOS Functions/Blocks with TASKS .....	4-14
Opening SERCOS Setup .....	4-15
Copying SERCOS Data .....	4-16
Saving a SERCOS Setup File .....	4-17
Printing SERCOS Setup .....	4-17
Making a SERCOS Setup Function .....	4-18
Inserting/Editing SERCOS Rings .....	4-20
Inserting/Editing SERCOS Slaves .....	4-21
Edit Startup IDN List .....	4-22
Inserting/Editing IDNs .....	4-23
Define Cyclic Data .....	4-24
Cyclic Data Structures .....	4-25
Define Operation Modes .....	4-26



Operation Mode Application Note .....	4-27
Battery Box .....	4-28
Battery Box Application Note .....	4-29
Receive IDNs .....	4-30
Upload Drive Information .....	4-31
Changing IDN Data and Uploading IDNs Note .....	4-32
Receive Continuous IDN Data .....	4-33
Inserting/Editing IDNs for Receive Continuous .....	4-34
Send IDNs .....	4-35
Inserting/Editing IDNs for Send .....	4-36
Execute Procedure Command .....	4-37
Ring State .....	4-38
Slave Status/Control .....	4-40
IDN Lists .....	4-41
Options for IDN Display .....	4-41
IDN Lists .....	4-41
Finding a Specific IDN in an IDN List .....	4-42
Printing IDNs from an IDN list .....	4-44
Viewing Variable Length Data from an IDN List .....	4-45
Specifying the SRS for Viewing the Drive IDN List .....	4-46
SERCOS Startup .....	4-47
Replacing Your Analog System with SERCOS .....	4-47
Using SERCOS Advanced Features .....	4-48
Axis Positioning and Referencing in SERCOS .....	4-49
Troubleshooting SERCOS .....	4-50
Communication Problems .....	4-50
Control Problems .....	4-50

## **CHAPTER 5-Working With TASKS and UDFBs**

---

Tasks .....	5-1
Comparing UDFBs and TASKs .....	5-1
Types of TASKs .....	5-1
Using the Task Template .....	5-2
Using Functions from the I/O and Motion Libraries .....	5-3
I/O Functions .....	5-3
Motion Functions .....	5-3
Comparing Declaration Rules for Main and Task LDOs .....	5-4
Hardware Declarations .....	5-4
Software Declarations .....	5-4
Creating a Task .....	5-5
Naming the Task .....	5-6
Interlocking Data in Tasks .....	5-6
Setting up Semaphore Flags .....	5-7
UDFBs .....	5-11
Creating a UDFB .....	5-11

Naming the UDFB .....	5-12
Size of UDFB Libraries .....	5-13
Data Handling in UDFBs .....	5-14
Compiling the UDFB .....	5-14
Editing a UDFB .....	5-14
Off-Line Editing .....	5-14
On-Line Editing .....	5-15
Viewing a UDFB .....	5-15
Viewing the Parent Ladder .....	5-15
UDFB Software Declarations .....	5-16
Naming the Variables .....	5-16
Order of UDFB Inputs and Outputs .....	5-17
Number of UDFB Inputs and Outputs .....	5-17
Marking UDFB Inputs and Outputs .....	5-18

## **CHAPTER 6-Ethernet - TCP/IP**

---

Ethernet .....	6-1
TCP/IP .....	6-1

## **Appendix A - Quick Reference to In/Out Function Data**

---

Function Input/Output Data Type Reference .....	A-1
---	-----

## **Appendix B - Errors**

---

Function Errors .....	B-1
General Function Errors .....	B-1
I/O Function Block Error Codes .....	B-1
String Functions Errors .....	B-5
Servo C-Stop, E-Stop, and Programming Errors .....	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
Compile Errors .....	B-12
Dependency List Errors .....	B-15
Library Error .....	B-16
Communications Errors .....	B-17
Hardware Errors/Messages .....	B-23
Compiling Errors .....	B-25
Master Rack, PiC CPU .....	B-42
MMC I/O Point Labels .....	B-43
Master Rack MMC CPU .....	B-44
Expansion Rack I/O .....	B-44
Block Expansion Rack I/O .....	B-44

Blown Fuse Status .....	B-46
Fast Inputs .....	B-46
Fieldbus Errors .....	B-51
Servo Errors .....	B-52
Force List Errors .....	B-59
SERCOS Errors .....	B-60
Ring Errors .....	B-60
Slave Errors .....	B-62

## **Appendix C - PiCPro Reference Card - Errors/Variables**

---

Card .....	C-1
------------	-----

## **Appendix D - Stepper Reference Card**

---

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

## **Appendix J - Module Filenames**

---

Module Filenames.....	J-1
-----------------------	-----

## **Appendix K - G&L DeviceNet Configuration Tool**

---

Overview of setting up a DeviceNet Network .....	K-1
Procedure for using the G&L DeviceNet Configuration Tool .....	K-2
Device Status Word .....	K-4
Device Status Code .....	K-4

## **Appendix L - G&L Ethernet - TCP/IP Configuration Tool**

---

Overview of setting up a Ethernet Network .....	L-1
Procedure for using the G&L TCP/IP Configuration Tool .....	L-2
Procedure for Creating a Configuration File .....	L-4

## **Application Note 1**

---

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

# CHAPTER 1 Getting Started with PiCPro for Windows

## Introduction to PiCPro for Windows

---

A PiC (Programmable industrial Computer) system provides 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 to run your application(s). It includes the following programs:

<b>Program</b>	<b>Function</b>
Ladder Diagram	Allows you to program a ladder and save it as an .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 .SCR file

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 is a tutorial 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).

Included with the PiCPro package is the Function/Function Block Reference Guide, and the PiC900 Hardware Manual and the MMC Hardware Manual.

## 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.

### **Bulletin Board System (BBS)**

If you have a modem and communications software, you can access PiC's BBS. Here you can access application notes, drive sizing programs, example ladders, sales presentations, demo programs, and the user file exchange where you can transfer files to customer support.

#### **To access the PiC Bulletin Board**

1. Dial 920/906-7682. The maximum baud rate is 14,400 Baud.
2. After connecting to the modem, you will be asked to log in. If you are a new user, you will enter your name, define your personal password, and answer some questions such as your company name and setup information on how you want to use the bulletin board. Subsequent log ins will only require your name and password.
3. To access the file(s) you are interested in, select the File Menu from the main BBS menu.

### **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 95 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.

## The Computer Workstation

---

The PiCPro for Windows software runs on IBM PC compatible platforms. The requirements for a workstation are found in Table 1-1.

**Table 1-1. Work station requirements**

	<b>Recommended</b>
<b>Computer</b>	A 486 or Pentium processor with Windows 95 or NT
<b>Memory</b>	32 MB of RAM, minimum; 64 MB of RAM, recommended
<b>Monitor</b>	VGA or higher resolution display adapter
<b>Disk drives</b>	Typically, 10 M of hard disk space required
<b>Serial communications port to PiC900</b>	RS232 port (COM 1 or COM2)

## 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 specified in the Computer dialog box and connect the “PiC900” end to the PiCPro Port on the PiC.

### IMPORTANT

Be careful to plug the “Computer” end of the cable into the PC serial port and the “PiC900” end into the PiCPro Port on the CPU or CSM/CPU module of the PiC.

The pin-out for the CPU serial port is given in the Hardware Manual. The pin-out for the PC serial port should be in the computer manual.

## Installing PiCPro for Windows

---

PiCPro for Windows is available on CD or on 3.5” floppy disks.

To install from the CD, follow the instructions on the CD jacket.

To install from the set of floppy disks, insert disk 1 in Drive A, select Run from the Start menu, and type a:/setup.

## The Work Area in PiC Pro 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 on the desktop.

### 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 or PiCPro is closed. With certain data (i.e. 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 include:

- You can undock 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.
- You can print the contents of the information window.
- You can hide the information window from view in various ways:

Pressing <Esc>

Choosing View, Information Window from the menu

Right clicking the mouse and choosing Hide

Choosing Window, Close All from the menu (NOTE: This will close all open files.)



## 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 version 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 is dimmed.

**PiCPro for Windows: Title Bar**



**PiCPro for Windows MMC Edition: Title Bar**



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 Restore, Move, Size, Minimize, Maximize, or Close 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 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 Menu Bar contains the names shown below.

**PiCPro for Windows: Menu Bar**



**PiCPro for Windows MMC Edition: Menu Bar**



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



## 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:

- Customize the toolbars
- Set preferences
- Choose animation color
- Display/hide the status bar

### Setting Options

Options for user preferences and for choosing animation color can be set under

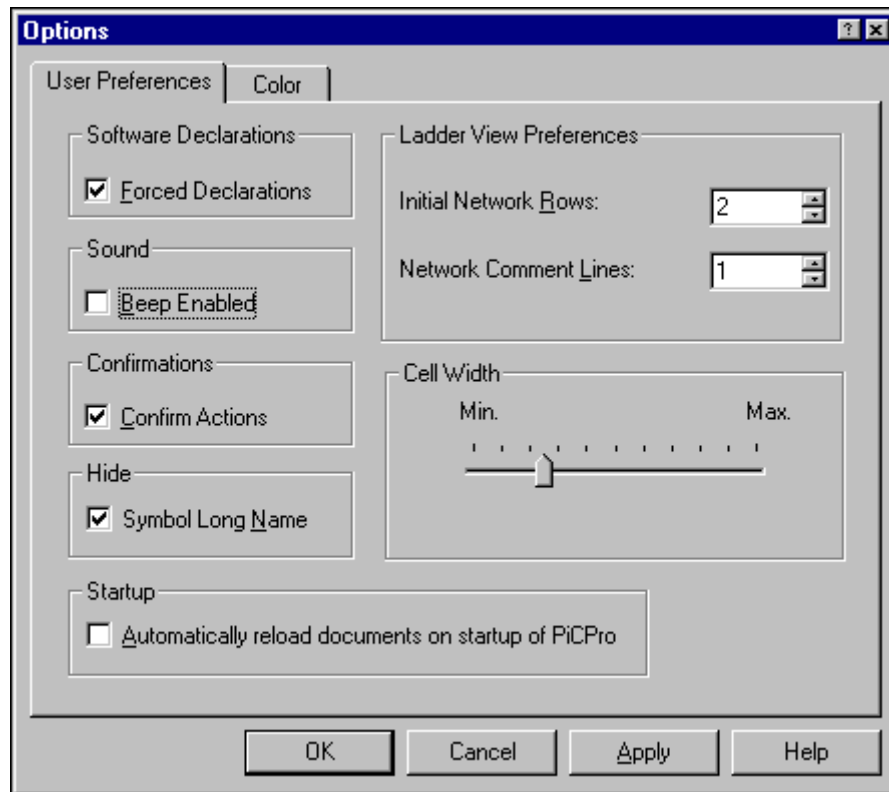
View Options from the menu or by pressing the  button from the Standard toolbar.

### 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 what you have just added to your ladder to the software declarations table.
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.
5. **Start-up - Automatically reload documents on startup of PiCPro:** If this choice is checked, any documents (.ldo, .srv, .src) opened using File/Open or File/New that are still open when PiCPro exits, will be reopened the next time PiCPro starts. Files that were opened using View/UDFB/Task..., View/Sercos Function or View /Servo Function will not be reopened.
6. **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.
7. **Cell Width:** How wide the cells in the networks appear on your screen.

The Options box below shows all the default settings on the User Preferences page.



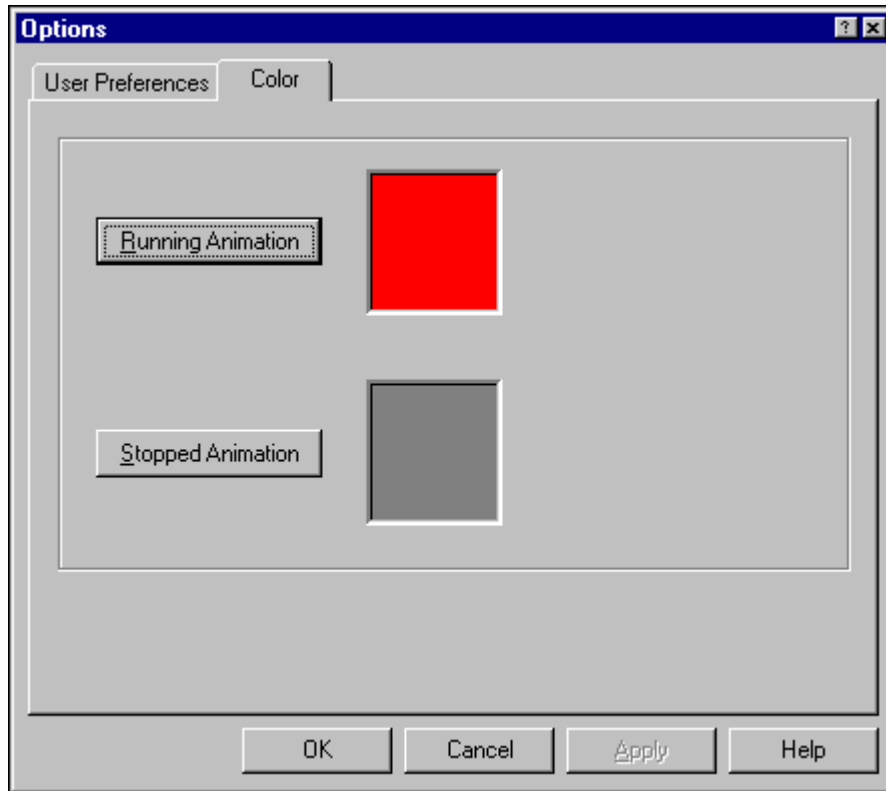
## Animation Color

You can select a color that will outline power flow when animation is running and another color to display when animation is stopped.

### To set animation color

Press the option  button on the Standard toolbar or choose View Options to open the Option box.

- Click on the Color tab.
- Click on Running Animation and choose a solid color from the Color box that appears.
- Click on Stopped Animation and choose a solid color from the Color box that appears.
- Click OK to save your choices.





## Status Bar

---

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



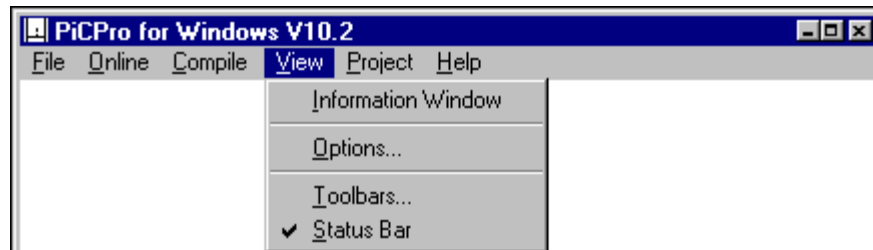
- Lists the location of the focus in your ladder.
- Indicates whether or not the PiC is connected to your PC. (Shown connected.  indicates connection to PiC.)
- 065 is the local node number of the PiC connected to PC through the RS232 programming port. All other PiCs in a network are assigned a node number from 1 to 255 (excluding 65). The network node number will appear at this location when you connect the PC to a PiC on the network using the Connect to Node command.
- Blank until forcing is turned on. The  symbol will appear in this location when forcing is on.
- Dynamically updates the time when the scan is running.

## Displaying/Hiding the Status Bar

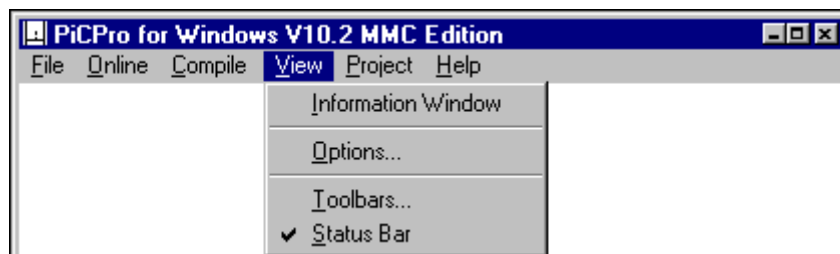
You can choose to display or hide the status bar.

- Choose View, Status Bar from the 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.

**PiCPro for Windows:**



**PiCPro for Windows MMC Edition:**



## 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 that PiCPro requires to proceed. Typically, you will see the following in a dialog box.

- OK button
- Cancel button
- Option buttons
- Check boxes
- Display boxes
- List boxes
- Spin boxes
- Tabs

### Toolbars

The toolbars give you quick access to some of the frequently used tools. You can access these functions in various ways such as from View and Toolbars under the menu bar, through hot keys, or right-clicking the mouse.

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 and Toolbars are selected from the menu, as many as seven 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:

#### Horizontally



Along the top or bottom of your PiCPro screen

#### Vertically



Along the left or right edge of the PiCPro screen

#### Labeled



Anywhere within the PiCPro screen or on the desktop

**NOTE: When the toolbar has a label, it is possible to resize it.**

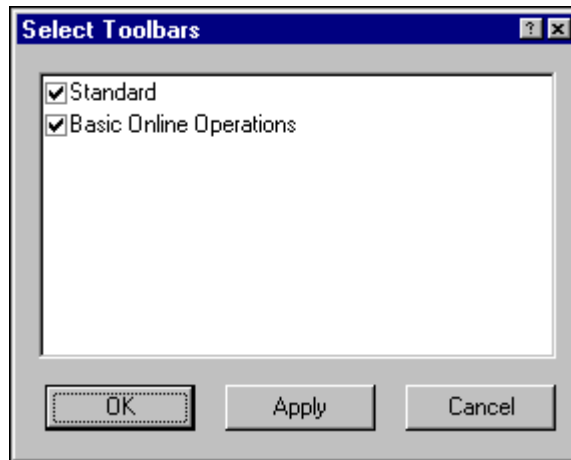
## Docking Tips

- 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 View, Toolbars from the 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

One method of organizing your files when working with PiCPro is to create subdirectories in the PiCPro for Windows VXX.X subdirectory. The PiCPro for Windows VXX.X subdirectory will already contain the Libraries subdirectory holding all the standard functions and functions blocks.

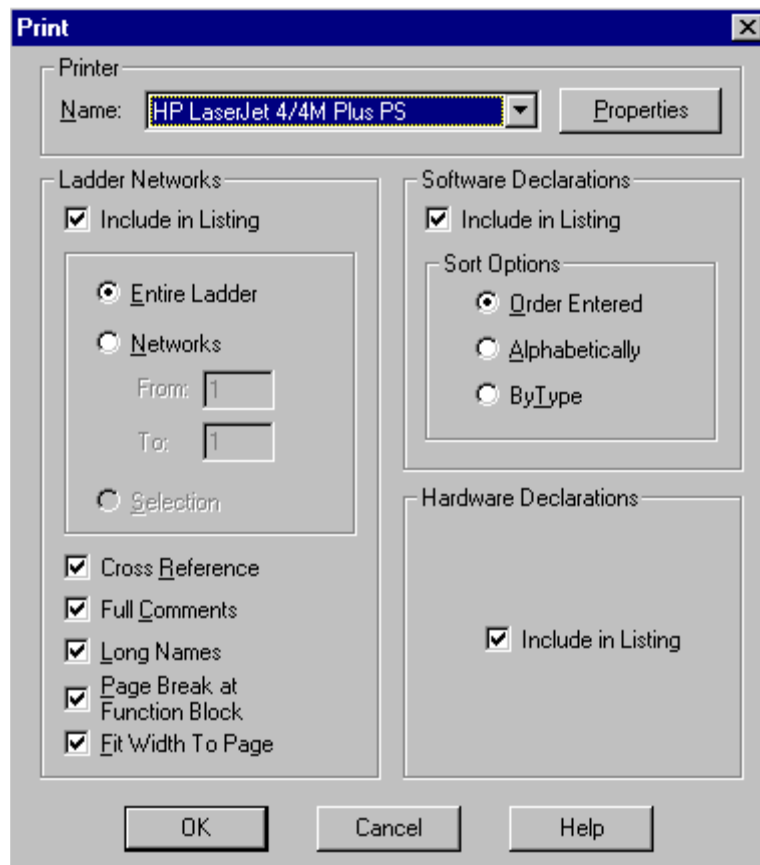
You can create the following subdirectories as required.

- An ASFB subdirectory to hold any Application Specific Function Blocks you may be using.
- A USER subdirectory to hold any User Defined Function Blocks you may create.
- A PROJECT subdirectory for each project you develop.

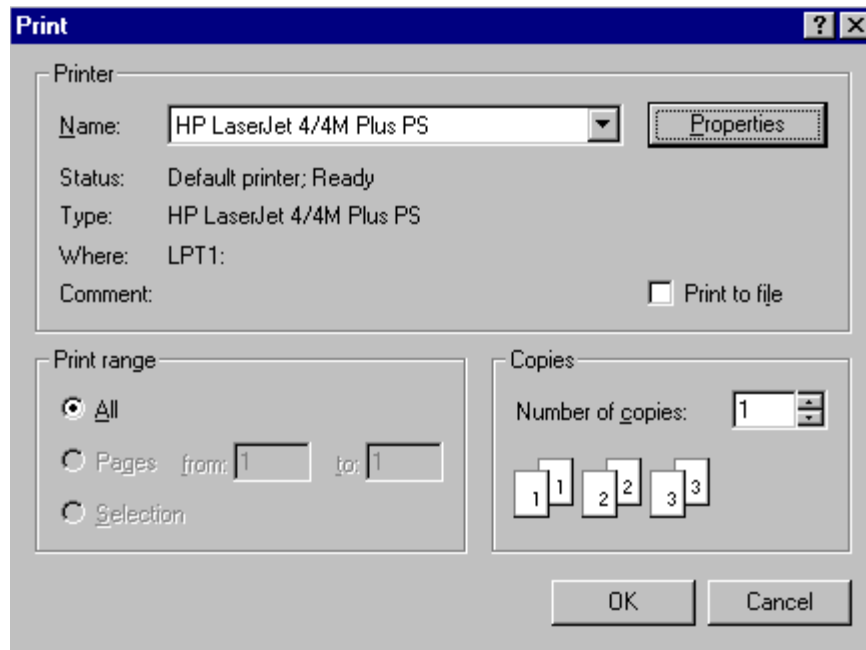
## Printing

---

You can print your LDO file. The print dialog box below shows all the default settings. Choose File, Open and select the file. Next, choose File, Print. The Properties button allows you to select choices in the areas of Paper, Graphics, Device Options, and PostScript.



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 View/View List/File/Print, View/Force List/File/Print or View/Information Window/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

- Make the window that contains the ladder active.
- Choose any of the following to bring up the print dialog box:

File, Print

Ctrl+P

Right click the mouse

The  button

Make your selections or accept the defaults and click OK in the print dialog box.

## **Selections in the Ladder Network Box**

### **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/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.

- Beneath contacts/coils representing direct inputs/outputs, the I/O point is displayed.
- Beneath all other contacts, the source network(s) is displayed.

### **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 that you have entered a long name for in software declarations.

### **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.

## **Selections in the Software Declarations Box**

### **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.

## **Selections in the Hardware Declarations Box**

### **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**

- Ensure that the list or window you want to print is active by clicking in it. If you want to print part of the list or window, highlight that portion before choosing the print command.
- Choose any of the following to bring up the print dialog box:

File, Print

Ctrl+P

Right click the mouse

The  button

Make your 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.
I04.01	Appears under a direct input to indicate the following:
or	Input in master rack (I), slot 4 (04), channel 1 (01)
I2.04.01	or
	Input in expansion rack 2 (I2), slot 4 (04), channel 1 (01)
O03.01	Appears under a direct output to indicate the following:
or	Output in master rack (O), slot 3 (03), channel 1 (01)
O2.03.01	or
	Output in expansion rack 2 (O2), slot 3 (03), channel 1 (01)

## 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

## Using Help

---

The Help documentation includes an on-line Help system, a context-sensitive or What's This? System, and the Function/Function Block Reference Guide. Some of the ways you can access the Help system is 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 will display information that is relevant to the current status of the application.

### Using 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 page.

#### To print a specific Help topic

- Click the Print button that appears along the bottom right-side of the Help window.  
or
- Right click the mouse and click Print Topic

#### To print an entire section of Help

1. Display the Contents page of the Help system and highlight the section you want to print.
2. Click the Print button that appears along the bottom right-side of the window.



## CHAPTER 2 Working with PiCPro for Windows

### A PiCPro Session

---

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.

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.

This section also covers information about basic operations you need to be familiar with to use PiCPro.

#### 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, you must first launch the program.


#### To start PiCPro

- Click the Start button in the Windows 95 Start menu.
- Click the PiCPro entry in Programs.

#### Creating New Files

Once PiCPro is started, you can create a new LDO file or a new setup file depending on which program you have chosen to open.

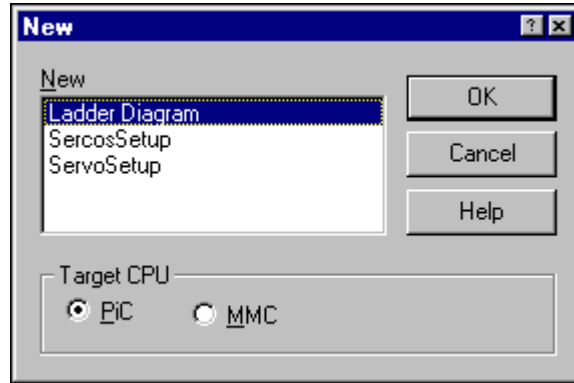
#### To create a new file

- Do one of the following:  
Choose File, New  
or  
Click the  button in the Standard toolbar.

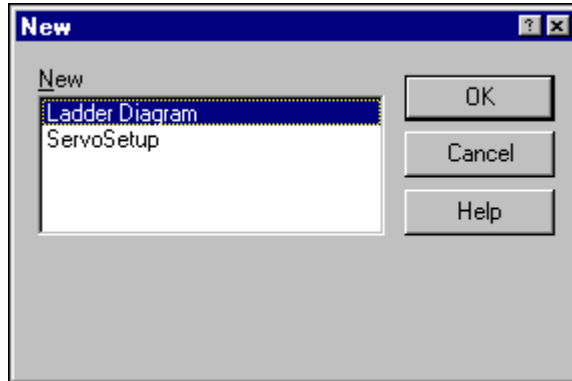
The New box appears and you can choose Ladder Diagram, Servo Setup, or SERCOS setup. When creating new files in the full version of PiCPro for Windows, you can also select the CPU type (PiC or MMC). The default selection is the last selection made. In the PiCPro for Windows MMC Edition the CPU type is always MMC.

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

**PiCPro for Windows: File / New**



**PiCPro for Windows MMC Edition: File / New**



## Opening an Existing Ladder

Once you start PiCPro, you can open an existing file. The Open command opens files that have already been saved.

### To open an existing file

- Do one of the following:

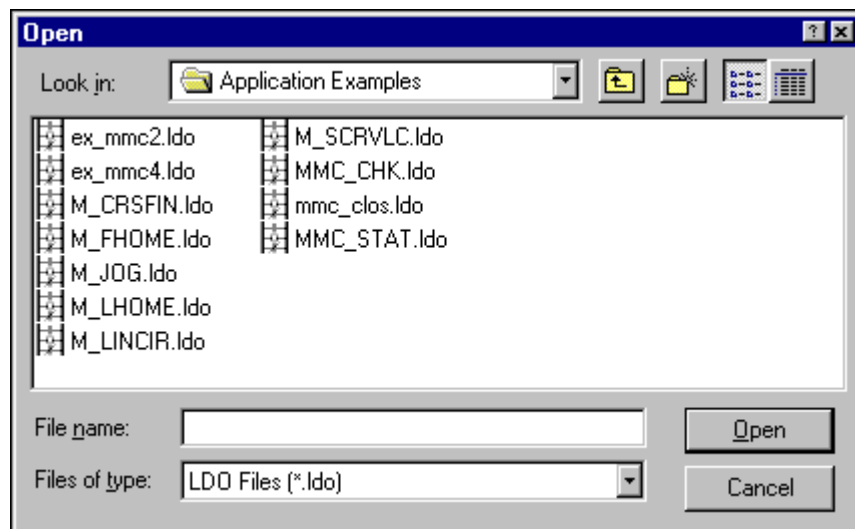
Choose **F**ile, **O**pen

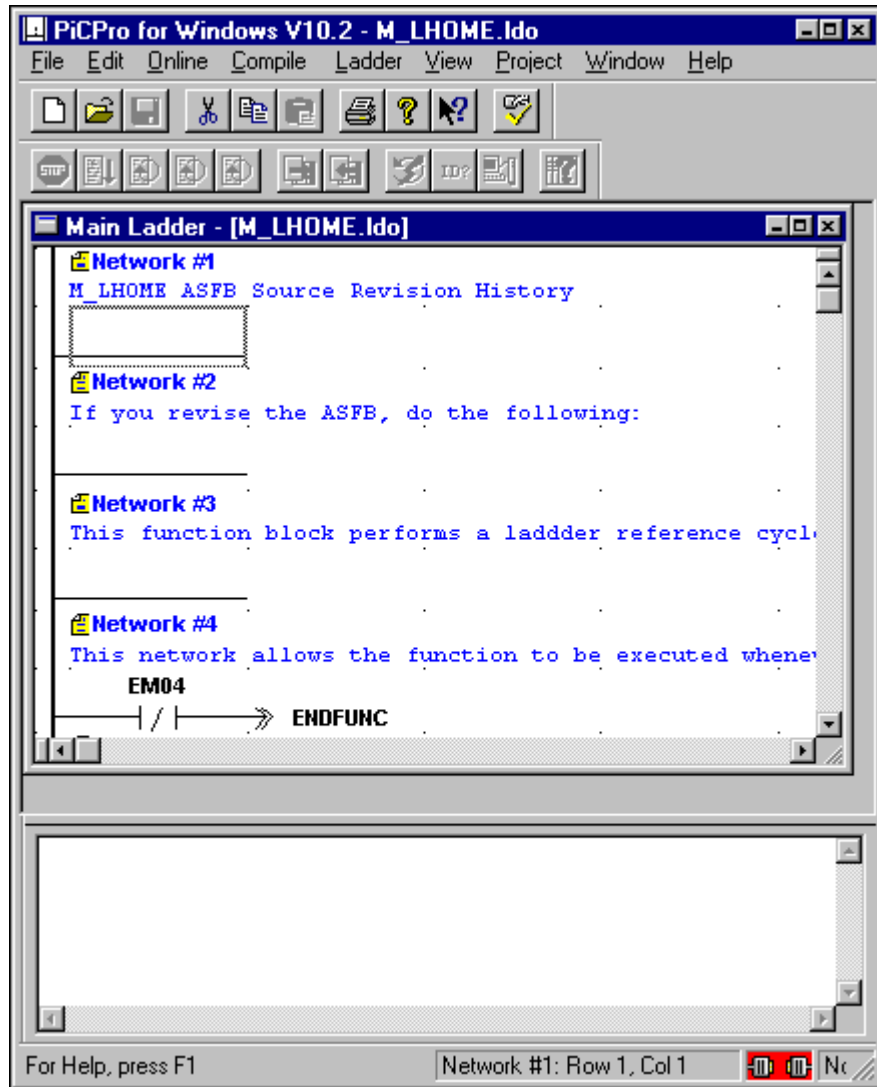
or

Click the  button

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

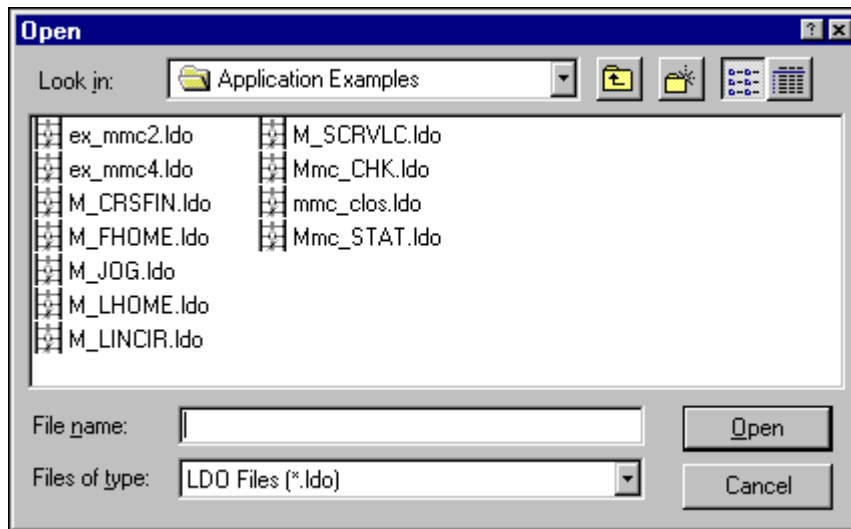
### PiCPro for Windows: File / Open



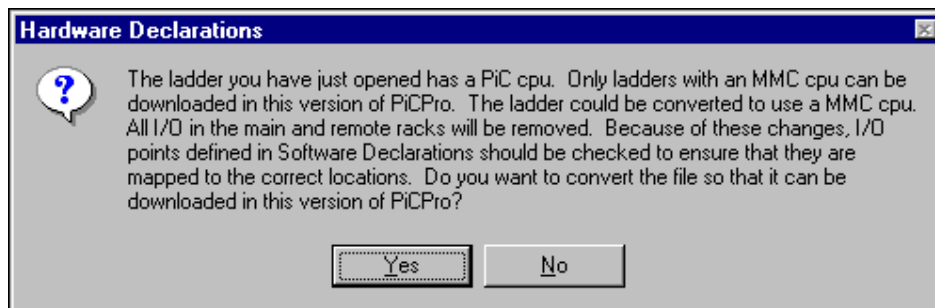


Opening an LDO file with a PiC CPU in PiCPro for Windows MMC Edition will result in a confirmation prompt as shown below. If you choose not to convert the LDO, the file will be opened but cannot be downloaded, compiled etc..

#### PiCPro for Windows MMC Edition:File / Open



#### PiCPro for Windows MMC Edition:File / Open / Open with a PiC CPU



#### To open a recently opened file

- Choose **F**ile.  
A list of the last four opened files appears at the bottom of the menu. These will be either LDO, SRV, or SRC files.
- Click on the file you want to open.

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

### To clear application memory (for use with MMC CPU)

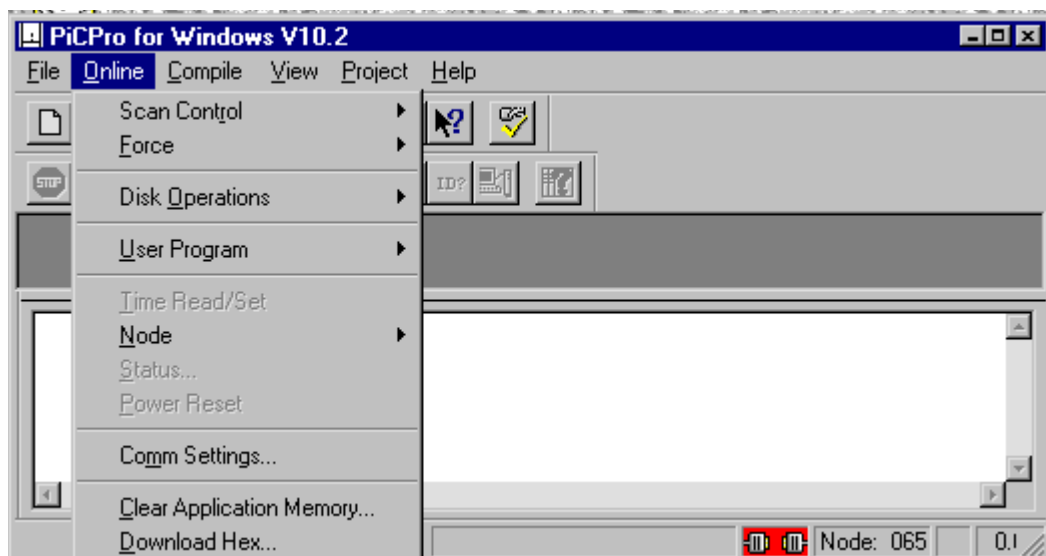
If you are using an MMC control, the Clear Application Memory command provides a software solution to clearing out a ladder application that may not be scanning correctly. When you choose the command from the Online menu, the Download Hex dialog box appears. The entry field for a hex file is grayed because PiCPro automatically enters a file called clapp.hex. This file is down loaded to the MMC when you choose Start and turn the control off and on again. It clears the application memory in the MMC.

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 MMC control, communication is re-established and you can proceed to download a ladder file.

1. Click on Online.
2. Click on Clear Application Memory.



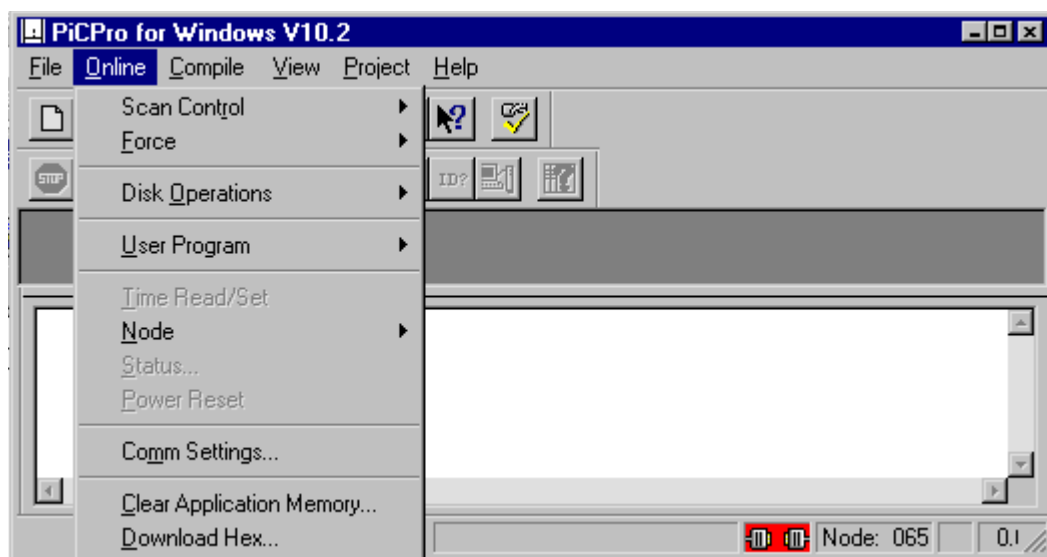
## To Download a Hex File

The Download Hex File command is available in PiCPro for Windows and PiCPro for Windows MMC Edition and allows you to do any of the following;

- Update the control's firmware
- Load the ladder flash memory on a PiC 9041, PiC 9043 or MMC CPU
- Clear the ladder flash memory.
- Clear the ladder memory.
- Configure application/RAMDISK size.
- Update TCP/IP firmware.
- Update the SERCOS module.

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 opened, this command will be grayed and inaccessible. The following procedure is used to download a hex file:

1. Click Online.
2. Click Download Hex.




3. The Download Hex dialog box appears. Use Browse to select the Hex file you want to download.
4. Click on the appropriate Baud Rate.
5. Select the port if different from the default (PiCPro port).
6. Click on Start to begin downloading the hex file. Follow the prompts. 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.

The toolbars give you quick access to some of the frequently used tools. These are the toolbars available in PiCPro.

- Standard
- Basic Online Operations
- Advanced Operations
- Ladder
- View Navigator
- Functions
- Compiler

### Using the Pointer Tool

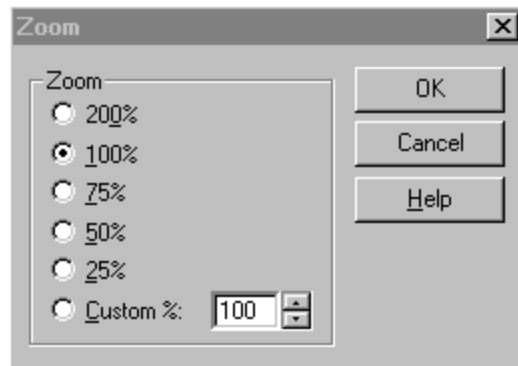
The Pointer Tool  button is found on the Ladder Toolbar. It is the default selection allowing you to point and click to select locations in the ladder. You can exit the drop mode for any of the other Ladder toolbar buttons by selecting the Pointer Tool.

### Zooming In and Out

Zooming in and out allows you to view and work on your program from 5% to 500% magnification.

#### To use the zoom command from the view menu



1. Choose View, Zoom. The Zoom 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.



## To use the zoom command from the 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.

Keys	Focus Moves:
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
Ctrl + End	To the End of Module marker
Tab	One element to the right of the current element
Shift + Tab	One element to the left of the current element
Page Down/Up	One vertical viewable page down/up
Ctrl + Page Down/Up	One horizontal viewable page left/right

### 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.

You can also select cells using the keyboard.

<b>Keys</b>	<b>Selects</b>
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.
Shift + Ctrl + End	The focus cell and all cells to the last row in the network if focus cell is not 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.
Ctrl + A	All cells in the ladder

### **Dragging and Dropping**

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

### **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 a Network

---

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.

### **Adding, Changing, or Deleting Networks**

#### **To insert a network**

1. Position the focus where you want to insert a network. The network will be inserted above the focus cell.
2. Choose Ladder, Network Insert or press <Shift + Insert>.
3. A new network appears above the focus cell. All the networks below the insertion are renumbered.

#### **To change an element in a network**

1. Position the focus over the element you want to change.
2. Select the new element from the toolbar and click on the focus cell.
3. The new element will be placed in the focus location.

#### **To delete a network**

1. Position the focus anywhere in the network you want to delete.
2. Choose Ladder, Network Delete 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 Ladder, Row Delete or press <Delete>.

## Copying and Deleting

---

The Cut, Copy, and Paste commands allow you to use the Clipboard to make copies of items from your ladder.

The Copy command places a copy of the selected item on the Clipboard.

The Cut command removes the item from your ladder and places it on the Clipboard.

Once an item is on the Clipboard, you can use the Paste or Paste Insert commands to place the item within or between LDO files. 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.

The Delete 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.

### Cutting an Item

When you cut an item, it is removed from the LDO file and placed on the Clipboard.

#### To cut an item

1. Select one or more cells containing 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 Edit, Cut from the menu.  
Press <Ctrl + X>.  
Right-click the mouse.

### Copying an Item

When you copy an item, it remains in the LDO file and is placed on the Clipboard.

