



PS69-DPS

**CompactLogix or MicroLogix
Platform**

Profibus DP Slave Communication
Module

May 9, 2014

Your Feedback Please

We always want you to feel that you made the right decision to use our products. If you have suggestions, comments, compliments or complaints about our products, documentation, or support, please write or call us.

How to Contact Us

ProSoft Technology

5201 Truxtun Ave., 3rd Floor
Bakersfield, CA 93309
+1 (661) 716-5100
+1 (661) 716-5101 (Fax)
www.prosoft-technology.com
support@prosoft-technology.com

Copyright © 2014 ProSoft Technology, Inc., All rights reserved.

PS69-DPS User Manual

May 9, 2014

ProSoft Technology[®], ProLinx[®], inRAx[®], ProTalk[®], and RadioLinx[®] are Registered Trademarks of ProSoft Technology, Inc. All other brand or product names are or may be trademarks of, and are used to identify products and services of, their respective owners.

ProSoft Technology[®] Product Documentation

In an effort to conserve paper, ProSoft Technology no longer includes printed manuals with our product shipments. User Manuals, Datasheets, Sample Ladder Files, and Configuration Files are provided on the enclosed DVD, and are available at no charge from our web site: www.prosoft-technology.com

Printed documentation is available for purchase. Contact ProSoft Technology for pricing and availability.

North America: +1.661.716.5100

Asia Pacific: +603.7724.2080

Europe, Middle East, Africa: +33 (0) 5.3436.87.20

Latin America: +1.281.298.9109

Throughout this manual, you will see references to other product names such as:

- RIF 1769-DPM
- SYCON.net

These product names (RIF 1769, SYCON.net) are legacy versions, and are mentioned for backward compatibility with existing implementations. These products are now supported and maintained by ProSoft Technology.

The ProSoft and legacy versions of these products may not be interchangeable.

Contents

Your Feedback Please.....	2
How to Contact Us	2
ProSoft Technology® Product Documentation.....	2

Guide to the PS69-DPS User Manual **7**

1 Start Here **9**

1.1	Software Requirements.....	10
1.2	Hardware Requirements	10
1.3	Reference Systems	10
1.4	Programmable Controller Functionality.....	11
1.5	Package Contents	12
1.6	Installing the Module in the Rack	13
1.7	Connecting Your PC to the Processor	16
1.8	PS69-DPS Sample Add-On Instruction Import Procedure.....	17
1.8.1	Create a new RSLogix5000 project	17
1.8.2	Create the Module.....	18
1.8.3	Import the Ladder Rung	20
1.8.4	Adding Multiple Modules (Optional)	25
1.9	Downloading the Sample Program to the Processor	32
1.9.1	Configuring the RSLinx Driver for the PC COM Port	33
1.10	Adapter (PROFIBUS-DP-Slave)	35

2 Configuration and Start-Up **37**

2.1	RSLogix 5000.....	38
2.1.1	Module Selection.....	38
2.1.2	Module Properties 1	40
2.1.3	Module Properties 2	41
2.2	RSLogix 500.....	42
2.2.1	Module Selection.....	42
2.2.2	Expansion General Configuration	43
2.2.3	Generic Extra Data Config	44
2.3	Slave Configuration	45
2.3.1	General.....	45
2.3.2	GSD File.....	45
2.3.3	Configuration by Master	45
2.3.4	Configuration by Controller Application.....	46
2.3.5	Explanation of settable configuration values.....	47

3 RSLogix Example Program **49**

3.1	CompactLogix I/O Example.....	50
3.2	CompactLogix Messaging Example	52

4	Diagnostics and Troubleshooting	55
4.1	Hardware Diagnostics (LED)	56
4.1.1	CompactLogix	56
4.1.2	MicroLogix 1500	56
4.1.3	PS69 LEDs	57
4.2	Troubleshooting	58
4.2.1	CompactLogix I/O LED	58
4.2.2	MicroLogix Fault LED	58
4.2.3	SYS and COM Status LEDs	58
4.2.4	Error Sources and Reasons	58
4.2.5	Cable	60
5	Reference	61
5.1	Specifications	62
5.1.1	General Specifications	62
5.1.2	Hardware Specifications	63
5.1.3	Functional Specifications	64
5.1.4	PROFIBUS Interface	65
5.2	RSLogix5000 User Defined Data Types	66
5.2.1	Input: DPS_INPUT_ARRAY	66
5.2.2	Input: DPS_DEV_STATUS_REGISTER	66
5.2.3	Input: DPS_FW_REVISION	66
5.2.4	Input: DPS_STATUS_FIELD	67
5.2.5	Output: DPS_OUTPUT_ARRAY	67
5.2.6	Output: DPS_DEV_COMMAND_REGISTER	67
5.2.7	APP_CONSTANT_PATTERN	68
5.2.8	APP_DPV1_PROG_CONTROL	68
5.2.9	APP_DPV1_STAT_COUNTER	68
5.2.10	DPS_DIAGNOSTIC_CONFIRM	69
5.2.11	DPS_DIAGNOSTIC_REQUEST	69
5.2.12	DPS_DPV1C1_ALARM_CONFIRM	70
5.2.13	DPS_DPV1C1_ALARM_REQUEST	70
5.2.14	DPS_DPV1C1_RW_INDICATION	71
5.2.15	DPS_DPV1C1_RW_RESP_CONFIRM	71
5.2.16	DPS_DPV1C1_RW_RESP_REQUEST	72
5.3	PROFIBUS Functionality	73
5.3.1	DPV0 Services	73
5.3.2	DPV1 Services	74
5.3.3	Start/Stop Communication	74
5.4	Communication	75
5.4.1	IO Communication and IO Memory Map	75
5.4.2	CIP Messaging	84
5.5	Constructing a Bus Cable for PROFIBUS DP	99
6	Support, Service & Warranty	103
	Contacting Technical Support	103
6.1	Warranty Information	105
	Index	107

Guide to the PS69-DPS User Manual

Function		Section to Read	Details
Introduction (Must Do)	→	Start Here (page 9)	This section introduces the customer to the module. Included are: package contents, system requirements, hardware installation, and basic configuration.
Diagnostic and Troubleshooting	→	Diagnostics and Troubleshooting (page 55)	This section describes Diagnostic and Troubleshooting procedures.
Reference Product Specifications Functional Overview	→	Reference (page 61) Product Specifications (page 62) Functional Overview	These sections contain general references associated with this product, Specifications, and the Functional Overview.
Support, Service, and Warranty Index	→	Support, Service and Warranty (page 103) Index	This section contains Support, Service and Warranty information. Index of chapters.

1 Start Here

In This Chapter

❖ Software Requirements	10
❖ Hardware Requirements.....	10
❖ Reference Systems	10
❖ Programmable Controller Functionality	11
❖ Package Contents	12
❖ Installing the Module in the Rack.....	13
❖ Connecting Your PC to the Processor.....	16
❖ PS69-DPS Sample Add-On Instruction Import Procedure	17
❖ Downloading the Sample Program to the Processor	32
❖ Adapter (PROFIBUS-DP-Slave).....	35

To get the most benefit from this User Manual, you should have the following skills:

- **Rockwell Automation® RSLogix™ software:** launch the program, configure ladder logic, and transfer the ladder logic to the processor
- **Microsoft Windows:** install and launch programs, execute menu commands, navigate dialog boxes, and enter data
- **Hardware installation and wiring:** install the module, and safely connect PROFIBUS DP and CompactLogix or MicroLogix devices to a power source and to the PS69-DPS module's application port(s)

The PS69-DPS module expands the functionality of Rockwell Automation's CompactLogix or MicroLogix to include PROFIBUS DP V0/V1. The PS69-DPS is a more cost-effective option offering more features than the PS69-PDPS, and supports both I/O control and messaging. Explicit ladder logic CIP message blocks provide slave status diagnostic data and acyclic messaging.

The PS69-DPS interface appears to the CompactLogix or MicroLogix controller as a standard I/O module allowing it to be configured via RSLogix5000, or configuration can be transferred from the Master to the PS69-DPS. For third party configuration a GSD file is supplied. The slave interface possesses a diagnostic interface and has rotary switches for setting of the bus address. Complete program examples for simple and quick start-up are available.

Each module is equipped with LEDs to display communication and device status.

1.1 Software Requirements

Follows are the software requirements for using the PS69-DPS module within a CompactLogix system. You must have the following software installed on your computer unless otherwise noted:

CompactLogix System

- RSLogix 5000, V13.00 or higher

MicroLogix 1500 System

- RSLogix 500, V6.30 or higher

1.2 Hardware Requirements

The following minimum hardware is required to use the PS69-DPS PROFIBUS module.

CompactLogix System

- Personal Computer
- 1769: Programmable Controller
- 1769: Power Supply
- 1769: Right or Left handed Termination End Cap
- Serial Cable for interface to the 1769-Programmable Controller.

MicroLogix 1500 System

- Personal Computer
- 1764: MicroLogix 1500 Programmable Controller
- 1769: Right handed Termination End Cap
- Serial Cable for interface to the 1764-Programmable Controller.

1.3 Reference Systems

The firmware of the communication module PS69-DPS was developed and tested with following CompactLogix / MicroLogix Controller types and firmware revisions.

CompactLogix System

PS69-DPS	CompactLogix 1769-L20	CompactLogix 1769-L32E
Firmware V10.2	Firmware V13.18	Firmware V13.28

MicroLogix 1500 System

PS69-DPS	MicroLogix 1500 (Processor 1764-LRP/A Rev2.0)
Firmware V10.2	Firmware: OS 1510; Series C ; Revision 9.0

1.4 Programmable Controller Functionality

PROFIBUS-DP supports acyclic services through messages. These PROFIBUS-DP services are supported by the RSLogix5000 programming tool using CIP messages. Not all of the 1769 Programmable Controllers support CIP messaging. If your Controller does not support messaging, these services are not available.

The basic PROFIBUS-DP acyclic services Global Control or Slave Diag request are also executable in addition to the CIP method by using the I/O area. Follows is a matrix of 1769 Programmable Controllers and the functionality that they support.

CompactLogix System

Processor/ Features	1769-L20	1769 -L30	1769 -L31	1769 -L32E	1769- L35E
I/O	yes	yes	yes	yes	yes
CIP Messaging	no	no	yes	yes	Yes

MicroLogix 1500 System

Processor/ Features	1764 -LRP	1764 -LSP
I/O	yes	yes
CIP Messaging	no	no

yes = functionality supported
 no = functionality not supported

1.5 Package Contents

The following components are included with your PS69-DPS module, and are all required for installation and configuration.

Important: Before beginning the installation, please verify that all of the following items are present.

Qty.	Part Name	Part Number	Part Description
1	PS69-DPS Module	PS69-DPS	Profibus DP Slave Communication Module
1	ProSoft Solutions DVD	DVD-001	Contains sample programs, utilities and documentation for the PS69-DPS module.

If any of these components are missing, please contact ProSoft Technology Support for replacement parts.

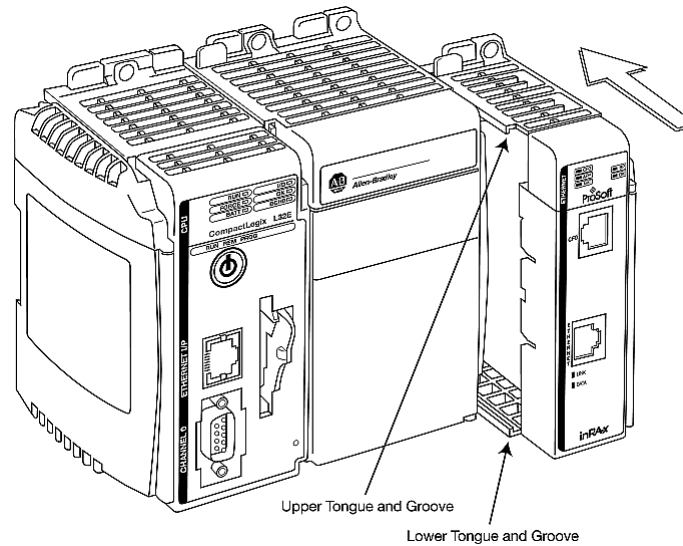
1.6 Installing the Module in the Rack

This section describes how to install the module into a CompactLogix or MicroLogix rack.

Before you attempt to install the module, make sure that the bus lever of the adjacent module is in the unlocked (fully right) position.

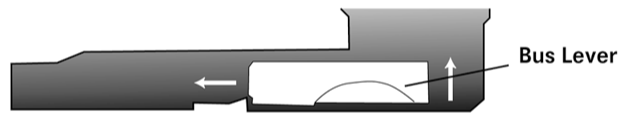
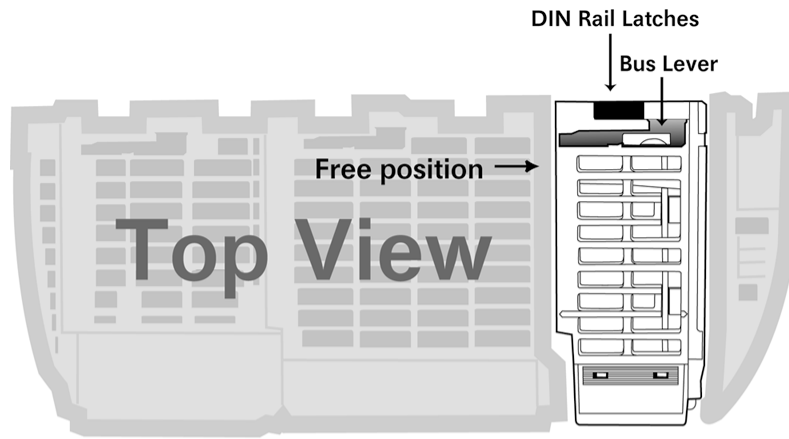
Warning: This module is not hot-swappable! Always remove power from the rack before inserting or removing this module, or damage may result to the module, the processor, or other connected devices.

- 1 Align the module using the upper and lower tongue-and-groove slots with the adjacent module and slide forward in the direction of the arrow.

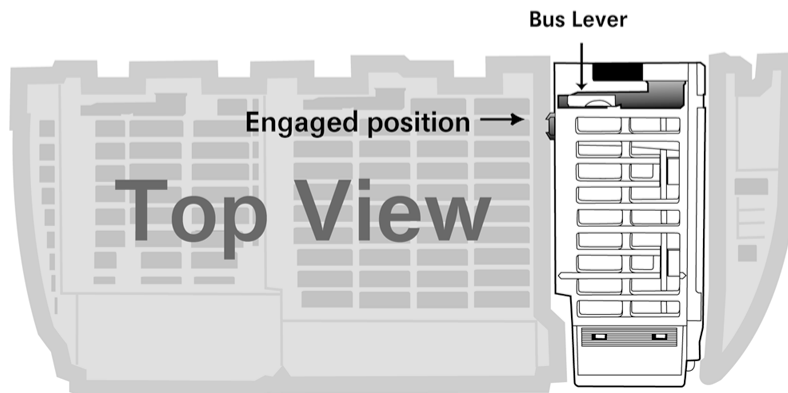


- 2 Move the module back along the tongue-and-groove slots until the bus connectors on the PS69 module and the adjacent module line up with each other.

- 3 Push the module's bus lever back slightly to clear the positioning tab and move it firmly to the left until it clicks. Ensure that it is locked firmly in place.

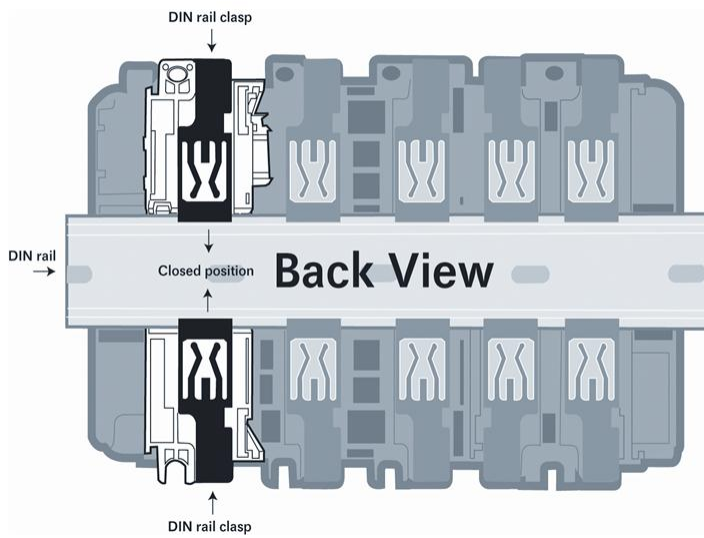
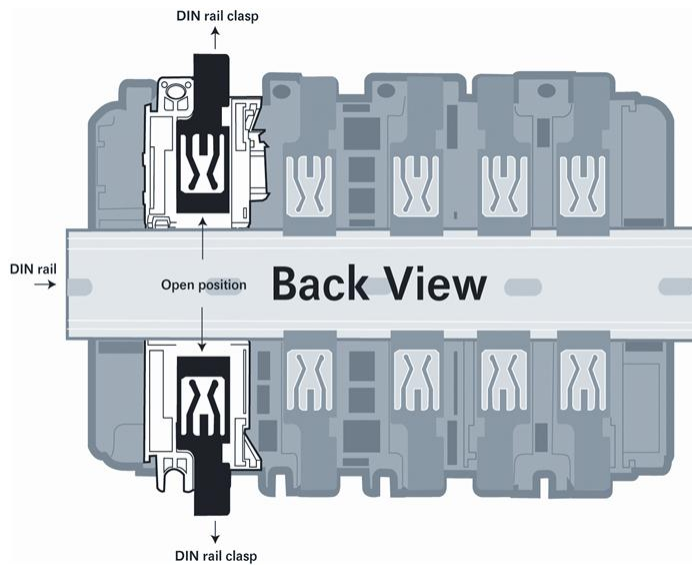


Move the Bus Lever to the left
until it clicks



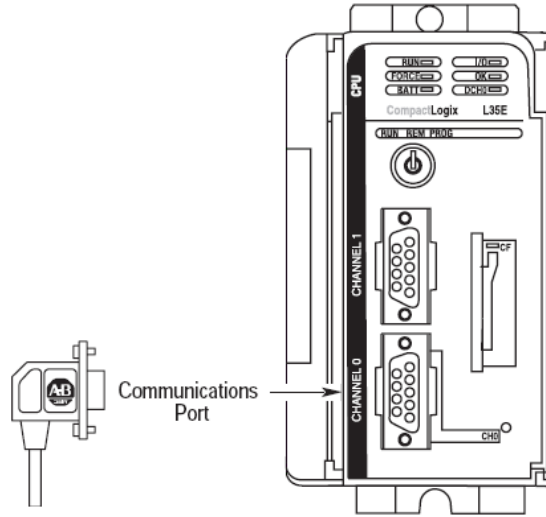
- 4 Close all DIN-rail latches.