#### To copy an item

1. Select one or more cells containing 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 Edit, Copy from the menu.  
Press <Ctrl C>.  
Right-click the mouse.

## Pasting an Item

To paste an item into the LDO file from the Clipboard, you can use either the Paste or the Paste Insert commands.

### To use the Paste command

1. Select a cell to be the target cell for the paste.
2. Choose one of the following methods to issue the Paste command:

Click the  button from the standard toolbar.  
Choose Edit, Paste from the menu.  
Press <Ctrl V>.  
Right-click the mouse.

## Deleting an Item

The Delete command removes the selected item. It does not place it on the clipboard.

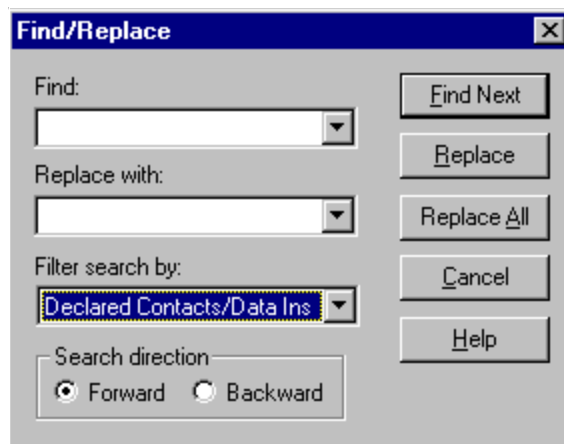
### To use the Delete command

1. Click on the item or highlight it.
2. Choose Edit, Delete or press <Ctrl + D>.

## Finding and Replacing

---

In PiCPro, you can search for any declared elements, for functions/function blocks, for network or jump labels, or for network numbers. You can also 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. The search is refined by applying a filter that limits the search. You can search 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.



Both the Find: and Replace with: boxes can hold 10 of the most recently entered names. On subsequent searches, you can choose from these lists.

NOTE: You must enter the complete name of the item you want to search for.

### Finding and Replacing

#### To search for an element in the ladder

1. Choose Edit, Find/Replace or press <Alt F3> to bring up 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 the first occurrence of the element you want to search for, it will appear in the Find box.
3. Be sure the correct filter is selected in the Filter search by: box.
4. Choose the search direction.
5. Select the Find Next button to move either forward or backward to the next location of the element in the ladder.
6. Continue selecting the Find Next button or press <Enter> if you want to continue to find occurrences of the element. When the ladder has been completely searched, a message will inform you.

## To search and replace an element in the ladder

1. Enter the name of the ladder element you want to search for in the Find box. If the focus in your ladder is on the first occurrence of the element you want to search for, it will appear in the Find box.
2. Enter the name you want to replace the variable with in the Replace with: box. If the replacement name has not been added to the software declarations table, you will be prompted to do so now.
3. Be sure the correct filter is selected in the Filter search by: box.
4. Choose the search direction.
5. First choose the Find/Next button to find the element.
6. 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 and the correct filter will be entered in the Filter search by: box.
- If the focus is on a wire or a return label, things that cannot be searched for, the Filter search by: box defaults to Declared Contacts/Data Ins. 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.

### Types of Filters

Declared Contacts/Data Ins  
Declared Coils/Data Outs  
Declared Function Blocks  
All Declared Elements  
Function/Block Titles

### Ladder Elements to Find

Contact or data input

Coil or data output

Function block

Contact, coil, data input, or  
data output

Functions

Network or jump label

Numbered network

## Finding Duplicates

PiCPro can search for duplicated function blocks, variables, contacts, and coils. When PiCPro finds a duplicated element, a message appears 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 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.

## Saving Files

It is a good idea to save your file at regular intervals as you work on it. Using the Save command, you can save the file under its existing name. Using the Save As command, you can specify a new filename and/or a location where you want the file stored.

### To save a new file

1. Choose File, Save or the  button.
2. In the Save As box, choose a drive and folder where you want to save your LDO.
3. Enter a name in the File Name box.
4. Click Save.

### To save an existing file

- Choose File, Save or the  button.



### To save all files open on the desktop

- Choose File, Save All.

NOTE: 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 File Name box and click Save.

### 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

Before you close a file, save the file if you want to keep the changes you have made since the last save. If you do not want to save those changes, close without saving.

#### To close a file

- Choose File, Close

or



- Click on the corner box on the menu bar to drop down list. Choose Close from this list or press <Ctrl + F4>.

### 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>.

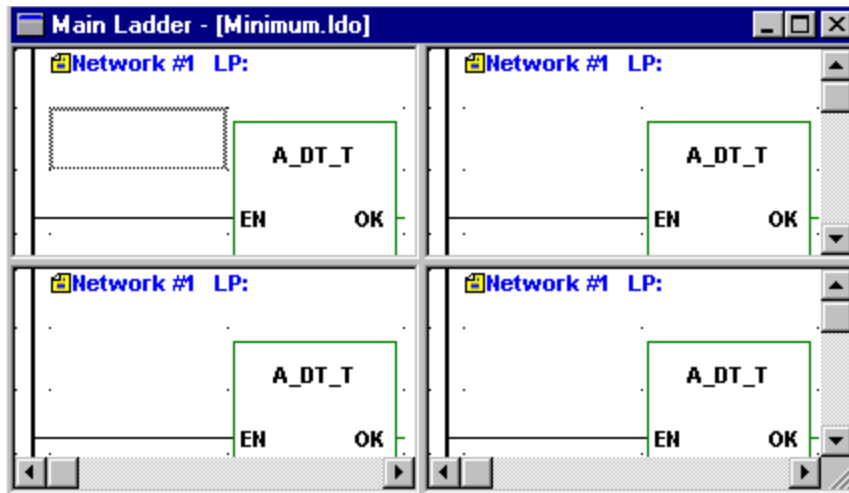
PiCPro asks if you want to save any unsaved changes in any open files.

- Click Yes to save changes before exiting.
- Click No to exit without saving changes.
- Click Cancel to exit the dialog box and continue working on 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. 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

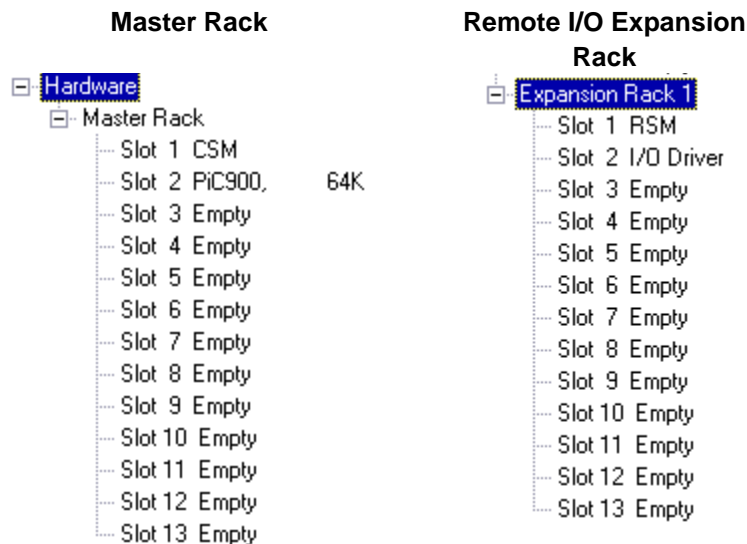
---

In the hardware declarations table, you define what type of hardware module occupies each slot or block in the control's master rack and in either the remote I/O expansion racks or block I/O modules if used. NOTE: From one to seven remote I/O expansion racks can be added when the CPU is a PiC.

The hardware configuration in the hardware declarations table is compared to the software configuration 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. The hardware configuration is different based on the CPU type.

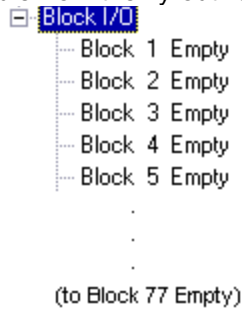
### Hardware Declarations Table - PiC CPU

Master Rack/Remote I/O Expansion Rack	
Slot 1	In the Master Rack, reserved for the Central Services Module (CSM) In the Remote I/O Expansion Rack, reserved for the Remote Services Module (RSM)
Slot 2	In the Master Rack, reserved for the CPU module. In the Remote I/O Expansion Rack, reserved for the I/O driver module.
Slots 3 - 13	Defines the type and location of the I/O modules.



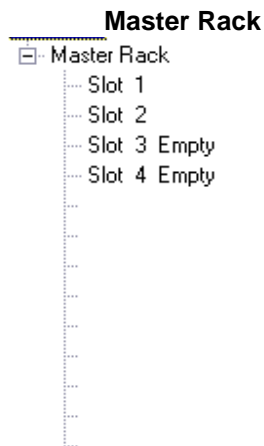
### Block I/O Expansion

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



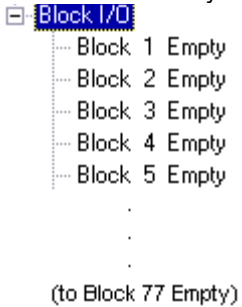
### Hardware Declarations Table - MMC CPU

Master Rack	
Slot 1	2 1/2 or 4 1/2 Axes Servo
Slot 2	MMC CPU and General I/O
Slots 3 - 4	Defines the type and location of the additional I/O modules.




### Block I/O Expansion

Blocks1 - 77 Defines the type and location of the I/O modules.  
Select the module type and then the correct module from the fly-out list provided



### Entering Hardware Declarations

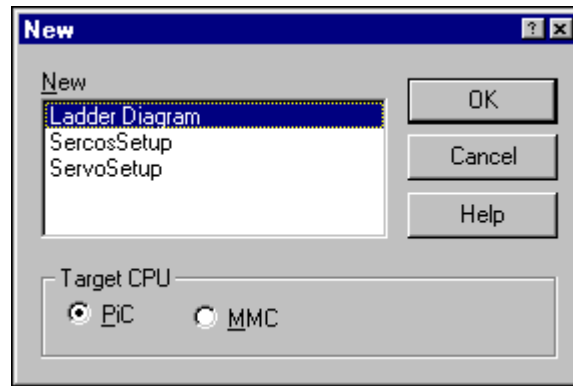
When you are ready to enter your hardware declarations, keep the following in mind.

- To bring up the Hardware Declarations table, choose View, Hardware Declarations or select the  hardware button from the View Navigator toolbar.
- You can switch to using the keyboard for selecting the modules you want to insert by pressing the spacebar.
- You can switch to using the keyboard for the editing commands by pressing <Shift> <F10>.

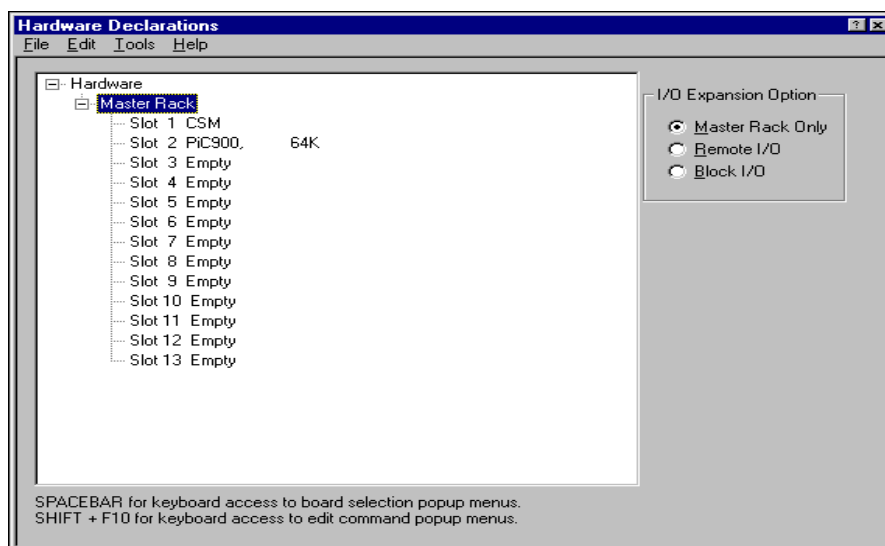
## Master Rack Declarations


To enter hardware declarations for Master Rack for a PiC CPU:

1. Choose File, New, Ladder Diagram, and click on PiC at Target CPU.



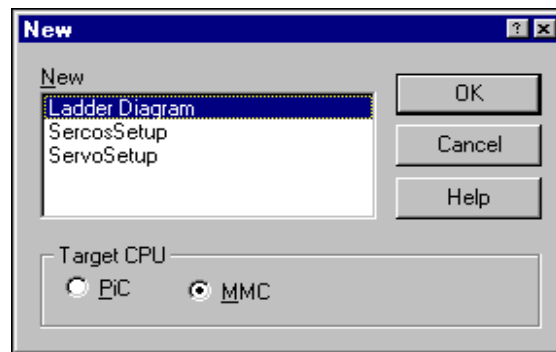
2. Choose View and Hardware Declarations.



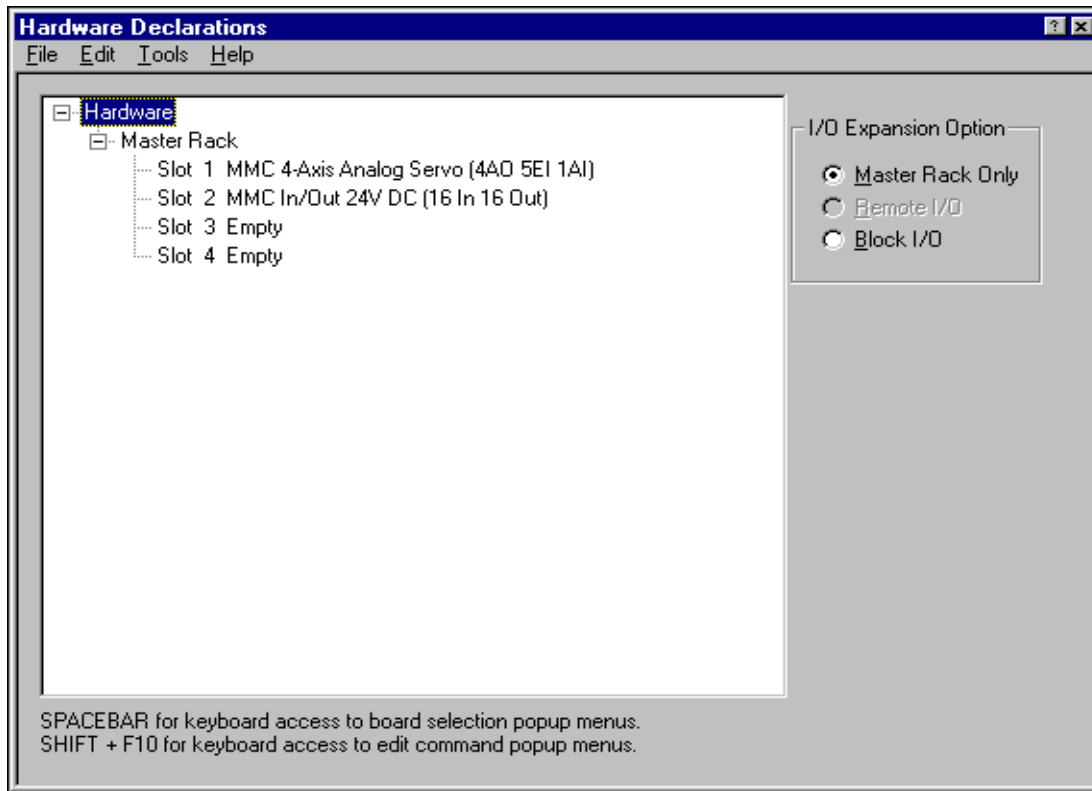
3. Position the cursor on the slot you want to declare a hardware module in and click. A fly-out box appears. If an arrow  is at the end of the item on the fly-out list, there are more choices.
4. Slide the mouse over each fly-out list highlighting the correct description of the module you want to declare.
5. When you are at the final fly-out list, click the mouse to make your selection. Your choice will be listed in the master rack list.

**To enter hardware declarations for Master Rack for an MMC CPU:**

1. Choose File, New, Ladder Diagram, click on MMC at Target CPU and click on OK.



2. Choose View and Hardware Declarations.



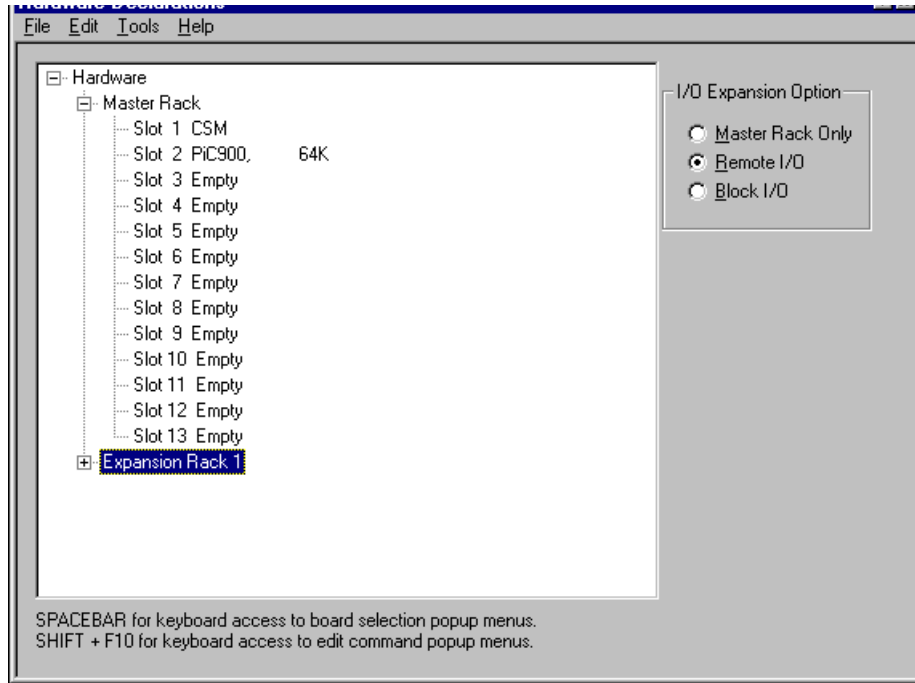
3. Position the cursor on the slot you want to declare a hardware module in and click. A fly-out box appears. If an arrow ► is at the end of the item on the fly-out list, there are more choices.
4. Slide the mouse over each fly-out list highlighting the correct description of the module you want to declare.
5. When you are at the final fly-out list, click the mouse to make your selection. Your choice will be listed in the master rack list.



## Remote I/O Expansion Rack Declarations

### To add declarations for a remote I/O expansion rack for PiC CPU:

1. Choose File, New, Ladder Diagram, click on PiC at Target CPU and click on OK.
2. Choose View, Hardware Declarations, and select Remote I/O in the I/O Expansion Option box.

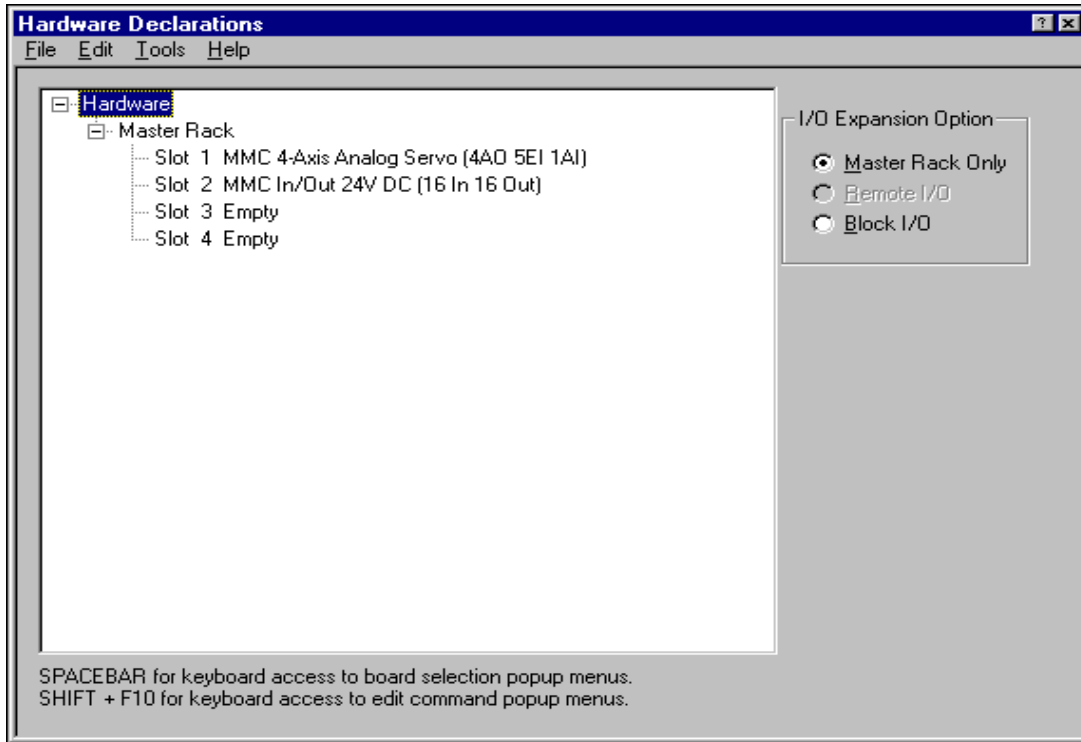


3. Expansion Rack 1 appears after the Master Rack list. Click on the + sign in front of Expansion Rack 1 and the list will appear. NOTE: You can add up to seven expansion racks to your system. Any additional remote I/O expansion racks will be numbered consecutively.
4. Note that slots 1 and 2 are defined and cannot be edited. Position the cursor on the first slot you want to declare a hardware module in and 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.
5. Slide the mouse over each fly-out list highlighting the correct description of the module you want to declare.
6. When you are at the final fly-out list, click the mouse to make your selection. Your choice will be listed in the expansion rack list.

### To add declarations for a remote I/O expansion rack for MMC CPU:

**Note:** When MMC is chosen as the Target CPU, it is not possible to add declarations for a remote I/O expansion rack.

1. Choose File, New, click on MMC at Target CPU and click on OK.
2. Choose View, Hardware Declarations, and notice that Remote I/O in the I/O Expansion Option box is not available.

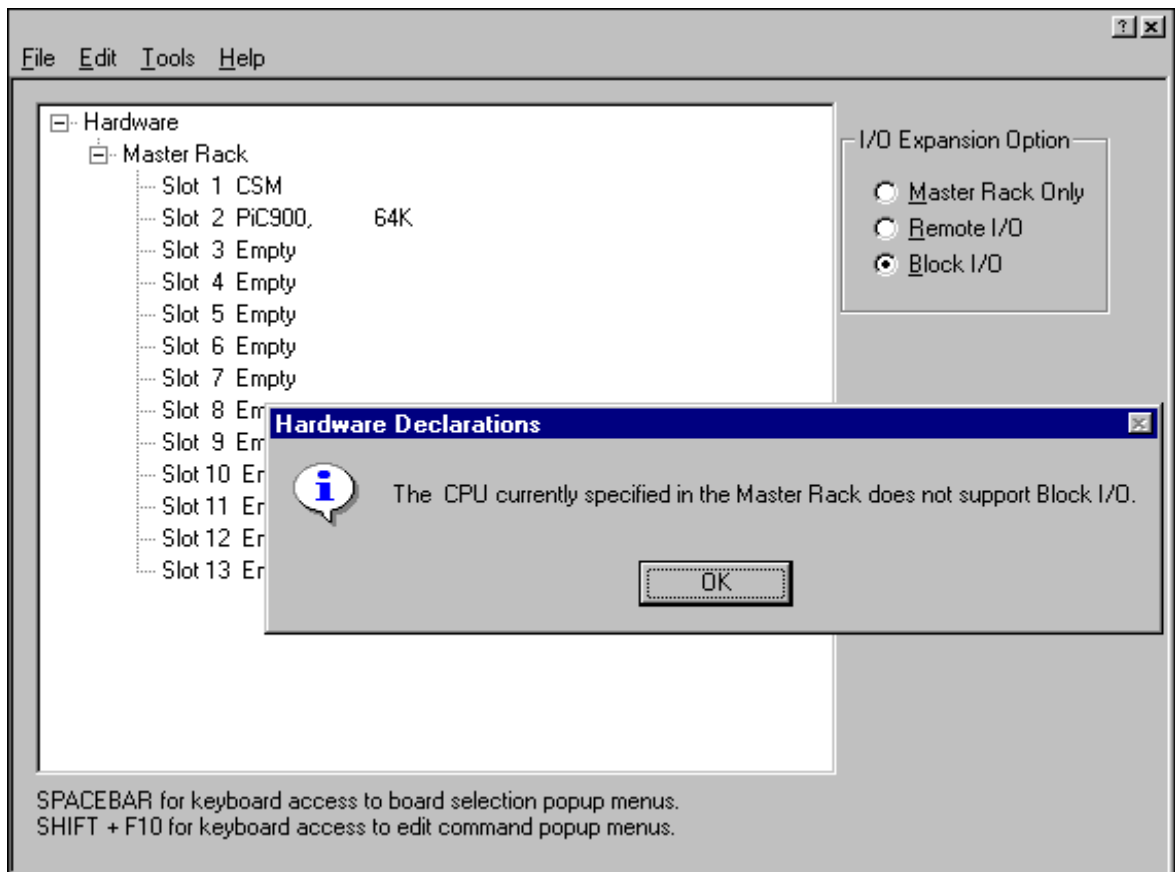


## Block I/O Declarations

### To enter declarations for block I/O expansion modules for a PiC CPU:

1. Choose File, New, Ladder Diagram, click on PiC at Target CPU and click on OK.
2. Choose View, Hardware Declarations, and select Block I/O in the I/O Expansion Option box.

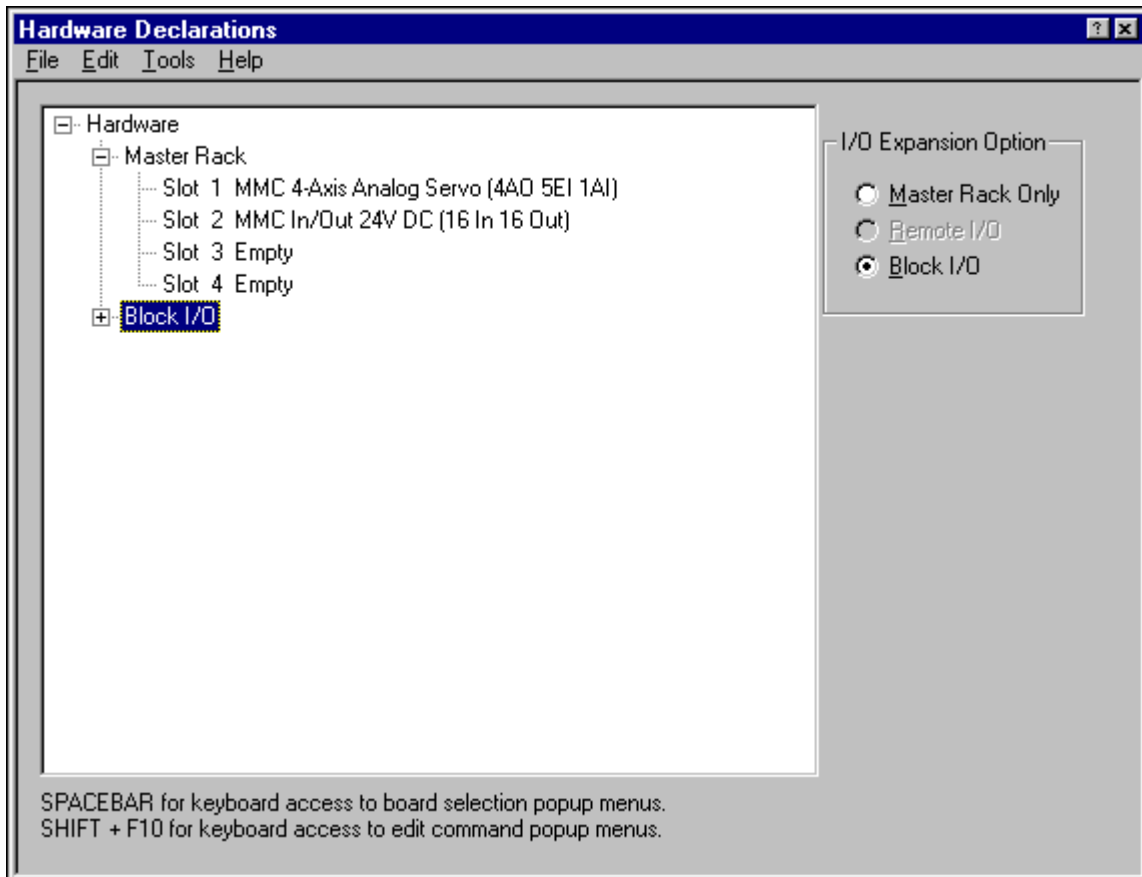
**NOTE:** A pop-up box may appear stating “The CPU currently specified in the Master Rack does not support Block I/O”. Click on OK to clear this message..



3. If the PiC CPU does support Block I/O the above message box will not appear. Proceed with step 4.
4. Block I/O appears after the Master Rack list. Click on the + sign in front of Block I/O and the list will appear.
5. Position the cursor on the first block you want to declare a hardware block module in and 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.
6. Slide the mouse over each fly-out list highlighting the correct description of the block module you want to declare.
7. When you are at the final fly-out list, click the mouse to make your selection. Your choice will be listed in the block I/O list.

**To enter declarations for block I/O expansion modules for MMC CPU:**

1. Choose File, New, Ladder Diagram, click on MMC at Target CPU and click on OK.
2. Choose View, Hardware Declarations, and select Block I/O in the I/O Expansion Option box.



3. Block I/O appears after the Master Rack list. Click on the + sign in front of Block I/O and the list will appear.
4. Position the cursor on the first block you want to declare a hardware block module in and 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.
5. Slide the mouse over each fly-out list highlighting the correct description of the block module you want to declare.
6. When you are at the final fly-out list, click the mouse to make your selection. Your choice will be listed in the block I/O list.

## Editing Hardware Declarations

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

### Inserting and Deleting in the Hardware Declarations Table

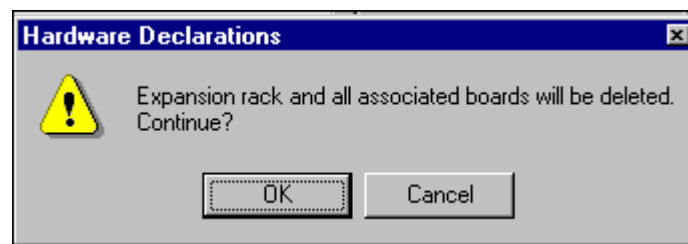
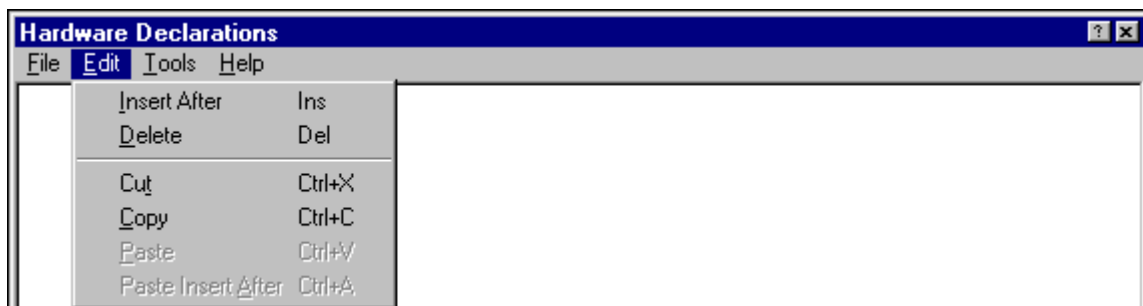
The following are choices available for inserting and deleting in the Hardware Declarations Table:

Item	Insert Remote Expansion I/O Racks	Delete Remote Expansion I/O Racks	Insert Block I/O Racks	Delete Block I/O Racks
PiC CPU	Yes	Yes	Yes	Yes
MMC CPU	No	No	Yes	Yes

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

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

1. Select the object you want to delete.
2. Choose Edit, Del~~e~~te or <DEL>. 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.



#### To insert additional remote I/O expansion racks

1. Select the expansion rack after which you want to insert another rack.

2. Choose Edit, Insert After or <Ins>. The new expansion rack will be placed directly after the one you selected. If there are any racks after it, they will be renumbered.

## Cutting and Copying Hardware Declarations

The Copy and Cut 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 Copy command places a copy of the selected object on the clipboard.

The Cut 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
  - CPU, I/O, Block Modules

Copy command **cannot** be used:

- 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

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 Edit, Copy 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 Edit, Cut or <CTRL>X to cut the selected object.

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

- When you cut an I/O or block module, the location reverts to Empty. With the block module, all remaining modules are moved up in the list.
- When you cut an expansion rack or a block branch, the rack or branch is removed. With expansion racks, any remaining racks are renumbered.

**Pasting and Inserting Paste in Hardware Declarations**

Once an object is on the clipboard, you can use the Paste command or the Paste Insert After 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 Edit, Paste or <CTRL>P. 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 Edit, Paste Insert After 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, a block module, or a block I/O branch. The selected object can be the master rack, an expansion rack, the block I/O root, or an individual block module.

**To copy an object or a default I/O to the clipboard for storage in software declarations, in the hardware declarations table using PiCPro for Windows MMC Edition:**

Right click on the MMC I/O board. The default I/O name for that slot will be copied to the clipboard.

## Printing Hardware Declarations

The hardware declarations table can be printed using the Print command from the File 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 File, Print or <CTRL P>.
2. The Print 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 File, Save & Close or <F10>, your table will close and any changes will be saved.
2. If you choose File, Close 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.

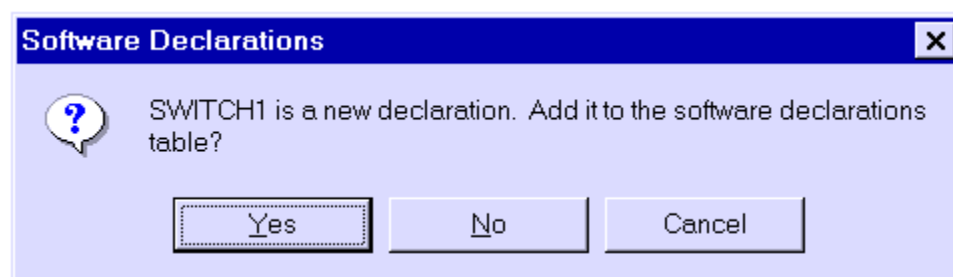
You can perform the following editing functions in the software declarations table:

- 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 Dclarations 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.



The software declarations table appears and you can add the variable to the declarations. There are six columns in the table in which you enter information.

Software Declarations						
File Edit Tools Help						
	Name	Type	A.	I/O Point	Initial Value	Long Name
	end list	void				

- 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. 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 eight alphanumeric/underscore characters.

- The first character must be alpha (A-Z) or the underscore (\_).
- The second through eighth character can be alpha, numeric (0-9), or underscore.

If the name is changed in this column, every occurrence of the variable in your ladder will 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 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 ladder.

## 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.
- For block I/O modules, the block 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 hardware module must be declared in the Hardware Declarations table.

## MMC I/O Point Labels

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.

PiCPro allows you to enter the I/O information in the software table for the general I/O on the CPU section and the servo I/O on the analog section. The information can be copied by:

- Selecting the CPU (Slot 2) or Analog (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 module CPU and for the axis and auxiliary connectors on the analog module. The information in column 2 Declared Names will be pasted in the Name column and the information in column 3 MMC I/O point 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)	PiC I/O Point
16 general DC Inputs	GENI1 – GENI16	IGEN.1 through IGEN.16	I3.1 – I3.16
16 general DC Outputs	GENO1 – GENO16	OGEN.1 through OGEN.16	O3.1 – O3.16
2 short circuit Inputs	SHORT1- SHORT2	ISGEN.1 through ISGEN.2	I3.17 – I3.18
6/12 auxiliary DC Inputs	AUXI1- AUXI12	IAUX.1 through IAUX.12	I4.1 – I4.12
Axis 1 DC input	AX1READY	IA1.1 (Axis 1, Input 1)	I4.13
Axis 2 DC input	AX2READY	IA2.1	I4.14

Axis 3 DC input	AX3READY	IA3.1	I4.15
Axis 4 DC input	AX4READY	IA4.1	I4.16
Axis 1 DC output	AX1ENABL	OA1.1 (Axis 1 Output 1)	O4.1
Axis 2 DC output	AX2ENABL	OA2.1	O4.2
Axis 3 DC output	AX3ENABL	OA3.1	O4.3
Axis 4 DC output	AX4ENABL	OA4.1	O4.4
Axis 1 DC output	AX1RESET	OA1.2 (Axis 1 Output 2)	O4.5
Axis 2 DC output	AX2RESET	OA2.2	O4.6
Axis 3 DC output	AX3RESET	OA3.2	O4.7
Axis 4 DC output	AX4RESET	OA4.2	O4.8
Axis 1 fast input	AX1FINPT	IFAUX.1 (aux port, Axis 1, Fast Input)	I4.17
Axis 2 fast input	AX2FINPT	IFAUX.2	I4.18
Axis 3 fast input	AX3FINPT	IFAUX.3	I4.19
Axis 4 fast input	AX4FINPT	IFAUX.4	I4.20
Axis 49 fast input	DIGFINPT	IFAUX.49 (aux port, Digitize 49, Fast Input)	I4.21
Expansion input (future)		I3.1 – I3.x	I5.1 – I5.x

## Numbering

### PiC CPU

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

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

### Master Rack MMC CPU

1 <sup>st</sup>	I or O	Input or Output
2 <sup>nd</sup>	Connector /Type	GEN, AUX, A1, A2, A3, A4, FAUX
	. (point)	Used as a separator
	First digit of channel number	
	Second digit of channel number	

### 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 an MMC CPU.

Enter five to eight characters.

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

### Block Expansion Rack I/O

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

Enter five to seven characters.

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

Name	Type	I/O Point
OUT1	BOOL	O4.1
OUT2	BOOL	O4.2
.	.	.
.	.	.
.	.	.
OUT16	BOOL	O4.16
FB1	BOOL	I4.1
FB2	BOOL	I4.2
FB3	BOOL	I4.3
FB4	BOOL	I4.4

### The 24V I/O 16/8 source module with two fuses

Name	Type	I/O Point
IN1	BOOL	I5.1
IN2	BOOL	I5.2
.	.	.
.	.	.
.	.	.
IN16	BOOL	I5.16
OUT1	BOOL	O5.1
OUT2	BOOL	O5.2
.	.	.
OUT8	BOOL	O5.8
FB1	BOOL	I5.17
FB2	BOOL	I5.18

## Fast Inputs

### PiCPro 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

### MMC CPU

The fast inputs available on the 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)

## 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

---

When a variable is declared in the software declarations table, you assign a data type to it in the Type column. The categories of data types available are listed below with their associated data types.

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/block

Structures/arrays

### 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	<b>BOOL</b> 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.	<b>BYTE</b>	8	eight 0s - 1s
	<b>WORD</b>	16	sixteen 0s - 1s
	<b>DWORD</b> double word	32	thirty-two 0s - 1s
	<b>LWORD</b> 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:
<b>On</b>	(hex) 16#80 or (decimal) 128
<b>Off</b>	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.	<b>SINT</b> short integer	8	-128 to +127
	<b>INT</b> integer	16	-32,768 to +32,767
	<b>DINT</b> double integer	32	-2,147,483,648 to +2,147,483,647
	<b>LINT</b> 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.	<b>USINT</b> unsigned short integer	8	0 to 255
	<b>UINT</b> unsigned integer	16	0 to 65,535
	<b>UDINT</b> unsigned double integer	32	0 to 4,294,967,295
	<b>ULINT</b> 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.	<b>REAL</b>	32	significant digits 6-7
	<b>LREAL</b> 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.	<b>STRING</b>	Two plus number of declared characters	1 to 255 ASCII characters

NOTE: STRINGS cannot be entered as constants.

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).	<b>DATE</b>	16	Jan. 1, 1988 - Dec. 31, 2051	Year/month/day D#1997-10-23
	<b>TIME_OF_DAY</b>	32	00:00:00 - 23:59:59	Hours/minutes/ seconds TOD#23:59:59
	<b>DATE_AND_TIME</b>	32	Same as the two above, combined	Year/month/dayhours/ 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.	<b>TIME</b>	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 Block

Constants can be inputs to any function/blocks except functions that operate on **STRING**s (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 **STRING**s 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 functions/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.

### 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.

## Entering an Array in Software Declarations

### 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.
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.
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. 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

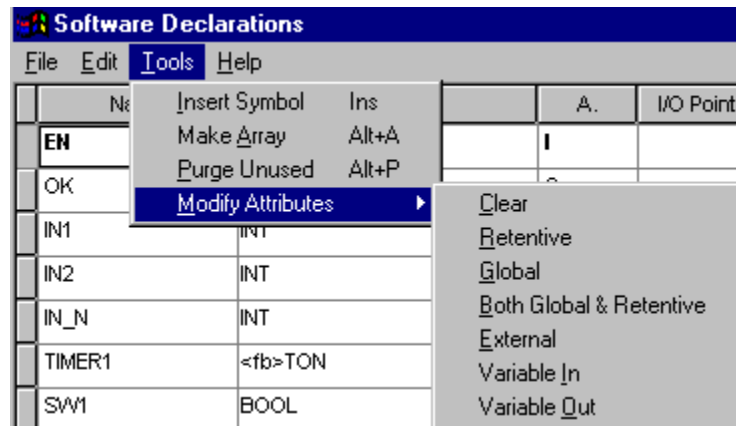
1. Position the selected cell anywhere on the variable declaration you want to change.
2. Press <Alt + A> or the Make Array command.
3. Enter a new array length.
4. The index numbers displayed behind the data type of the variable will be updated.
5. To initialize or change the initial values of the elements 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 Make Array 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.

## Working with Attributes

You can assign a retentive, global, global and retentive, external, variable in and 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 from the **Tools**, **Modify Attributes** command or by right clicking. The Clear choice will remove an attribute from a selected variable.