- 5 Press the DIN-rail mounting area of the controller against the DIN-rail. The latches will momentarily open and lock into place.

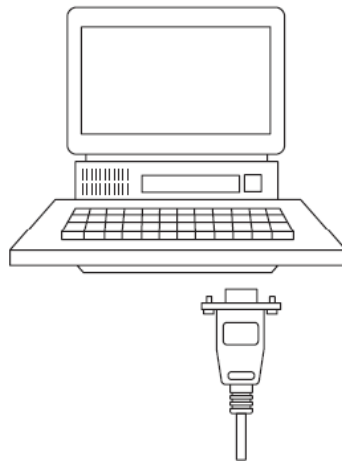


1.7 Connecting Your PC to the Processor

- 1 Connect the right-angle connector end of the cable to your controller at the communications port.



- 2 Connect the straight connector end of the cable to the serial port on your computer.



1.8 PS69-DPS Sample Add-On Instruction Import Procedure

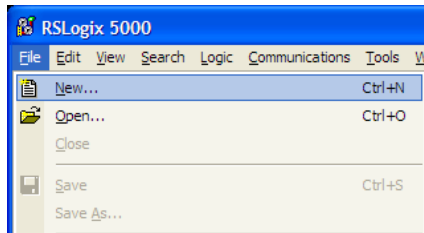
Note: this section only applies if you are using RSLogix 5000 version 16 or higher.

The following file is required before you start this procedure. Copy the file from the ProSoft Solutions DVD, or download it from www.prosoft-technology.com.

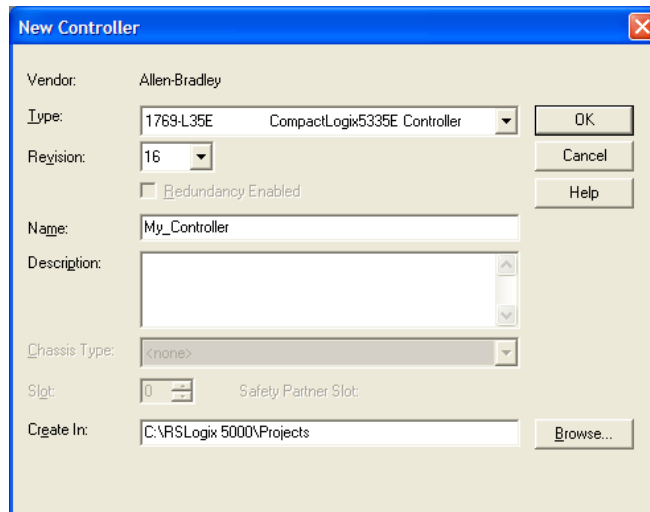
File Name	Description
AOIPS69DPS_<Version #>.L5X	L5X file contains the Add-On instruction, the user defined data types, data objects and ladder logic required to set up the PS69-DPS module

1.8.1 Create a new RSLogix5000 project

- 1 Open the **FILE** menu, and then choose **NEW...**

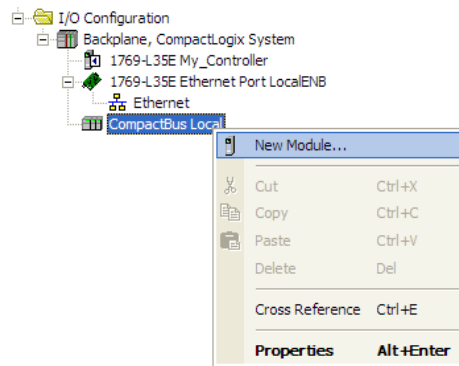


- 2 Select **REVISION 16**

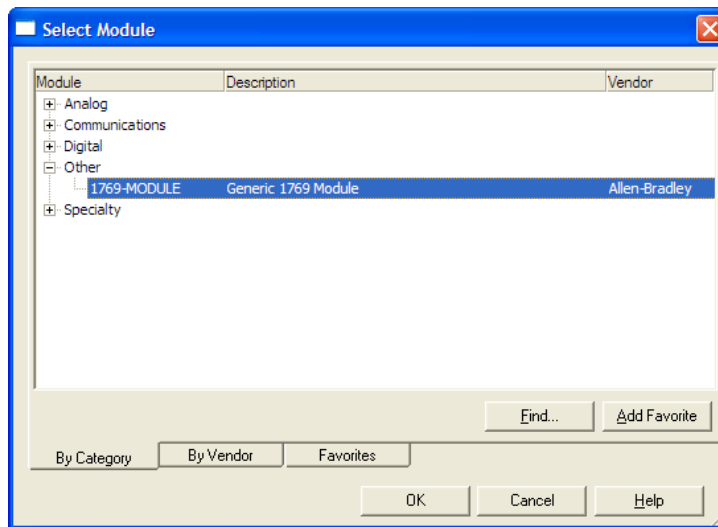


1.8.2 Create the Module

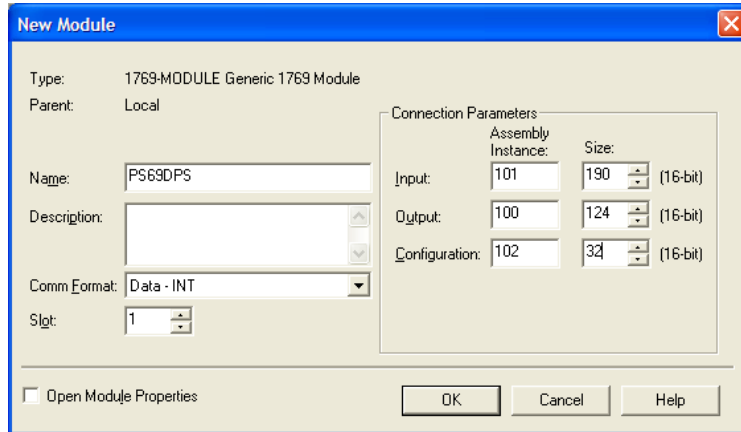
- 1 Right-click I/O Configuration and choose New Module...



- 2 Select 1769-MODULE

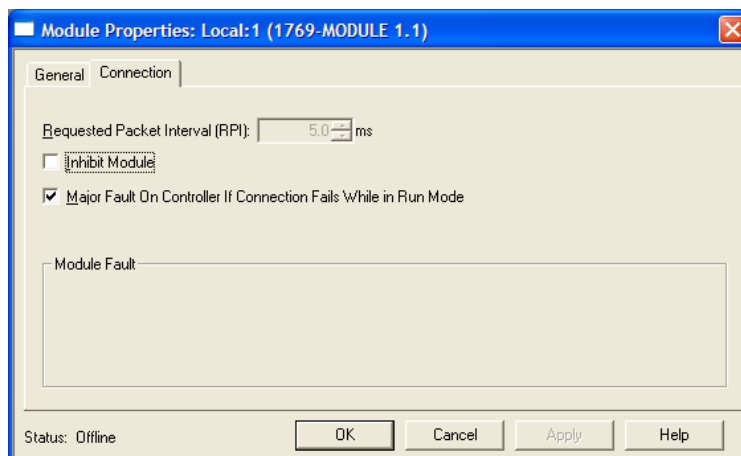


3 Set the Module Properties values as follows:

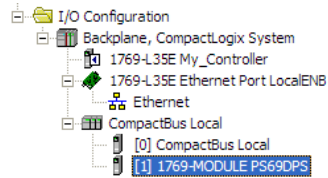


Parameter	Value
Name	Enter a module identification string. Example: PS69DPS
Description	Enter a description for the module. Example: Profibus DP Slave Communication Module.
Comm Format	Select Data-INT
Slot	Enter the slot number in the rack where the PS69-DPS module will be installed.
Input Assembly Instance	101
Input Size	190
Output Assembly Instance	100
Output Size	124
Configuration Assembly Instance	102
Configuration Size	32

4 On the Connection tab, check or un-check, as desired the Major fault option.

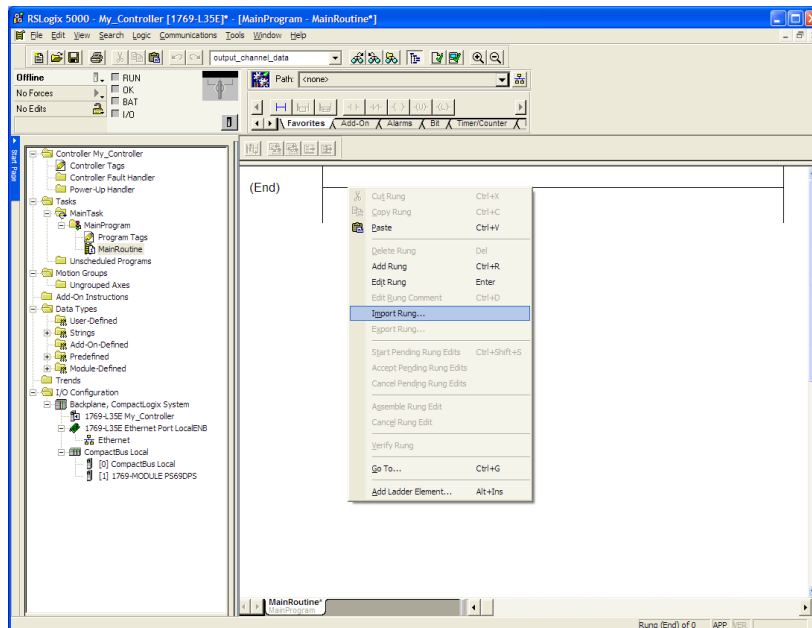


Now the PS69-DPS module will be visible at the I/O Configuration section

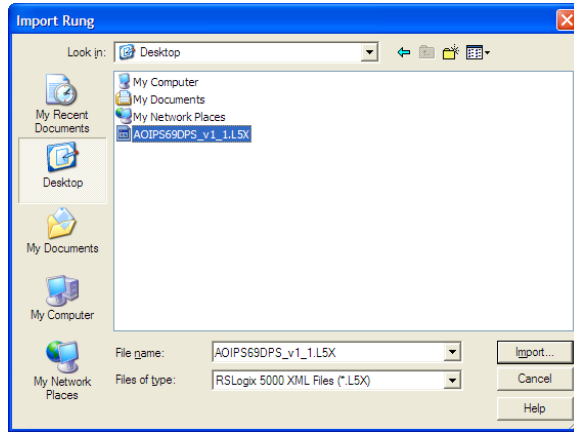


1.8.3 Import the Ladder Rung

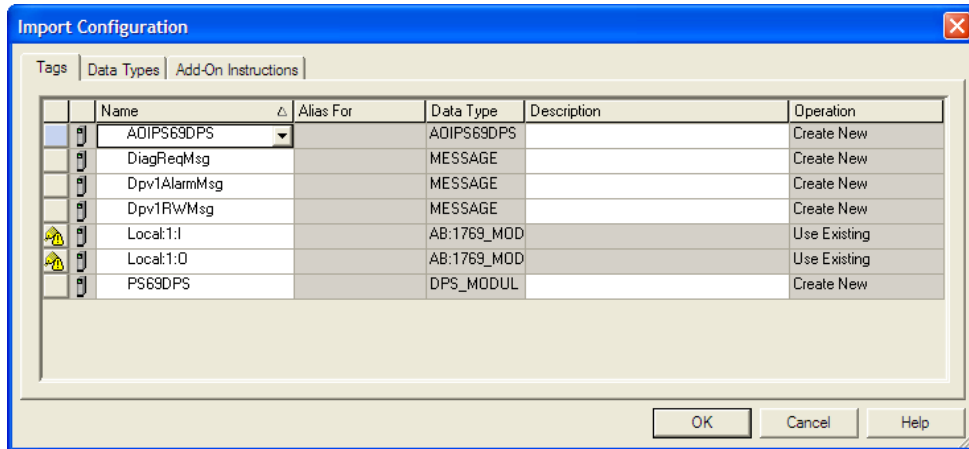
- 1 Open your application in RSLogix 5000.
- 2 To create a new routine, expand the **TASKS** folder, and then expand the **MAIN TASK** folder.
- 3 On the **MAIN PROGRAM** folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW ROUTINE**.
- 4 In the **NEW ROUTINE** dialog box, enter the name and description of your routine, and then click **OK**. In this example we are demonstrating the importing of the ladder rung using the default MainRoutine. In the case where you create a routine by an other name for placing the Add-On instruction, then in your original routine where your other ladder logic is located you need to add a rung with a jump instruction to the new routine holding the Add-On instruction.
- 5 Select an empty rung in the new routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose "**IMPORT RUNG...**".



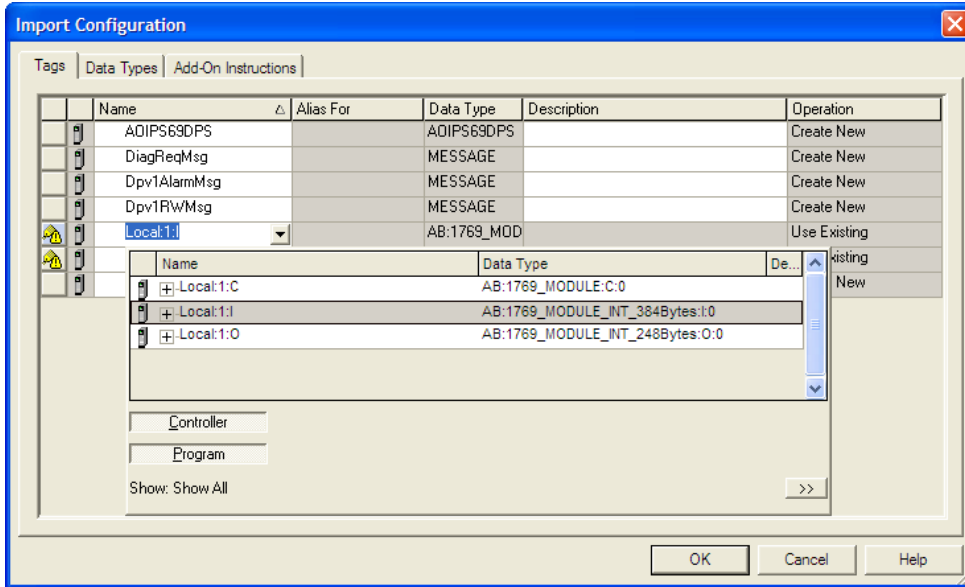
6 Select the AOIPS69DPS_<Version #>.L5X file



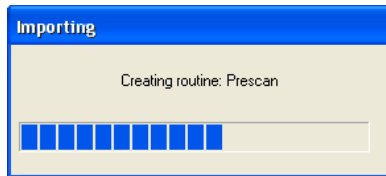
7 The following window will be displayed showing the controller tags to be created during the import procedure: If desired, the description, "PS69-DPS Interface AOI" may be typed into the description field for AOIPS69DPS_<Version #>.L5X file.



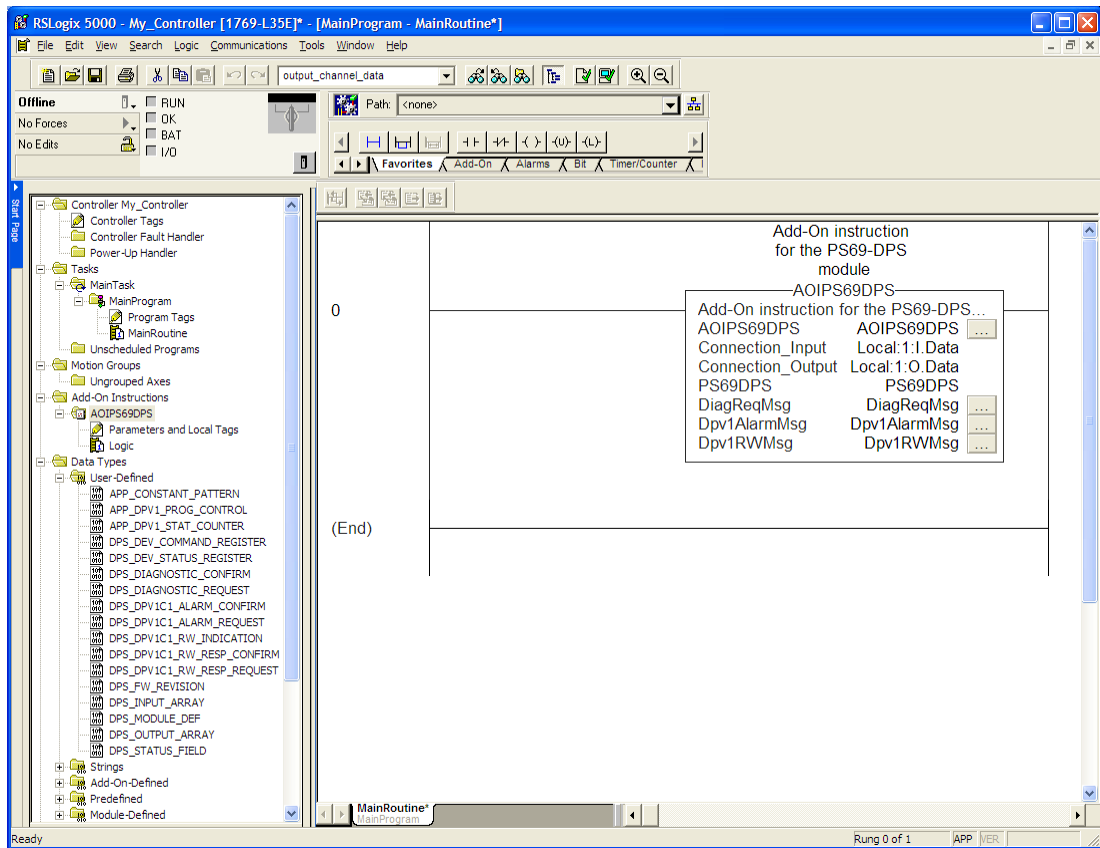
- 8 If you are using the module in a different slot (or remote rack) select the correct connection input and output variables associated to the module. If your module is located in slot 1 of the local rack this step is not required.



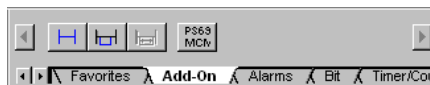
Click OK to confirm the import. RSLogix will indicate that the import is under progress:



When the import is completed, the new rung with the Add-On instruction will be visible as shown in the following illustration.