### 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.

### Global Attribute

The global attribute is used when DOS LDO merge is being 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

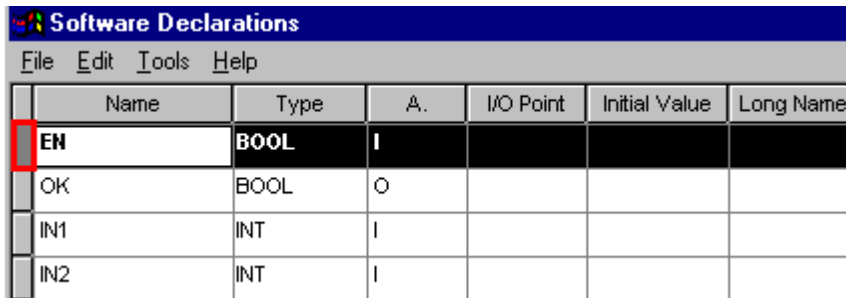
1. Click in the Name column of the table (or highlight the entire row) of the entry you want to delete.
2. Choose Edit, Delete or press <Delete>. A confirmation box will appear. You can choose Yes, No, or Cancel. If you choose Yes, the entry will be deleted if it is not used in the LDO.


## 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 in red 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 Edit, Cut from the menu  
or  
Press <Ctrl + X> on the keyboard  
The cut entry is placed on the clipboard.

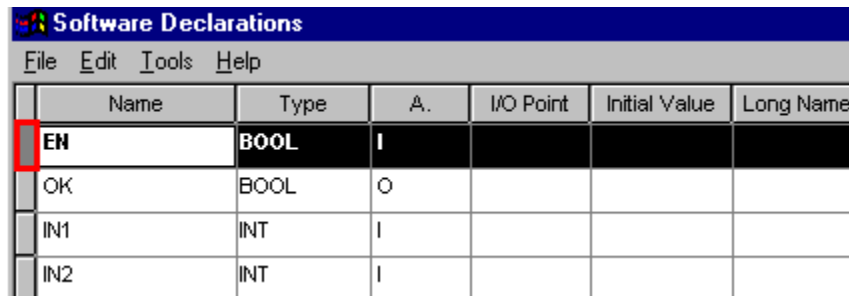
If the entry is not referenced in the LDO, the entry is removed from the table.

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.

### 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 in red 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 Edit, Copy from the menu  
or  
Press <Ctrl + C> on the keyboard  
The copy entry is placed on the clipboard.

### 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 Edit, Paste from the menu.  
or  
Press <Ctrl V> on the keyboard.

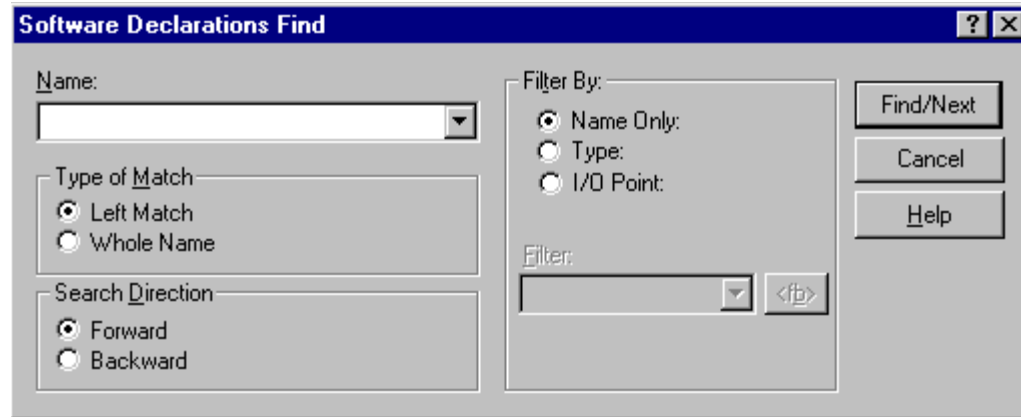
**Note:** Refer to the following table. If a ladder has an MMC declared in hardware declarations and all I/O points are copied and pasted into software declarations, the I/O points shown in column 1 will be displayed. If the CPU in hardware declarations is changed to a PiC CPU and then the software declarations are displayed, the I/O points will change to those in column 2.

Software Declaration I/O assignment (MMC I/O Point)	PiC I/O Point
IGEN.1 through IGEN.16	I3.1 – I3.16
OGEN.1 through OGEN.16	O3.1 – O3.16
ISGEN.1 through ISGEN.2	I3.17 – I3.18
IAUX.1 through IAUX.12	I4.1 – I4.12
IA1.1 (Axis 1, Input 1)	I4.13
IA2.1	I4.14
IA3.1	I4.15
IA4.1	I4.16
OA1.1 (Axis 1 Output 1)	O4.1
OA2.1	O4.2
OA3.1	O4.3
OA4.1	O4.4
OA1.2 (Axis 1 Output 2)	O4.5
OA2.2	O4.6
OA3.2	O4.7
OA4.2	O4.8
IFAUX.1 (aux port, Axis 1, Fast Input)	I4.17
IFAUX.2	I4.18
IFAUX.3	I4.19
IFAUX.4	I4.20
IFAUX.49 (aux port, Digitize 49, Fast Input)	I4.21
I3.1 – I3.x	I5.1 – I5.x

## 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 Edit, Find 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. Select the Name Only button if it is not already selected.
2. Enter the name (or portion of a name with Left Match selected) in the Name box. The list accessed through the down arrow will hold the most recently searched for names. You may choose from this list if applicable.

### Find by Type

To search by data type:

Select the Type button. This will enable the Filter box.

- 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 choose from the lists.

### Find by I/O Point:

**To search by I/O point:**

1. 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 with the purge unused command.

### To purge unused variables

Choose any of the following:

Tools, Purge Unused

or

<Alt + P>

or

Right click the mouse and select Purge Unused

## Working with Network Elements

---

Your LDO consists of executable networks entered by you. They hold the ladder logic and commands that run your application. They are numbered sequentially by the software as you enter them. The ladder logic is executed in this numerical order unless a jump command forces execution to be out of sequence. Execution occurs left to right, top to bottom within the network.

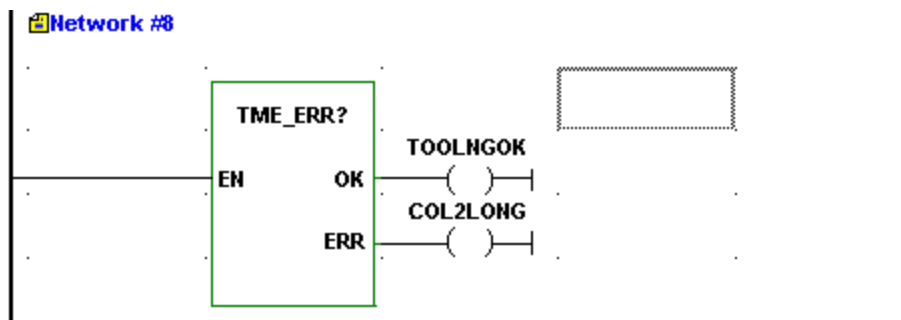
A label can be added to a network number line.

### Network Size

A 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.

PiCPro accepts a network with an area of from 1 to 255 cells. The area of a network is the product of its width (widest row) times its length (longest column).

A portion of a 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.



## Elements of a Network

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

### Contacts

Contacts 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 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

normally open positive transition

normally closed positive transition

normally open negative transition

normally closed negative transition

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

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

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

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

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

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

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)

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)

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
	combination

There are several ways to add wires to your network.

Buttons	Wire Types
	horizontal
	vertical
	combination

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 Vertical, Horizontal, or Both.
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
	(on menu)	Moves to cell above vertical wire
	Ctrl + I	Moves to cell below vertical wire
	Ctrl + M	
Horizontal	Ctrl + Shift + H	Remains over horizontal wire
	(on menu)	Moves to cell to left of horizontal wire*
	Ctrl + J	Moves to cell to right of horizontal wire*
	Ctrl + K	wire*

Combination	Ctrl + Shift + B (on menu)	Remains over the combination wire
-------------	-------------------------------	-----------------------------------

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

The horizontal wire provides a horizontal connection between elements in the network.

The vertical wire provides a vertical connection between elements in the network.

The combination wire provides a branch connection between elements in the network.

## Functions/Function Blocks

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.




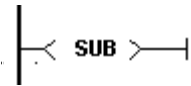


## To Enable Function/Function Block Connections

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.

## Jumps

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 Label	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 Subroutine	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.
	
Return	Places the symbol for the return command (<return>) into the network.
	Can be placed: <ul style="list-style-type: none"><li>■ In a subroutine where it returns you to the network from which the jump originated</li><li>■ In the main module (typically at the end of the program before any subroutines) where it takes you to the &lt;End Of Module&gt; in the ladder</li><li>■ In a UDFB source ladder (typically at the beginning) where it takes you to the &lt;End Of the Module&gt; if the EN (enable) is not set.</li></ul>
	

### 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 network you want to insert the jump into.
2. Under the Ladder menu, select Jumps. From the Jumps flyout choose either To Label or To Subroutine. Or choose the button from the Ladder toolbar.

### **Labels**

You can label a network with the Label command found in the Ladder menu under the Network 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 to a network**

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

### **Long Names**

You can add a long name to any variable used in a network. Long names can have up to 40 characters on four lines (10 per line). You can view, edit or add a long name to a variable in a 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 ladder.

### **Adding/Editing Long Names in a Network**

1. Place the cursor on the variable you want to add/edit a long name.
2. Under the Edit menu, choose Properties.
3. Select the Symbol tab.
4. Enter/edit the long name and click OK.

### **Adding/Editing Long Names in the Software Declarations Table**

1. Place the cursor on the variable you want to add/edit a long name. This will put you in the correct row of the software declarations table.
2. Under the View menu, choose Software Declarations.
3. Tab across the selected variable's row to the Long Name column.
4. Enter/edit the long name and exit the table.

## Viewing Long Names

1. Use the Long Name button on the View Navigator toolbar to view long names or proceed with steps 2 through 5.  
Note that all cells in your LDO increase in size when you display Long Names.
2. Under the View menu, choose Options.
3. Select the User Preference tab.
4. In the Hide box, be sure the Symbol Long Name is not checked.
5. Click OK.

## Comments

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

### To add comments to your ladder

1. Under the Edit menu, select properties.
2. Select the Network tab. In the Comment area type in the comments you want for the network you are working in.

### To choose the number of comment lines to display

1. Under the View menu, select Options.
2. Select the User Preferences tab. In the Ladder View Preferences 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 Cancel.

## Variables and Constants

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 can be used as input values to function/function blocks. They cannot be entered anywhere else in a 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.

Variables are used in three 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.

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

## 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

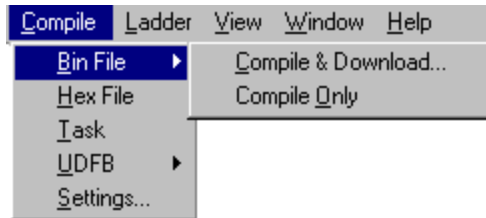
Downloading sends the active LDO on your PC to PiC memory after it has been compiled into an executable program. You can backup this executable form from the PiC to the PC by using the One, User Program, Backup command or the  button on the Basic Online Operations toolbar. You can restore any BIN file to the PiC memory with the One, User Program, Restore command or the  button on the Basic Online Operations toolbar.

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 PiC before they take effect. Always backup this source LDO to ensure that you will have a copy of your LDO in its source form. Include in the backup the filenames for the LDO with any of the following extensions:

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

## Compiling a Bin File


There are two options under the Compile, **B**in File command; Compile and Down-  
load... or Compile Only.

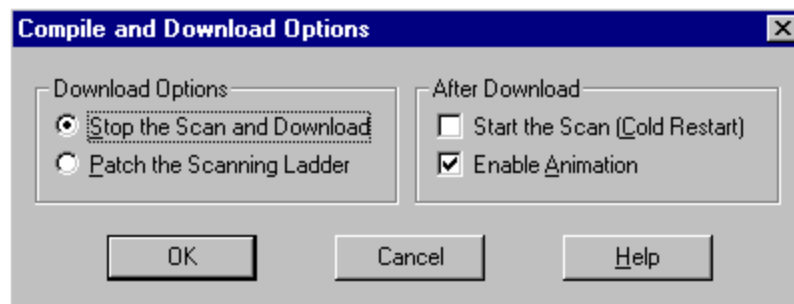


These commands can be accessed from the compile and download button or the compile only button on the Compiler toolbar.

### 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 PiC.

1. Ensure that your PC is connected to a PiC.
2. Choose the Compile, **B**in 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 window.

PiC Memory Usage  
115 of 8k data bits, 216 of 32k data bytes  
1669 ladder code bytes, 39440 total code bytes

There are:

8K bits for booleans

32K bytes for variables

ladder code bytes for your application ladder


total code bytes for your application ladder plus libraries\*



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

## Compile Only

The Compile Only command allows you to create a binary file without downloading it to the PiC. 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 a PiC.

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.

```
PiC Memory Usage
115 of 8k data bits, 216 of 32k data bytes
1669 ladder code bytes, 39440 total code bytes
```

There are:

8K bits for booleans

32K bytes for variables

ladder code bytes for your application ladder


total code bytes for your application ladder plus libraries\*

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

## Compiling and Downloading 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 EPROMs or loaded to FLASH memory on processes that support this option.

### To Compile a Hex File

1. Open the LDO you want to compile as a hex file.
2. Choose Compile, Hex 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 filename 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.

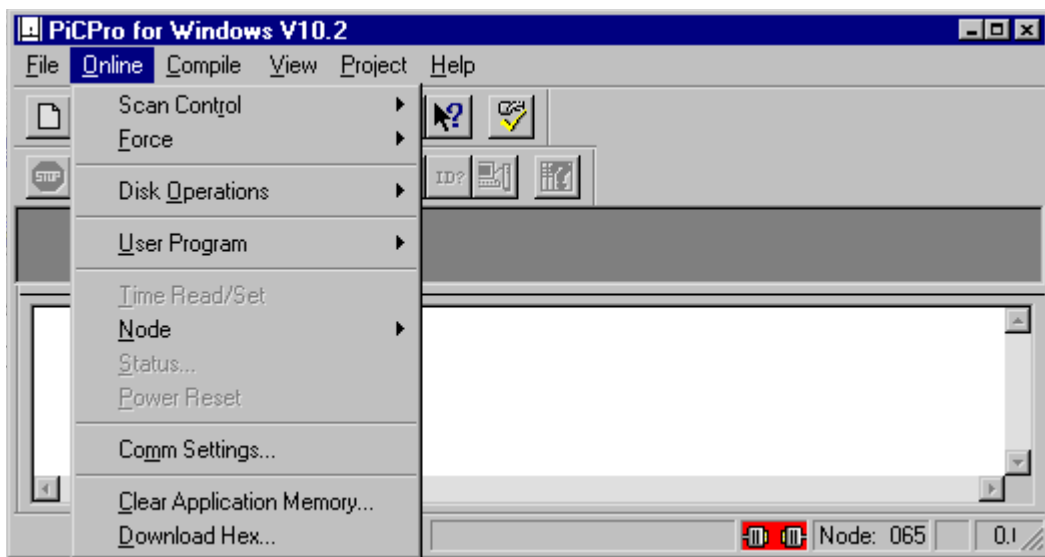
## To Download a Hex File

The Download Hex File command is available in PiCPro for Windows and PiCPro for Windows MMC Edition and allows you to do any of the following;

- Update the control's firmware
- Load the ladder flash memory on a PiC 9041, PiC 9043 or MMC CPU
- Clear the ladder flash memory.
- Clear the ladder memory.
- Configure application/RAMDISK size.
- Update TCP/IP firmware.
- Update the SERCOS module.

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 opened, this command will be grayed and inaccessible. The following procedure is used to download a hex file:

1. Click Online.
2. Click Download Hex.




3. The Download Hex dialog box appears. Use Browse to select the Hex file you want to download.
4. Click on the appropriate Baud Rate.
5. Select the port if different from the default (PiCPro port).
6. Click on Start to begin downloading the hex file. Follow the prompts. They will tell you to turn the control off and then back on again.

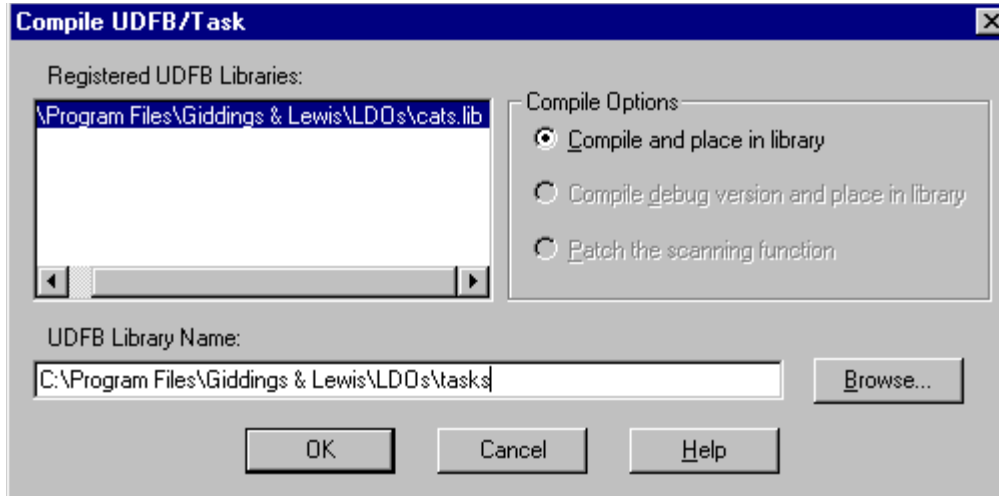
A status bar will show you the progress of the download. If any errors occur, they will be reported in the Information Window.

## 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.



You may choose from the Compile Options to Compile and place in library.  
NOTE: The other options are grayed and not available for TASKs.


You also 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 library box, you may select one of them.

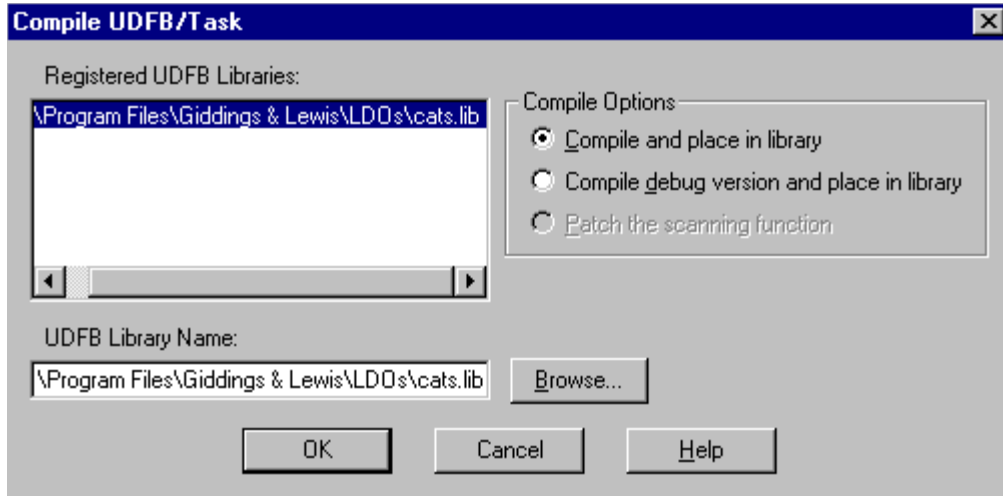
NOTE: You cannot place a TASK in any of the standard library files.

## 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.

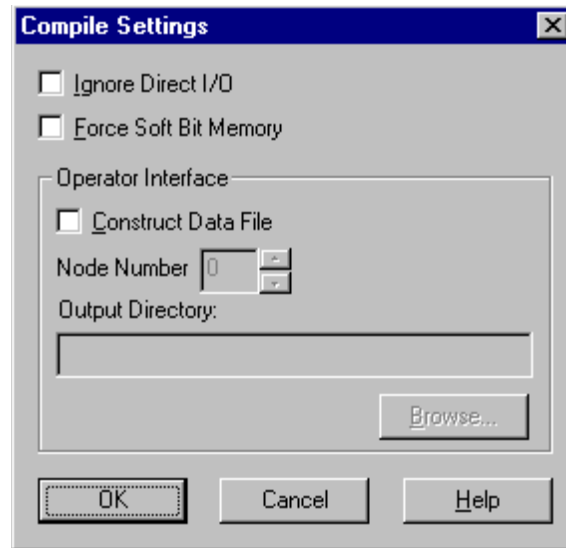
If there are existing libraries listed in the Registered UDFB library box, you may select one of them.

You are not allowed to place a UDFB in a standard library file.

## Settings

The Settings command under the Compile menu allows you to choose settings in three areas:

1. Ignore direct I/O
2. Force soft bit memory
3. Use an operator 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.

### 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.

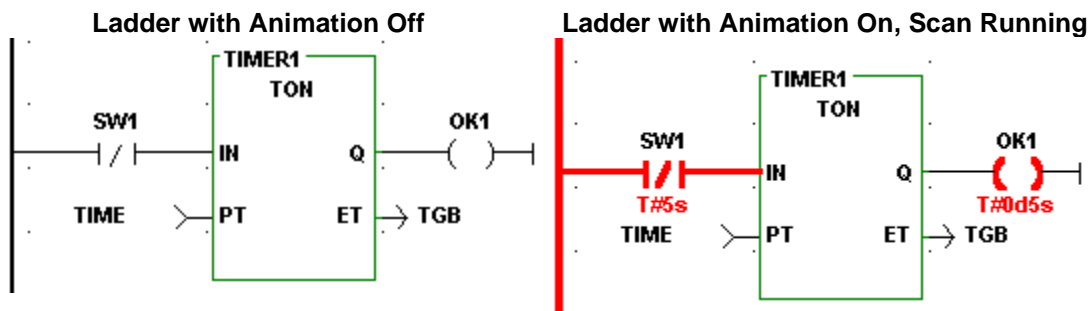
## Animating the Ladder

Animation allows you to monitor the power flow in the ladder diagram that has been compiled and downloaded to the PiC and in one or multiple UDFB and/or TASK LDOs that are displayed.

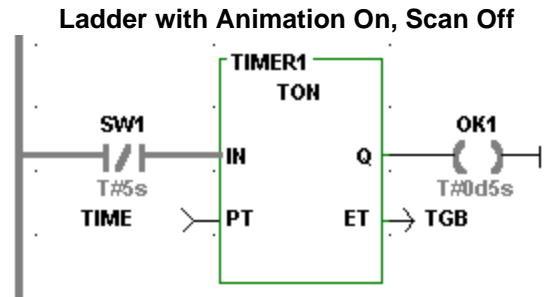
### To Turn Animation On

1. Compile and download the ladder.
2. Display any UDFB or Task LDOs you want to animate using the View, UDFB/Task command.
3. Ensure that the main ladder has focus.
4. Choose Online, Animate from the menu or 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</i> .#IND
Not-a-number	<i>digit</i> .#NAN
  - 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 PiC 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.
- Enter the edit mode for the LDO. Animation will not turn back on at the end of an edit. 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 Force List.

When animation is turned off, any numeric values shown above the variables are no longer displayed.

## Forcing 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 addition 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, 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 PiC.


### IMPORTANT

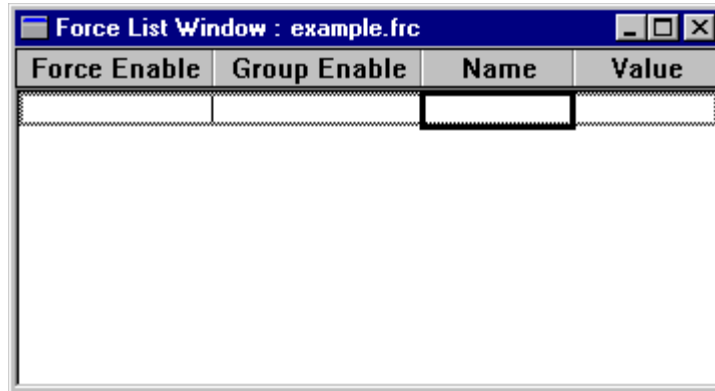
Animation may not reflect forced conditions. Physical inputs are forced at the beginning of the scan. All other variables are forced at the end of the scan. The writing of variable values by the LDO is not inhibited by forcing. Therefore, it is possible to change the state of a coil during the scan but not see the change in animation.



## Entering Variables in the Force List

With PiCPro 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 **V**iew, **F**orce from the menu or choose the  button from the View Navigator toolbar. The Force List appears with focus on the Name column. The Window and columns are sizable.



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 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 right click 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 PiC.
- 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 Enabled column or in the Group Enabled 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 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.
3. II. Turning On Forcing/Grouping from the Tool Bar Buttons
4. Check all variables that you want to force and/or group in the Force Enabled column or in the Group Enabled column respectively of the Force List.
5. Toggle forcing on by choosing the  button from the Advanced Operations toolbar.
6. 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 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 PiC after Force List is edited

While forcing, you can change any entries in the Force List. These changes can then be updated in the PiC.

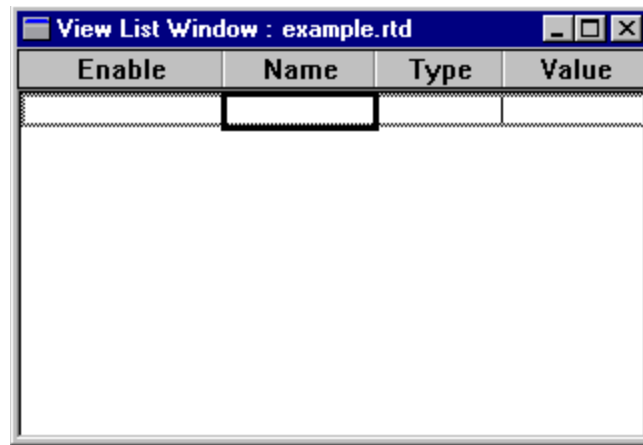
1. Edit the Force List.
2. Choose Online, Force, Update Force List from the menu or choose the  button from the Online menu.

NOTE: If you delete any or all entries from your force list without updating the PiC, the entries remain active.

## Viewing Enabled Variables

With PiCPro 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/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 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 View Enable and/or Group Enable column.

### 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. 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 View Enable and/or Group Enable column.

### 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 to View List**. The variable will be added to the end of the View 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 View Enable and/or Group Enable column.

### 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 right click 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 View Enable and/or Group Enable column.

### 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.
  - Choose Online, Animate from the menu or select the  button on the Advanced Operations toolbar to turn animation on in the View List.

## Running the Ladder

---

Once the cable has been connected between your PC and your PiC and your ladder program has been downloaded to the PiC, there are several things you can do that are described here.

- Controlling the scan
- Restarting the ladder
- On-line editing
- Animating the ladder
- Forcing variables
- Viewing variables
- Building a dependency list

### Connecting the PC to the PiC Controller

A cable is supplied with PiCPro software. One end of the cable is labeled for the PiC and the other for the PC. Connect the computer end to the serial port you designated in PiCPro and the PiC end into the PiCPro Port on the CPU or CPU/CSM module of the PiC control.

### Controlling the Scan

When the PiC is running, the CPU repeatedly scans the application program in its memory. A scan is the cyclical process of:

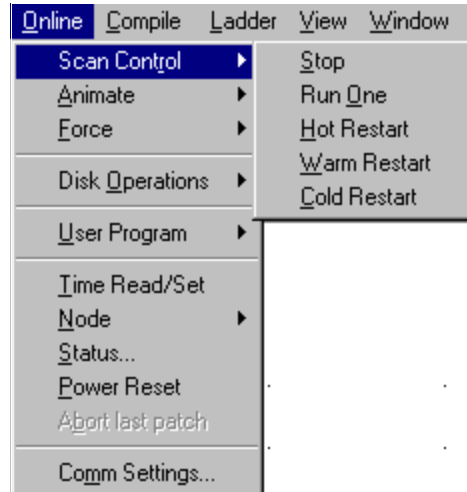
1. Reading data from the hardware input modules
2. Using the forcing list to modify input data
3. Executing ladder logic and using the data to update program variables
4. Using the forcing list to modify output data
5. Sending commands to the hardware output modules
6. Sending data to the PC to be displayed 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.

*The PiC system shuts down as a safety measure if either the ladder scan or the overhead takes more than 200 ms.*

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 PiC 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 One, 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 One, Scan Control, Run One from the menu.  
or
- Select the  button 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 One, Scan Control, Hot Restart from the menu.  
or
- Select the  button 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 One, Scan Control, Warm Restart from the menu.  
or
- Select the  button 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 One, Scan Control, Cold Restart from the menu.  
or
- Select the  button 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 PiC 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 PiC 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 PiC. 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.

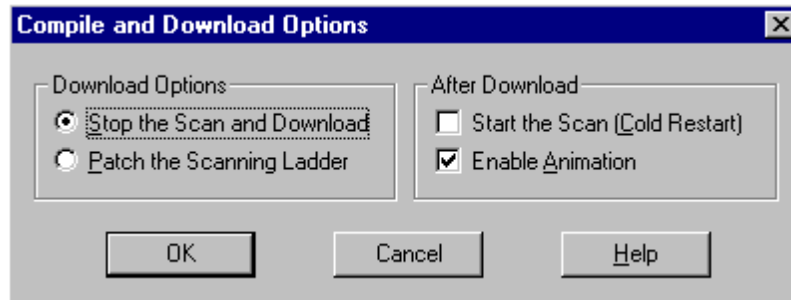
- Data Bits
- Data Bytes
- Code Bytes
- Number of Patches (out of 100)
- Number of Label/Function Links (out of 26)

Patch the Scanning Ladder is not available in the following situations.

- If any changes have been made to the hardware or software 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 26 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 Compile, Bin File, Compile and 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 Patch the Scanning Ladder and click OK.

## To abort a patch

You can undo the last patch downloaded to the PiC with the Abort Last Patch command. This will remove the patch from the PiC 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 PiC, choose Online, Abort last patch or press the  button on the Online toolbar.
  2. The last patch is removed from the PiC. 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 PiC.

## IMPORTANT

Aborting the last patch removes the operation but not necessarily the effect of that operation from the PiC. 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 network is removed from the PiC but the coil will remain on.

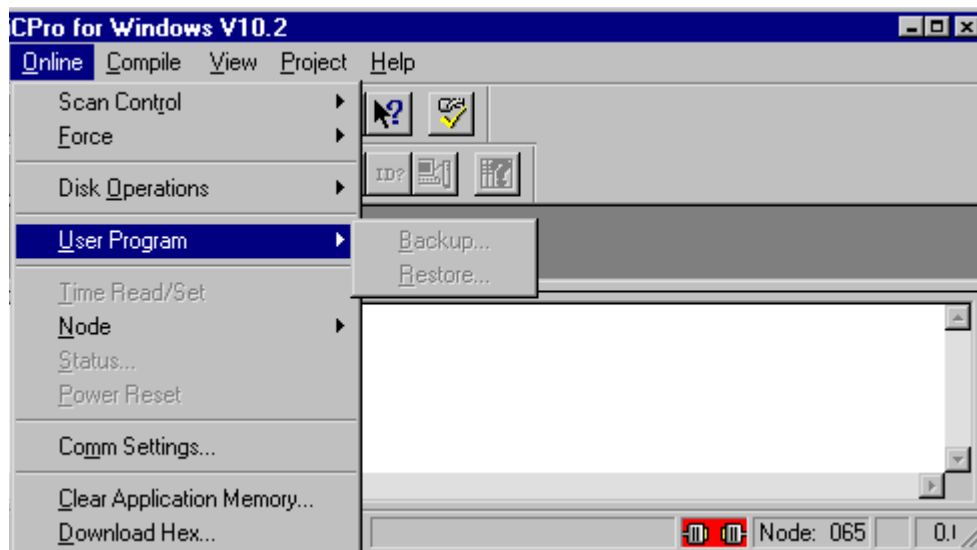
## Backing Up and Restoring User Programs

---

### Backing Up User Programs

Your program in the PiCPro 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 flyout. Backup will be bold and selectable if there is an existing communication link to your PC.



**NOTE:** The above display is the same for PiCPro for Windows and PiCPro for Windows MMC Edition.

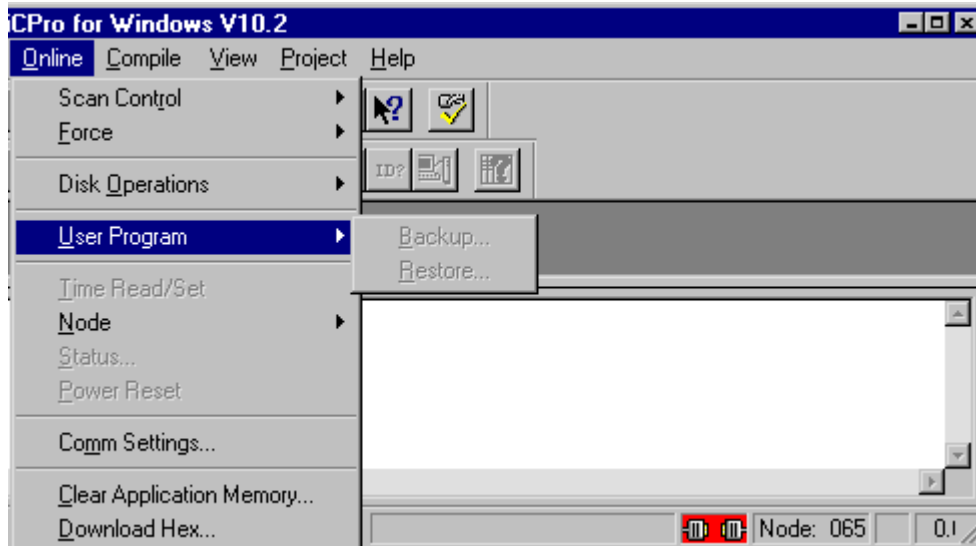
4. The Backup File to PC dialog box appears. The filename of the LDO currently in the control memory with a .bin extension will appear in the entry field.
5. Choose the location where you want to save the binary file and choose Save. A progress box will tell you the file is being saved to the PC.

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 Onlilne menu.
2. Choose User Program from the dropdown menu.
3. Select Restore from the from the flyout. 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.

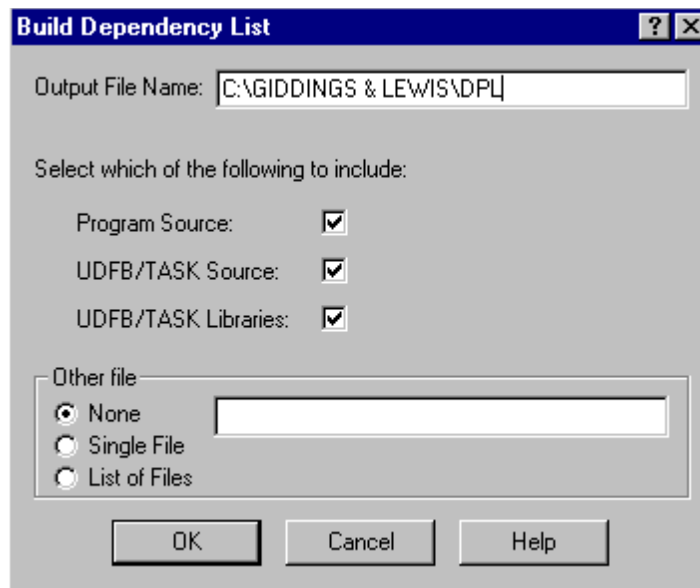
## Building a Dependency List

---

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.

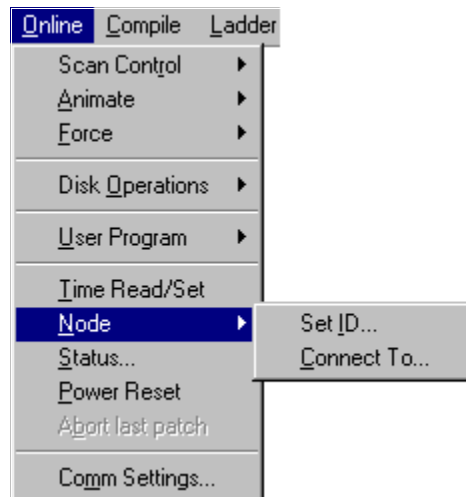


3. When you click OK, the list is generated and saved to the current folder of the active ladder. The filename is *ladder.dpl*.
4. To view your dependency list, open the .dpl file in a text editor program.

## Communications

---

Peer-to-peer communications using twisted pair wire or fiber optic cable can be set up between PiCs whose CPUs have network hardware. If you want to communicate to networked PiCs through PiCPro in order to program, monitor, force or tune, you will use the set ID command and the connect to node command. These commands are found under the On-line menu, under Node.



Each PiC on a network must be given a unique ID number identifying it as a node on your network. The range of numbers available is 1 to 255 excluding 65.


NOTE: 65 is the number used by the software to indicate the PiC that is physically connected to the PC via the serial RS232 connection to the PiCPro port. When you set your network up and if you leave one PiC physically connected to the PC, you must set the ID number on this PiC to a number other than 65 (from the range of 1 to 255) making this PiC a node also. You may, at any time, physically connect the PC to any PiC on your network and communicate to it serially by referring to it as 65.

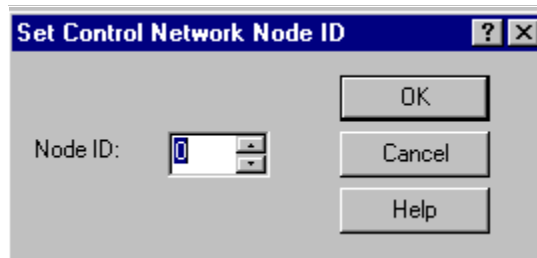


## Setting Node IDs

Initially, you must connect the PC through the RS232 programming port to each PiC you want on your network and set the node ID. The node ID will not have to be set again unless you want to change it or unless the CPU module in the PiC is changed. The scan is stopped when you set the node.

### To set the network node ID

1. Connect the PC to the PiC through the RS232 programming port and run PiCPro.
2. Choose Online, Node, Set ID from the menu or press the  button on the online toolbar.  
This box appears and you enter the ID number (from 1 to 255 excluding 65) that you want to assign to this node.




3. Press OK to accept the node ID or Cancel to ignore it.  
NOTE: Setting the node ID to zero clears the current ID and removes the PiC from the network.

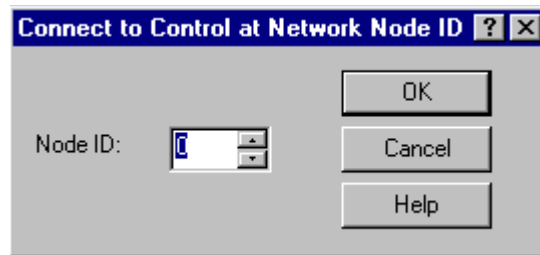
NOTE: If you are using the network function blocks in your LDO, the source ID number entered in the NETOPN function block must match the ID number assigned with this command. If it does not, an error is indicated.

## Connecting the PC to a PiC in the Network

Once a node ID has been assigned to every PiC in the network, you can connect your PC to any of the networked PiCs using the connect to node command.

### To connect your PC to a node on the network

1. Connect the PC to a PiC on the network through the RS232 programming port and run PiCPro.
2. Choose One, Node, Connect To from the menu or press the  button on the online toolbar.  
This box appears and you enter the ID number (from 1 to 255 excluding 65) of the node you want to connect to.



3. Press OK to connect to node or Cancel to ignore it.

NOTE: The node you are connected to will be listed at the bottom of the PiCPro screen in the status bar. This node number is not retentive and the display will read 065 when PiCPro is started again.

## CHAPTER 3    Servo Setup and Tuning

### Servo Setup

---

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 number. 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 follow 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 set-up data is called in a ladder, it is copied into the RAM memory of the PiC.


6. Download the application to the PiC. The setup data for your application is sent to the PiC in the form of the function. Read and write setup parameters to fine tune each axis in your application.

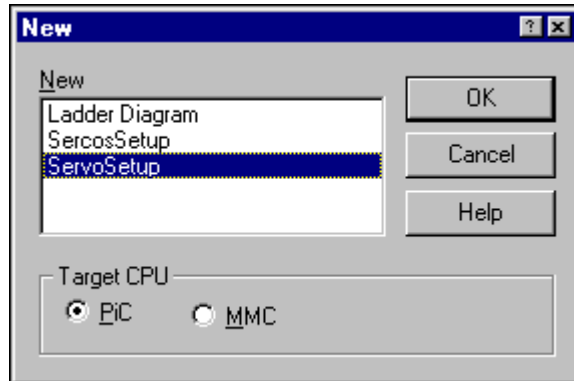
**NOTE:** This creates a .SVT file for the Servo View List and the Servo Force List.

## Opening Servo Setup

There are several ways to access the Servo Setup program.


### To create a new servo setup file for a PiC CPU

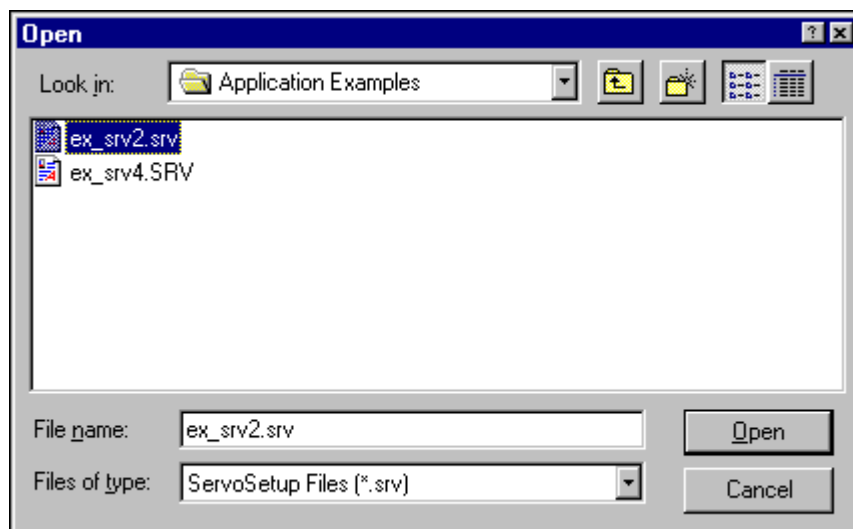
1. Choose File, New from the menu or click on the  button on the standard toolbar.
2. Select ServoSetup from the box that appears.
3. Select PiC as the Target CPU.