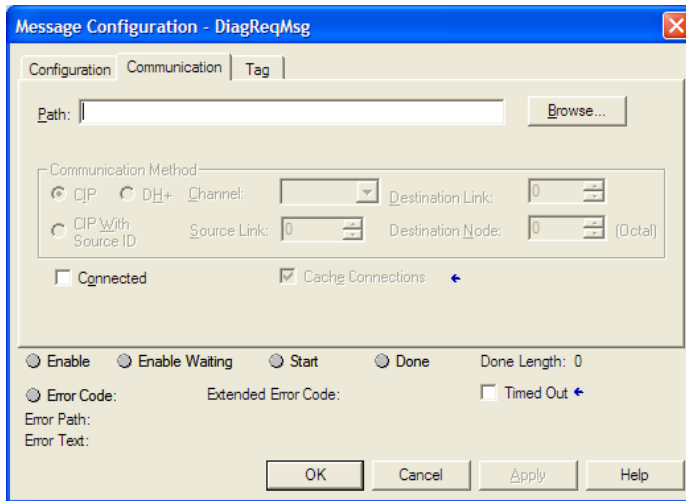


The procedure has also imported new user defined data types, data objects and the Add-On instruction to be used at your project.

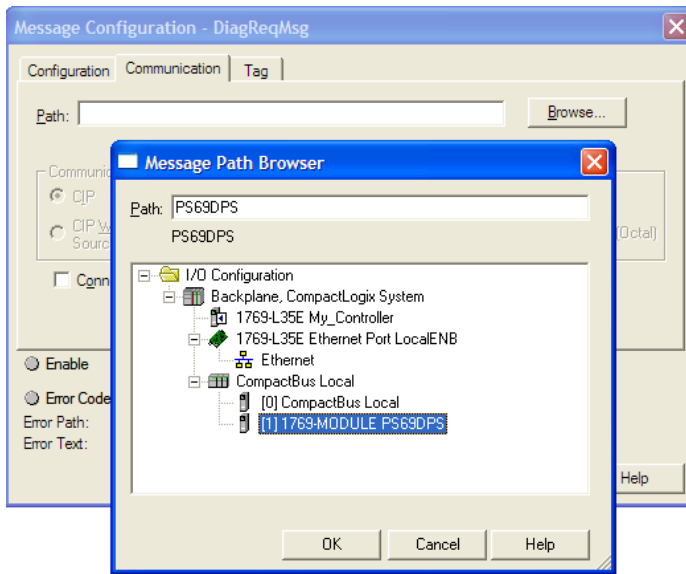


- 9 In the Add-On Instruction, click the [...] button next to each **MSG** tag to open the **MESSAGE CONFIGURATION TAG**.

10 Click the **COMMUNICATION** tab and click the **BROWSE** button as follows.



11 Select the module to configure the message path.

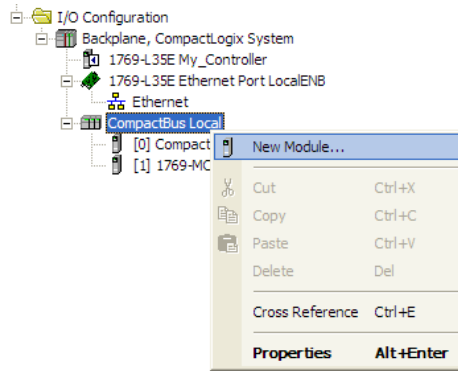


12 Repeat steps 9 through 11 to configure the message path for Dpv1AlarmMsg and DPv1RWMsg.

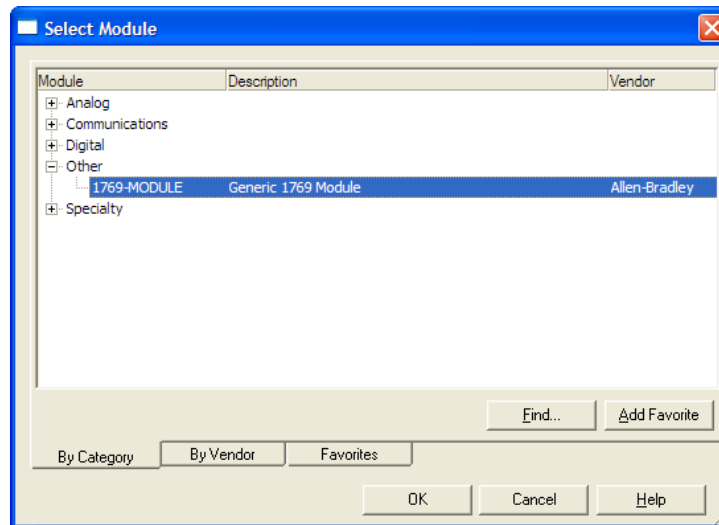
1.8.4 Adding Multiple Modules (Optional)

Important: If your application requires more than one PS69-DPS module into the same project, follow the steps below and make certain that both modules are assigned identical Block Transfer Sizes.

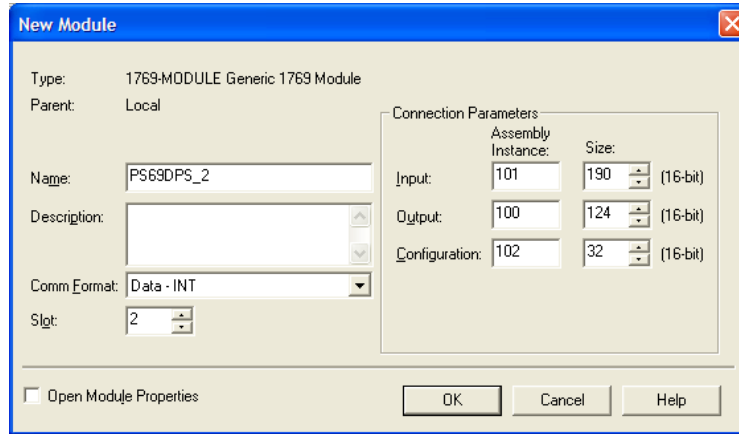
- 1 In the **I/O CONFIGURATION** folder, click the right mouse button to open a shortcut menu, and then choose **NEW MODULE**.



- 2 Select **1756-MODULE**

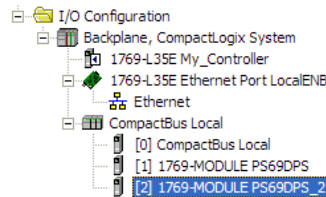


3 Fill the module properties as follows:



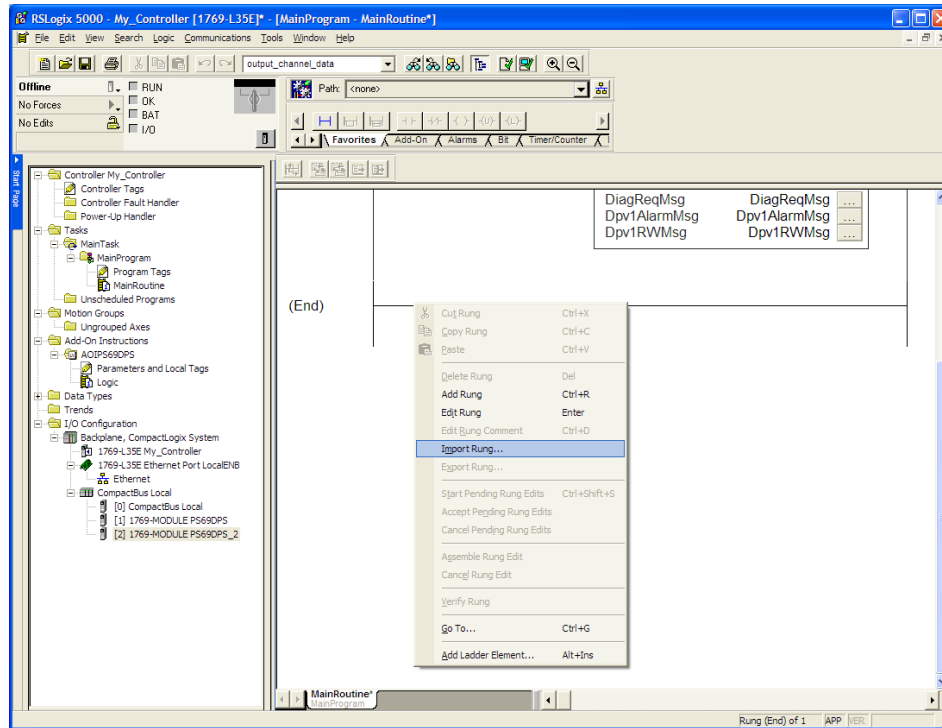
Parameter	Value
Name	Enter a module identification string. Example: PS69DPS_2
Description	Enter a description for the module. Example: Profibus DP Slave Communication Module
Comm Format	Select Data-INT
Slot	Enter the slot number in the rack where the PS69-DPS module will be installed.
Input Assembly Instance	101
Input Size	190
Output Assembly Instance	100
Output Size	124
Configuration Assembly Instance	102
Configuration Size	32