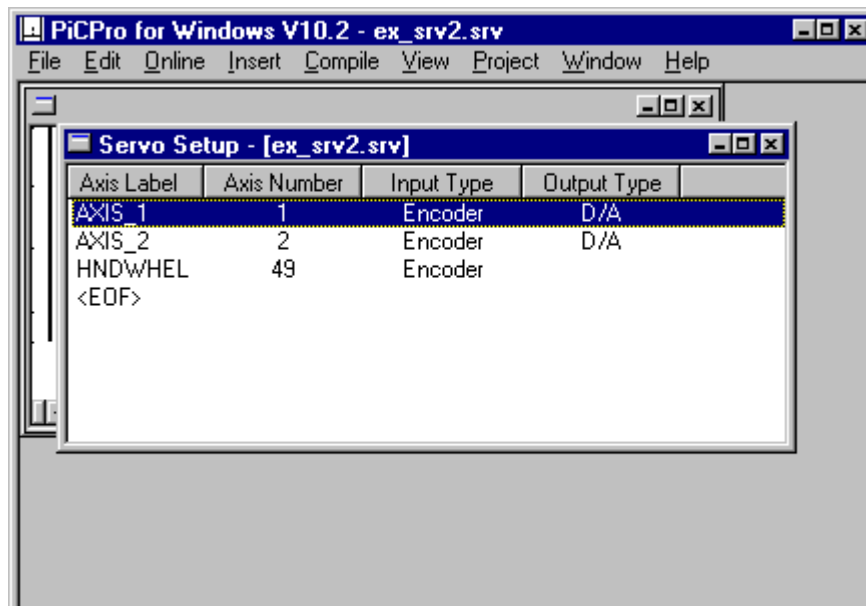
4. Choose OK.

### To open an existing servo setup file for a PiC CPU

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 Servo Setup 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



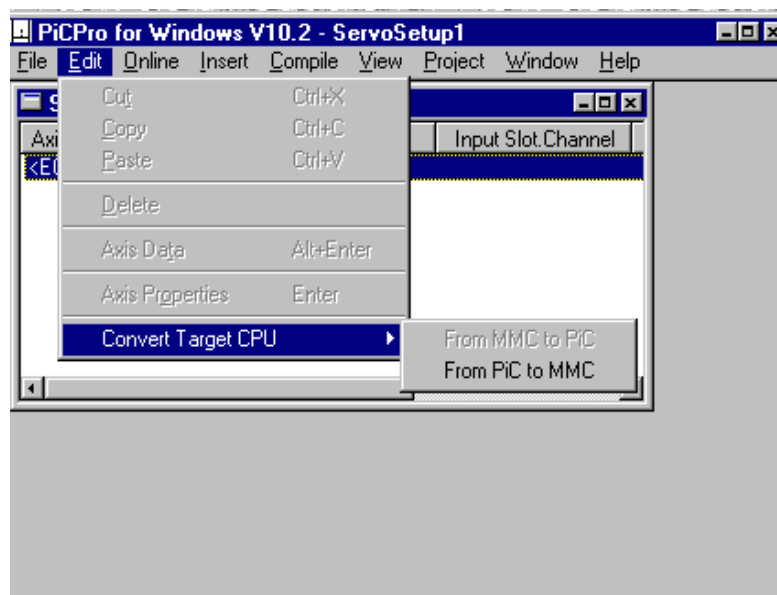
4. Click Open or press <Enter>.



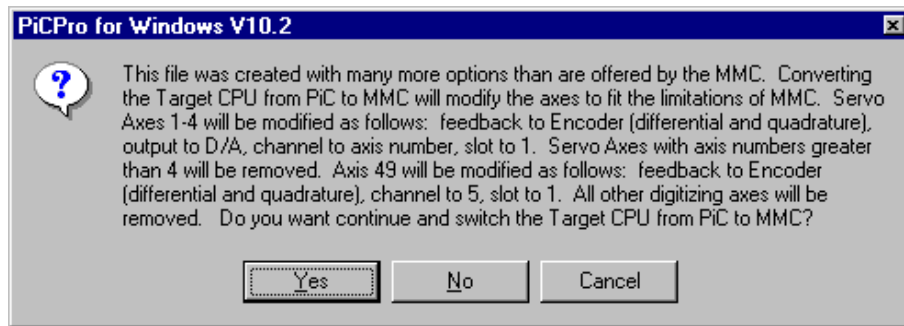
**To convert an open existing servo setup file for a PiC CPU to a servo setup for an MMC CPU**

**NOTE:** This function is not available in PiCPro for Windows MMC Edition.


1. Choose Edit / Convert Target CPU.

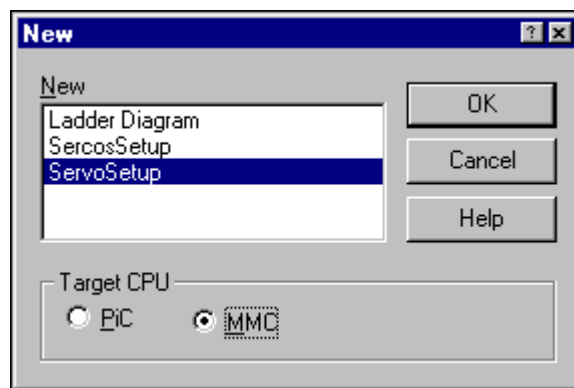


2. Click on the flyout menu From PiC to MMC. A confirmation prompt will be displayed that indicates the types of changes that will be made to the servo setup information for this file.




### To create a new servo setup file for an MMC CPU

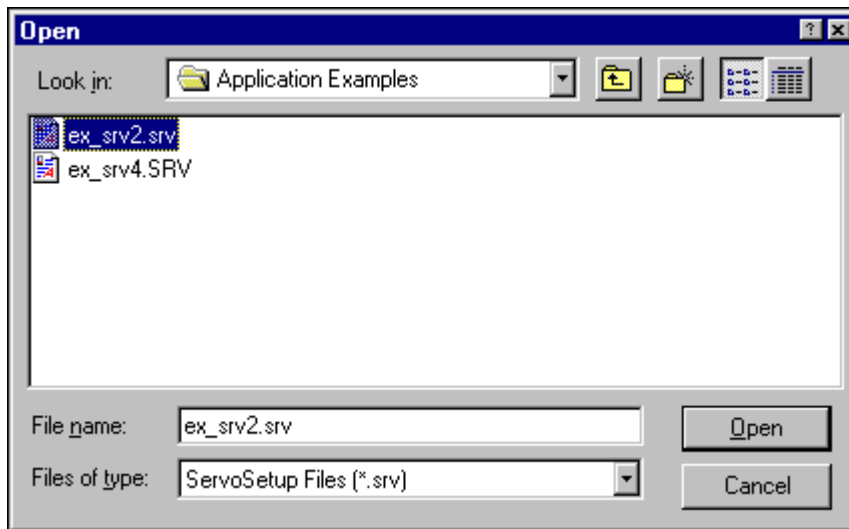
1. Choose File, New from the menu or click on the  button on the standard toolbar.
2. Select ServoSetup from the box that appears.



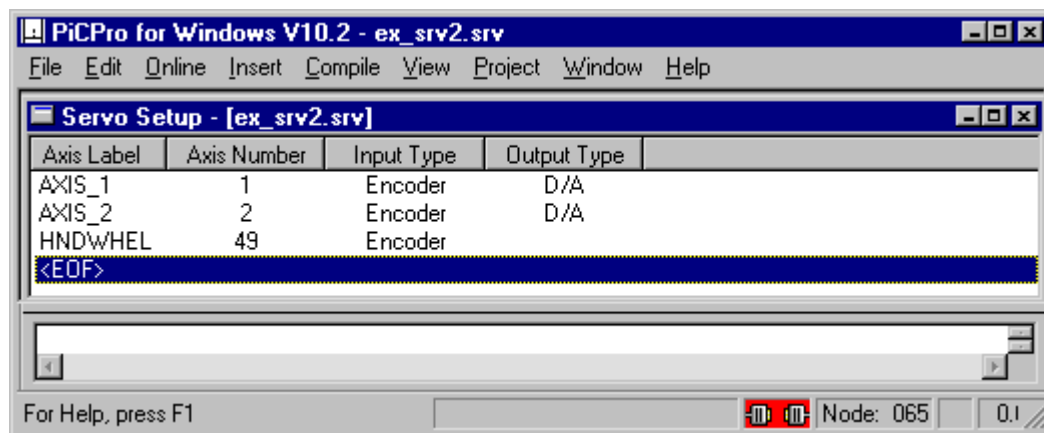
3. Select MMC as the Target CPU.
4. Choose OK.

## To open an existing servo setup file for an MMC CPU

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 Servo Setup 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.



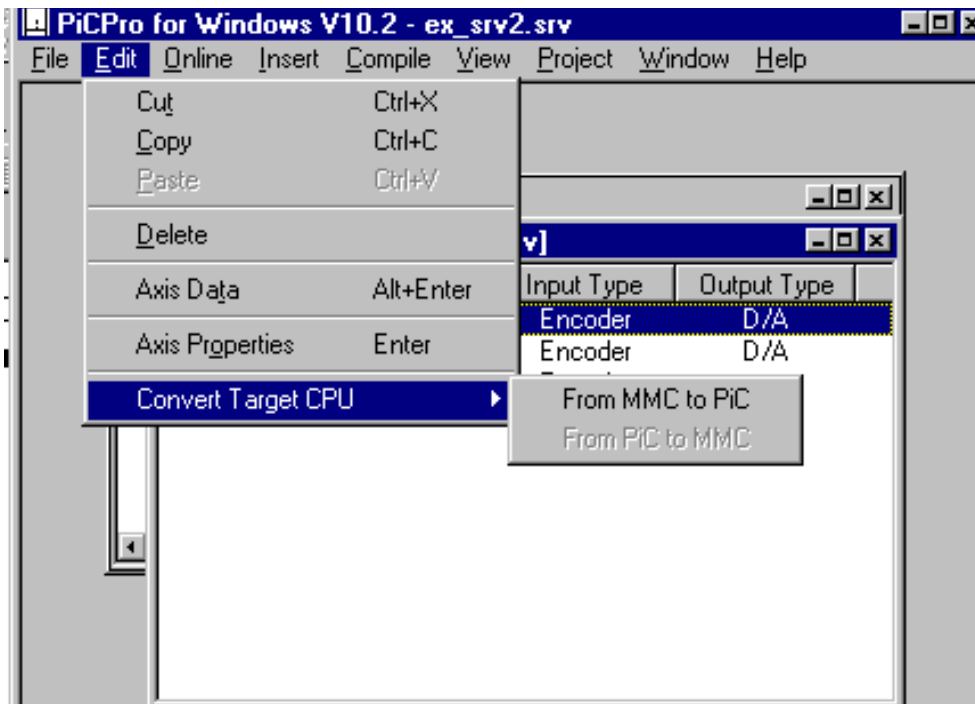
4. Click Open or press <Enter>.



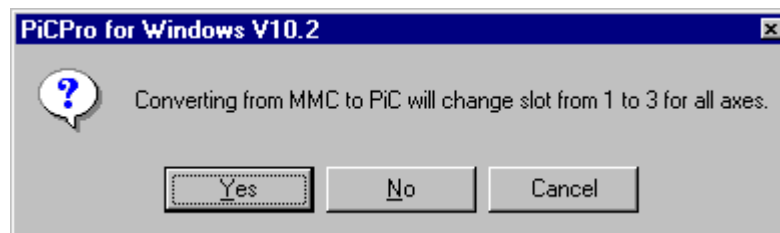
**NOTE:** When the Target CPU is MMC, the input slot channel and output slot channel columns do not appear.

To convert an open existing servo setup file from MMC CPU to a servo setup file for a PiC CPU

1. Choose Edit / Convert Target CPU.
2. Click on the flyout menu From MMC to PiC.




The following message box will appear.

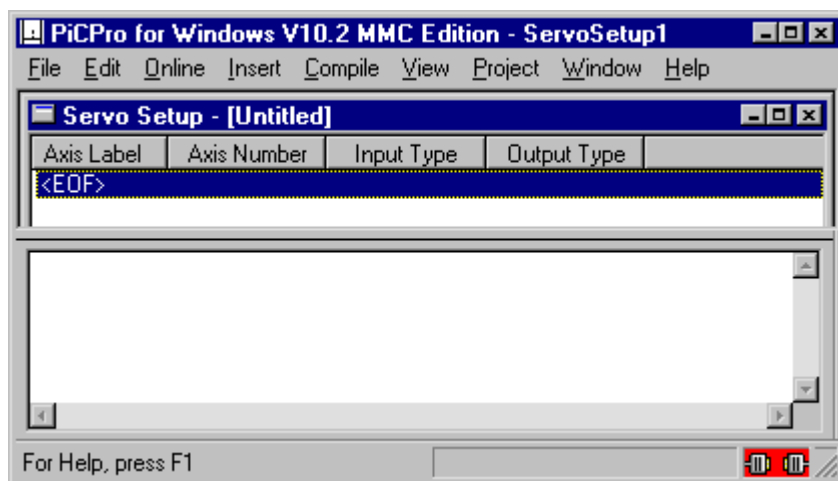
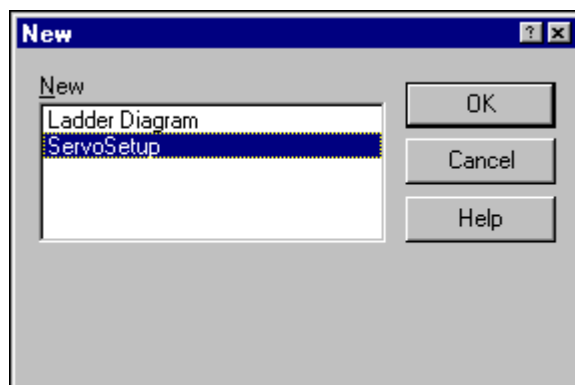





## To create a new servo setup file using PiCPro for Windows MMC Edition

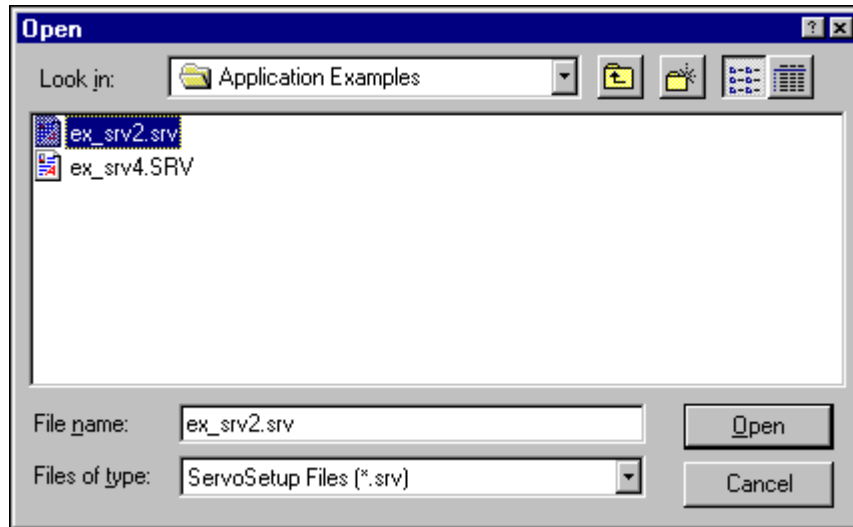
1. Choose File, New from the menu or click on the  button on the standard toolbar.
2. Select ServoSetup from the box that appears.
3. Click on OK.

**NOTE:** An empty servo setup file will be created with an MMC Target CPU.

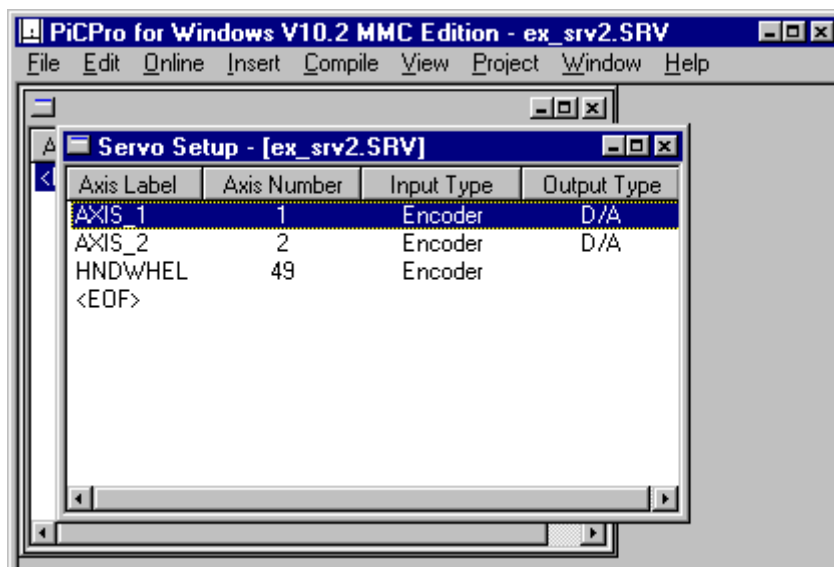


## To open an existing servo setup file using Windows for PiCPro MMC Edition

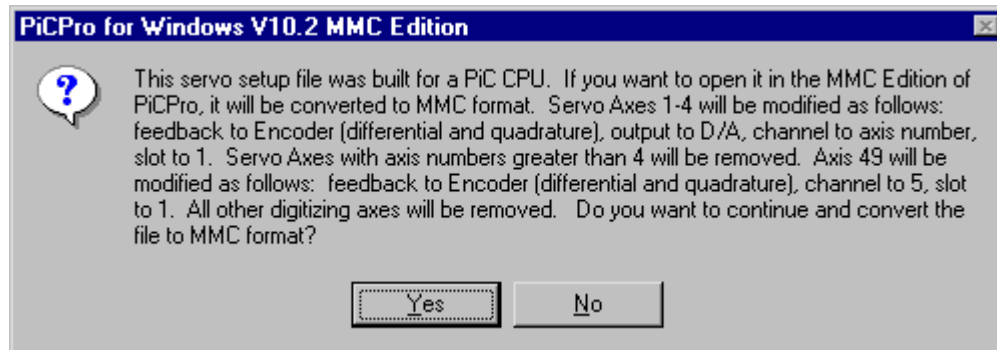
1. Choose File, Open from the menu or click on the  button on the standard toolbar.
2. The Open dialog box appears. At the bottom, choose Servo Setup Files (\*.srv) from the drop down list in the Files of Type box. This brings up a list of all existing servo setup files.
3. Highlight the file you want to open.



4. Click Open or press <Enter>.



**NOTE:** If you attempt to open in PiCPro for Windows MMC Edition, an existing SRV file that is saved for a PiC 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.



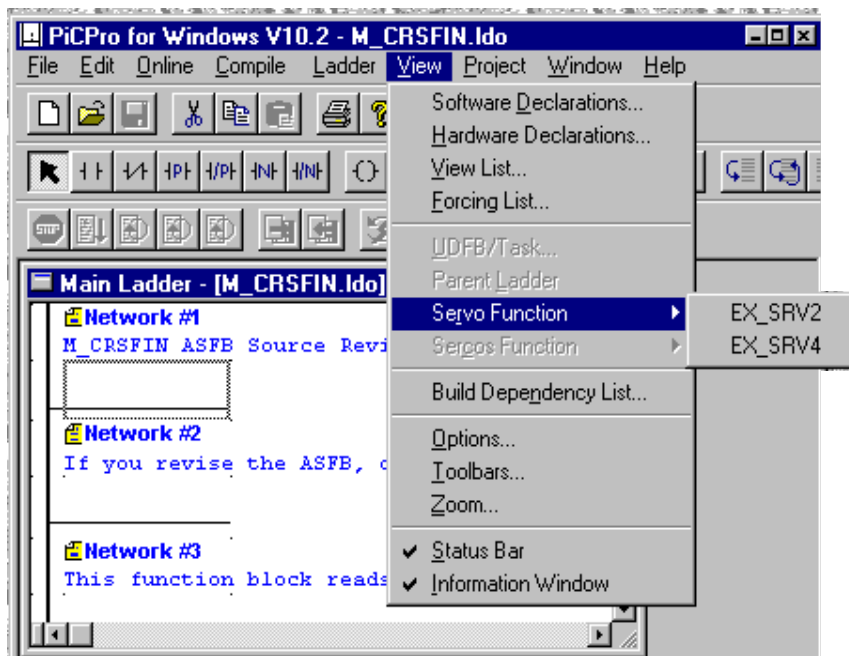
### To open an existing servo setup file from the setup function in your ladder

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 select View Servo Function

OR

Choose View, Servo Function from the menu and then select the desired setup function from the flyout list or right click the mouse and choose View Servo Function.



**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 an existing servo setup file from Windows explorer**

1. Open Windows explorer. Choose Find to locate the .srv file you want to open.
2. Double click on the .srv file you want to open and the Servo Setup window will appear.

#### **Copying Axis Data**

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 data.

#### **How to use the copy/paste commands**

1. Enter an axis and all its Axis Data in servo setup.
2. Click OK to accept and exit the Servo Axis Data box.
3. Select the axis you want to copy. Choose the copy command from the Edit menu or press <Ctrl + C> or right click and choose Copy.
4. Place the focus on the EOF line and choose the paste command from the Edit menu or press <Ctrl + V> or right click and choose Paste. You will be prompted to enter a new label for the pasted axis since you cannot have duplicate labels. The axis will be entered in numerical sequence.

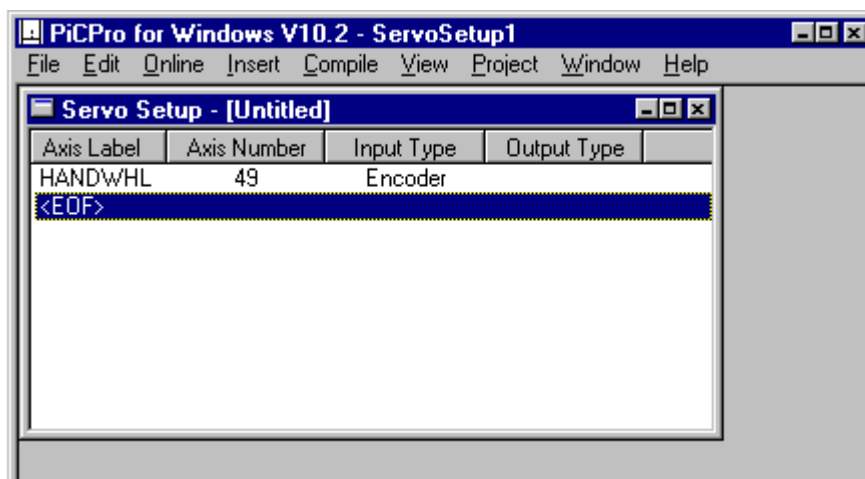
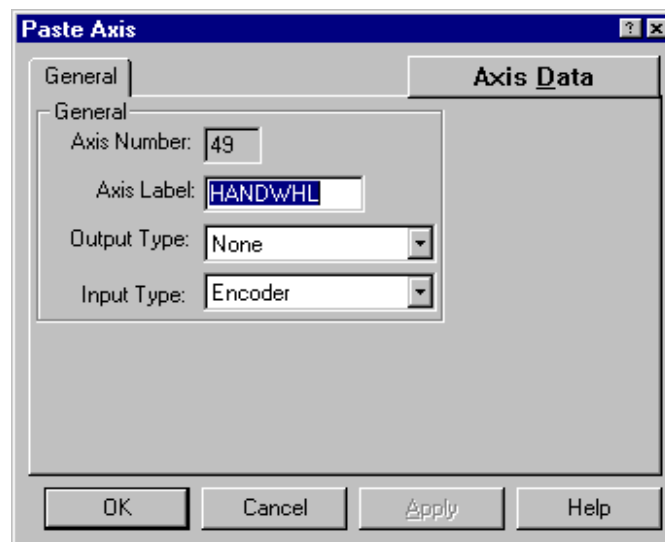
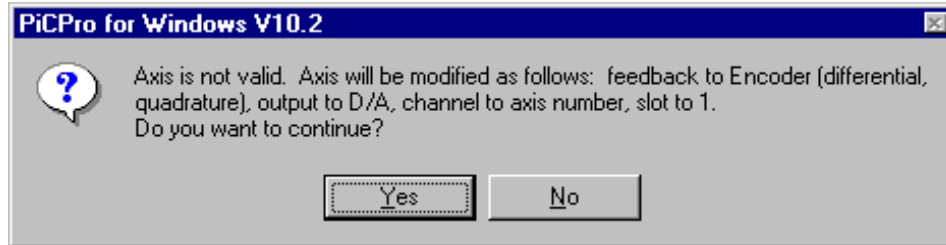
**NOTE:** If some of the axis parameters for the axis need to be changed, you can edit them after you have pasted the axis in the list.

## How to copy axis data from one axis to another

From the servo setup screen you can also copy the axis 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 change them after you have copied the axis data to the second axis.

1. Insert the axes into servo setup.
2. Select one axis and enter the Axis Data for that axis if you did not do so when you entered axis properties.
3. Click the <Apply> button.
4. Click OK to close the dialog box and return to the servo setup screen.
5. With the focus still on the axis you just entered axis data for, choose the copy command from the Edit menu or press <Ctrl + C> or right click and choose Copy.
6. Highlight the axis you want to copy the axis data to and choose the paste command from the Edit menu or press <Ctrl + V> or right click and choose Paste.
7. If you need to change some of the parameters for the second axis, choose Axis Data from the edit menu or press Alt + Enter to bring up the axis data box.
8. Choose Apply before leaving the axis data box.

**NOTE:** When you paste an axis from a servo setup list view for a PiC CPU into a servo list view for an MMC CPU or from a servo setup list view for an MMC CPU into a servo list view for a PiC CPU, a message box will appear and ask if the slot/channel and input/output types (if not Encoder/D/A) should be changed.



## 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 listed in numerical order. 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 Edit, Axis Properties from the menu or press the <Enter> key.
- With the axis selected, right click and choose Axis Properties.
- 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 Edit, Axis Data from the menu or press the <Alt + Enter> key.
- With the axis selected, right click and choose Axis Data.

Make the necessary changes to Scaling, Iterator, or Position Loop Data and choose OK.

## Saving a Servo Setup File

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

Using the Save command, you can save the file under its existing name.

Using the Save As 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 File, Save.
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.
4. Click Save.

### To save an existing file

- Choose File, Save.

You can use the Save As 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.

### **To save a file under a different name or in a different location**

1. Choose File, Save As from the menu.
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 Save.

### **To save a Servo Setup File in a format that can be read by earlier versions of PiCPro for Windows (prior to V10.2)**

SRV files that are created or saved in PiCPro for Windows V10.2 cannot be opened directly by earlier versions of PiCPro for Windows. To save an SRV file for use with an earlier version of PiCPro for Windows do the following:

1. Choose File, Save As from the menu.
2. At the “Save as type:” dropdown list select “Servosetup - Prior to PiCPro V10.2 (.srv)”.
3. Click Save.

### **Printing Axis Data**

You can print the axis data (scaling, iterator, and position loop information) for one or all axes in your servo setup program.

#### **To print axis data**

1. Open the Servo Setup file (.srv) that you want to print. If you just want to print the axis data for one axis, select that axis.
2. Choose File, Print 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 All 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 PiC 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.

#### **To create a servo setup function**

1. After all the setup information for all your axes has been entered, choose Compile, Make Function from the menu.



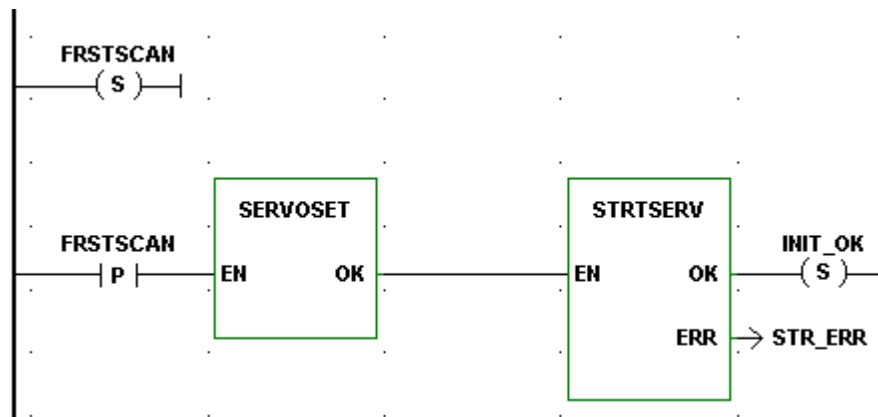
2. The Save As box will appear if you have not previously saved your setup file. Choose the location you want to save your .SRV file to. There is a default that is entered in the File Name box beginning with ServoSetup1.SRV. You can accept this name or enter your own.

**NOTE:** The second setup file you create will have the default name ServoSetup2.SRV, the third will be ServoSetup3.SRV and so on.

3. The Compile SRV box appears. The Registered Servo Setup 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 Servo Setup Library Name: box.
4. 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. 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.

### 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 STRSERV with your setup function. Below is a network designed to do this. The example setup function is labeled SERVOSET.



## Inserting an Axis in Servo Setup

An axis is added to Servo Setup by using the Insert command from the menu. You can insert either a servo or digitizing (read-only) axis. Digitizing axes have no output.

Servo setup will number the axes sequentially as they are inserted. Axis numbers may not be changed.

Each time you insert an axis you must enter the axis properties for that axis.

## **Servo Axis Data**

### **PiC CPU**

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.

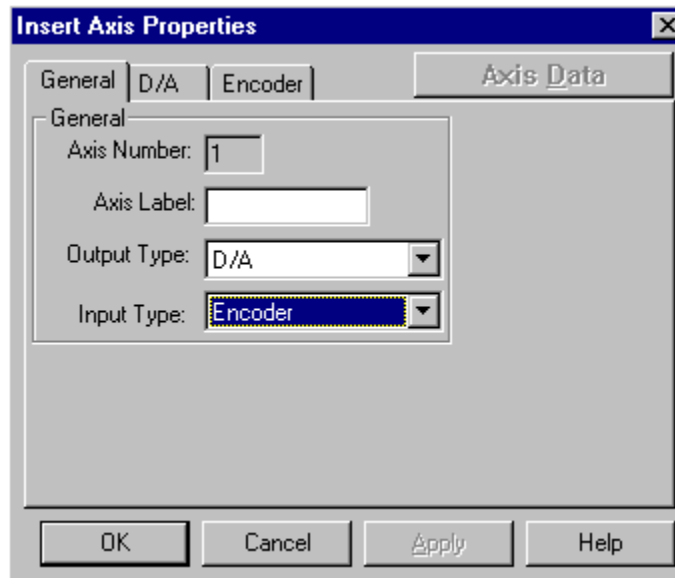
### **MMC CPU**

The maximum number of servo axes is 4. The servo axes are numbered from 1 to 4.

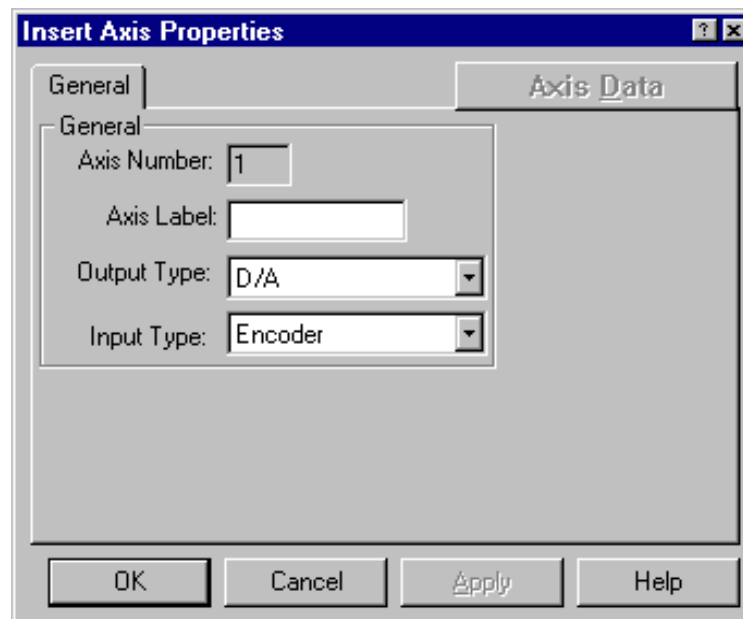
The maximum number of digitizing axes is 1. The digitizing axes is numbered 49.

## To insert a servo axis

1. Choose Insert, Servo. The Insert Axis Properties box for a servo axis appears.  
The Axis Number is entered by PiCPro in sequential order. You cannot change this number.



**NOTE:** The window that appears in PiCPro for Windows with an MMC Target CPU chosen and PiCPro for Windows MMC Edition does not contain the D/A and Encoder tabs and appears as follows:



2. Enter a name for the axis in the Axis Label box.
3. Enter the Output 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. If you need to change the defaults, change the Output Type first since the Input Type choices vary based on your output selection.
4. 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.

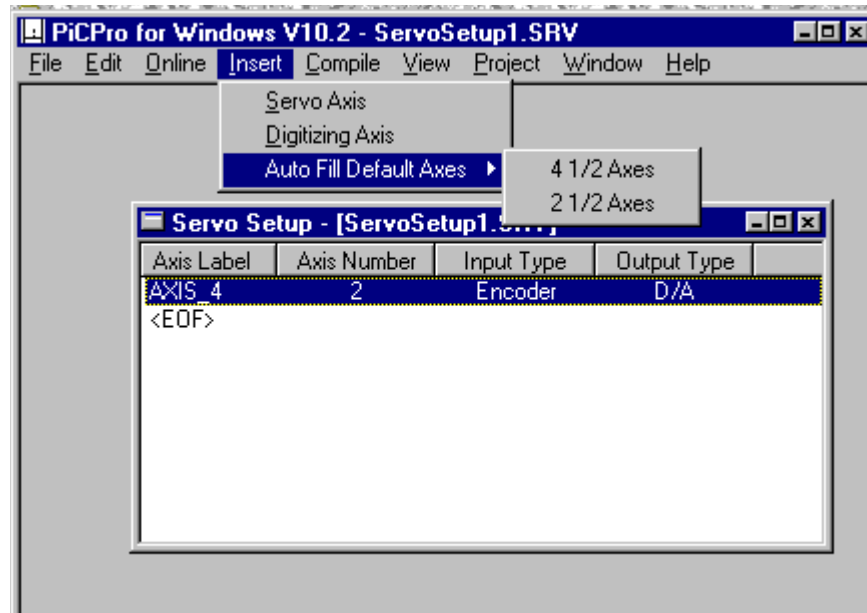
The various combinations of Input and Output types are listed below.

CPU Type	For a Servo Axis		For a Digitizing Axis		For a Stepper Axis		For a SERCOS Axis	
	Inputs	Output	Inputs	Output	Input	Output	Input	Output
PiC CPU	Encoder Resolver Analog TTL	D/A	Encoder Resolver Analog TTL	No Output	No Input	Stepper	SERCOS	SERCOS
MMC CPU	Encoder	D/A	Encoder	No Output	Not Applicable	Not Applicable	Not Applicable	Not Applicable

## To insert default axis using Auto Fill

Auto Fill is only available in PiCPro when MMC is the Target CPU.

1. Choose Insert, Auto Fill Default Axis. Flyout menus for 4 1/2 Axes and 2 1/2 axes will appear.
2. Click on the appropriate flyout menu.

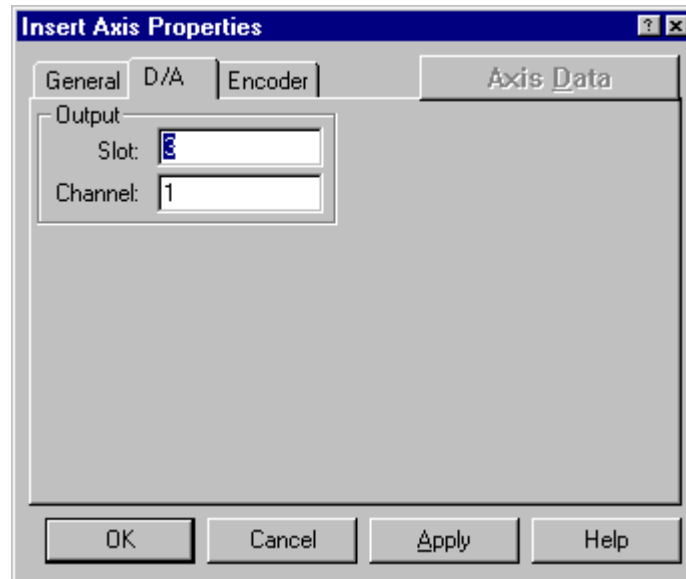


If axes are present when one of these menu items is selected, a message will be displayed to indicate that all axes must be removed before default axes can be inserted. If the axis count is 0, 4 or 2 servo axes will be added and 1 digitizing axis will be added. The servo axis will have an input type of Encoder and an output type of D/A. The digitizing axis will have an input of type of encoder. The servo axes will be named AXIS1-4. The digitizing axis will be named AXIS49.

## D/A Output Setup

### To configure the D/A output using PiCPro with PiC Target CPU

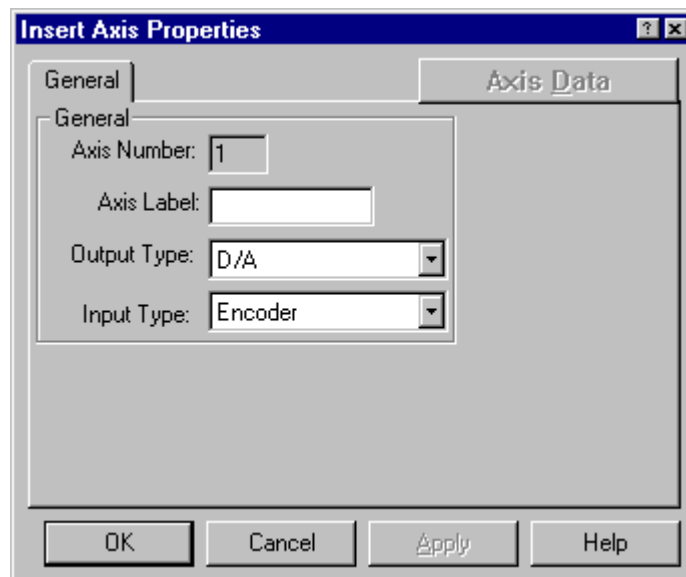
1. Enter the slot and channel location of the D/A module.
2. Click OK.



The dialog box titled "Insert Axis Properties" has four tabs: "General", "D/A", "Encoder", and "Axis Data". The "D/A" tab is selected. Under the "Output" section, there are two input fields: "Slot:" with the value "3" and "Channel:" with the value "1". At the bottom are four buttons: "OK", "Cancel", "Apply", and "Help".

### To configure the D/A output using PiCPro with MMC Target CPU

The channel in this window is represented by the Axis Number shown. The slot entry is not necessary when using MMC.



The dialog box titled "Insert Axis Properties" has two tabs: "General" and "Axis Data". The "General" tab is selected. Under the "General" section, there are four input fields: "Axis Number:" with the value "1", "Axis Label:" (empty), "Output Type:" with a dropdown menu showing "D/A", and "Input Type:" with a dropdown menu showing "Encoder". At the bottom are four buttons: "OK", "Cancel", "Apply", and "Help".

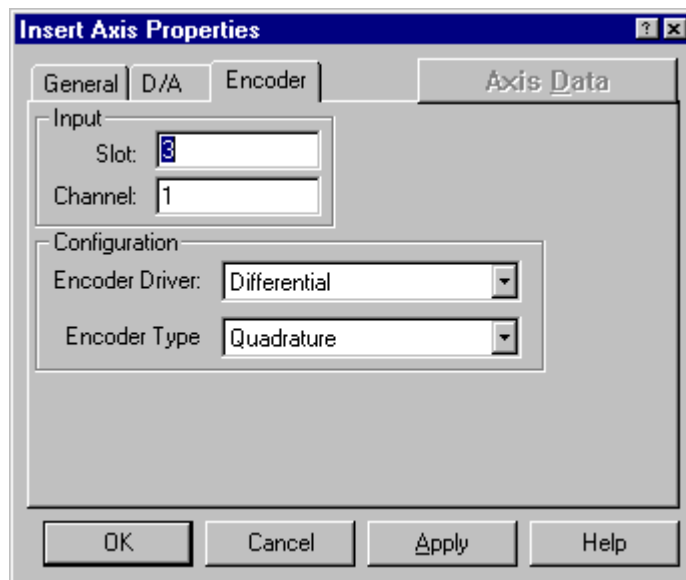
## Encoder Input Setup

### To configure the encoder input using PiCPro with PiC Target CPU

1. Enter the slot and channel location of the encoder module.
2. Select the correct encoder driver, differential or single-ended, from the drop down list.
3. Select the correct encoder type, quadrature or pulse, from the drop down list.

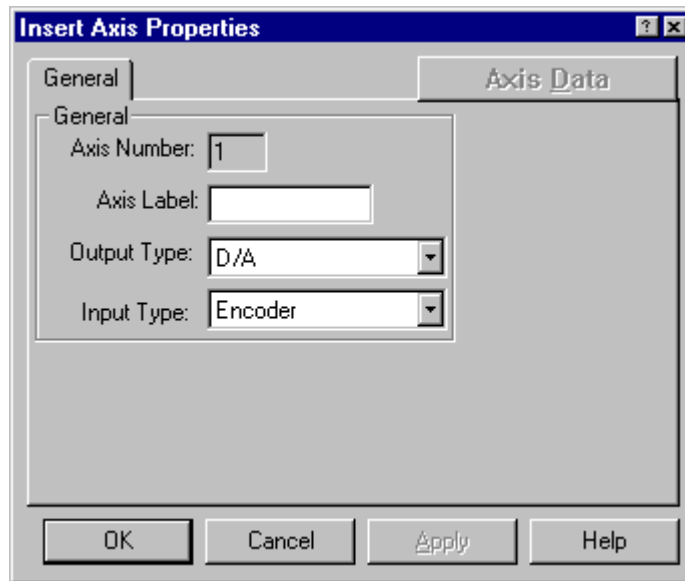
**NOTE:** Quadrature type assumes that there are four counts for each quadrature cycle. The rising and falling edges of both channel A and channel B are counted. Pulse type counts either channel A or channel B so that one count is recorded per cycle.

4. Click OK.



## To configure the encoder input using PiCPro with MMC Target CPU

The channel in this window is represented by the Axis Number shown. The slot entry is not necessary when using MMC.



The encoder is automatically configured to the following when the Target CPU is PiC:

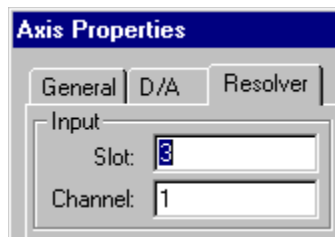
Driver: Differential

Type: Quadrature

## 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.

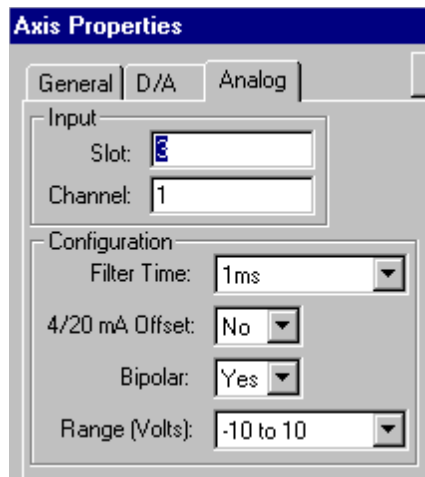
## Analog Input Setup (Only for PiC CPU)

### 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 mode, select Yes from the drop down list. NOTE: The bipolar box will be changed to No and the unipolar input range of 0 to 5 is inserted in the software. 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 mode, select the bipolar or unipolar voltage range from the drop down list.
5. Click OK.

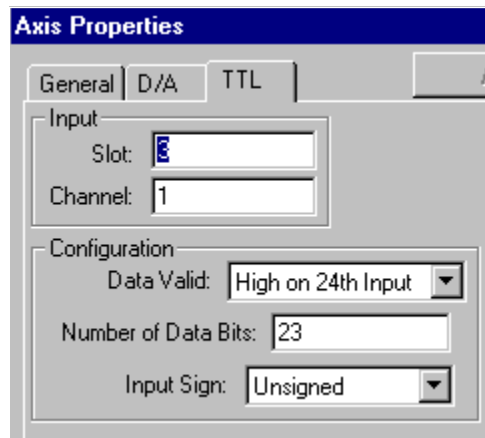


## TTL Input Setup (Only for PiC CPU)

### 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 24<sup>th</sup> input is not used.  
A maximum of 23 bits can be used if High or Low is selected with the 24<sup>th</sup> 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 <sup>th</sup> input is not used.
Gray Code	Alternative to binary encoding. Only one bit of a gray code number changes at a time. The 24 <sup>th</sup> input is not used.

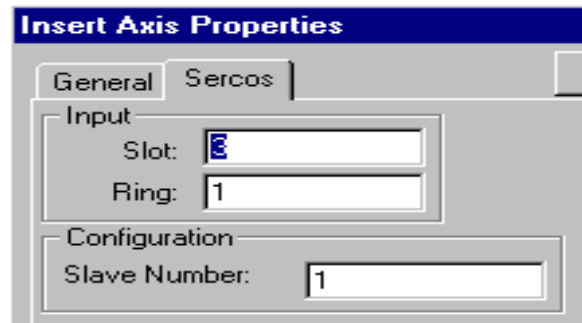
## SERCOS Setup (Only for PiC CPU)

When a SERCOS slave is connected to a servo axis, SERCOS is chosen as the feedback module. NOTE: Only a servo loop axis may be connected to a SERCOS slave.

### To configure a SERCOS axis

1. Enter the slot and ring numbers for the SERCOS axis. The slot number identifies which slot the SERCOS module is installed in the PiC rack. 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 slaves on a ring must be numbered sequentially. For example, if you have three slave axes on a ring, number them 1, 2, 3: not 1, 2, 4. The number entered here must match the address switch set on 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.

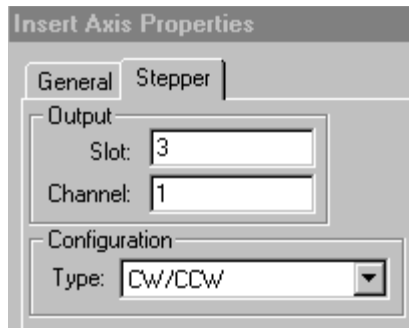


The screenshot shows a software window titled "Insert Axis Properties" with a blue header bar. Below the header are two tabs: "General" and "Sercos". The "Sercos" tab is selected. Inside the "Sercos" tab, there are two sections. The first section is labeled "Input" and contains two text boxes: "Slot:" with the value "E" and "Ring:" with the value "1". The second section is labeled "Configuration" and contains one text box: "Slave Number:" with the value "1".

## 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.

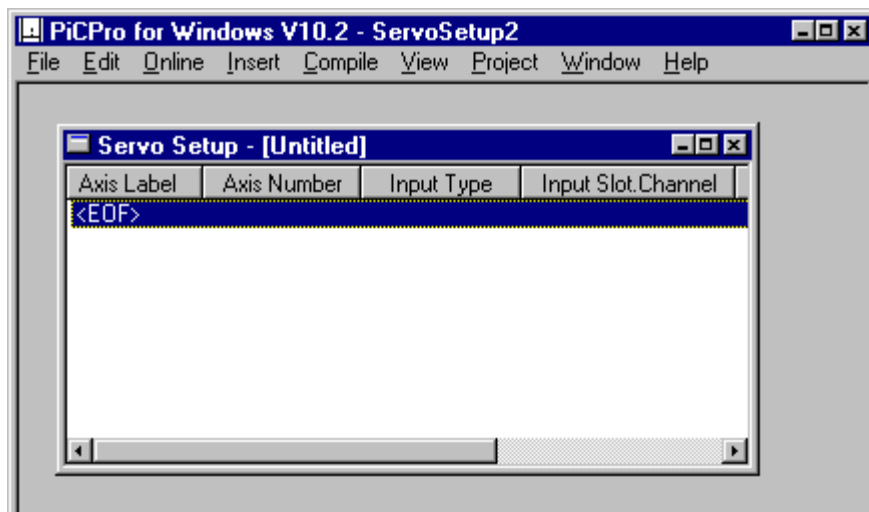


## Entering servo setup data

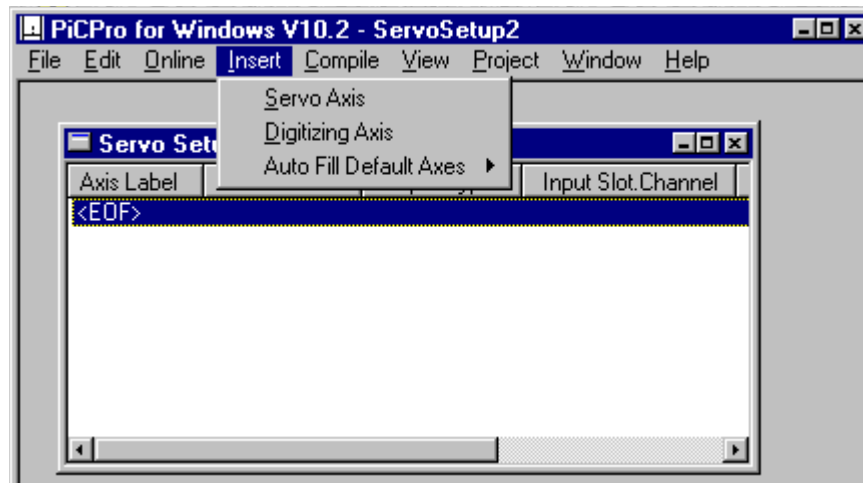
---

### Entering servo setup data for PiC CPU

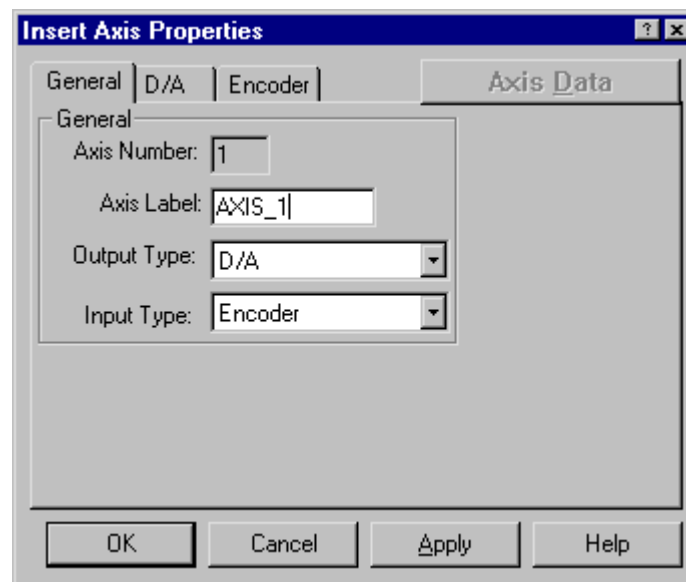
1. Bring up the Servo Setup window.



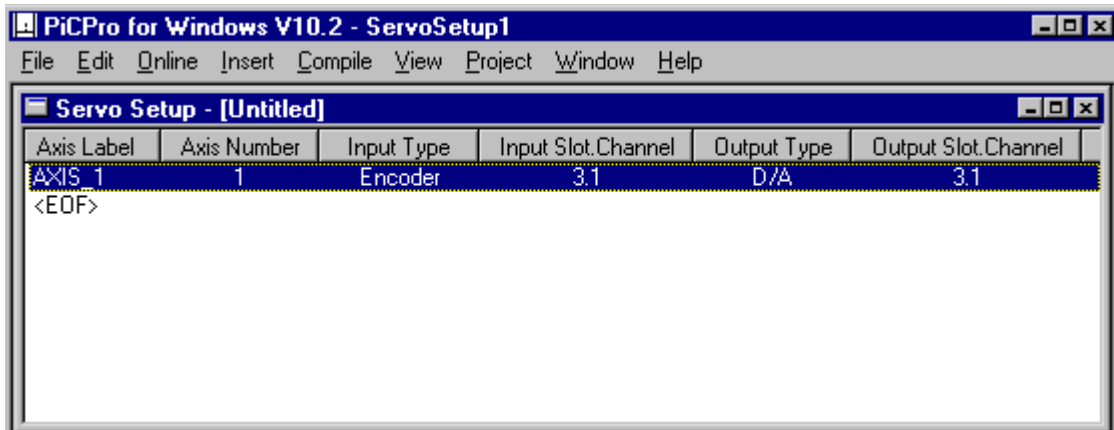
2. Click on Insert and click on Servo Axis.



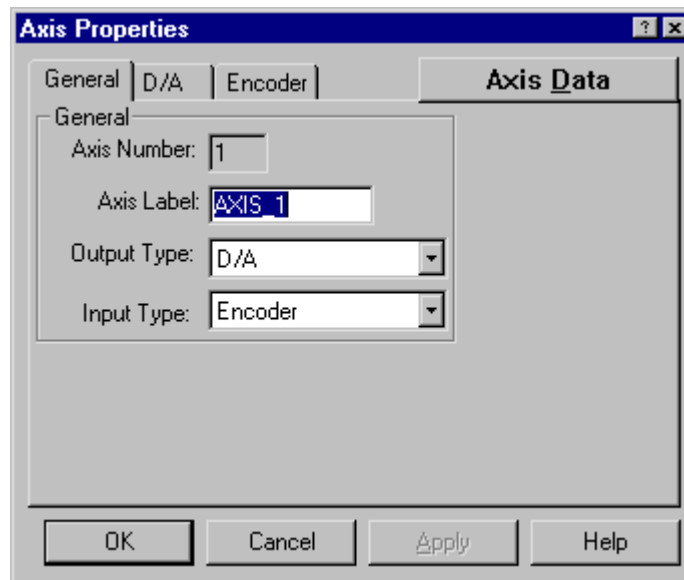
3. At Axis Label, insert a label if one is not already assigned.



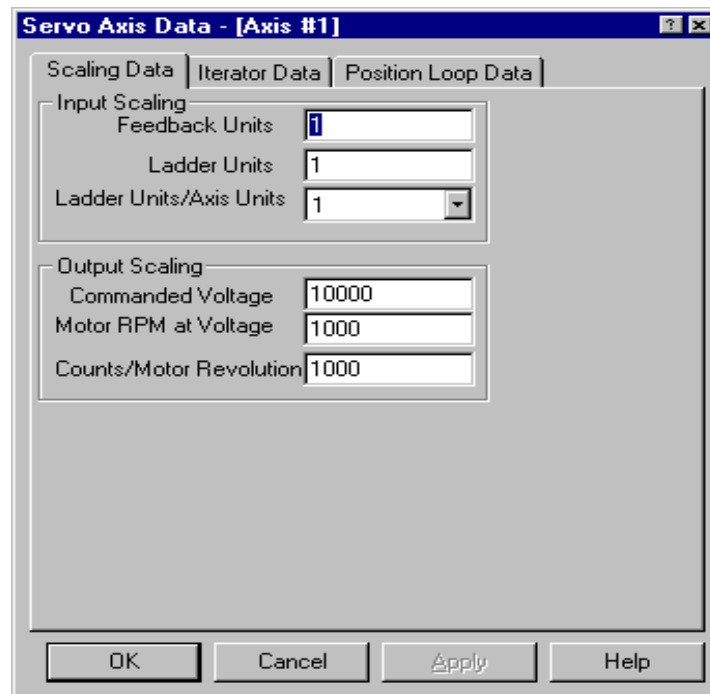
4. Click on OK.
5. Double click on the highlighted axis listing in the Servo Setup window.



6. Click on Axis Data.



7. The Servo Axis Data window will appear.



The image shows a dialog box titled "Servo Axis Data - [Axis #1]". It has three tabs: "Scaling Data", "Iterator Data", and "Position Loop Data". The "Scaling Data" tab is selected. It contains two sections: "Input Scaling" and "Output Scaling".

**Input Scaling:**

- Feedback Units: 1
- Ladder Units: 1
- Ladder Units/Axis Units: 1 (dropdown menu)

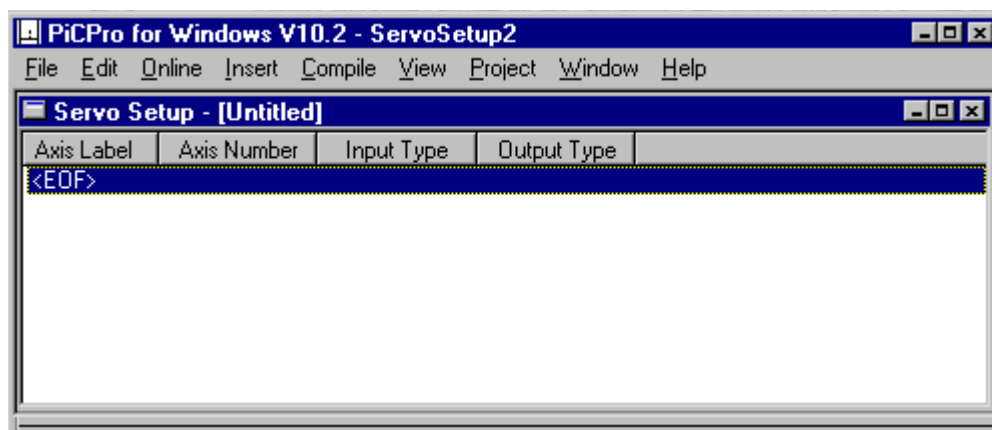
**Output Scaling:**

- Commanded Voltage: 10000
- Motor RPM at Voltage: 1000
- Counts/Motor Revolution: 1000

At the bottom are four buttons: "OK", "Cancel", "Apply", and "Help".

## Entering servo setup data for MMC CPU

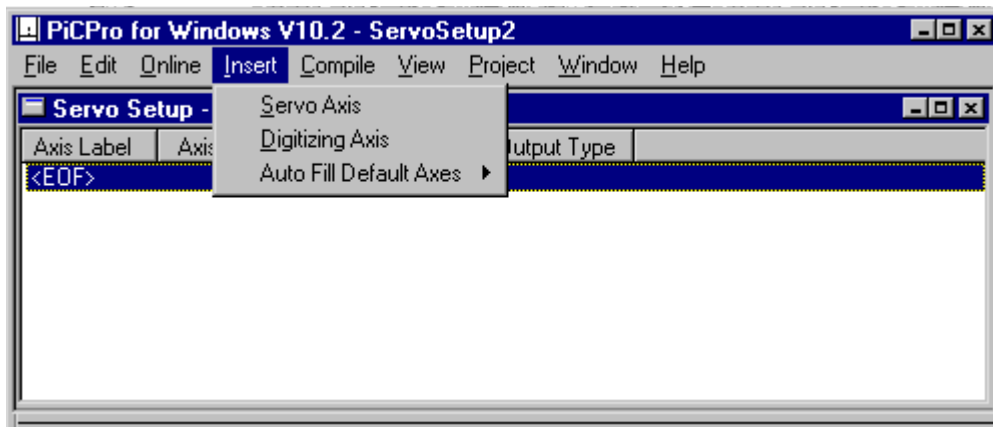
1. Bring up the Servo Setup window.



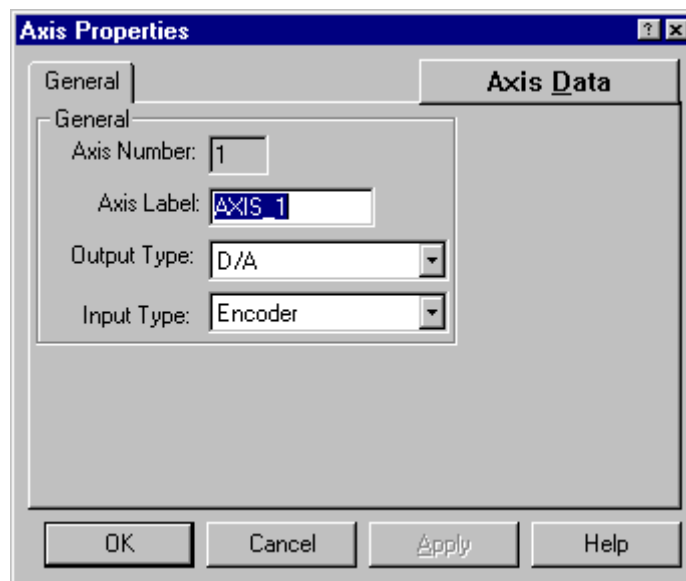
The image shows a window titled "PiCPro for Windows V10.2 - ServoSetup2". It has a menu bar with "File", "Edit", "Online", "Insert", "Compile", "View", "Project", "Window", and "Help". Below the menu bar is a tabbed interface with a tab titled "Servo Setup - [Untitled]".

Inside the "Servo Setup" tab, there is a table with the following columns: "Axis Label", "Axis Number", "Input Type", and "Output Type". The first row of the table contains the text "<EOF>".

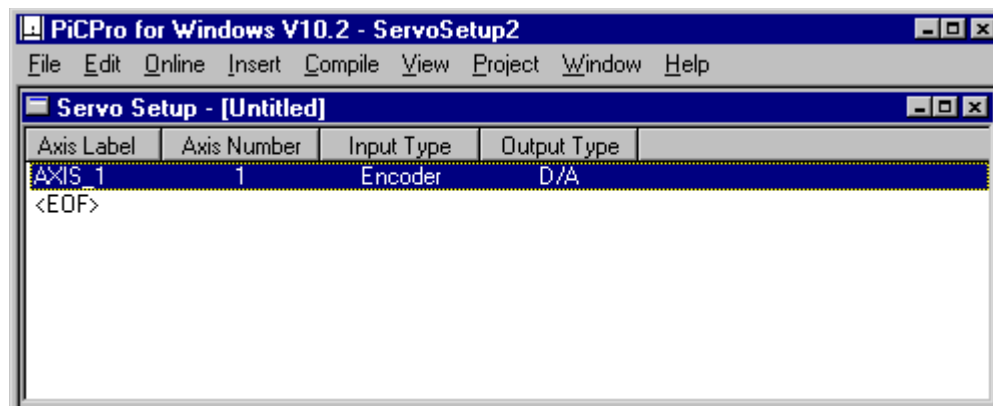
2. Click on Insert and click on Servo Axis.



3. At Axis Label, insert a label if one is not already assigned.

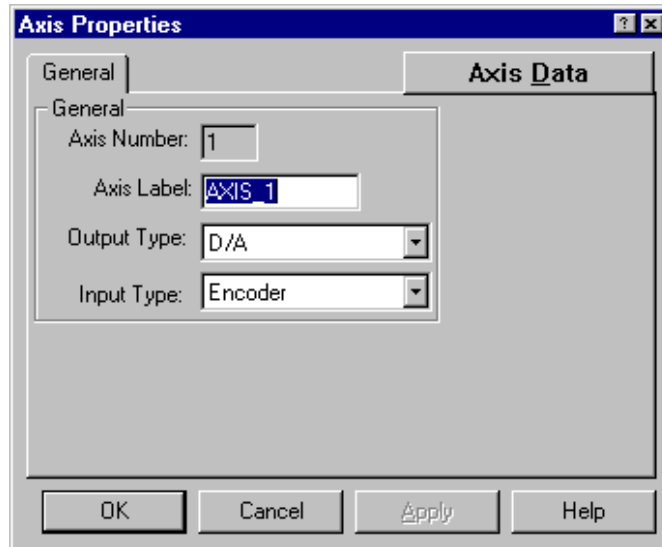


4. Click on OK.
5. Double click on the highlighted axis listing in the Servo Setup window.

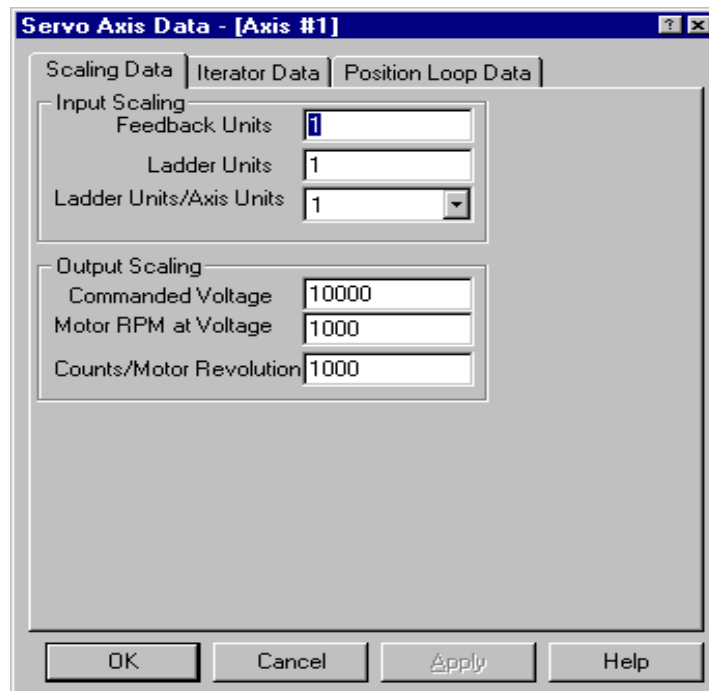




- Click on Axis Data.



- The Servo Axis Data window will appear.



## Servo Axis Setup Data Categories for PiC and MMC CPU

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.

The screenshot shows a Windows-style dialog box titled "Servo Axis Data - [Axis #1]". It has three tabs: "Scaling Data", "Iterator Data", and "Position Loop Data". The "Scaling Data" tab is selected. Inside this tab, there are two main sections: "Input Scaling" and "Output Scaling".

**Input Scaling:**

- Feedback Units: A text box containing the value "1".
- Ladder Units: A text box containing the value "1".
- Ladder Units/Axis Units: A dropdown menu showing "1".

**Output Scaling:**

- Commanded Voltage: A text box containing the value "10000".
- Motor RPM at Voltage: A text box containing the value "1000".
- Counts/Motor Revolution: A text box containing the value "1000".

At the bottom of the dialog, there are four buttons: "OK", "Cancel", "Apply", and "Help".

## Entering Scaling Data

The 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.

### 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.

The ratio of the feedback device signal to the PiC feedback units is:

One pulse of an encoder = one feedback unit

NOTE: One cycle of a quadrature type encoder equals four pulses.

NOTE: You cannot enter fractional values here.

**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 a 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: These parameters are used for scaling servo calculations by the servo software. They do not represent limits.

## 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 Servo Force List.
2. Add the Feed Forward Percent parameter to the list and enter 100% for the value.
3. Move the axis at a fixed velocity in either always the positive or always the negative direction.
4. Observe the following error in the Servo View List. If the following error is not zero, the velocity scaling is incorrect.
5. 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.
6. Make the servo setup function and download the ladder to the PiC.
7. Again, observe the following error in the Servo View List.
8. Repeat the above steps until the following error is close to zero.
9. Finally, remake the servo setup function and perform a complete download of the ladder.

## Entering Iterator Data

**Servo Axis Data - [Axis #1]**

Scaling Data | **Iterator Data** | Position Loop Data

Iterator Data

Velocity Limit:

Acceleration Ramp:

Deceleration Ramp:

Controlled Stop Ramp:

Slow Velocity Filter:

Fast Velocity Filter:

Slow/Fast Velocity Threshold:

Rollover on Position?:

Rollover Position:

Software Limit (Upper):

Software Limit (Lower):

Ignore Limits until Referenced?:

OK Cancel Apply Help

The following parameters comprise the iterator data for an axis.

**Velocity Limit** 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.

### WARNING

*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.

**Acceleration Ramp** 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** 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	<p>Enter the rate in axis units/minute/second at which the servo software will iterate the axis to reach a controlled stop.</p> <p>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.</p>
Slow Velocity Filter	<p>Enter in milliseconds the filter 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.</p> <p>Range = 0 - 10,000 ms</p>
Fast Velocity Filter	<p>Enter the milliseconds the filter 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.</p> <p>Range = 0 - 10,000 ms</p>
Slow/Fast Velocity Threshold	<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 on Position	<p>This parameter acts as a reference reset.</p> <p>Enter Yes if you want the axis position to roll over to zero when the Rollover Position entered on the next line is reached.</p> <p>Enter No if you do not want the axis position to roll over. The commanded position will then accumulate.</p>
Rollover Position	<p>Enter the rollover position in axis units.</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 Upper Limit	<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 Lower Limit	<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>
Ignore Limits Until Referenced?	<p>A Yes 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.</p> <p>After the machine reference is complete, the software limits are in effect.</p>

## Entering Position Loop Data

The screenshot shows a software window titled "Servo Axis Data - [Axis #1]". It has three tabs: "Scaling Data", "Iterator Data", and "Position Loop Data". The "Position Loop Data" tab is active. Inside this tab, there are several settings:

- Input Polarity:** A dropdown menu set to "Positive".
- Output Polarity:** A dropdown menu set to "Positive".
- Analog Output Offset:** A text box containing "0".
- Feed Forward Percent:** A text box containing "0".
- Proportional Gain:** A text box containing "100".
- Integral Gain:** A text box containing "0".
- (+) Integral Error Limit:** A text box containing "0".
- (-) Integral Error Limit:** A text box containing "0".
- Derivative Gain:** A text box containing "0".
- Following Error Limit:** A text box containing "0".
- Update Rate:** A dropdown menu set to "4ms".
- In Position Band:** A text box containing "0".

At the bottom of the dialog are four buttons: "OK", "Cancel", "Apply", and "Help".

Position loop data provides the servo software with information on how the axis is set up for the position loop.

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	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	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	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	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	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	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	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.



**Update Rate** Enter in milliseconds, (1, 2, 4, or 8) how often the servo upgrade 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.

The iteration rate is eight times the servo upgrade rate:

SUG (ms)	Iteration Rate (ms)
.25	2
.5	4
1	8
2	16
4	32
8	64

For digitizing axes, the update rates are .25, .5, 1, 2, 4, 8, or 16 ms.

A rate can be selected for each axis.

NOTE: In master/slave moves, the iteration rate is the same as the servo upgrade rate. The update rate for the master axis and the slave axis must be the same in any master/slave move.

**In-Position Band** 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.

## 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 PiC, 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. You can also select the Update button in the Servo Force List box.

**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 PiC using the Servo Force List.

#### To force an axis servo variable in the Servo Force list

1. Open the Servo Setup (.srv) file or, if your ladder containing a servo function is open, 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 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 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. Choose Update Forcing to send the updated values to the PiC.

## Axis Tuning using Viewing

You can read values from certain servo variables for an axis running in the PiC using the Servo View List.

### To view an axis servo variable in the Servo View list

1. Open the Servo Setup (.srv) file or, if your ladder containing a servo function is open, 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 TUNEWRIT 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

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.

## CHAPTER 4 PiC and SERCOS

### SERCOS Setup Background Information

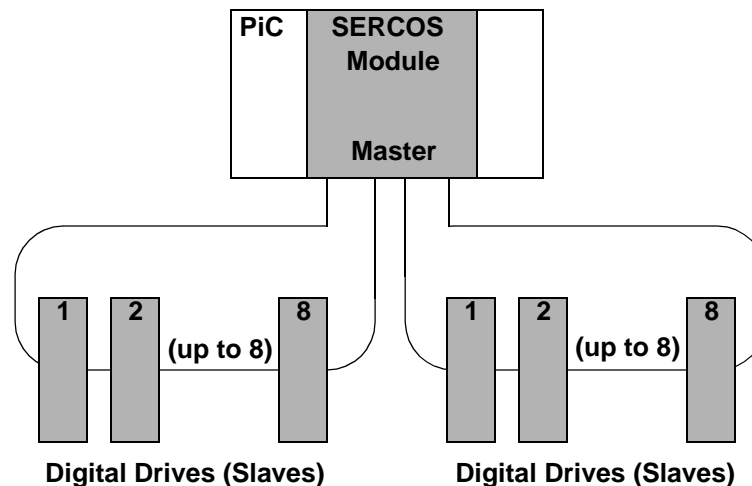
---

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.

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.

**FIGURE 4 - 1. PiC/SERCOS Network**



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 configuration supports up to eight slaves on the ring each slave controlling a single drive. It does not 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.

## 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.

### NOTE

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_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
REF Group	
SCA_ACKR	Acknowledges the reference cycle for a servo axis
SCA_REF	Starts 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

## 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 the Motion.Lib, 4 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.



## 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            | 6. Definitions of telegram contents |
| 2. Diagnostics                   | 7. Drive operation modes            |
| 3. Lists of operation data       | 8. Standard operation data          |
| 4. Internal and external signals | 9. Drive parameters                 |
| 5. Manufacture specifications    |                                     |

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, Tncyd
	00002	Communication Cycle Time, Tscyc
	00006	AT Transmission Starting Time (T1)
Phase 3	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
Phase 3	00127	CP3 Transition Check
	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 ( $\geq 8$ ), 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 Application Note

---

### 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 Procedure CMD	Receive IDN	Receive Continuous	Send IDN	Upload Drive Information
-----------------------	-------------	--------------------	----------	--------------------------

Keep the following in mind:

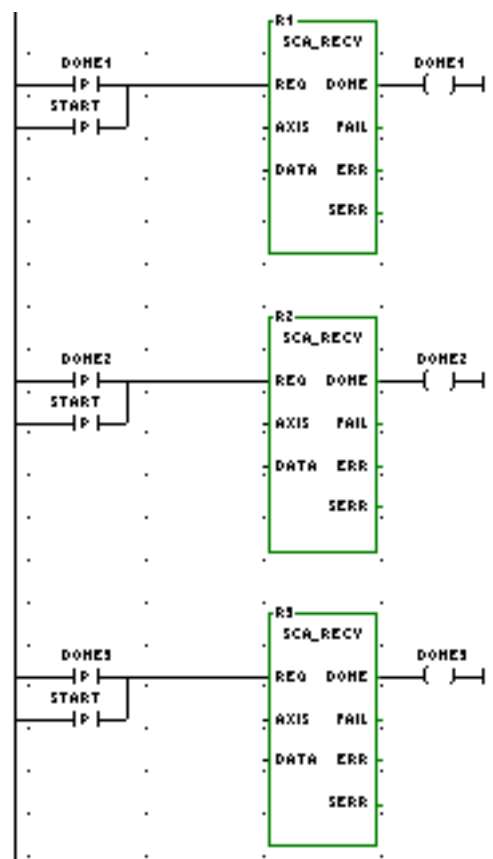
- The LDO requests are used for machine control.
- SERCOS setup requests are used for diagnosis.
- Only one request is processed on the ring at a time. It must be completed before the next one begins.
- The SERCOS module allows the LDO to queue up to 16 requests to the service channel.
- The LDO requests 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.

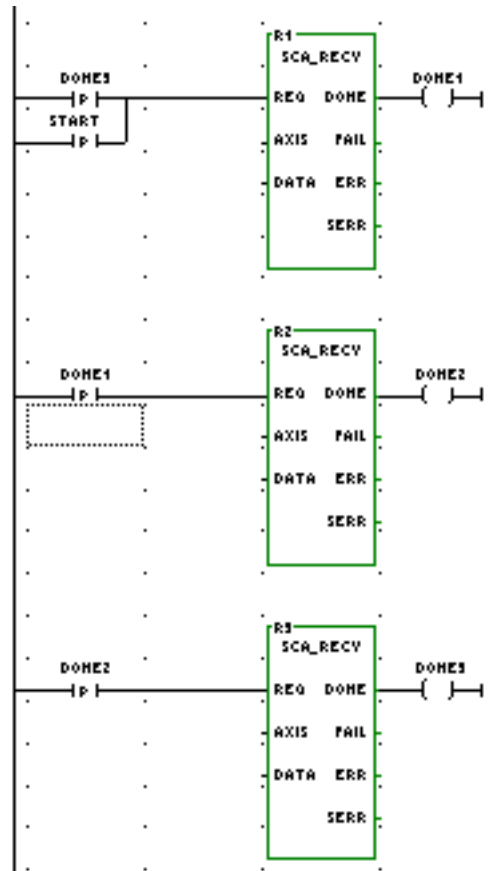
### Example 1 Logic that gives priority to the LDO requests

In this example, you want to read IDN 1, read IDN 2 when IDN 1 is complete, read IDN 3 when IDN 2 is complete, read IDN 1 when IDN 3 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.



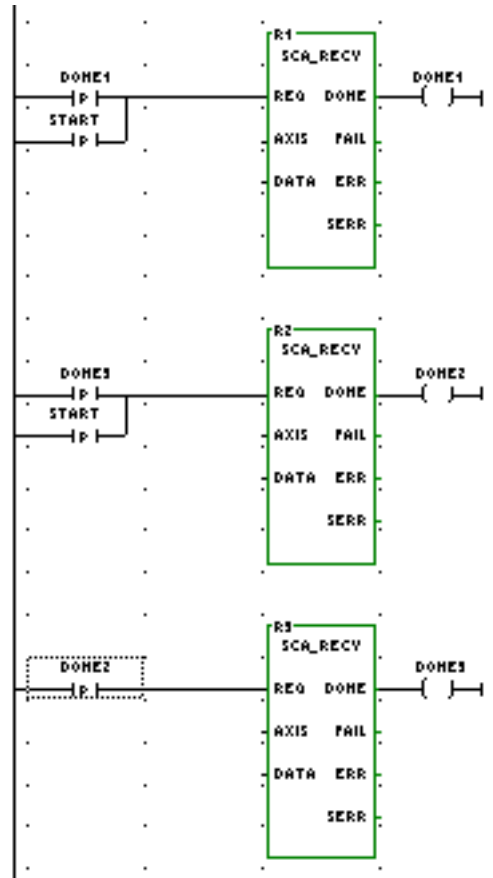
## Example 2 Logic that allows SERCOS Setup commands to be read sooner

In this example, you want to read IDN 1. 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.





## 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
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

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.

### REGIST Function

The registration function cannot be used on a SERCOS axis.

### Reference-Related Functions

The SCA\_REF (servo) and the SCS\_REF (non-servo) function blocks are used to perform referencing functions with a SERCOS axis.

The reference-related functions in PiCPro on the left below cannot be used with a SERCOS axis. The functions on the right can be used with a SERCOS axis.

**Not available with  
SERCOS**

FAST\_REF  
LAD\_REF  
REF\_DNE?  
REF\_END

**Available with  
SERCOS**

PART\_REF  
PART\_CLR

**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


## Opening SERCOS Setup

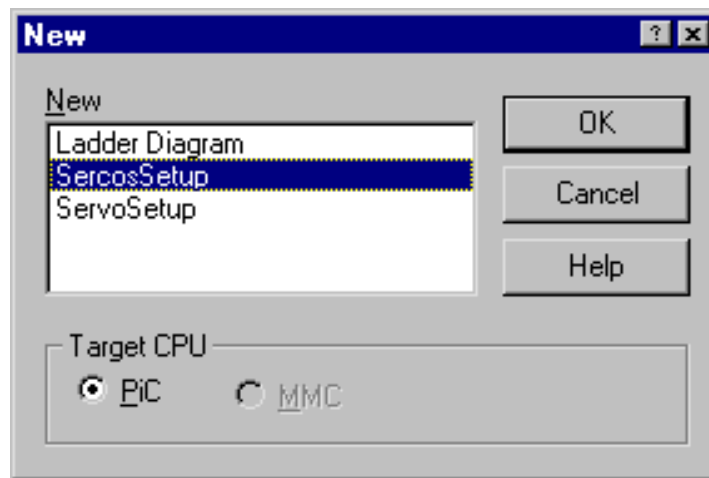
---

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 PiC so that the SERCOS slaves can be initialized. There are several ways to access the SERCOS Setup program.

**Note:** SERCOS setup cannot be accessed from the MMC Edition. SRC files cannot be created in the MMC Edition of PiCPro.


### 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. Choose OK.

### 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>.

### To open an existing SERCOS setup file from the setup function in your 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 and right click your mouse. Choose View SERCOS Function.  
or

2. Place focus on the setup function in your ladder and choose View, SERCOS Function from the menu and then select the desired setup function from the fly-out list or right click the mouse and choose View SERCOS Function.

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.

## **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/paste commands**

1. Enter the SERCOS configuration.
2. Select the ring(s), slave(s), or IDN(s) (within the Insert Slave box) you want to copy or cut. Choose the copy or cut command from the Edit menu or toolbar buttons, press <Ctrl + C> (copy) or <Ctrl + X> (cut), or right click and choose Copy or Cut.
3. Place the focus on the location you want to paste into and choose the paste command from the Edit menu or toolbar buttons or press <Ctrl + V> or right click and choose Paste. The configuration will be pasted above the selected location.
  - Double click on the slave.

Make the necessary changes and choose OK.

## **Saving a SERCOS Setup File**

---

It is a good idea to save your file at regular intervals as you work on it. Using the Save command, you can save the file under its existing name. Using the Save As 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 File, Save.
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.
4. Click Save.

### **To save an existing file**

- Choose File, Save.

You can use the Save As 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.

### **To save a file under a different name or in a different location**

1. Choose File, Save As from the menu.
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 Save.

## **Printing SERCOS Setup**

---

You can print the selection you have highlighted 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.
2. Choose File, Print 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 All 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.

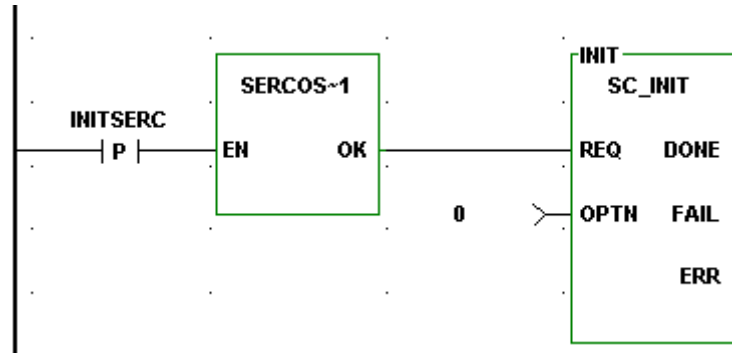
### To create a SERCOS setup function

1. After all the setup information has been entered, choose Compile, Make 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 .SRC file to.  
There is a default that is entered in the File Name box beginning with SERCOSSetup1.SRC. You can accept this name or enter your own. NOTE: The second setup file you create will have the default name SERCOSSetup2.SRC, the third will be SERCOSSetup3.SRC and so on.
3. The Compile SRC box appears. The Registered SERCOS Setup 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 SERCOS Setup Library Name: box.
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. 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.

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.

## 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.



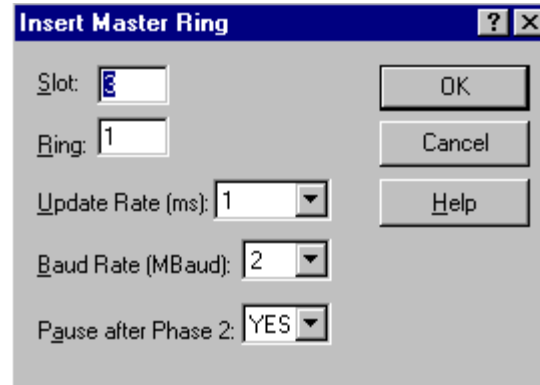
## Inserting/Editing SERCOS Rings

---

A ring is added to SERCOS Setup by using the Insert command from the 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. Open the SERCOS setup file.
2. Choose Insert, Ring from the menu. The Insert Master Ring box appears.



3. Enter the slot number (3-13) identifying the slot in which your SERCOS module is installed in the PiC rack.
4. Enter the port number (1-2).
5. The default update rate is 4 ms. Choose an update 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 PiC update rate. The PiC 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. All other update rates allow two rings with eight slaves on each.
6. Choose the baud rate from the drop down list (2 or 4).
7. Choosing YES for Pause 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. When finished entering ring data, choose OK to accept your ring configuration.
8. 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 Edit menu and selecting Ring Data. This brings up the Edit Master Ring Data box.
9. 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.