4 Click **OK** to confirm. The new module is now visible:

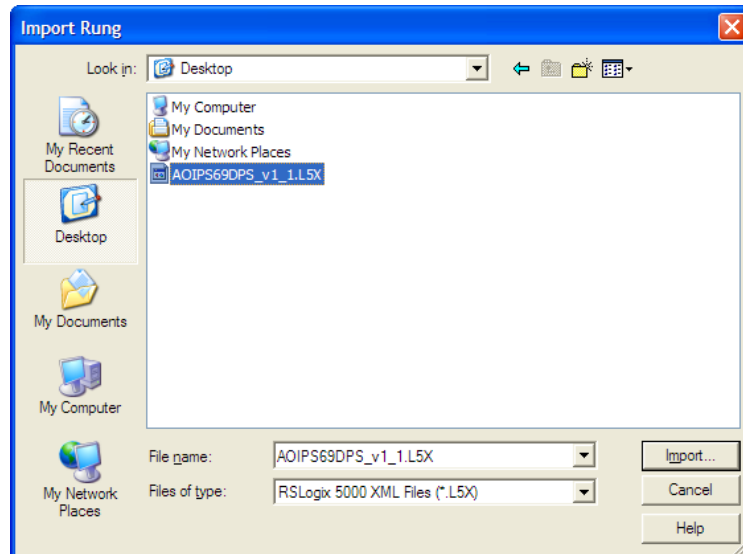


- 5** Expand the **TASKS** folder, and then expand the **MAINTASK** folder.
- 6** On the **MAINPROGRAM** folder, click the right mouse button to open a shortcut menu. On the shortcut menu, choose **NEW ROUTINE**. As an alternative to creating a separate New Routine, you could skip to Step 8 and import the AOI for the second module into the same routine you created for the first module.
- 7** In the **NEW ROUTINE** dialog box, enter the name and description of your routine, and then click **OK**.

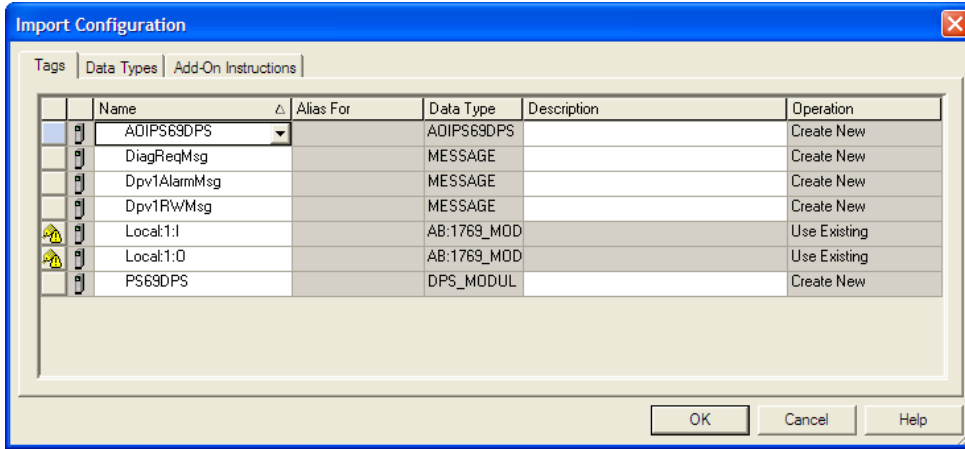
- 8 Select an empty rung in the new routine or an existing routine, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose **IMPORT RUNG...**



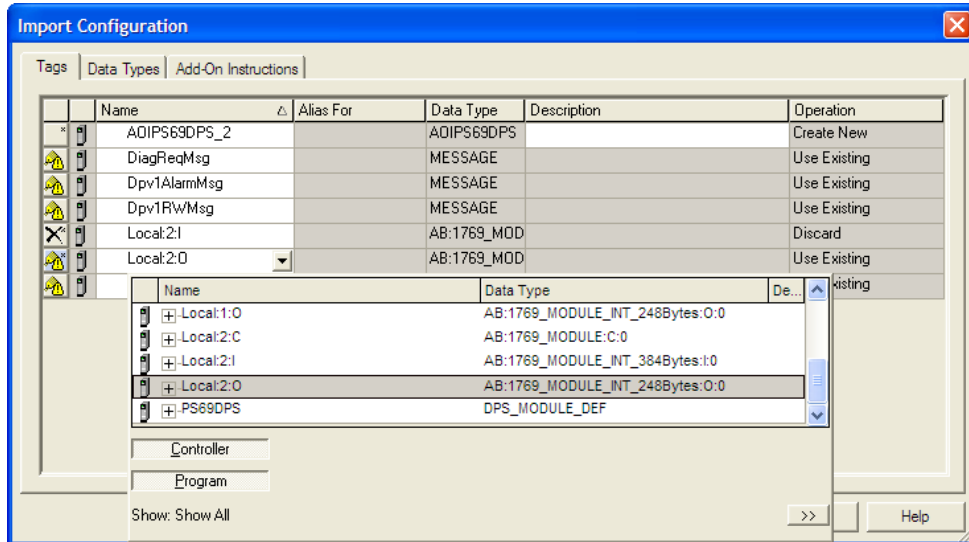
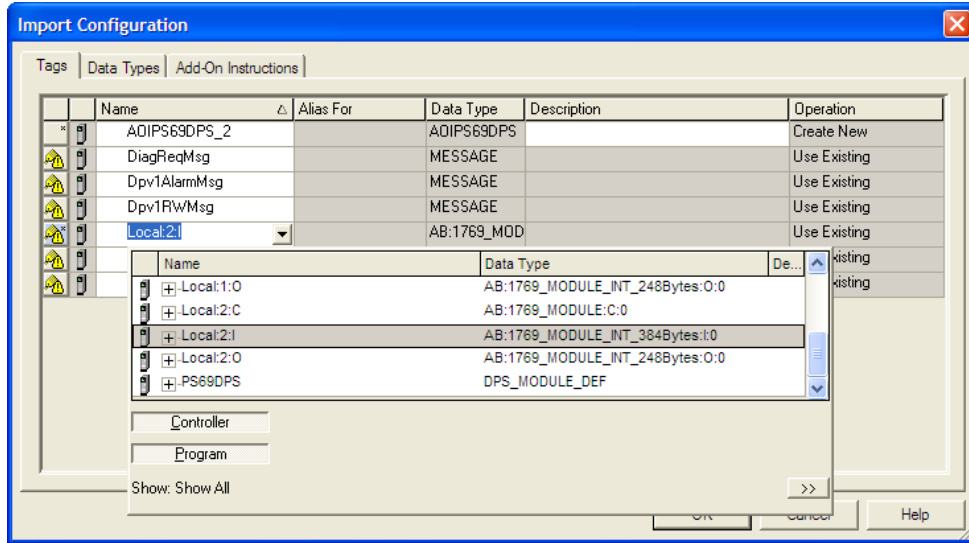
- 9 Select the *AOIPS69DPS5X* file



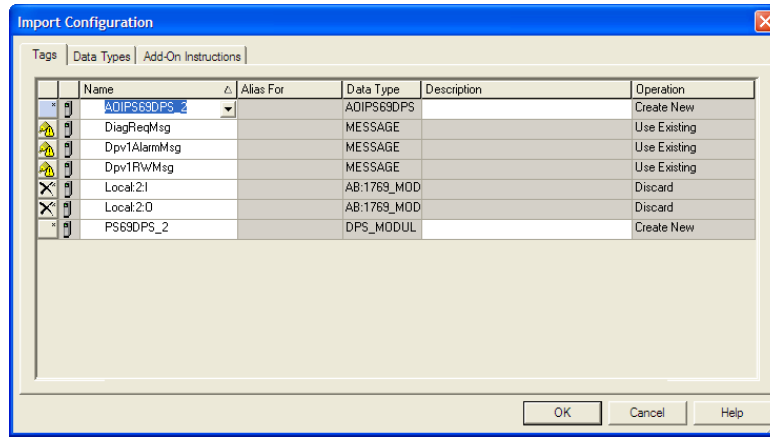
10 The following window will be displayed showing the tags to be imported:



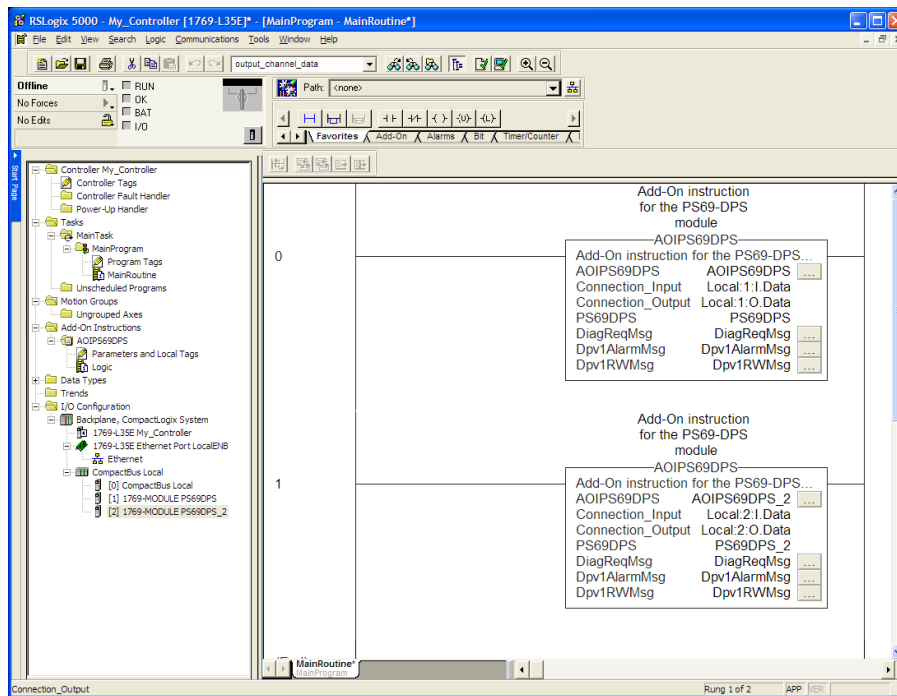
- Associate the I/O connection variables to the correct module. The default values are Local:1:I and Local:1:O. These require re-assignment to the new module's location.