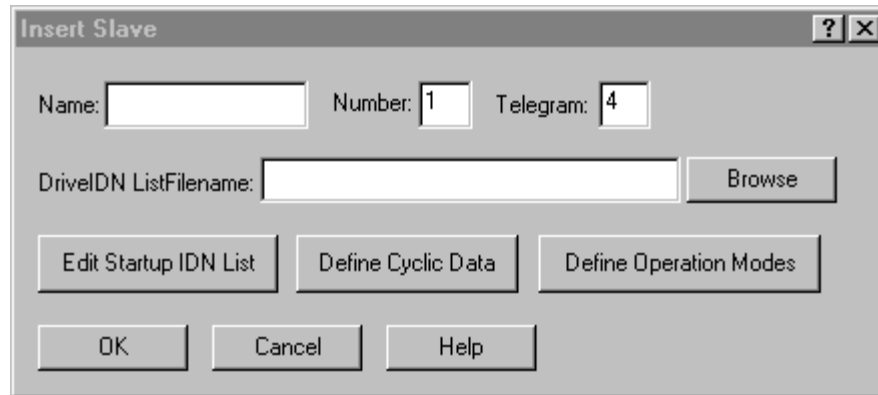


## Inserting/Editing SERCOS Slaves

---

### 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. Enter the slave number (1-8). The slaves on a ring must be numbered sequentially. For example, if you are entering three slave axes on this ring, number them 1, 2, 3; not 1, 2, 4.  
The number entered here must match the address switch set on the slave.
3. 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

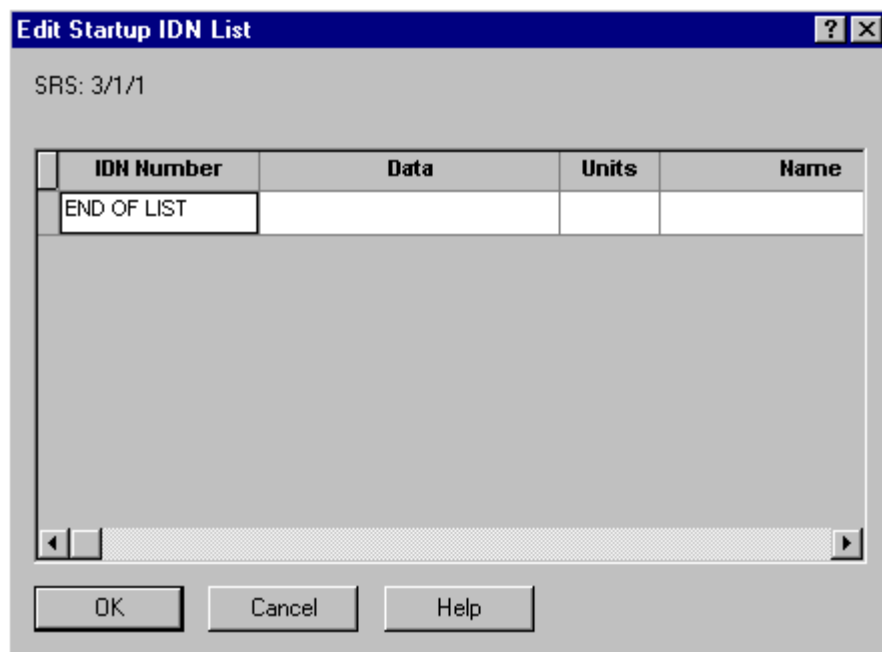
0	No data
1	Torque control
2	velocity control, velocity feedback
3	Velocity control, position feedback
4	Position control, position feedback
5	Position and velocity control, position and velocity feedback
6	Velocity control
7	User-defined

4. Enter the name and location of the Drive IDN List File for this slave. Use the Browse button to locate the file if necessary.
5. 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 box appears.

## Edit Startup IDN List

---

When you edit the 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 column or press <Insert>. The Insert IDN box appears.



You can cut/copy/paste entries within this list using the <Ctrl + X, Ctrl + C, and Ctrl + V > keys.

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.

## Inserting/Editing IDNs

---

Enter the data for the IDN to be added to the startup list.

The screenshot shows the 'Insert Idn' dialog box. The 'Slot/Ring/Slave Information' section contains the text 'SRS: 6/1/1'. The 'IDN Information' section has two buttons: 'System IDNs' and 'Drive IDNs'. Below these are fields for 'Number:' (a dropdown), 'Name:' (a text box), 'Length:' (a dropdown), and 'Element:' (a dropdown showing 'Operation Data'). At the bottom of this section is a 'Data:' label and a large text area labeled 'IDN Data' containing the number '0'. To the right of the 'IDN Data' box are labels for 'Min:', 'Units:', and 'Max:'. At the very bottom are 'OK', 'Cancel', and 'Help' buttons.

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 Edit Startup List. Cancel allows you to exit without saving any data.

## Define Cyclic Data

---

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.

Define Cyclic Data

Slot/Ring/Slave Information  
6/6/1

Cyclic Data

Telegram

#	MDT Description	AT Description	Intended Mode
4	IDN 47 Position Cmd	IDN 51 Position F3bk 1	Position

MDT/AT Description

MDT: S00047 MDT...

AT: S00051 AT...

OK Cancel Cyclic Data Structures Help

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.

The telegrams you choose are held in IDN 15. The MDT and AT telegram boxes hold the IDNs connected to the telegram you select.

If you chose telegram 0, 1, 2, or 6, you do not have to make any other choices.

If you chose telegram 3, 4, or 5, you are using feedback 1 on the drive.

If you chose telegram 11, 12, or 13, you are using feedback 2 on the drive.

If you use 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 an IDN in the description to remove it.

NOTE: When you are working with motion.lib, either select telegram 4 or telegram 7. If you chose 4, nothing further needs to be done. If you choose 7, IDN 47 (position read) must appear first in the MDT description and IDN 51 for feedback 1 (IDN 53 for feedback 2) (position write) must appear first 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.

The list of IDNs for AT become IDN 16.

## Cyclic Data Structures

If you are using the SCA\_RCYC and SCA\_WCYC functions from motion.lib, you will need the structures that are placed in the Cyclic Data Structures box by PiCPro. 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.

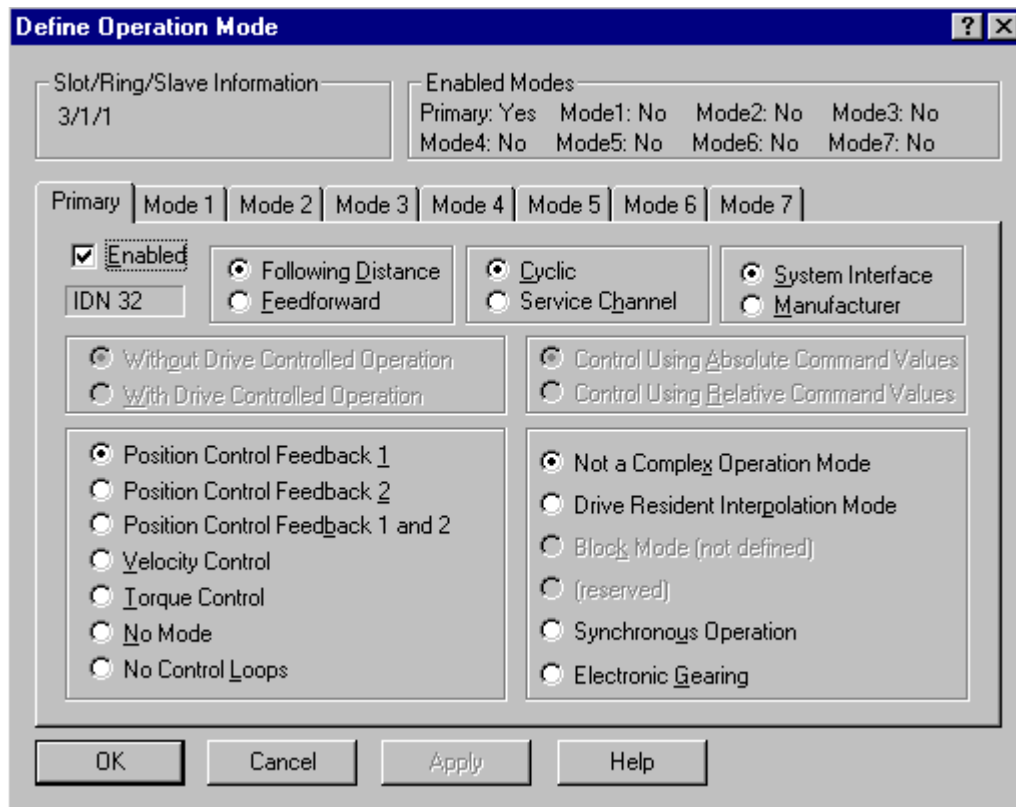
Name	Type	A.	I/O Point	Initial Value	Long Name
ILISTW	STRUCT				
.S00047	DINT				
.SIZE	USINT			4	
	END_STRUCT				
ILISTR	STRUCT(0..1)				
.S00051	DINT				
.SIZE	USINT			4	
	END_STRUCT				

OK Copy to Clipboard Help

## Define Operation Modes

---

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 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.

## 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.

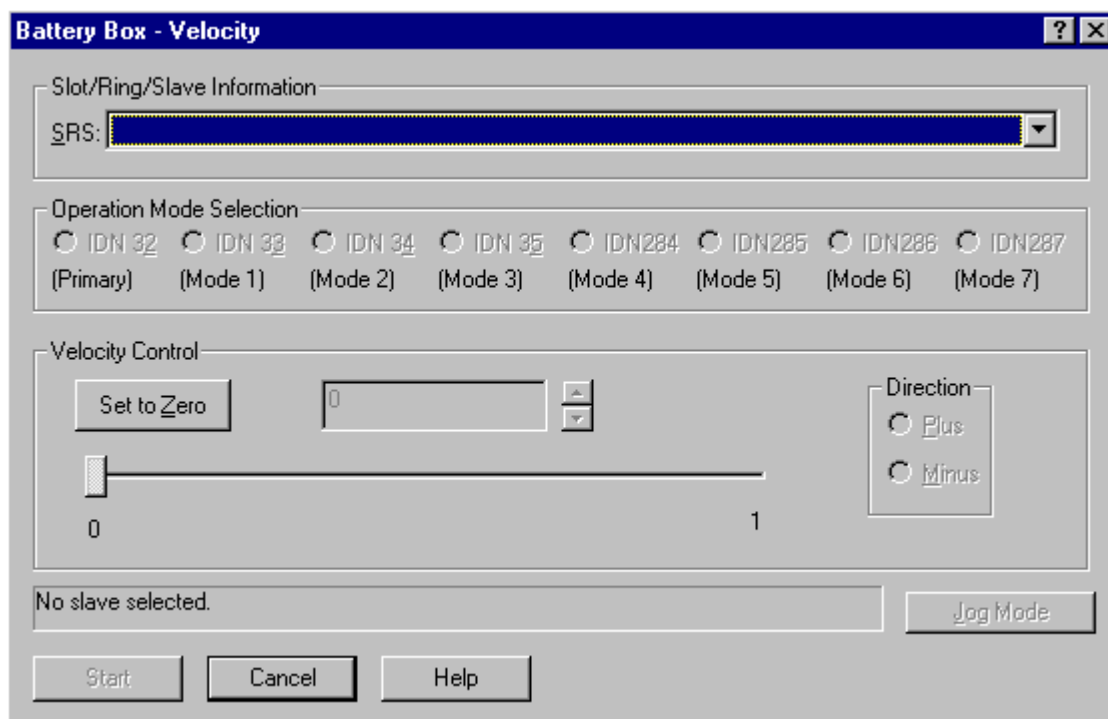
The mode can be set by calling the SCA\_CTRL function with the OPTN input set to 1. Prior to calling SCA\_CTRL, the default is the primary mode. You can disconnect motion.lib from the drive by writing a 1 to WRITSV variable 48. Then you use the SCS\_CTRL function to write the loop state, the cyclic data, 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.

## Battery Box

---

The battery box feature allows you to command velocity motion of a SERCOS slave from PiCPro. Once started, the motion continues either until you click on the stop button in the Battery Box-Velocity box or until you release the right or left arrow key when using the Battery Box-Velocity Jog Mode. The latter is accessed via the Jog Mode button.





## Battery Box Application Note

---

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.
2. Ensure that all motion is stopped.
3. 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.
4. Call the CLSLOOP? function to ensure that both loops have been opened.
5. 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.
6. When you are finished using the battery box, exit it.
7. 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.
8. Ensure that all motion is stopped.
9. Call the SCA\_CLOS function to close the loop.
10. Call the CLSLOOP? function to confirm that both loops are closed.

You are now ready to resume control using motion.lib in the PiC.

## Receive IDNs

---

You can receive information on an IDN from the drive with this feature. 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.

The screenshot shows the 'Receive IDN' dialog box. It has a title bar with the text 'Receive IDN' and standard window controls. The main area is divided into sections. The first section, 'Slot/Ring/Slave Information', contains a dropdown menu labeled 'SRS:'. The second section, 'IDN Information', contains four buttons: 'System IDNs', 'Drive IDNs', 'Cyclic Data IDNs', and 'Startup IDNs'. Below these buttons are four input fields: 'Number:' (a dropdown menu), 'Name:' (a text box), 'Length:' (a dropdown menu), and 'Element:' (a dropdown menu with 'Operation Data' selected). Below these fields are three buttons: 'Receive', 'Status:' (a text box), and 'Help'. The bottom half of the dialog is a large area labeled 'Data' with a scroll bar.

You can choose from four categories of IDNs; system, drive, cyclic data, 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. Press it to read the requested IDN from the drive. The Receive button changes to Abort.

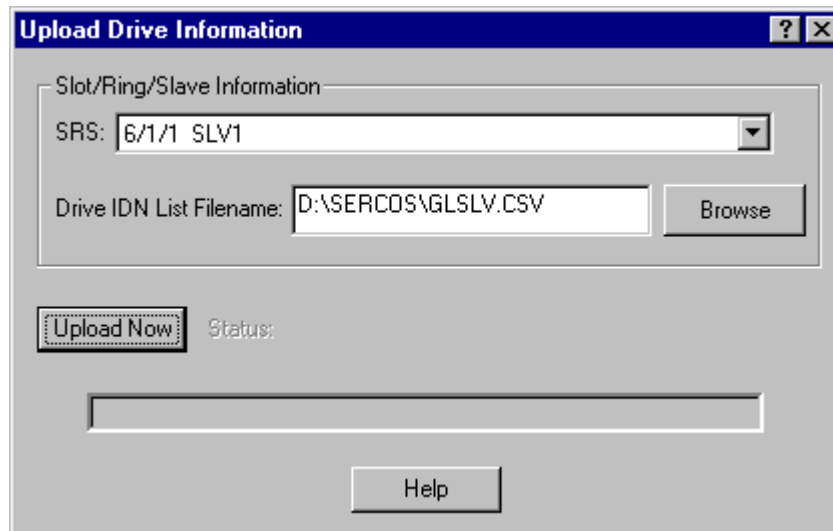
The Status box reports on the progress of the receive command.

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.

## 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.



Available slave axes will be listed in the drop down list for SRS. Choose the slave you want to upload the drive information for.

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.

## 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

## 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.

Enable	SRS	IDN Number	Element	Data
<input type="checkbox"/>	6/M/1 SLV1	S00001	Minimum	
<input checked="" type="checkbox"/>	6/M/1 SLV1	S00003	Operation Data	
<input type="checkbox"/>	6/M/1 SLV1	S00004	Operation Data	
<input type="checkbox"/>	6/M/1 SLV1	S00001	Operation Data	
<input type="checkbox"/>	End List			

Receive      Status:      Help

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 <Enter>. 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 question mark and a close button. 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 with 'Operation Data' selected), 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 after phase 2 is completed, 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.

Enable	SRS	IDN #	Elem...	Data	Units	Status	
<input type="checkbox"/>	6/M	SLV	S00011	Opera...	2#1010		Class 1 die
<input checked="" type="checkbox"/>	6/M	SLV	S00041	Opera...	9.994645		Homing ve
<input checked="" type="checkbox"/>	6/M	SLV	S00011	Opera...	2#1010		Class 1 die
<input type="checkbox"/>	6/M	SLV	S00041	Opera...	9.994645		Homing ve
<input type="checkbox"/>	6/M	SLV	S00011	Opera...	2#1010		Class 1 die
<input type="checkbox"/>	6/M	SLV	S00011	Opera...	2#1010		Class 1 die
<input checked="" type="checkbox"/>	6/M	SLV	S00011	Opera...	2#1111		Class 1 die
<input type="checkbox"/>	6/M	SLV	S00002	Opera...	1000	us	Communic
<input type="checkbox"/>	6/M	SLV	S00001	Opera...	1000	us	Control uni
<input type="checkbox"/>	6/M	SLV	S00003	Opera...	1000	us	Shortest A
<input type="checkbox"/>	6/M	SLV	S00003	Opera...	1000	us	Shortest A

Status:

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.

**Insert Idn**

Slot/Ring/Slave Information  
SRS: 6/1/1 SLV1

IDN Information  
System IDNs Drive IDNs Cyclic Data IDNs Startup IDNs

Number: [dropdown] Name: [text box]  
Length: [dropdown] Element: Operation Data [dropdown]  
Data: [IDN Data text area]  
Min: [text box]  
Units: [text box]  
Max: [text box]

OK Cancel 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.

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 Send List. Cancel allows you to exit without saving any data.

## Execute Procedure Command

---

The Execute Procedure 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.

The screenshot shows a Windows-style dialog box titled "Execute Procedure Command Function". It has a standard title bar with a question mark icon and a close button (X). The dialog is divided into two main sections. The first section, "Slot/Ring/Slave Information", contains a label "SRS:" followed by a text box and a dropdown arrow. The second section, "IDN Information", contains two buttons: "System PCFs" and "Drive PCFs". Below these buttons are two labels, "Number:" and "Name:", each followed by a text box and a dropdown arrow. At the bottom of the dialog, there is an "Execute" button on the left and a "Help" button on the right.

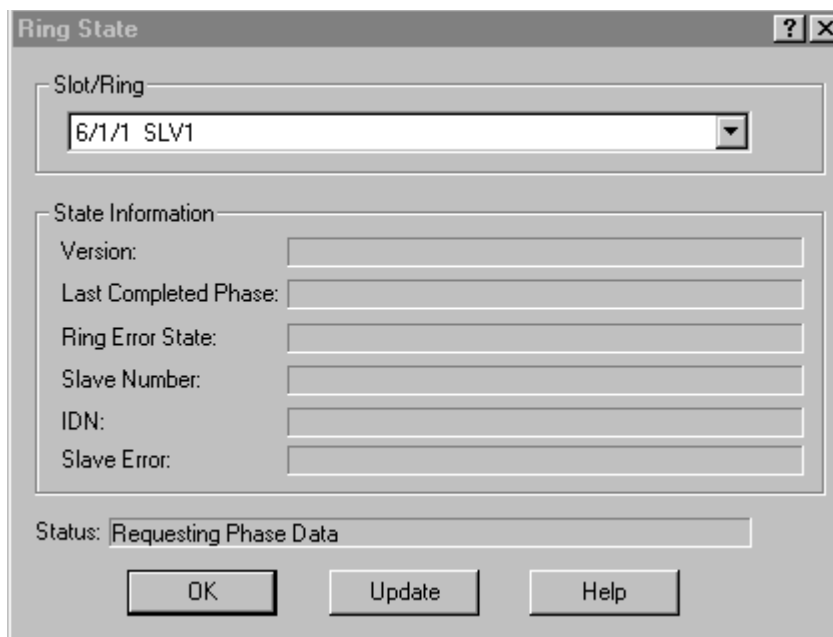
Once the PCF has been entered, the Execute button is active. Press it to send the PCF IDN to the drive and begin the procedure. The progress of the procedure will be noted to the right of the Execute button.

The Execute button converts to a Cancel button when progress begins.

## Ring State

---

You can read the State Information shown below for a ring.

A screenshot of a software dialog box titled "Ring State". At the top, there is a "Slot/Ring" dropdown menu showing "6/1/1 SLV1". Below this is a section titled "State Information" containing six text input fields: "Version:", "Last Completed Phase:", "Ring Error State:", "Slave Number:", "IDN:", and "Slave Error:". At the bottom of the dialog, there is a "Status:" label followed by a text box containing "Requesting Phase Data". Three buttons are at the very bottom: "OK", "Update", and "Help".

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"><li>▪ SERCOS board in correct slot</li><li>▪ SR structure members correct</li></ul>
17	The SERCOS module did not receive an expected AT response. Cable could be disconnected.	<ul style="list-style-type: none"><li>▪ Check connection</li></ul>
20	Phase 0 detected that the ring is not complete.	<ul style="list-style-type: none"><li>▪ Check connection</li><li>▪ Ensure drive is turned on</li></ul>
65	Error occurred calculating when MDT should occur.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ One or more drives cannot accommodate required MDT</li></ul>

66	Error occurred calculating when drive data valid.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ One or more drives cannot accommodate command times</li> </ul>
67	Error occurred calculating when feedback data valid.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ One or more drives cannot accommodate feedback capture times</li> </ul>
68	Error occurred calculating total time required for communication cycle.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ Cyclic data on slaves too long</li> <li>■ Update rate too fast</li> </ul>
69	Error occurred calculating cyclic data memory for SERCON processor.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ Cyclic data on slaves too long</li> </ul>
70	Error occurred calculating cyclic data memory for internal memory map.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ Cyclic data on slaves too long</li> </ul>
71	Error occurred calculating service channel memory map.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ Cyclic data on slaves too long</li> </ul>
74	CPU on SERCOS module has too many tasks during update.	<ul style="list-style-type: none"> <li>■ Too many slaves on one ring</li> <li>■ Cyclic data on slaves too long</li> </ul>
128	Slave error occurred. Read SERR output to identify error. The SLV output indicates the slave number.	<ul style="list-style-type: none"> <li>■ SLV output contains slave number</li> <li>■ IDN output contains the IDN transfer that caused the error</li> <li>■ SERR output contains the drive generated error number</li> </ul>
136	Individual slave will not respond. The SLV output indicates the slave number.	<ul style="list-style-type: none"> <li>■ Read Drive diagnostic IDN 95</li> <li>■ Address switch on drive does not match slave number</li> <li>■ Baud rate switch on drive does not match rate in ring definition</li> <li>■ SLV output contains slave number that does not respond</li> </ul>
144	Individual slave cannot carry out a Procedure Command Function. The SLV output indicates the slave number.	<ul style="list-style-type: none"> <li>■ SLV output contains slave number</li> <li>■ IDN output contains the Procedure Command Function that caused the error</li> <li>■ For IDN = 127, read IDN 22 to read list of IDNs still required by the drive</li> <li>■ For IDN = 128, read IDN 23 to read list of IDNs still required by the drive</li> <li>■ Read Drive diagnostic IDN 95</li> </ul>

The Slave Number can be 1 to 8.

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". The "Control" section contains input fields for "Drive" (three stacked boxes), "Operation Mode", "Realtime Bit 1" (two stacked boxes), and a "Value" field. The "Status" section contains input fields for "Drive", "Error", "Operation Mode", "Realtime Bit 1", "Realtime Bit 2", and a "Value" field. At the bottom of the window are two buttons, "Start" and "Help", followed by a "Status:" label and 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

---

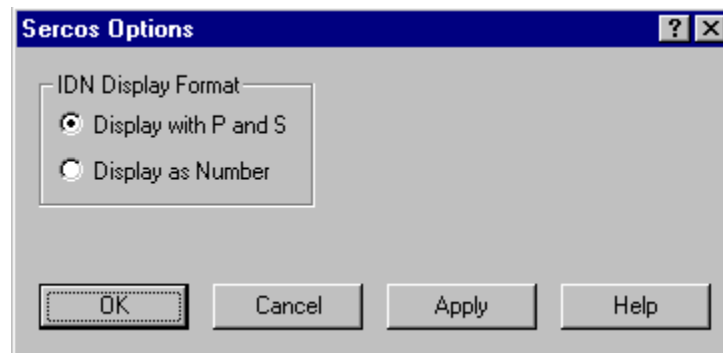
There are lists of IDNs for the system, the drive, the cyclic data, the startup, the system PCFs, and the drive PCFs. While working in SERCOS setup, there are several dialog boxes that allow you to access these lists of IDNs. You can also view the system and drive IDN lists from under the View menu.

### Options for IDN Display

There is an Option command found under the View 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 - C:\PROGRAM FILES\GIDDINGS & LEWIS\PICPRO FOR WINDOWS V10.0\idnssystem.... ? X

Number	Name	Attribute	Value	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			

OK Find Name String Print Help

You can use the Find Name String button to bring up an IDN Search box.

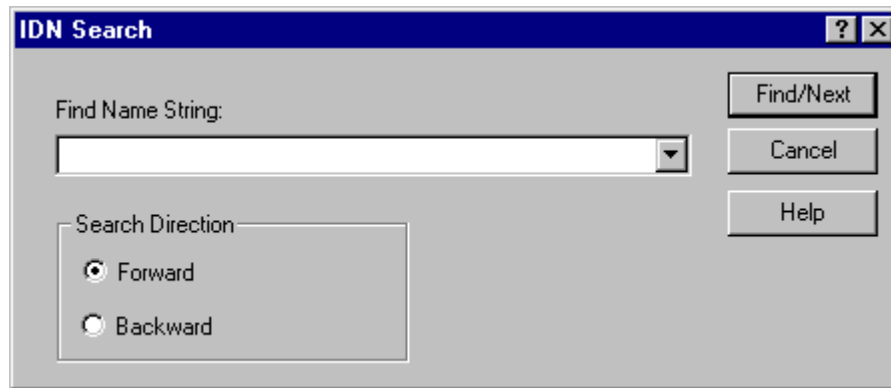
The Drive IDNs button displays a list of IDNs that you uploaded from the drive. You can sort by number or name, toggle the sort order, search by name, and print this drive IDN list.

The Cyclic Data IDNs button displays a list of IDNs that you entered as cyclic data.

The Startup IDNs button 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.

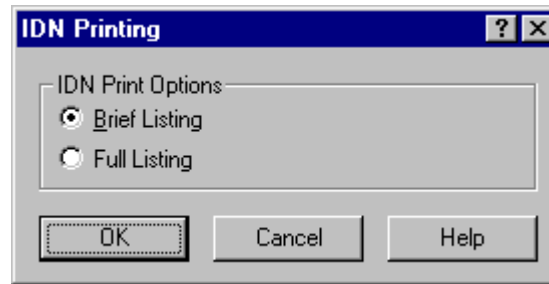


To find specific text in an IDN list, follow these steps:

1. Press the Find Name String button on the IDN list you are viewing. This will display the box above.
2. Type the desired test 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. Press 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, press 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.



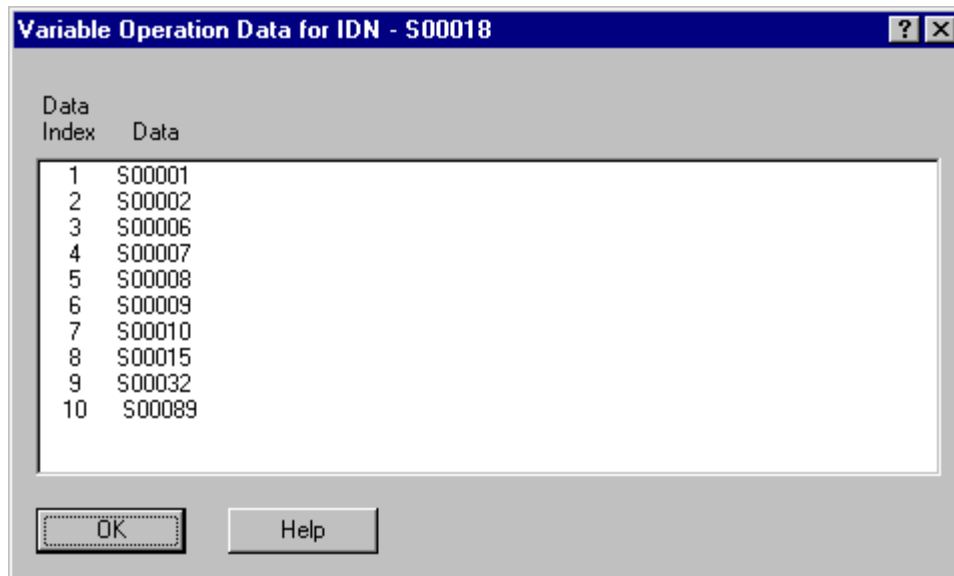
To print an IDN list, follow these steps:

1. Press 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. Press 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.

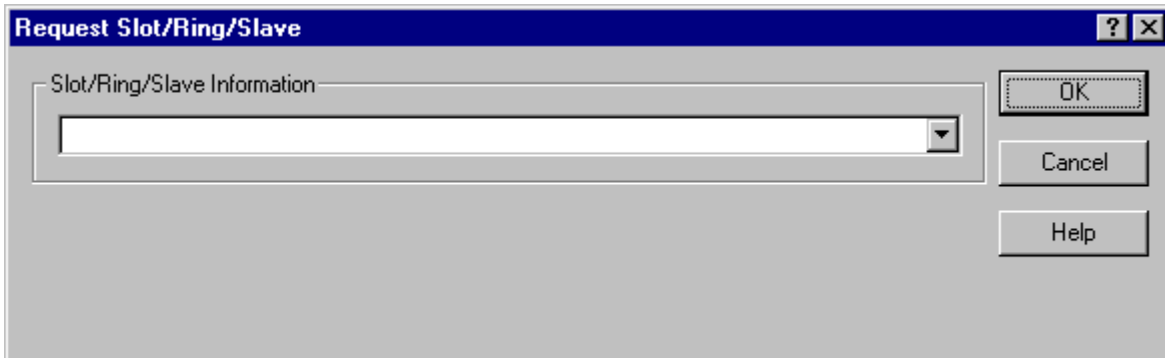


To view variable length operation data, follow these steps:

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, press 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 View menu on the main menu bar, you must indicate which drive you want. Using the Request SRS dialog box, select the appropriate SRS (Slot/Ring/Slave) which uniquely identifies the desired drive.



To specify the SRS, follow these steps:

1. From SERCOS Setup, select the View/View Drive IDN 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, press the OK button. The Drive IDN list for the specified drive will be displayed.

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 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.

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 PiC 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 to/from 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 provides the feature description for the drive. Knowing the drive features available will aid you in your LDO design.

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 current position from the drive must be read. The reference cycle within the drive defines the value for IDN 47. The SCA\_REF 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.

# 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/Read 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 5 Working With TASKS and UDFBs

Two programming tools are covered in this chapter; TASKs and UDFBs (User Defined Function Blocks).

### 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 affect 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

<b>Inputs:</b>	EN (BOOL) - enables execution
	SERV (TIME) - 1, 2, 4, 8, or 16ms constant for servo task. Enter in the T# format.
<b>Outputs:</b>	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.
	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.

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 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?

PID

### 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

MEASURE  
SCA\_RECV  
SCS\_SEND

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  
SCS\_READ

CAPTINIT  
C\_ERRORS  
HOLD  
RATIOSCL  
TME\_ERR?  
SCR\_CONT  
SCS\_STAT

CAPTSTAT  
C\_STOP  
HOLD\_END  
READ\_SV  
TUNEREAD  
SCR\_ERR  
SCS\_WRIT

CLOSLOOP  
E\_ERRORS  
OPENLOOP  
READ\_SV  
TUNEWRT  
SCR\_PHASE

CLSLOOP?  
E\_STOP  
P\_ERRORS  
REF\_DNE?  
WRITE\_SV  
SCS\_CTRL

The following 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.

## STATUSSV

### 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, 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.

## 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.

## 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.

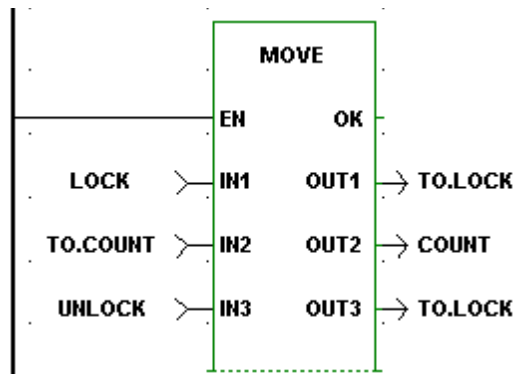
### Semaphore Flag Example 1 - Data Transfer from Task LDO to Main LDO

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

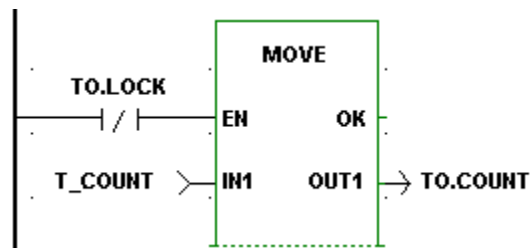
The declarations in the main LDO software declarations table are:

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	

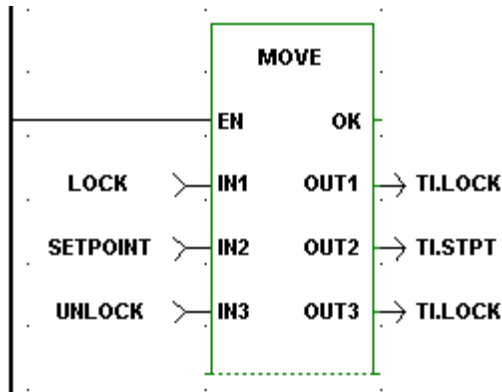
### Semaphore Flag Example 2 - Data Transfer from Main LDO to Task LDO

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

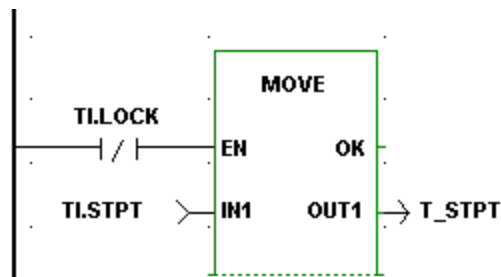
The declarations in the main LDO software declarations table are:

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	



## 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.).

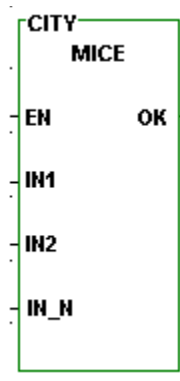
The UDFB follows the same conventions as the standard function blocks found in PiCPro. It must be declared in the software declarations table when it is used in the network of a module. You assign a name to it when it is declared.

### 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.)

### UDFB Template Example



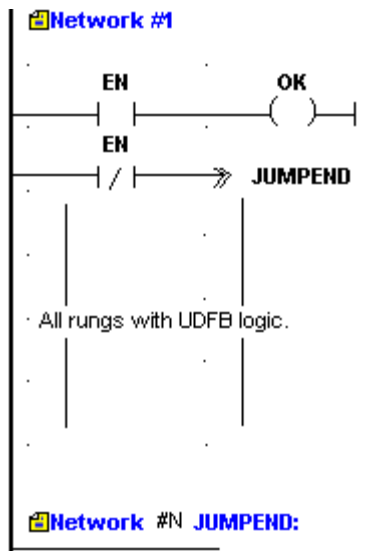
The EN input and the OK output are required. The remaining inputs and outputs are determined by you when creating the module.

## 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.

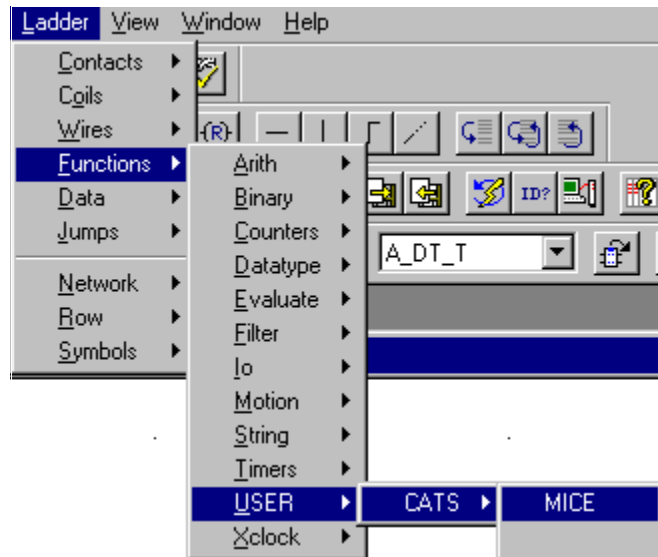
## Contents of the UDFB LDO module

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.

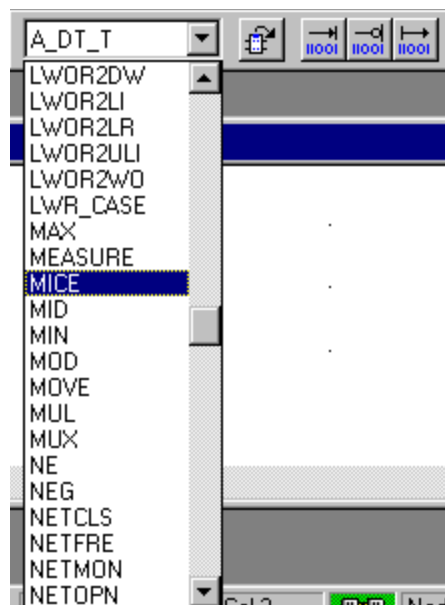
## Size of UDFB Libraries

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

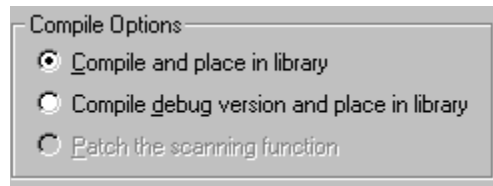
Data is copied and pointers are passed into a UDFB only when there is power flow to the first input (EN). *String, array, or structure pointers within a UDFB must not be accessed until power flow to the EN has occurred.*

## 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.

## 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.

## Viewing the Parent Ladder

When you are working in the UDFB ladder, you can view the parent ladder by choosing View, Parent.

## 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.

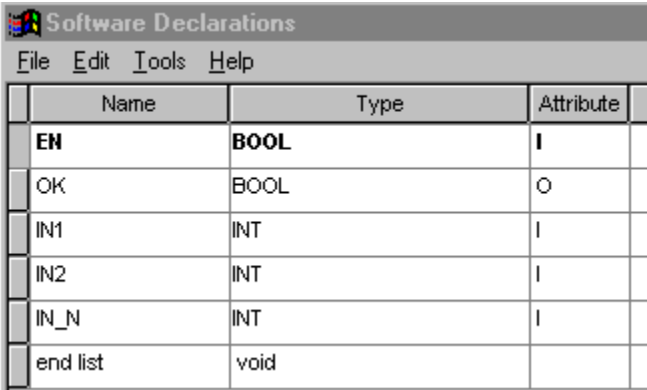
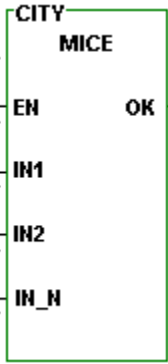
The variables which will become the inputs for the UDFB may be any data type except function blocks.

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		CITY	
EN	BOOL	I		MICE	
OK	BOOL	O		EN	OK
IN1	INT	I		IN1	
IN2	INT	I		IN2	
IN_N	INT	I		IN_N	
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. They must 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**

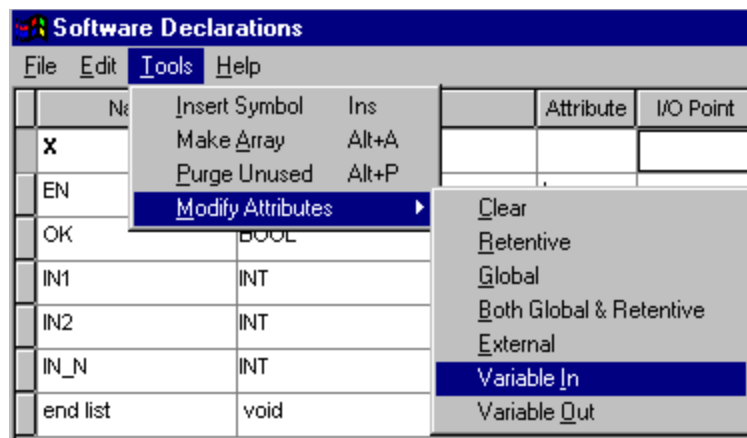
The maximum number of inputs or outputs for a UDFB is 64. It is recommended that you keep the number to a minimum (under 16). More can be declared if necessary, but transferring all the inputs and outputs to and from a function block does use scan time. There is also the constraint of the 255 element 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

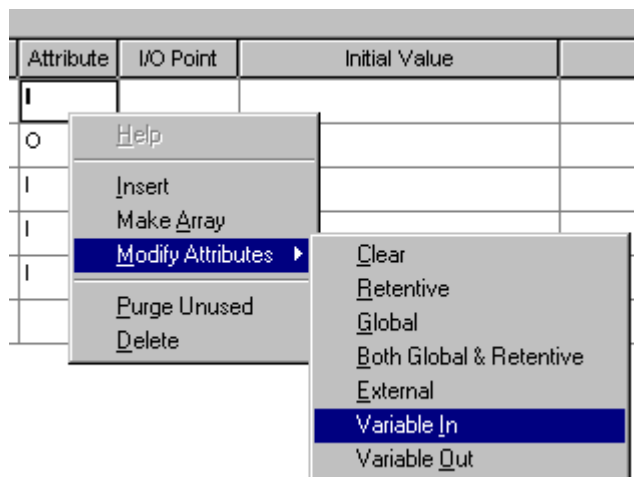
You must mark all the inputs and outputs for the function block in the software declarations table using the attribute tool. When you compile your UDFB ladder, these inputs and outputs will appear on the UDFB template in the order you have entered them. Remember that the EN input and the OK output must 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 alternatively, right click your mouse to bring up the following.



Either method will enter the **I** or **O** symbol in the Attribute column.



## 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 D-1 and D-2.

4mA0..... BOOL	CNTL..... UINT	NETOPEN, NETRCV, NETSND, OPEN, READ, SEEK, STATUS, WRITE, DELFIL, FRESpace, RENAME, COORD2RL, TUNEWRIT
10ms..... BOOL	CU .....BOOL	
100ms..... BOOL	CV ..... INT	
<b>A</b>	<b>D</b>	
ACCL..... UDINT	DABL.....BOOL	ERR .....SINT
ACT ..... DINT	DATA ..... DINT	ERR ..... USINT
PID	DAY..... UINT	A_INCHIT, A_INCHRD, A_INMDIT, ARTCHIT, ARTDCHRD, ARTDMDIT, ATMPCHIT, ATMPCHRD, ATMPMDIT, STRTSERV, CAPTINIT, SERVCLK
ACT .....INT	DBUF ..... ARRAY, STRUCT, STRING	
NETRCV, NETSND, READ, WRITE	DCNT ..... INT	
ACTV .....WORD	DECL .....UDINT	ERRS..... WORD
ANGL..... REAL/LREAL	DEN..... INT	ESTP ..... BOOL
AXIS .....USINT	DEST .....ARRAY OF STRUCT	ET ..... TIME
<b>B</b>	DID .....USINT	<b>F</b>
BEG..... BOOL	DIFF .....(same as IN0)	FAHR ..... BOOL
BIPO..... BOOL	DIM..... DINT	FAIL ..... BOOL
BKPR..... BOOL	DIR.....STRING	FAST..... USINT
BTVL ..... DINT	DIST ..... DINT	<b>G</b>
BUFR .....ARRAY, STRUCT, STRING	DISTANCE, FAST_QUE	G ..... BOOL
<b>C</b>	DIST .....UDINT	<b>H</b>
CAM..... ARRAY OF STRUCTURES	REGIST	HILT ..... BOOL
CD ..... BOOL	DVND ..... NUMERIC or TIME	HNDL..... INT
CFGZ.....STRING	DONE.....BOOL	<b>I</b>
CHAN .....USINT	DROP ..... DINT	IGNR..... UDINT
CLRC..... UINT	DTST.....STRING	INPS ..... BOOL
CLSD..... BOOL	DVSR ..... same NUMERIC as DVND or DINT if DVND = TIME	INs..... (See Table D-1.)
CMD .....UINT	<b>E</b>	IST ..... STRUCT
CNT .....INT	ELEM ..... UINT	<b>K</b>
COMN .....STRUCT	EN .....BOOL	K..... NUMERIC
CONF .....STRUCT	ERR.....BOOL	<b>L</b>
COS..... REAL/LREAL	TME_ERR?, ANLG_OUT	L ..... INT
CSTOP ..... BOOL	ERR..... INT	LD ..... BOOL
	ASSIGN, CLOSE, CONFIG,	

LEN..... INT	PLUS..... BOOL	SDST..... DINT
LGTH..... UDINT	PNT..... USINT	SEG1..... STRUCT
LOG..... REAL/LREAL	PNUM..... USINT	SET..... BOOL
LOLT..... BOOL	POS..... DINT	SID..... USINT
LN..... REAL/LREAL	PROD..... same as MCDN	SIN..... REAL/LREAL
<b>M</b>	PT..... TIME	SIZE..... DINT
MAN..... BOOL	PV..... INT	SIZE..... USINT
MAST..... USINT	<b>Q</b>	CAPTINIT
MAX..... same as MIN	Q..... BOOL	SLOT..... USINT
MCND..... NUMERIC or TIME	QAVL..... BOOL	SLPE..... ARRAY OF STRUCTURES
MDST..... DINT	QTY..... DINT	SPT..... DINT
µsec..... UINT	QUE..... USINT	SQR..... UDINT, UINT, USINT, or constant
MIN..... any	QUOT..... same as DVND	SRCE..... ARRAY OF STRUCT
MODE..... INT	<b>R</b>	SSTR..... DINT
MOVE..... STRUCT	R..... BOOL	STAT..... INT
MPLR..... same NUMERIC as MCND or DINT if MCND = TIME	RACK..... USINT	NETMON, STATUS
MSTR..... DINT	RATE..... UDINT	STAT..... WORD
<b>N</b>	RATE..... TIME	STATUSSV
N..... USINT	SERVOCLK	STR..... STRING
NAME..... STRING	RDNE..... BOOL	STRC..... STRUCT
NAMZ..... STRING	REAL..... ARRAY OF STRUCT	SUM..... same as IN1
NUM..... INT	REFD..... DINT	<b>T</b>
RATIOSCL	REM..... same as DVND	TAN..... REAL/LREAL
NUM..... NUMERIC	REQ..... BOOL	TBUF..... ARRAY, STRUCT, STRING
NUM2STR, STR2NUM	REV..... BOOL	TCNT..... INT
NUM..... REAL/LREAL	ROOT..... same as SQR	TIME..... UINT
EXP, LOG, LN	RNGE..... USINT	TOLR..... UDINT
NUM..... USINT	RPER..... USINT	TYPE..... USINT
USIN2STR, STR2USI	RPTP..... BOOL	<b>V</b>
<b>O</b>	RSCD..... STRUCT	VALU..... INT
OFF..... DINT	RSLT..... DINT	VAR..... SINT
OFST..... UINT	RVAL..... BOOL	VARS..... STRUCT
OK..... BOOL	<b>S</b>	<b>W</b>
ON..... DINT	SDIR..... BOOL	WEEK..... BOOL
OPTN..... WORD		
ORG..... INT		
OUTs..... (See Table D-2.)		
<b>P</b>		
P..... NUMERIC		
DELETE, INSERT, MID, REPLACE		

**TABLE A-1. IN inputs**

[Inputs for extensible functions are followed with (ext).

Input	Data type	Functions
<b>IN</b>	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
<b>IN0</b>	NUMERIC or TIME	SUB
	any except STRUCT	SEL
<b>IN0 (ext)</b>	any except STRUCT	MUX
<b>IN1</b>	any except BOOL or STRUCT	NE
	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
<b>IN1 (ext)</b>	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
<b>IN2</b>	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
<b>IN2 (ext)</b>	same as IN1	ADD, AND, EQ, GE, GT, LE, MAX, MIN, OR, XOR
	STRING	CONCAT, REPLACE

**TABLE A-2. OUT outputs**

Output	Data type	Functions
OUT	BOOL	CAM_OUT, EQ, GE, GT, LE, LT, NE
	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.

# Appendix B: Errors

When errors occur in PiCPro, they are reported to you in the information window or in a message box.

## Function Errors

---

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: Using I/O functions with an unopened handle <ul style="list-style-type: none"><li>■ Attempting to do a READ on a handle opened for WRITE ONLY</li><li>■ Attempting to do a WRITE to a handle opened for READ ONLY</li></ul>
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"> <li>■ DISK driver detects a checksum error on the DISK. Media error on DISK reading disk FCB (File Control Block).</li> <li>■ A parity, overrun, or framing error exists in the data in the input buffer of the COM port.</li> </ul>
07	<p>write_error</p> <p>Error writing to the device</p>
08	<p>write_protect</p> <p>Reserved for Giddings &amp; Lewis</p>
09	<p>device_error</p> <p>Caused by any of the following:</p> <ul style="list-style-type: none"> <li>■ The Seek origin is not A00, A01, or A02.</li> <li>■ Improper device name (Choices are PICPRO:, RAMDISK:, FMDSK:, or USER:)</li> <li>■ Attempting to do a STATUS on a file</li> <li>■ Attempting to do a file operation on a non-DOS device</li> <li>■ Bad parity in the configuration string or a hardware error in configuring the port</li> <li>■ Attempting to do a second or subsequent operation before the first one was completed</li> </ul>
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:, FMDSK:, 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 &amp; Lewis. Internal GLOS error</p>
14	<p>too_many_connections</p> <p>Reserved for Giddings &amp; 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"> <li>▪ Attempting to CLOSE a file that was not open</li> <li>▪ SEEK has a problem seeking a new location.</li> <li>▪ Attempting to write to a file opened for read only</li> </ul>
19	file_not_in_dir Caused by one of the following: <ul style="list-style-type: none"> <li>▪ Filename not found in directory</li> <li>▪ Volume name not found in directory</li> </ul>
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"> <li>▪ Invalid mode inputted to OPEN</li> <li>▪ Invalid mode inputted to SEEK</li> </ul>
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	<b>Io_dev_aborted</b> 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.

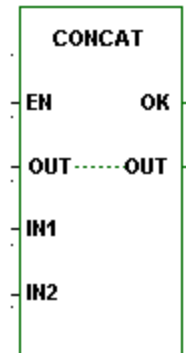


## String Functions Errors

If an error occurs when the string functions execute, the following occurs.

- The output at OK is not energized.
- The STRING variable output will be null (have a length of zero).

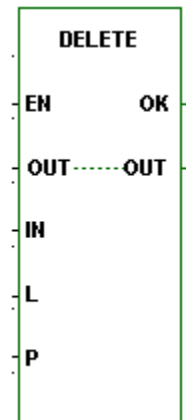
### CONCAT Function



#### 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

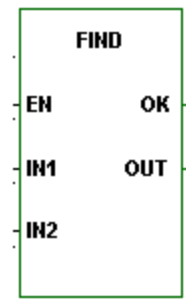
### DELETE Function



#### An error occurs if

P = 0  
 P > 255  
 P > length of IN  
 L > 255  
 The length of IN - L > the length of OUT

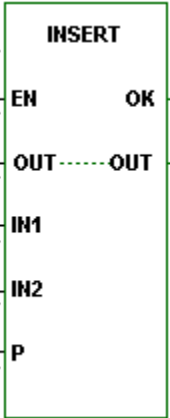
### FIND Function



#### An error occurs if

The length of IN1 = 0  
 The length of IN2 = 0  
 The length of IN2 > the length of IN1

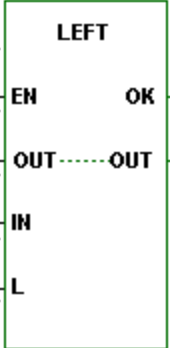
**INSERT Function**



**An error occurs if**

$P = 0$   
 $P > 255$   
 $P > \text{length of IN}$   
 $\text{IN2} = \text{OUT}$   
The length of IN1 + the length of IN2 > the length of OUT

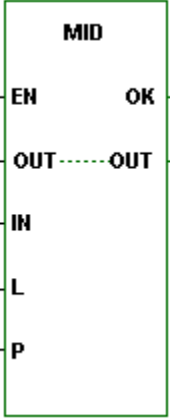
**LEFT Function**



**An error occurs if**

$L > 255$   
 $L > \text{the length of OUT}$

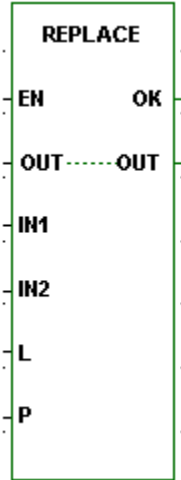
**MID Function**



**An error occurs if**

$P = 0$   
 $P > 255$   
 $P > \text{length of IN}$   
 $L > 255$   
 $L > \text{the length of OUT}$

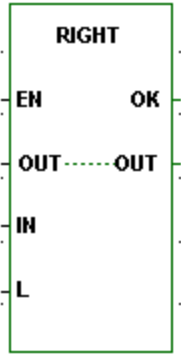
**REPLACE Function**



**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 Errors

There are four types of errors that can occur when working with servo control. The first three apply to individual axes. The fourth, timing error, 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 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	<b>Part reference error</b> Move was in progress when a part reference or a part clear function was called.	8080 (32896)
7	<b>Part reference dimension error</b> When the dimension for the part reference was converted to feedback units, it was too big to fit into 29 bits.	8040 (32832)
6	<b>Distance or position move dimension error</b> When the dimension for the move was converted to feedback units, it was too big to fit into 31 bits.	8020 (32800)
5	<b>Feedrate error**</b> When the feedrate for the move was converted to feedback units per servo up-grade, it was too big to fit into 32 bits or it exceeds 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	<b>Machine reference dimension error</b> When the dimension for the machine reference was converted to feedback units, it was too big to fit into 29 bits.	8008 (32776)
3	<b>User-defined C-stop</b> When this bit is set, a user-defined C-stop has occurred.	8004 (32772)
2	<b>Negative software limit exceeded</b> The command position exceeded the user-defined negative software end limit.	8002 (32770)
1	<b>Positive software limit exceeded</b> 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.

Bit Location (low byte)	Error Description	Hex * Value (decimal)
8, 7	(Not Used)	in LDO-
6	<b>SERCOS error</b>  Cyclic data synchronization error	8020 (32800)
5	<b>SERCOS error</b>  SERCOS drive E-stop - Status word bits 15, 14, and 13 not equal to 1 1 0 respectively.	8010 (32784)
4	<b>User-set</b>  An E-stop on a servo axis has occurred which was called in the ladder using the ESTOP function.	8008 (32776)
3	<b>Overflow error</b>  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	<b>Excess error</b>  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	<b>Loss of feedback</b>  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\_QUEUE 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	<b>The FAST axis in the FAST_QUEUE function moved too far in the wrong direction</b>  The axis traveled more than 65,535 FU in the opposite direction of the value entered in DIST of the FAST_QUEUE function.	8080 (32896)
7	<b>Profile number not found</b>  Data for a profile move is not valid.	8040 (32832)
6	<b>Master axis not available</b>  This error can occur when using the FAST_QUEUE 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	<b>Not Used</b>	---
1	<b>Master start position for lock on</b>  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	<b>The FAST axis in the FAST_QUEUE function moved too far in the wrong direction</b>  The axis traveled more than 65,535 FU in the opposite direction of the value entered in DIST of the FAST_QUEUE function.	8000 (32768)
7, 6, 5	(Not Used)	---
4	<b>Master axis beyond start point</b>  The master axis is beyond its starting point for a ratio synchronization (RATIOSYN) move.	8800 (34816)
3	<b>Slave axis beyond start point</b>  The slave axis is beyond its starting point for a ratio synchronization (RATIOSYN) move.	8400 (33792)

2	<b>Master distance not valid</b>  When the master distance is converted to feedback units, it is greater than 16 bits.	8200 (33280)
1	<b>Slave distance not valid</b>  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 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.

### Name Change

The Windows filename \_ changed to \_ in the PiC.

## Compile Errors

---

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.

### 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.

## 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 of 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.

## Dependency List Errors

---

These errors can occur when building a dependency list.

### **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

---

### **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 editing and recompiling 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 Errors

---

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 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 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 in 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 a Windows 95 manual on the procedure to restore a registry.

#### **2033 COMM: SETTINGS**

In the communications settings dialog box, you can do the following:

Communication Ports - Select the serial port the PiC is connected to.

Baud Rates - Select the transmission rate for communications.

Changing the settings will initiate a power reset in the PiC.

#### **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.

#### **2041 COMM: SET TIME**

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.

#### **An example of set TZ for your CONFIG.SYS**

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.

#### **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 + P 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 + V 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.

## Hardware Errors/Messages

---

There are some errors that can occur and there are some messages that appear in certain dialog boxes when working with hardware declarations. These are described next.

### **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 + P 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 77<sup>th</sup> block module is not an Empty module. Performing any Insert function will remove this module.

#### **4018 HWD: HARDWARE DIALOG**

The hardware declarations define what type of module occupies each slot in a PiC system rack and what type of module occupies a particular block in a block I/O configuration.

The hardware configuration is compared to the software declarations when the module is downloaded. PiCPro can detect errors between hardware declarations and software declarations in addition to validating the declared CPU type.

#### **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 + V 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 software bit memory.

## Compiling Errors

---

These errors can occur when compiling.

### 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.

#### Tip

Enter the correct data type.

### 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.

#### Tip

Delete this label and enter a new label.

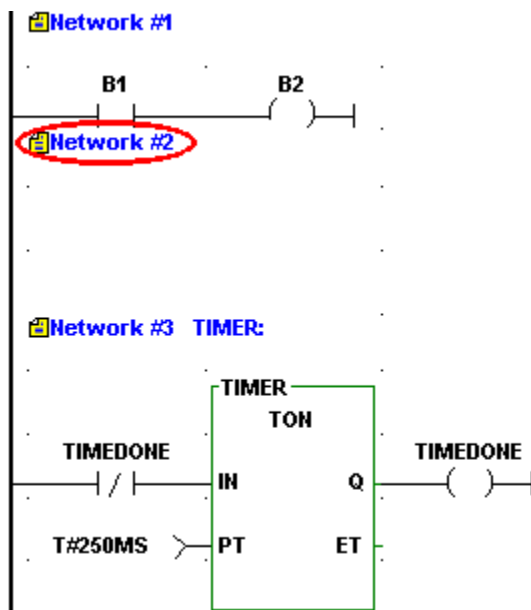
### 5002 COMPILE: EMPTY NETWORK

**Empty network not allowed.**

An empty network is not allowed. Delete the empty network if it is not needed.

#### Example of Error

Network 2 is empty.



#### Tips

Whenever an empty network is included in a ladder, an error message will appear.

- Delete the empty network if it is not required
- Enter ladder logic

#### NOTE

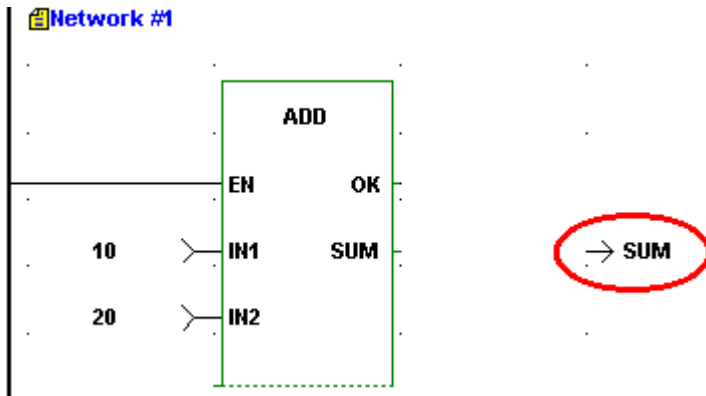
If a network has a label assigned to it but no logic to execute (empty), the label will be ignored and may cause an undefined label error message to appear. If you need the network (i.e. when designing UDFBs), you can prevent this error by simply adding a horizontal wire in the empty network. The error 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.



#### Tips

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

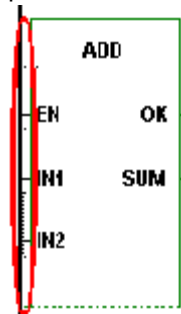
**'Function': Function or Function Block cannot be directly connected to left rail.**

You cannot connect a function/function block directly to the left power rail.

#### Example of Error

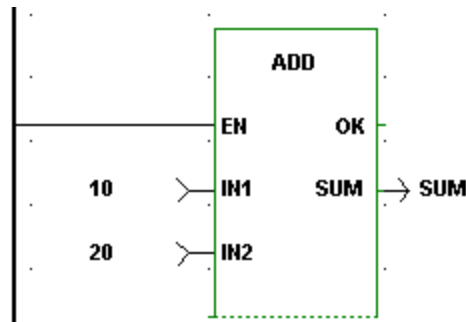
##### Incorrect

Function incorrectly placed at left power rail.



##### Correct

Function correctly positioned.



#### Tip

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.

#### Tip

Whenever you delete a network that contains a task, you must stop the scan and perform a full download.

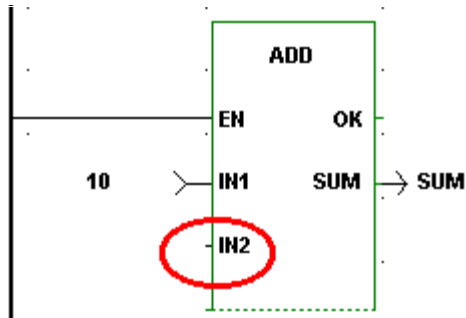
### 5006 COMPILE: NOT ENOUGH INPUTS

**'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.



#### Tip

Ensure that the correct number of inputs have been supplied to the function.

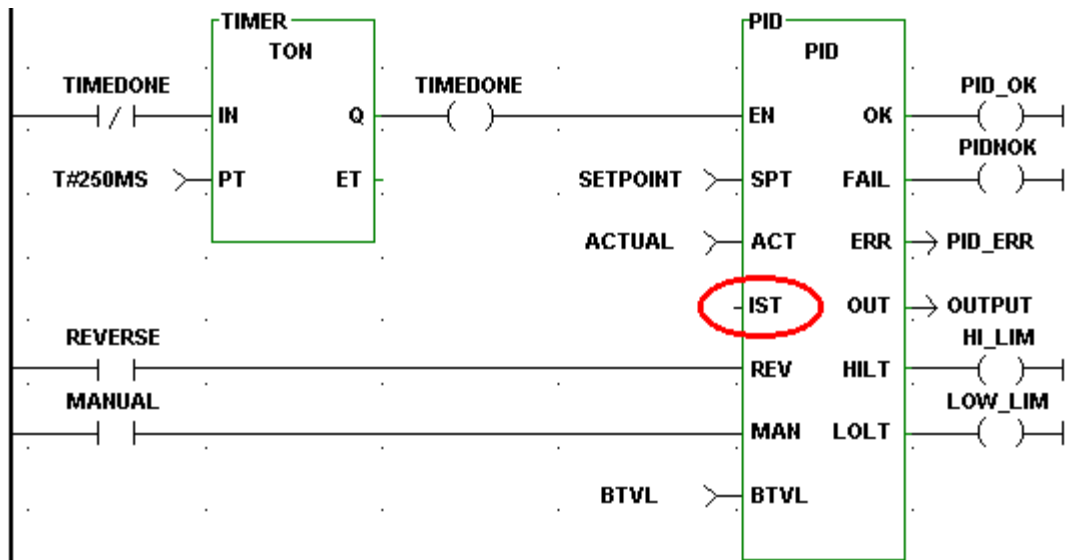
### 5007 COMPILE: INPUT REQUIRED

#### 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.



#### Tip

Enter the required function block input.

### 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.

#### Tip

Whenever you make changes to 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.**

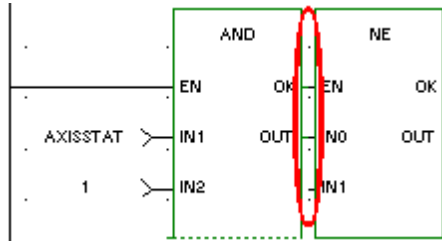
When connecting function/function blocks, you must leave at least one empty column between them and connect the outputs of the first to the inputs of the second with wires or contacts/coils.

### Example of Error

Do not place a second function/function block directly next to an existing one.

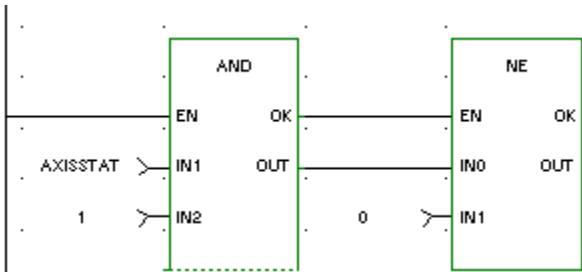
### Incorrect

NE function incorrectly placed directly next to the AND function.



### Correct

NE function correctly positioned and connected.



### Tips

Function/function blocks cannot be directly connected to each other. Leave a column between them and use wires or contacts/coils to make the required connections.

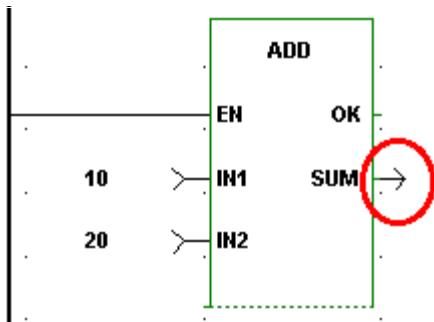
## 5010 COMPILE: VARIABLE REQUIRED

**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.



### Tips

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 Declara-



tions. PiCPro will then prompt you to declare a variable each time you are required to enter one.

### 5011 COMPILE: DATAOUT CONNECTION ERROR

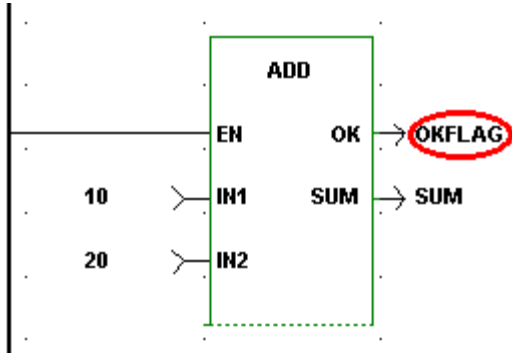
**Connection Error. DATA OUT cannot be connected to first output.**

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

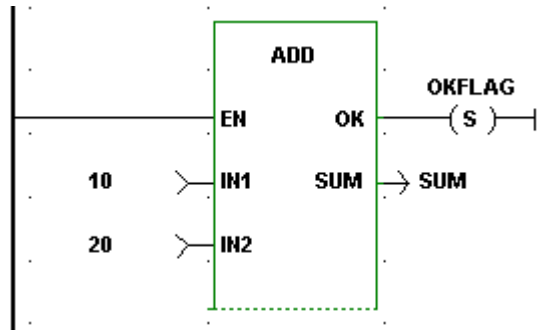
##### Incorrect

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.



#### Tips

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.

#### Tips

- 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.

#### **Tips**

- Enter a valid boolean variable.
- Declare variable as a boolean in the software declarations table.

### 5014 COMPILE: LABEL REQUIRED

#### **Label required.**

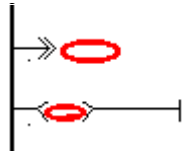
The label of the network you want to jump to with the Jump to Label or Jump to Subroutine commands must be entered with the jump command.

#### **Example of Error**

The Jump to Label and the Jump to Subroutine entries require a label of the network the jump is going to.

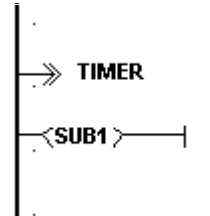
#### **Incorrect**

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.



#### **Tips**

- 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.

#### **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.

### 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.

#### **Tips**

- 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

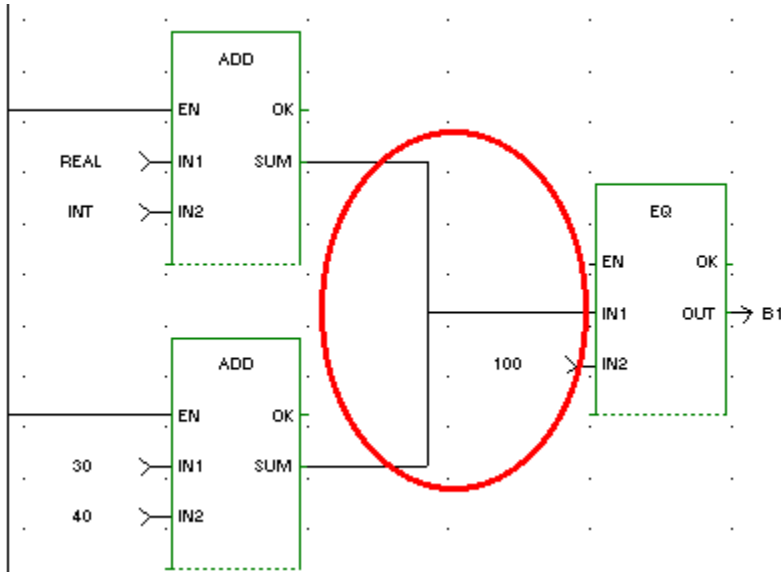
### Invalid branch.

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

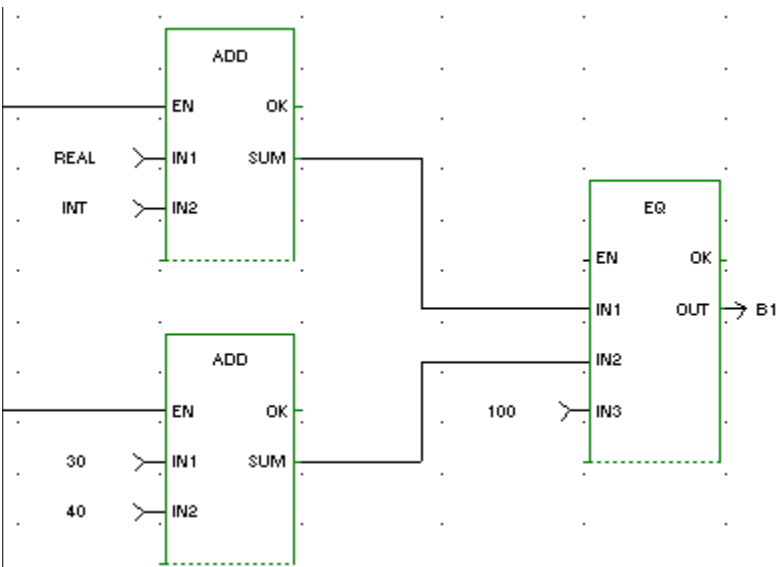
#### Incorrect

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.



### Tip

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.

#### **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.

### **5019 COMPILE: INTERNAL LABEL NOT FOUND**

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.

### **5020 COMPILE: INTERNAL EMPTY LIST**

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.

### **5021 COMPILE: INTERNAL UNKNOWN ELEMENT**

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.

### **5022 COMPILE: INTERNAL NO TEMPLATE**

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.

### **5023 COMPILE: INTERNAL NO DESTINATION**

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.

### **5024 COMPILE: INTERNAL INVALID OUTPUT COUNT**

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.

### **5025 COMPILE: INTERNAL LDO ERROR**

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.

### **5026 COMPILE: INTERNAL INVALID DATA TYPE**

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.

### **5027 COMPILE: NO NETWORKS**

**Ladder does not contain any networks.**

The ladder you are attempting to compile does not contain any networks.

#### **Tips**

- Load a ladder which does contain networks.
- Program network(s) into your ladder and compile again.

### **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.

#### **Tips**

- Use a variable or constant that is a USINT or UINT.
- Change the variable's data type to USINT or UNIT.

## 5029 COMPILE: PARAMETER ERROR

### Parameter error.

The parameter does not meet the necessary requirements for this function or function block.

### Tip

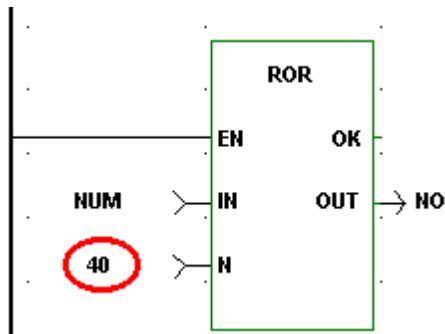
Review the documentation associated with the function/function block you are working with and correct the input.

## 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.



### Tip

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. This is not allowed.

### Tip

Program a task within the main ladder only.

### 5032 COMPILE: TOO MANY TASK INPUTS

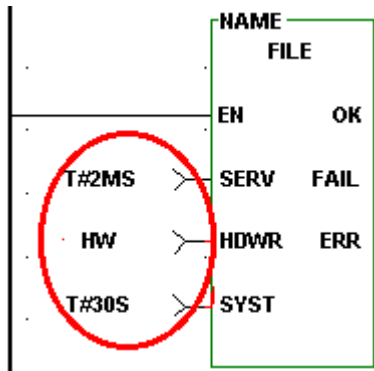
**Too many task inputs specified. Task inputs SERV, HDWR, and SYST are mutually exclusive. Only one may be specified.**

Task inputs SERV, HDWR, and SYST are mutually exclusive. You may specify only one of these inputs for a task.

#### Example of Error

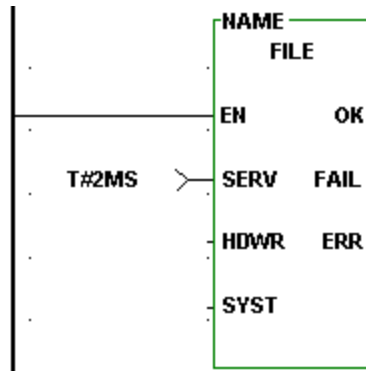
##### Incorrect

All three TASK inputs (SERV, HDWR, and SYST) have been entered.



##### Correct

Only one TASK input has been entered.



#### Tip

Select only one of the three inputs for a task function block.

### 5033 COMPILE: TASK INPUT REQUIRED

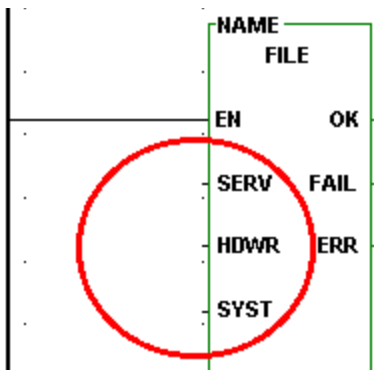
**An input is required at one of the task inputs.**

You have not connected any input at any of the task inputs of SERV, HDWR, or SYST.

#### Example of Error

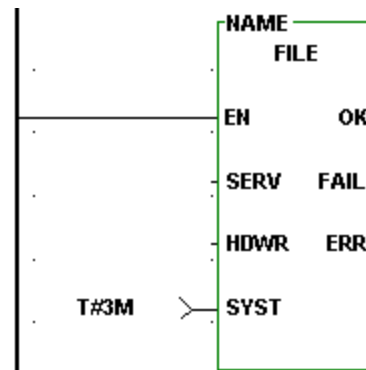
##### Incorrect

None of the three TASK inputs (SERV, HDWR, and SYST) has been entered.



##### Correct

One TASK input must be entered.



#### Tip

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

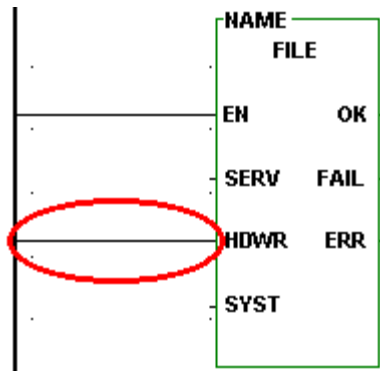
#### Input must be DATA IN or DATA INVERTED.

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

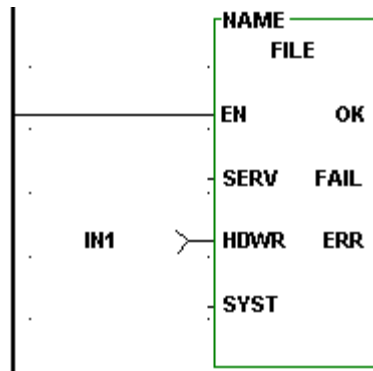
##### Incorrect

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.



#### Tip

- 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

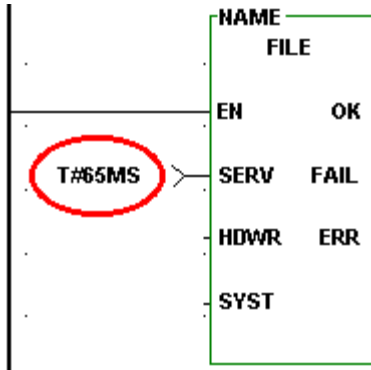
**Input must be one of the following constants:**  
T#1MS, T#2MS, T#4MS, T#8MS or T#16MS.

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

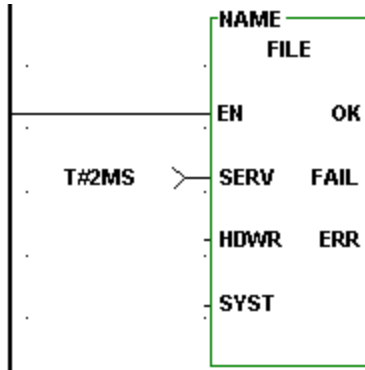
##### Incorrect

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.



#### Tip

Enter #1MS, T#2MS, T#4MS, T#8MS or T#16MS 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.

#### Tip

Only enter 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.

#### Tip

Enter an 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.

#### Tip

Enter any boolean variable instead of a constant.

**Parameter error.**

The parameter does not meet the necessary requirements for this function or function block.

#### Tip

Review the documentation associated with the function/function block you are working with and correct the input.

## 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.

#### **Tips**

- 1 Declare a CPU with an NPX processor in the Hardware Declarations table.
- 2 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.

#### **Tips**

- 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 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.

#### **Tips**

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.

#### **Tips**

- 1 Declare a CPU with an NPX processor in the Hardware Declarations table.
- 2 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.

#### **Tip**

Always perform a full module download with the scan stopped when you add a new member to a structure.

## 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.

#### **Tip**

Always perform a full module download with the scan stopped when you modify an existing member of a structure.

#### 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.

##### Tip

Always perform a full module download with the scan stopped when you modify a function block.

#### 9005 COMPILE: INVALID ATTRIBUTE

**Invalid symbol attribute in '\_\_\_'.**

There is an invalid symbol attribute in the software declarations table.

##### Tip

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.

##### Tip

Always perform a full module download with the scan stopped when you modify an existing variable.

#### 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.

##### Tip

Always perform a full module download with the scan stopped when you enter a new variable with the retained attribute.

#### 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.

##### Tips

Whenever a change is made that requires initialization, you must:

- 1 Recompile the function block.
- 2 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.

##### Tip

After adding a new or moving an existing input or output, choose Compile, Bin File, Compile & Download from the .

#### 9010 COMPILE: I/O NOT AN INPUT

**'\_\_\_\_\_' calls for an input point '\_\_\_\_\_' in rack '\_\_\_\_\_' slot '\_\_\_\_\_' which is not configured as an input point.**

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is downloaded. PiCPro will detect an error if you attempt to define an input point at an output location.

##### Tip

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 as an output point.**

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is downloaded. PiCPro will detect an error if you attempt to define an output point at an input location.

#### **Tip**

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.

#### **Tip**

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.**

#### **Tip**

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.

#### **Tip**

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.

#### **Tip**

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.

#### **Tip**

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.

#### **Tip**

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.

#### **Tip**

Do not enter a function block as an input to a UDFB.

### **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.

#### **Tip**

Do not enter a function block, structure, array, or string as an output to a UDFB.

## 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.

### Tip

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.

### Tip

When creating an UDFB, ensure in the software declarations table that the first input:

- 1 Is a boolean.
- 2 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.

### Tip

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.

### Tip

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, or block I/O as listed below.

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

## MMC I/O Point Labels

PiCPro for Windows MMC Edition allows you to enter the I/O information in the software table for the general I/O on the CPU section and the servo I/O on the analog section. The information can be copied by:

- Selecting the CPU (Slot 2) or Analog (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 CPU section and for the axis and auxiliary connectors on the analog module. The information in column 2 Declared Names will be pasted in the Name column and the information in column 3 MMC I/O point will be pasted in the I/O Point column of the software declarations table.

Discrete MMC Point	Declared Name	MMC I/O Point	PIC I/O Point*
16 general DC inputs	GENI - GENI 16	IGEN.1 - IGEN.16	I3.1 - I3.16
16 general DC outputs	GEN01 - GEN016	OGEN.1 - OGEN.16	O3.1 - O3.6
2 short circuit inputs	SHORT1 - SHORT 2	ISGEN.1 - ISGEN.2	I3.17 - I3.18
6/12 aux DC inputs	AUXI1 - AUXI12	IAUX.1 - IAUX.12	I4.1 - I4.12
Axis 1 DC input	AX1READY	IA1.1 (Axis 1, Input 1)	I4.13
Axis 2 DC input	AX2READY	IA2.1	I4.14
Axis 3 DC input	AX3READY	IA3.1	I4.15
Axis 4 DC input	AX4READY	IA4.1	I4.16
Axis 1 DC output	AX1ENABL	OA1.1 (Axis 1, Output 1)	O4.1
Axis 2 DC output	AX2ENABL	OA2.1	O4.2
Axis 3 DC output	AX3ENABL	OA3.1	O4.3
Axis 4 DC output	AX4ENABL	OA4.1	O4.4
Axis 1 DC output	AX1RESET	OA1.2 (Axis 1, Output2)	O4.5
Axis 2 DC output	AX2RESET	OA2.2	O4.6
Axis 3 DC output	AX3RESET	OA3.2	O4.7
Axis 4 DC output	AX4RESET	OA4.2	O4.8
Axis 1 fast input	AX1FINPT	IFAUX.1 (aux port, A1, F1)	I4.17
Axis 2 fast input	AX2FINPT	IFAUX.2	I4.18
Axis 3 fast input	AX3FINPT	IFAUX.3	I4.19

Axis 4 fast input	AX4FINPT	IFAUX.4	I4.20
Axis 49 fast input	DIGFINPT	IFAUX.49 (aux port, D49,Fi	I4.21

\* The PiC I/O point column is what will appear in the software declarations table if you change your CPU from an MMC to a PiC.

### Master Rack MMC CPU

1 <sup>st</sup>	I or O	Input or Output
2 <sup>nd</sup>	Connector /Type	GEN, AUX, A1, A2, A3, A4, FAUX
	. (point)	Used as a separator
	First digit of channel number	
	Second digit of channel number	

### 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 an MMC CPU.

Enter five to eight characters.

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

### Block Expansion Rack I/O

**Note:** Block Expansion Rack I/O is only available in PiCPro for MMC or if MMC CPU is chosen in PiCPro for Windows. Block Expansion Rack I/O is not available if PiCPro CPU is chosen in PiCPro for Windows.

Enter five to seven characters.

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



7<sup>th</sup>      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**

<u>Name</u>	<u>Type</u>	<u>I/O Point</u>
OUT1	BOOL	O4.1
OUT2	BOOL	O4.2
.	.	.
.	.	.
OUT16	BOOL	O4.16
FB1	BOOL	I4.1
FB2	BOOL	I4.2
FB3	BOOL	I4.3
FB4	BOOL	I4.4

**The 24V I/O 16/8 source  
module with two fuses**

<u>Name</u>	<u>Type</u>	<u>I/O Point</u>
IN1	BOOL	I5.1
IN2	BOOL	I5.2
.	.	.
.	.	.
IN16	BOOL	I5.16
OUT1	BOOL	O5.1
OUT2	BOOL	O5.2
.	.	.
OUT8	BOOL	O5.8
FB1	BOOL	I5.17
FB2	BOOL	I5.18

## Fast Inputs

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**  
X.1

**For Channel 2**  
X.3

**For Channel 3**  
X.5

**For Channel 4**  
X.7

IFAUX.1  
IFAUX.2  
IFAUX.3  
IFAUX.4  
IFAUX.49

Channel 1  
Channel 2  
Channel 3  
Channel 4  
Channel 5

(AXIS 1)  
(AXIS 2)  
(AXIS 3)  
(AXIS 4)  
(AXIS 49)

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

**For Channel 1**  
X.1

**For Channel 2**  
X.3

**For Channel 3**  
X.5

**For Channel 4**  
X.7

**For Channel 5**  
X.9

### **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 TOD#23:59:59
TIME_OF_DAY	TOD#00:00:00	
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.