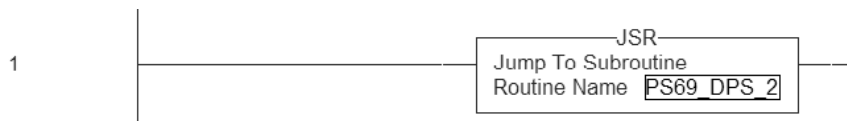
12 Change the default tags PS69DPS and AOIPS69DPS to avoid conflict with existing tags. This example procedure will append the string "_2" as follows:



13 You will be prompted to confirm your change. Click OK to continue.

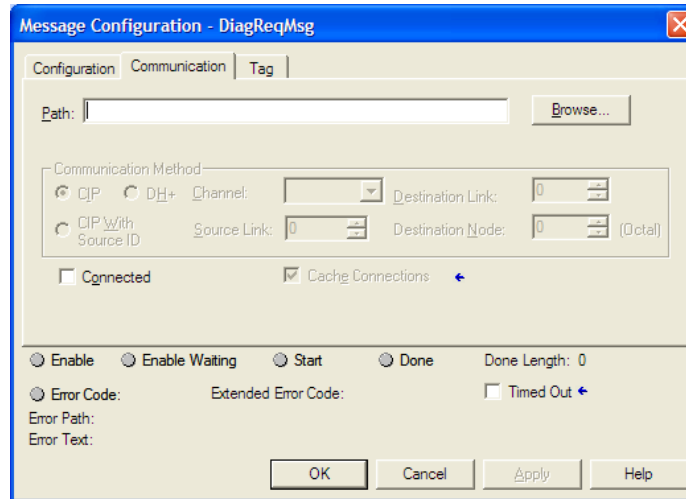


If the second module's logic was created in a new routine, enter a rung in the Main routine with a JSR instruction to the new routine to enable the PLC logic to communicate with both modules.

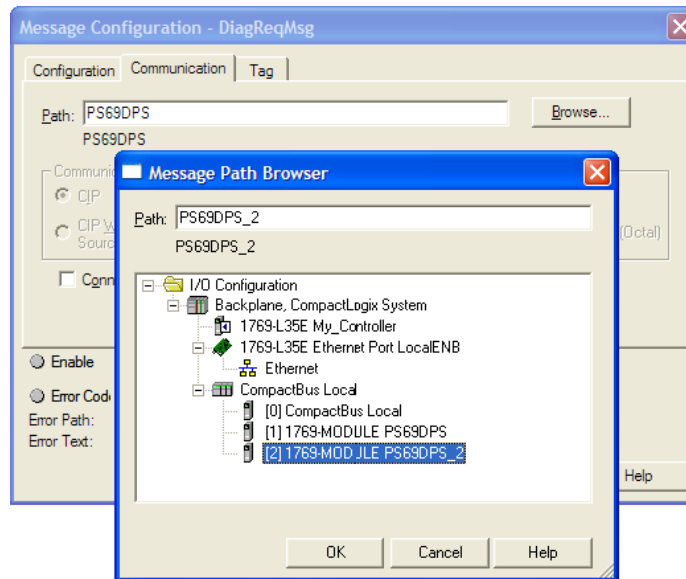


14 In the Add-On Instruction, click the [...] button next to each **MSG** tag to open the **MESSAGE CONFIGURATION TAG**.

15 Click the **COMMUNICATION** tab and click the **BROWSE** button as follows.



16 Select the module to configure the message path.



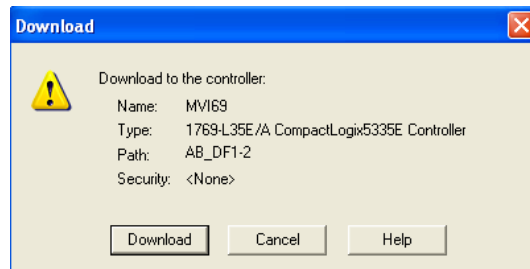
17 Repeat steps 14 through 16 to configure the message path for Dpv1AlarmMsg and DPv1RWMsg.

The setup procedure is now complete. Save the project and download the application to your CompactLogix processor.

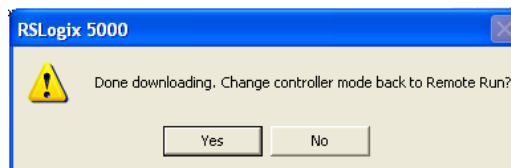
1.9 Downloading the Sample Program to the Processor

Note: The key switch on the front of the *CompactLogix* processor must be in the REM or PROG position.

- 1 If you are not already online to the processor, open the **COMMUNICATIONS** menu, and then choose **DOWNLOAD**. *RSLogix* will establish communication with the processor.
- 2 When communication is established, *RSLogix* will open a confirmation dialog box. Click the **DOWNLOAD** button to transfer the sample program to the processor.



- 3 *RSLogix* will compile the program and transfer it to the processor. This process may take a few minutes.
- 4 When the download is complete, *RSLogix* will open another confirmation dialog box. Click **OK** to switch the processor from PROGRAM mode to RUN mode.

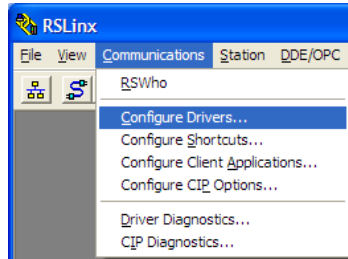


Note: If you receive an error message during these steps, refer to your *RSLogix* documentation to interpret and correct the error.

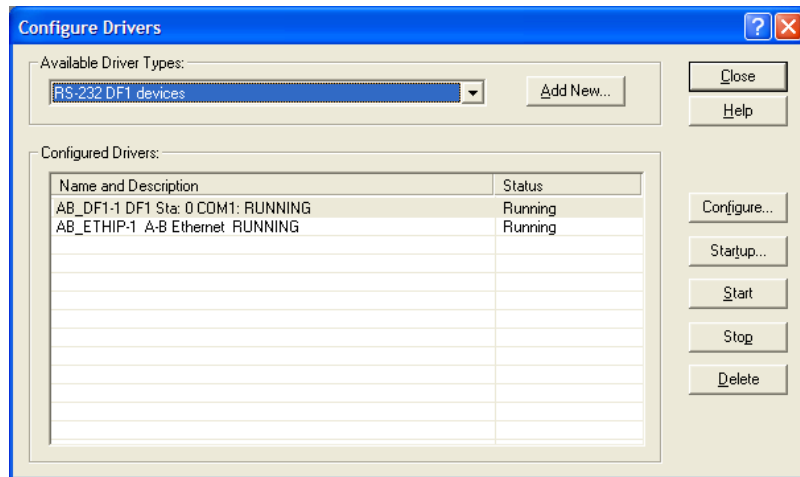
1.9.1 Configuring the RSLinx Driver for the PC COM Port

If RSLogix is unable to establish communication with the processor, follow these steps.

- 1 Open *RSLogix*.
- 2 Open the **COMMUNICATIONS** menu, and choose **CONFIGURE DRIVERS**.

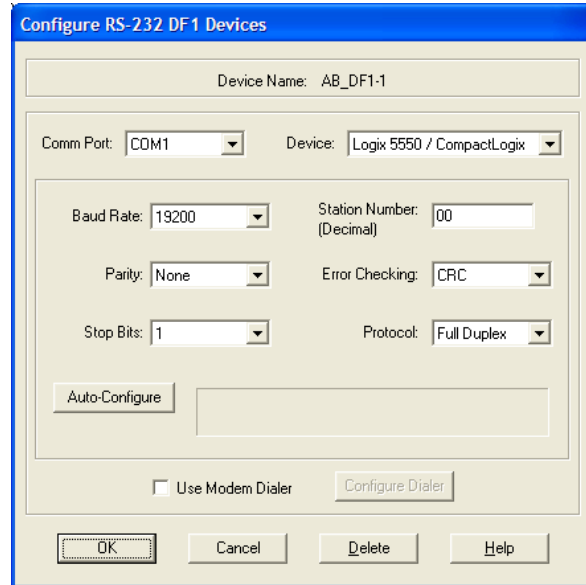


This action opens the *Configure Drivers* dialog box.



Note: If the list of configured drivers is blank, you must first choose and configure a driver from the Available Driver Types list. The recommended driver type to choose for serial communication with the processor is *RS-232 DF1 Devices*.