### Tip

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.

#### Tip

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.

#### Tip

When creating a UDFB, ensure in the software declarations table that the first output:

- 1 Is a boolean.
- 2 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.

#### Tip

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 as an input point.**

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is downloaded. PiCPro will detect an error if you attempt to define an input point at an output location.

#### Tip

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 as an output point.**

The hardware configuration defined by the hardware declarations table is compared to the software configuration when a module is downloaded. PiCPro will detect an error if you attempt to define an output point at an input location.

#### Tip

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.

#### Tip

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.

## Fieldbus Errors

---

### **9071 FIELDBUS: LIBRARY NOT AVAILABLE**

Check that you have indicated the correct directory for the libraries. Select File from the main menu and then select PICPro Libraries.

### **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 editing 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 "Polled Output" (0X2). 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 for 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 Errors

---

### 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.

$$\frac{\text{software upper limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of -536870912 to 536870911 FU)}$$

#### **Tip**

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.

$$\frac{\text{software lower limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of -536870912 to 536870911 FU)}$$

#### **Tip**

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.

$$\frac{\text{excess error limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of 0 to 536870911 FU)}$$

**Tip**

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.

$$\frac{\text{in position band} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of 0 to 536870911 FU)}$$

**Tip**

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.

$$\frac{\text{rollover position} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of 0 to 536870911 FU)}$$

**Tip**

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.

$$\frac{\text{plus integral error limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of 0 to 536870911 FU)}$$

**Tip**

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.

$$\frac{\text{minus integral error limit} * \text{FU}}{\text{LU}} = N \text{ (where N must be within range of -536870912 to 0 FU)}$$

#### **Tip**

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.

$$\frac{\text{velocity limit} * \text{FU} * 8 * \text{update rate}}{\text{LU}} = N \text{ (where N must be within range of 0 to 32767 FU/iteration)}$$

#### **Tip**

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

## 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.

$$\frac{\text{integral gain value} * 3253 * \text{update rate}}{\text{FU/min-volt}} = N \text{ (where N must be within range of 0 to 32767 )}$$

#### **Tip**

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.

## 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.

$$\frac{\text{proportional gain value} * 762601}{\text{FU/min-volt}} = N \text{ (where N must be within range of 0 to 32767 FU)}$$

#### **Tip**

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.

$$\frac{\text{derivative gain value} * 178734}{\text{FU/min-volt} * \text{update rate}} = N \text{ (where N must be within range of 0 to 32767 FU)}$$

#### **Tip**

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.

### 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.

$$\frac{\text{feed forward percent} * 457560436}{\text{FU/min-volt} * \text{update rate}} = N \text{ (where N must be within range of 0 to 524272)}$$

#### **Tip**

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

### 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.

$$\frac{\text{cstop ramp} * \text{FU} * \text{update rate} * \text{update rate}}{\text{LU} * 937500} = N \text{ (where N must be within range of 1 to 65535 FU)}$$

#### **Tip**

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.

### 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.

$$\frac{\text{acceleration ramp} * \text{FU} * \text{update rate} * \text{update rate}}{\text{LU} * 937500} = N \text{ (where N must be within range of 1 to 65535 FU)}$$

#### **Tip**

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.

### 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.

$$\frac{\text{deceleration ramp} * \text{FU} * \text{update rate} * \text{update rate}}{\text{LU} * 937500} = N \text{ (where N must be within range of 1 to 65535 FU)}$$

#### Tip

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.

### 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.

$$65535 * \frac{\text{update rate}}{\text{slow filter} * (1 - e^{-\text{slow filter}}}) = N \text{ (where N must be within range of 0 to 65535 FU)}$$

#### Tip

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

### 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.

$$65535 * \frac{\text{update rate}}{\text{fast filter} * (1 - e^{-\text{fast filter}}}) = N \text{ (where N must be within range of 0 to 65535 FU)}$$

#### Tip

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

### 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.

$$\frac{\text{velocity threshold} * \text{FU} * \text{update rate}}{\text{LU} * 60000} = N \text{ (where N must be within range of 1 to 65535 FU)}$$

**Tip**

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.

### 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.

$$\frac{\text{velocity threshold} * \text{FU} * \text{update rate}}{\frac{\text{update rate}}{\text{slow filter}}} \text{ LU} * 65535 * (1 - e^{-\text{slow filter} * \text{update rate}}) = N \text{ (where N must be within range of 0 to 65535)}$$

**Tip**

Adjust the axis data information so that the result of the conversion calculation will fall within the acceptable range.

### 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.

**Tip**

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.

**Tip**

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.

**Tip**

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.

### 11097 CONFIRM FROM PIC TO MMC

Check that you want to cut the

**11101 AXIS: PASTE DUP CONVER2MMC SERVO**

This message is displayed when attempting to paste a PiC servo axis into an MMC servo setup file where there is already another servo axis with the same axis number.

**11103 AXIS: PASTE DUP CONVER2PIC SERVO**

This message is displayed when attempting to paste a MMC servo axis into a PiC servo setup file where there is already another servo axis with the same axis number.

**11104 AXIS: PASTE DUP CONVER2PIC DIGIT**

This message is displayed when attempting to paste an MMC servo axis into a PiC servo setup file where there is already another servo axis with the same axis number.

**11105 AXIS: PASTE DUP CONVER2MMC SERVO**

This message is displayed when attempting to paste a PiC servo axis into a PiC servo setup file where there isn't another servo axis with the same axis number.

**11106 AXIS: PASTE DUP CONVER2PIC SERVO**

This message is displayed when attempting to paste a PiC digitizing axis into a PiC servo setup file where there isn't another servo axis with the same axis number.

**11107 AXIS: PASTE DUP CONVER2MMC DIGIT**

This message is displayed when attempting to paste a PiC digitizing axis into a PiC servo setup file where there isn't another digitizing axis with the same axis number.

**11108 AXIS: PASTE DUP CONVER2PIC DIGIT**

This message is displayed when attempting to paste an MMC digitizing axis into a PiC servo setup file where there isn't another digitizing axis with the same axis number.

**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.

## Force List Errors

---

### **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.

## SERCOS Errors

---

This section covers errors that can occur when working with SERCOS features.

### 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"><li>▪ SERCOS board in correct slot</li><li>▪ SR structure members correct</li></ul>
17	The SERCOS module did not receive an expected AT response. Cable could be disconnected.	<ul style="list-style-type: none"><li>▪ Check connection</li></ul>
20	Phase 0 detected that the ring is not complete.	<ul style="list-style-type: none"><li>▪ Check connection</li><li>▪ Ensure drive is turned on</li></ul>
65	Error occurred calculating when MDT should occur.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ One or more drives cannot accommodate required MDT</li></ul>
66	Error occurred calculating when drive data valid.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ One or more drives cannot accommodate command times</li></ul>
67	Error occurred calculating when feedback data valid.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ One or more drives cannot accommodate feedback capture times</li></ul>
68	Error occurred calculating total time required for communication cycle	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ Cyclic data on slaves too long</li><li>▪ Update rate too fast</li></ul>
69	Error occurred calculating cyclic data memory for SERCON processor.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ Cyclic data on slaves too long</li></ul>
70	Error occurred calculating cyclic data memory for internal memory map.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ Cyclic data on slaves too long</li></ul>
71	Error occurred calculating service channel memory map.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ Cyclic data on slaves too long</li></ul>
74	CPU on SERCOS module has too many tasks during update. Too many slaves on one ring.	<ul style="list-style-type: none"><li>▪ Too many slaves on one ring</li><li>▪ Cyclic data on slaves too long</li></ul>



128	Slave error occurred. Read SERR output to identify error. The SLV output indicates the slave number.	<ul style="list-style-type: none"> <li>▪ SLV output contains slave number</li> <li>▪ IDN output contains the IDN transfer that caused the error</li> <li>▪ SERR output contains the drive generated error number</li> <li>▪ Read Drive diagnostic IDN 95</li> </ul>
136	Individual slave will not respond. The SLV output indicates the slave number.	<ul style="list-style-type: none"> <li>▪ Address switch on drive does not match slave number</li> <li>▪ Baud rate switch on drive does not match rate in ring definition</li> <li>▪ SLV output contains slave number that does not respond</li> </ul>
144	Individual slave cannot carry out a Procedure Command Function. The SLV output indicates the slave number.	<ul style="list-style-type: none"> <li>▪ SLV output contains slave number</li> <li>▪ IDN output contains the Procedure Command Function that caused the error</li> <li>▪ For IDN = 127, read IDN 22 to read list of IDNs still required by the drive</li> <li>▪ For IDN = 128, read IDN 23 to read list of IDNs still required by the drive</li> <li>▪ Read Drive diagnostic IDN 95</li> </ul>

## 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.

### **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 SERVOS: 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/Port \_\_\_\_\_. Define slaves in SERCOS setup.

### **12010 SERCOS: DUPLICATE**

There is a duplicate slot/port definition at \_\_\_\_\_. Slot/port 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.

# Appendix C - PiCPro Reference Card - Errors/Variables

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 STATUSSV 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 dimension 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
1	Actual position	S, T, D	T
2	Move type 11 pos 18 ratiopro 12 dist 20 ratiosyn/gr 14 vel st 22 ratiocam 16 lad ref or 23 ratioslp fast ref 24 ratio real	S	
3	Command position	S, D	
4	Position error	S	
5	Slow velocity filter error	S	
6	Command velocity	S, T	T
7	Position change	S, D	
8	Feedback last	S, D	
9	Fast input position	S, D	
10	Reg/ref position change	S, D	
11	Consecutive bad marks	S, D	S, D
12	Rollover on position	S, T, D	S, T, D
13	Slave offset incremental	S	S
14	Master offset incremental	S	S
15	Slave offset absolute	S	S
16	Master offset absolute	S	S
17	Slave offset filter		S
18	Master offset filter		S
19	Fast input direction		S, D
20	Fast input distance	S, D	
21	Reversal not allowed	S	S
22	Fast input position (software)	S, D	
23	Position (software) of axis 1 with fast input on axis 2	S, D	S, D
24	Registration switch	S, D	S, D

V#	Variable Description	Read	Write
25	Fast queuing	S	S
26	Synchronized slave start	S, D, T	S, D, T
27	Backlash compensation	S	S
28	TTL feedback	S, D	S, D
29	Reference switch position	S	
30	Filter time constant	S	S
31	Filter error limit	S	S
32	Velocity compensation flag	S	S
33	Filter lag	S	
34	Position change over several interrupts	S, D	S, D
35	Part reference offset	S, D	
36	Software upper limit	S	S
37	Software lower limit	S	S
38	Commanded position (before slow velocity filter)	S, D	
39	Following error limit	S	S
40	In-position band	S	S
41	Current segment number	S	
42	Slave distance into segment	S	
43	Master distance into segment	S	
44	Set user iteration command	S	S
45	User iteration command	S	S
46	Set user PID command	S	S
47	User PID command	S	S
48	Disable servo software	S	S
49	(Reserved)		
50	Override endlimit check	S	S
51	Write SERCOS position	S	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 transistion 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.
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.
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.

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 and descriptions for the 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

## Card

#	Profile Commands	Range
1	Distance move	$\pm 2,147,352,575$ steps
2	Position move	$\pm 2,147,352,575$ steps
3	Velocity move	$\pm 1,000,000$ 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	$\pm 2,147,352,575$ 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 PiC900 is running, the DIAG LED on the CPU module 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.
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 <b>3</b> times and the third digit flashes <b>10</b> times, the module is <b>30</b> .

\* Errors connected with I/O expansion. Refer to the I/O Driver Module write-up in the 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	ß	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	ı	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	ı	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	ı	225	0xE1	·
130	0x82	é	162	0xA2	ó	194	0xC2	¬	226	0xE2	,
131	0x83	â	163	0xA3	ú	195	0xC3	Ð	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	å	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	ı	200	0xC8	»	232	0xE8	Ë
137	0x89	ë	169	0xA9	©	201	0xC9	...	233	0xE9	Ë
138	0x8A	è	170	0xAA	™	202	0xCA	þ	234	0xEA	¾
139	0x8B	ï	171	0xAB	´	203	0xCB	À	235	0xEB	Î
140	0x8C	î	172	0xAC	¨	204	0xCC	Ã	236	0xEC	×
141	0x8D	ì	173	0xAD	ı	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	ÿ	249	0xF9	•
154	0x9A	Ü	186	0xBA	½	218	0xDA	/	250	0xFA	
155	0x9B	ç	187	0xBB	ª	219	0xDB	¤	251	0xFB	Ð
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	
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.

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` function 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. The variable is in ladder units.

Range: +2,147,483,647 to -2,147,836,648 ladder units

### Controlling Time Axis Velocity

You can use either the `S_CURVE` function or the command velocity variable 6 to move a time axis.

`S_CURVE` Function

When using the `S_CURVE` function with a time axis, you can use the distance, position, or velocity moves to move the axis. The `S_CURVE` function must be called first when using these moves. See the `S_CURVE` description in the Function/Function Block Reference Guide.

### Command Velocity (Variable 6)

If you are not using the `S_CURVE` function, the command velocity variable can be used to define how fast the time axis will travel. It is programmed in ladder units per second. When the `WRITE_SV` function is called with variable 6, the time axis will step to the programmed velocity.

Each ladder unit of the time axis travel represents the portion of time programmed by this variable. For example, if a value of 1000 is programmed as the number of ladder units per second for the velocity, then the time axis would move one ladder unit 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. In effect, you have stopped time. 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 ladder units/sec

### **Rollover on Position with a Time Axis**

The rollover on position variable allows you to select where the time 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 counts continues to zero and back up to 2,147,483,647, etc.

Range: 1 to 536,870,912 ladder units (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, define the servo function. This function will be stored in the servo library and can then be called into 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.

<b>Not Available with Stepper</b>	<b>Available with Stepper</b>
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 toolbars available in PiCPro.

- Standard
- Basic Online Operations
- Advanced Operations
- Ladder
- View Navigator
- Functions
- Compiler











The PiCPro toolbars each have a set of tool buttons on the bar. They are listed here as a reference.

## 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
	Warm Restart
	Cold Restart
	Backup User Program
	Restore User Program
	Reset Power
	Set Node ID
	Connect to Node
	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                  |

## 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 variable 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

## NOTES



## Appendix K - G&L DeviceNet Configuration Tool

The diagram below illustrates a typical DeviceNet set up.

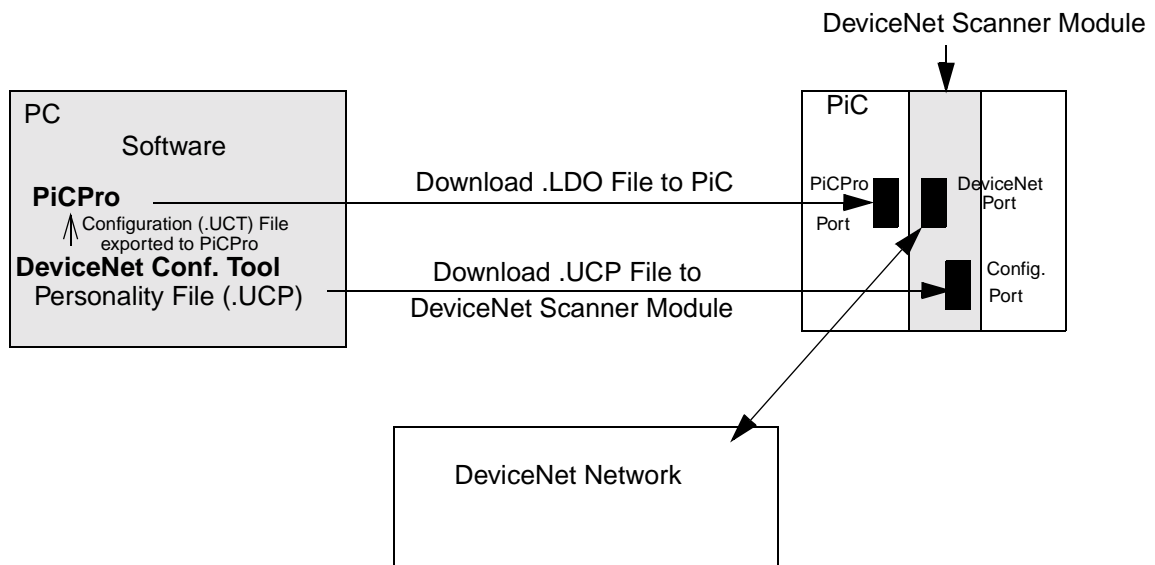
### Overview of setting up a DeviceNet Network

---

1. Use the G&L DeviceNet Configuration Tool to provide information about your DeviceNet system. (See procedure that follows this section.)
2. Always use the “export” command with this information so that
  - a. your PC will have a current configuration file (.UCT) for PiCPro.
  - b. the DeviceNet scanner module will have a current personality file (.UCP).
3. Download the personality file to the DeviceNet scanner module using the PiCPro cable connected to the Configuration Port.\*
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. Connect the DeviceNet cable to the DeviceNet Port on the DeviceNet scanner module.

\*NOTE: The PiCPro cable is first used to download the personality file by connecting it to the Configuration Port on the DeviceNet scanner module. You then need to connect it to the PiCPro Port on the CPU module in order to download the ladder.

**FIGURE K- 1. Block Diagram of DeviceNet Setup**



## Procedure for using the G&L DeviceNet Configuration Tool

---

There are several ways to accomplish the following. Once you become familiar with the operations, you may choose the method you prefer.

1. Select the G&L DeviceNet Configuration Tool icon to get to the application window.
2. Select File, New 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 name of your .LDO file.
3. Select The Network (Network Configuration) symbol and double click to bring up the Node Properties 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.  
Exit this menu.
4. Right click to bring up another menu and select New. This will display the DeviceNet scanner module. Right click on this symbol to bring up a menu and select Properties. A menu with three tabs appears. Enter appropriate information.

### **General TAB**

Enter a name for the scanner module or leave it as the default name Scanner1.

Check “Use Net Config” 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 presently used for the G&L DeviceNet scanner mode)

### **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 New. This brings up a Select Device menu.  
Select Basic Device and select OK to enter a Basic Device or node symbol.  
Right click on this to bring up a menu and select Properties.  
A menu with three tabs appears.

### **Identity TAB**

Assign a “Name” and a “Description”.

Select a MAC ID.

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 associated with the node.  
(COS and Cyclic are not supported yet.)

Select Apply.

**Tag TAB**

Select New and then Edit. 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.

**Tag TAB**

Select the appropriate I/O Type and select a Data Type to match the data type for the 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 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 New and then Edit. 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 appendix.
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 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 Tool when downloading the personality file to the DeviceNet scanner module.
9. 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 connected the PiCPro cable between your PC and the configuration port on the DeviceNet scanner module. Select the Download button and wait for the operation to be completed.
10. Return to the application screen and save your file before exiting.

## Device Status Word

A 16-bit device status word for each slave device can optionally be tagged. It 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 table below)
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)

## Appendix L - G&L Ethernet - TCP/IP Configuration Tool

The diagram below illustrates a typical Ethernet - TCP/IP set up.

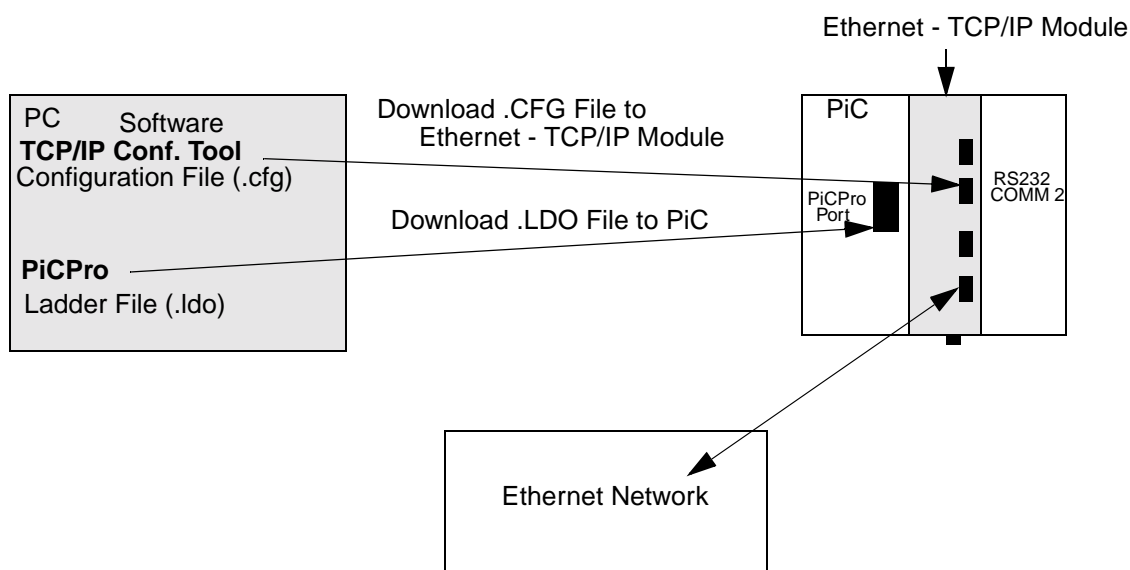
### Overview of setting up a Ethernet Network

---

1. Use the G&L TCP/IP Configuration Tool 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.

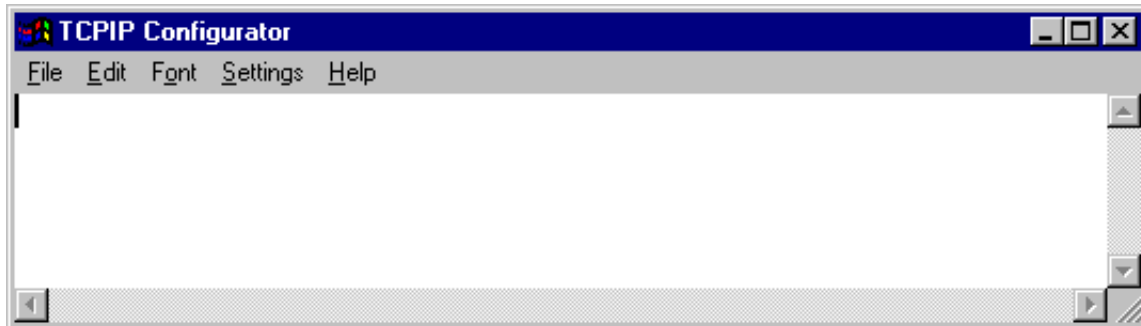
\*NOTE: The PiCPro cable is first used to download the configuration file by connecting it to the RS232 Come 2 port on the Ethernet - TCP/IP module. You then need to connect it to the PiCPro Port on the CPU module in order to download the ladder.

**FIGURE L- 1. Block Diagram of Ethernet - TCP/IP Setup**

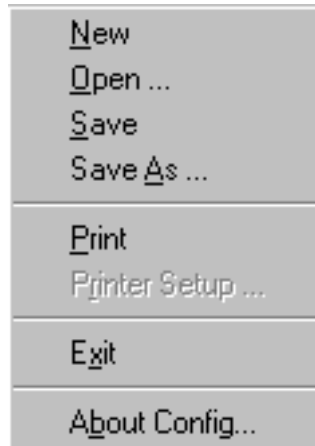


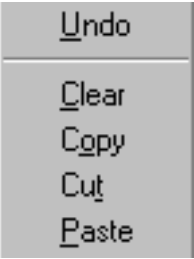

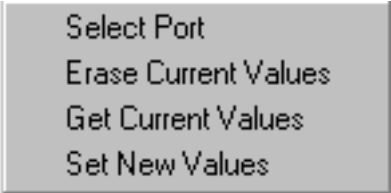

## Procedure for using the G&L TCP/IP Configuration Tool

The TCP/IP configuration tool is installed from the PiCPro CD. Double click on the configtcp.exe file to start the configurator. The configurator opens with the following menu bar displayed.



Under each menu bar item, the menus in the table below drop down. The commands found in these drop down menus are explained briefly here.

<b>File</b>  The New command allows you to open a new configuration file.  The Open command allows you to open an existing configuration file.  The Save command saves your file and any changes you may have made to it.  The Save As command allows you to choose a name and/or an extension for the file you want to save.  The Print commands allows you to print your configuration file.  The Printer Setup command allows you to change your printer configuration.  The Exit command allows you to leave the TCP/IP configuration program without saving any changes.  About Config... tells you what version of the configurator you are running.	
--	---

<p><b>Edit</b></p> <p>The Undo command allows you to undo your last edit.</p> <p>The Clear command removes the selected item from the file.</p> <p>The Copy command allows you to copy the selected item to the clipboard.</p> <p>The Cut command allows you to remove the selected item to the clipboard.</p> <p>The Paste command allows you to paste whatever is on the clipboard to the location after the cursor.</p>	
<p><b>Font</b></p> <p>You can change the font type, style, and size of the configuration data displayed on the screen. NOTE: This change cannot be saved.</p>	
<p><b>Settings</b></p> <p>The Select Port command brings up the <i>Choose a COM port</i> box that allows you to designate the communication port (typically, COM2) and the baud rate between 9600 to 57600). NOTE: Currently, must be set at 9600.</p> <p>The Erase Current Values command allows you to remove the values residing on the TCP/IP module.</p> <p>The Get Current Values command allows you to read the values that are currently residing on the TCP/IP module.</p> <p>The Set New Values allows you to send the values you have entered in your configuration file.</p>	
<p><b>Help</b></p> <p>The Help command brings up help information.</p>	

## Procedure for Creating a Configuration File

To set up a configuration file for your application, follow these steps.

1. Double click on the configtcp.exe icon.
2. Enter the configuration strings for your application as explained below.

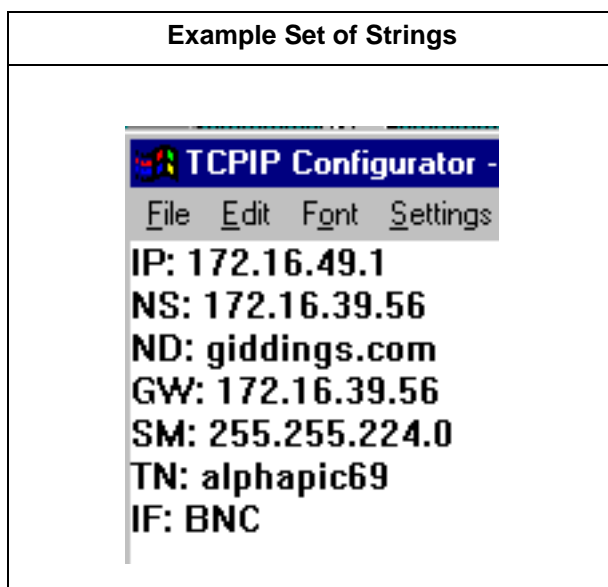
Each configuration string has two parts separated by a space:

1. an ID consisting of two uppercase characters and a colon
2. the string itself

The IDs and their definitions are found in the table below.

ID	Description
IP:	Address of this host in dotted decimal notation
SM:	Subnet mask in dotted decimal notation
IF:	The ethernet port to use (BNC, AUI, or UTP)
GW:	Gateway node by name (if nameserver is present) or by address (dotted decimal notation)
TN:	Name of this host
NS:	Address of domain nameserver in dotted decimal notation
ND:	Domain name of this host

Below is an example of the configuration strings entered into the configuration tool. Note that the order of entry does not matter.



3. After all the information is entered, choose the Save As command to name and save your file. You may add an extension to the file (for example, *.cfg*). It is not required.



4. Connect the PiCPro cable between the your PC and the COM2 port on the Ethernet - TCP/IP module.
5. With the TCP/IP Configurator running, choose Settings, Set New Values from the menu. This will send your configuration data to the Ethernet - TCP/IP module.
6. Cycle power.

## 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:

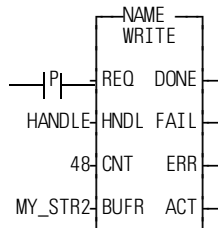
1. Consider a ladder with a structure declared as:

```

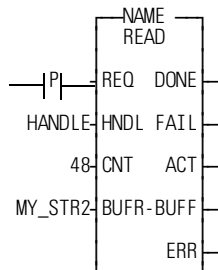
MY_STR2  STRUCT
.NAMES   STRING[10](0..3)
END_STRUCT

```

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)*

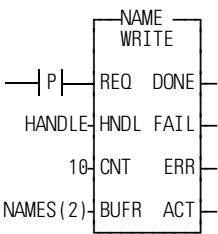
BYTES of the STRING itself. (*In this example 10 Bytes. Even if the actual number of characters in the STRING is less than the Declared Length, **the STRING must be padded up to the declared length.** You can use any character you choose to pad the STRING.*)

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'.

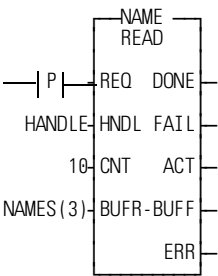
2. Consider a STRING declared as:

```
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 2-84
- acceleration ramp 3-35
- advanced operations toolbar I-2
- analog input setup 3-22
- analog output offset 3-38
- animating the ladder 2-70
- animation
  - color 1-7
- application note
  - reading strings from a structure AN-3
- arrays 2-45
  - entering 2-46
- ASCII
  - chart F-1
- AT - Amplifier Telegram 4-4
- attribute 2-47
  - external 2-47
  - global and retentive 2-47
  - retentive 2-47
  - variable in/out 2-48
- axis tuning 3-40
  - using forcing 3-40
  - using viewing 3-41
  - variables 3-41

## B

- basic online operations toolbar I-2
- battery box 4-28
  - application note 4-29
- Bin file 2-64
- block expansion rack I/O
  - numbering 2-37, B-44
- block I/O 2-27, 2-28
- blown fuse status 2-38, B-46
- build a dependency list 2-87

## C

- cable 2-78
- calling Technical Support 1-2
- Clear application memory 2-6
- closing files 2-17
- codes
  - diagnostic E-1
- coil 2-56
- cold restart 2-81
- comments 2-61
- communications errors B-17
- comparing to cyclic data 4-7

- comparing to service channel 4-7
- comparing UDFBs and TASKs 5-1
- compiler toolbar I-4
- compiling
  - Bin file 2-64
  - errors B-12, B-25
  - Hex file 2-65
  - LDO 2-63
  - Task 2-67
  - UDFB 2-68, 5-14
- computer workstation 1-3
- configuration tool
  - DeviceNet K-1
  - Ethernet - TCP/IP L-1
- connecting PC to network node 2-90
- constants 2-62
- contacts 2-55
- controlled stop ramp 3-36
- copying
  - axis data
    - servo setup 3-10
    - Copying an item 2-12
- copying and deleting 2-12
- creating
  - new files 2-1
  - servo setup file 3-2, 3-4, 3-7, 3-16
  - servo setup function 3-14
  - Task 5-5
  - UDFB 5-11
- C-stop error B-8
- customer service 1-2
- customizing PiCPro 1-6
- Cutting an item 2-12
- cyclic data structures 4-25
- cyclic operation 4-7

## D

- D/A output setup 3-20
- data handling in UDFBs 5-14
- data type
  - arrays 2-45
  - functions 2-45
  - numeric 2-42
  - structures 2-45
  - time 2-44
- data types 2-40
  - inputs/outputs A-1
- deceleration ramp 3-35
- define cyclic data 4-24
- Deleting an item 2-13

- dependency list 2-87
  - errors B-15
- derivative gain 3-38
- determining scaling information 3-34
- device status code K-4
- device status word K-4
- DeviceNet configuration tool K-1
  - procedure K-2
- diagnostic error codes E-1
- dialog boxes 1-9
- displaying/hiding toolbars 1-12
- Download a hex file 2-7, 2-66
- dragging and dropping 2-10
- duplicate 2-16
- E**
- editing
  - servo setup 3-13
  - UDFB 5-14
- encoder input setup 3-21
- entering
  - forcing variables 2-73
  - iterator data 3-35
  - network 2-11
  - position loop data 3-37
  - scaling data 3-33
  - servo setup data 3-26
- error codes, diagnostic E-1
- errors B-1
  - communications B-17
  - compiling B-12, B-25
  - C-stop B-8
  - dependency list B-15
  - E-stop B-9
  - force list B-59
  - function B-1
  - hardware B-23
  - LED codes E-1
  - library B-16
  - name change B-11
  - programming B-10
  - SERCOS B-60
  - servo B-51, B-52
  - servo timing B-11
  - stepper H-2
- E-stop errors B-9
- execute procedure command 4-37
- exiting 2-17
- expansion rack I/O
  - numbering 2-37, B-44

- external attribute 2-47
- F**
- fast inputs 2-38, B-46
- fast velocity filter 3-36
- faults, diagnostic E-1
- feed forward percent 3-38
- filter search 2-15
- find
  - IDNs 4-42
- finding and replacing 2-14
- Finding and replacing an item 2-14
- finding duplicates 2-16
- focus 2-9
- following error limit 3-38
- force list errors B-59
- forcing
  - entering variables 2-73
  - variables 2-72
- function
  - SERCOS 4-3
  - SERCOS setup 4-18
- function/function blocks 2-45, 2-58
  - data 2-58
- functions
  - errors B-1
  - inputs
    - data types A-1
  - outputs
    - data types A-1
  - with Stepper Axis Module H-1
- G**
- G&L scanner symbol K-3
- G&L TCP/IP configuration tool L-1
  - procedure L-2
- global and retentive attribute 2-47
- grouping 2-74
- H**
- Hardware declarations 2-19
- hardware declarations 2-19, 2-27, 2-28
  - closing 2-32
  - copying 2-30
  - cutting 2-30
  - deleting 2-29
  - editing 2-29
  - entering 2-21
  - inserting 2-29
  - pasting 2-31
  - printing 1-17, 2-32
  - remote I/O expansion rack 2-25, 2-26

- saving 2-32
- Hardware declarations, editing 2-29
- hardware errors B-23
- Hex file 2-65

## **I**

- I/O point 2-35
- I/O Points, numbering 2-36
- IDNS

- insert for receive continuous 4-34

- IDNs 4-5

- insert for send 4-36
  - insert startup 4-23
  - parameter data 4-6
  - PCF 4-6
  - print 4-44
  - receive 4-30
  - receive continuous 4-34
  - search 4-42
  - send 4-35

- ignore limits until referenced? 3-36

- information window 1-4

- initial values 2-39

- prefixes 2-39

- initialize

- servo setup data 3-15

- in-position band 3-39

- input polarity 3-38

- input scaling 3-33

- inserting an axis in servo setup 3-15

- integral error limit 3-38

- integral gain 3-38

- interlocking Task data 5-6

- iterator data 3-35

## **J**

- jump command 2-59

## **L**

- label 2-60

- ladder toolbar I-3

- LED error codes E-1

- library

- error B-16

- long names 2-34, 2-60

## **M**

- MDT - Master Data Telegram 4-4

- menu bar 1-5

- MST - Master Synchronization Telegram 4-4

## **N**

- name change error B-11

- naming

- Task 5-6

- UDFB 5-12

- NEMA SERCOS Specification 4-1

- network

- coil 2-56

- comments 2-61

- contacts 2-55

- elements 2-55

- function/function blocks 2-58

- jump command 2-59

- label 2-60

- long name 2-60

- wires 2-57

- numeric 2-42

## **O**

- off-line editing 5-14

- on-line editing 2-81, 5-15

- online Help system

- printing 1-20

- opening an existing ladder 2-3

- operation mode 4-26

- application note 4-27

- operator interface 2-69

- options

- setting 1-6

- options for IDN display 4-41

- organizing your PiCPro files 1-13

- output polarity 3-38

- output scaling 3-33

## **P**

- parameter data IDNs 4-6

- Pasting an item 2-13

- patching the ladder 2-81

- PC/PiCPro

- connecting 2-78

- PCF

- IDNs 4-6

- peer-to-peer communications 2-88

- phases

- SERCOS 4-7

- PiC

- and SERCOS 4-2

- PiCPro

- closing 2-17

- copying in 2-12

- creating files 2-1

- customizing 1-6

- define 1-1

- deleting in 2-12

- entering networks 2-11
- exiting 2-17
- filtering search 2-15
- finding and replacing 2-14
- finding duplicates 2-16
- getting started 1-1
- opening a file 2-3
- organizing files 1-13
- printing 1-14
- programs 1-1
- reference card C-1
- running 2-78
- saving 2-16
- scanning 2-78
- searching in 2-15
- splitting the screen 2-18
- starting 2-1
- tools 2-8
- work area 1-4
- working in 2-1
- pointer tool 2-8
- position loop data 3-37
- printing 1-14
  - comments 1-16
  - cross reference 1-16
  - hardware declarations 1-17
  - IDNs 4-44
  - ladder 1-15
  - long names 1-16
  - online Help system 1-20
  - SERCOS 4-17
  - servo setup axis data 3-14
- procedure
  - creating a TCP/IP configuration file L-4
  - DeviceNet configuration tool K-2
  - G&L TCP/IP configuration tool L-2
- procedure command 4-37
- Procedure Command Function 4-6
- programming errors B-10
- proportional gain 3-38
- purging unused variables 2-53
- R**
- racks
  - numbering 2-36
- receive continuous IDNs 4-33
- receive IDNs 4-30
- reference card
  - PiCPro C-1
- requirements 1-3
- resolver input setup 3-22
- retentive attribute 2-47
- ring
  - errors 4-38
  - state 4-38
- rollover on position 3-36
- rollover position 3-36
- running one scan 2-80
- S**
- saving
  - files 2-16
- saving servo setup 3-13
- scaling information 3-34
- scanner symbol K-3
- scanning 2-78
  - cold restart 2-81
  - hot restart 2-80
  - stopping 2-79
  - warm restart 2-80
- seach
  - IDNs 4-42
- searching 2-15
- selected cells 2-9
- semaphore flags 5-7
- SERCOS
  - advanced features 4-48
  - and PiC 4-2
  - battery box 4-28
  - changing IDN data and uploading IDNs 4-32
  - copying data 4-16
  - cyclic data structures 4-25
  - define cyclic data 4-24
  - define operation modes 4-26
  - errors B-60
  - execute procedure command 4-37
  - functions 4-3
  - IDNs 4-5
  - initialize setup data in your ladder 4-19
  - options for IDN display 4-41
  - phases 4-7
  - printing 4-17
  - receive continuous IDNs 4-33
  - receive IDNs 4-30
  - replacing analog system 4-47
  - ring state 4-38
  - rings 4-20
  - send IDNs 4-35
  - setup 3-22, 3-25, 4-1, 4-15

- setup file 4-17
- setup function 4-18
- slave status/control
  - slave status/control 4-40
- slaves 4-21
- specifying the SRS 4-46
- standard Motion Functions and Variables 4-13
- startup 4-47
- startup IDN List 4-22
- telegrams 4-4
- troubleshooting 4-50
- upload drive information 4-31
- using Functions/Blocks with TASKS 4-14
- variable length data 4-45
- service channel 4-7
  - accessing 4-10
  - application note 4-9
- servo
  - errors B-51, B-52
  - timing errors B-11
- servo setup 3-1
  - copying axis data 3-10
  - creating
    - file 3-2, 3-4, 3-7, 3-16
    - function 3-14
  - editing 3-13
  - entering
    - data 3-26
    - scaling data 3-33
  - initializing 3-15
  - inserting an axis 3-15
  - printing 3-14
  - saving 3-13
- setting node IDs 2-89
- settings command 2-69
- size of UDFB Libraries 5-13
- slow velocity filter 3-36
- slow/fast velocity threshold 3-36
- software declarations 2-33
  - attributes 2-47
  - copying 2-49
  - cutting 2-49
  - editing 2-48
  - entering 2-33
  - fast inputs 2-38, B-46
  - I/O point 2-35
  - initial values 2-39
  - long names 2-34
  - pasting 2-49
  - purging unused variables 2-53
  - searching 2-52
- Software declarations, entering 2-33
- software lower limit 3-36
- software upper limit 3-36
- specification
  - NEMA SERCOS 4-1
- Split screen 2-18
- split screen 2-18
- standard toolbar I-1
- starting PiCPro 2-1
- startup
  - SERCOS 4-47
- startup IDN list 4-22
- status
  - blown fuse 2-38, B-46
- status bar 1-8
  - displaying/hiding 1-9
  - stepper
    - axis module H-1, I-1, J-1
    - error handling H-2
    - reference card D-1
    - using functions with H-1
- stopping the scan 2-79
- STRING
  - read/write from STRUCT AN-3
- structures 2-45
- support services 1-2
- T**
- Task 5-1, 6-1
  - comparing to UDFB 5-1
  - compiling 2-67
  - creating 5-5
  - hardware declarations 5-4
  - interlocking data 5-6
  - naming 5-6
  - semaphore flags 5-7
  - software declarations 5-4
  - template 5-2
  - types 5-1
  - using Functions 5-3
- technical support 1-2
- telegrams
  - SERCOS 4-4
- time 2-44
- time axis F-1
  - synchronizing slave axes with F-2

- title bar 1-5
- toolbar
  - advanced operations I-2
  - basic online operations I-2
  - compiler I-4
  - customizing 1-10
  - displaying/hiding 1-12
  - docking 1-12
  - ladder I-3
  - moving 1-11
  - standard I-1
  - view navigator I-4
- troubleshooting
  - SERCOS 4-50
- TTL input setup 3-24
- U**
- UDFB 5-11
  - comparing to Task 5-1
  - compiling 2-68, 5-14
  - creating 5-11
  - data handling 5-14
  - edit 5-14
  - inputs and outputs
    - marking 5-18
    - number of 5-17
    - order of 5-17
  - library size 5-13
  - naming 5-12
  - software declarations 5-16
  - template 5-11
  - view 5-15
- UDFBs 5-1, 6-1
- update rate 3-39
- upload drive information 4-31
- user preferences 1-6
- V**
- variable in/out attribute 2-48
- variable length data
  - viewing 4-45
- Variables 2-34
- variables 2-40, 2-62
- Variables, names and long names 2-34
- velocity limit 3-35
- view navigator toolbar I-4
- viewing
  - enabled variables 2-76
  - parent ladder 5-15
  - UDFB 5-15
- W**
- Window elements 1-5
- wires 2-57
- work area
  - PiCPro 1-4
- workstation 1-3
- Z**
- zooming 2-8