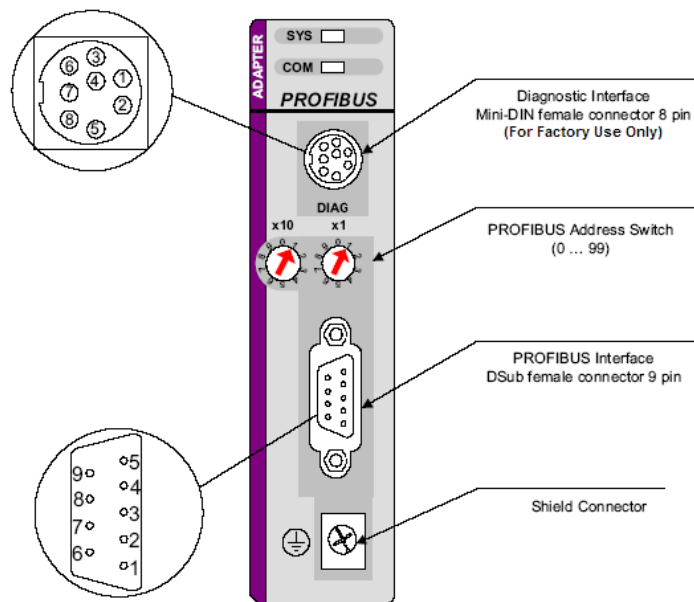
- 3 Click to select the driver, and then click **CONFIGURE**. This action opens the *Configure RS-232 DF1 Devices* dialog box.



- 4 Click the **AUTO-CONFIGURE** button. *RSLinx* will attempt to configure your serial port to work with the selected driver.
- 5 When you see the message *Auto Configuration Successful*, click the **OK** button to dismiss the dialog box.

Note: If the auto-configuration procedure fails, verify that the cables are connected correctly between the processor and the serial port on your computer, and then try again. If you are still unable to auto-configure the port, refer to your *RSLinx* documentation for further troubleshooting steps.

1.10 Adapter (PROFIBUS-DP-Slave)



2 Configuration and Start-Up

In This Chapter

❖ RSLogix 5000.....	38
❖ RSLogix 500.....	42
❖ Slave Configuration	45

The following sections will describe the individual steps for configuration and start-up of the PS69-DPS module. Install the PROFIBUS Slave module into a free slot in the CompactLogix or MicroLogix controller. Information regarding installation of communication modules in CompactLogix or MicroLogix systems can be found in the section Installation and Wiring or in the Rockwell installation manual for the appropriate controller system. The slave module must be within 6 modules of the I/O bank's power supply.

- Configuration and parameterization of the module is carried out in three steps
- Configuration of the module in a CompactLogix or MicroLogix project of the RSLogix5000 or RSLogix 500 programming tool.
- Determine configuration method to be used by the Slave module during startup.
- Creating the data objects and the ladder diagram in RSLogix5000

2.1 RSLogix 5000

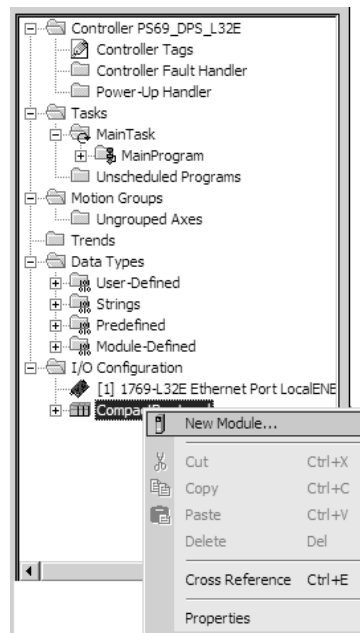
This section contains instructions for configuring the PS69-DPS module in a CompactLogix system using RSLogix5000.

Note: The simplest way to startup the module in RSLogix5000 project is to use the "PS69_DPS_L32E.ACD" example project. In this example project, the slot number in the configuration dialog of the module may have to be changed to match the users system.

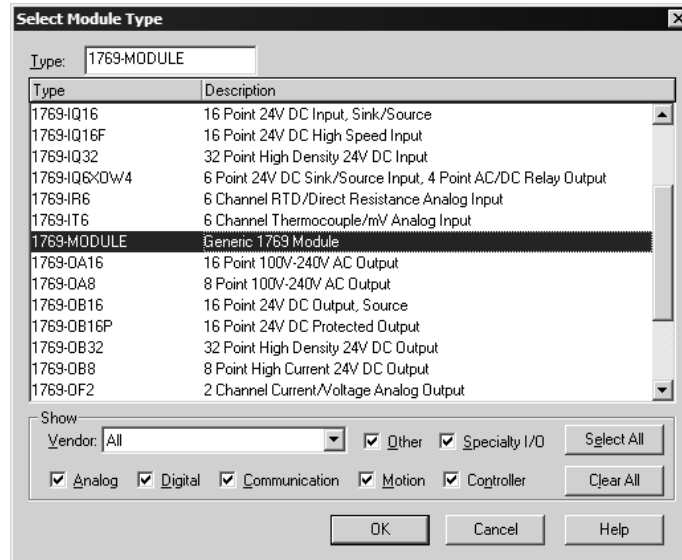
2.1.1 Module Selection

Create a new project in RSLogix5000 using a CompactLogix controller.

In the Controller Organization window, select CompactBus Local, and then click the right mouse button to open a shortcut menu. On the shortcut menu, choose New Module.



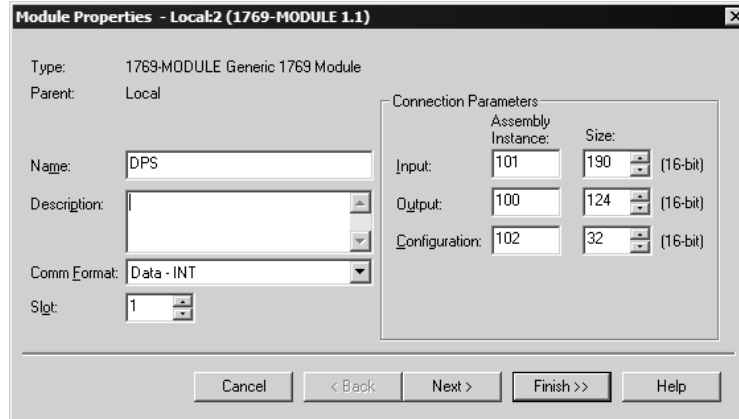
This action opens the following dialog box.



Select "**1769-MODULE Generic 1769 module**", and then click then OK.

2.1.2 Module Properties 1

The communications parameters for the module should be set as shown in the dialog below.



Select a name and enter a short description for the module. Select the slot number in which the module is installed in the controller. Select **Data - INT** as the **Comm_Format**. Set the connection parameters as they are shown in the dialog.

Connection Parameter	Assembly Instance	Size (in Words)
Input	101	68 + X ... 190
Output	100	2 + Y... 124
Configuration	102	32

X = Number of words configured for slave modules (PROFIBUS output data); output size can be in the range between 68 and 190 words

Y = Number of words configured for slave modules (PROFIBUS input data); input size can be in the range between 2 and 124 words

- **Input Size:** The input size must be at least 68 Words (136 Bytes). It must be large enough to accommodate the status information required by the module, which is 68 Words (136 Bytes) plus the number of PROFIBUS output data. The user can increase the size of this area using the size of each Output module connected. The PROFIBUS Output area starts with Word 68 (Byte 136).
- **Output Size:** The output size must be at least 2 Words (4 Bytes). It must be large enough to accommodate the command information required by the module, which is 2 Words (4 bytes), plus the number of PROFIBUS input data. The user can increase the size of this area using the size of each Input module connected. The PROFIBUS Input data image starts with byte 4.
- **Configuration Size:** The size for the configuration array must be always 32 Words.

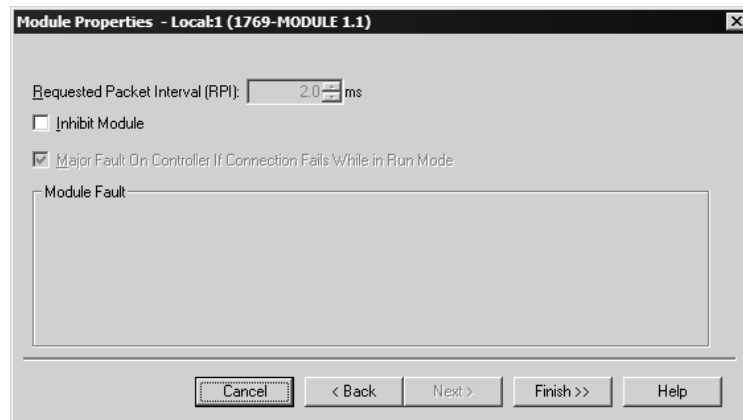
Note: If the parameters do not correspond to the template values, then the controller cannot establish communication with the module.

Select **Next >>** for the next configuration dialog.

2.1.3 Module Properties 2

The Requested Packet Interval RPI is shown in the following dialog box. Within this time interval, the I/O data between module and controller are exchanged.

It is not possible to change the RPI in this dialog separately for each module. The RPI can be changed in the properties dialog of the "CompactBus Local" for all I/O modules. Values in 1.0 ms steps are possible. The PROFIBUS PS69-DPS module supports all possible RPI values.



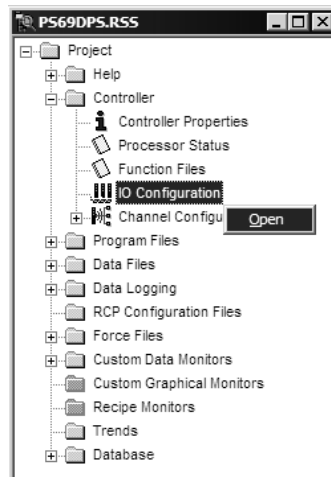
End the configuration of the module with **Finish>>**.

2.2 RSLogix 500

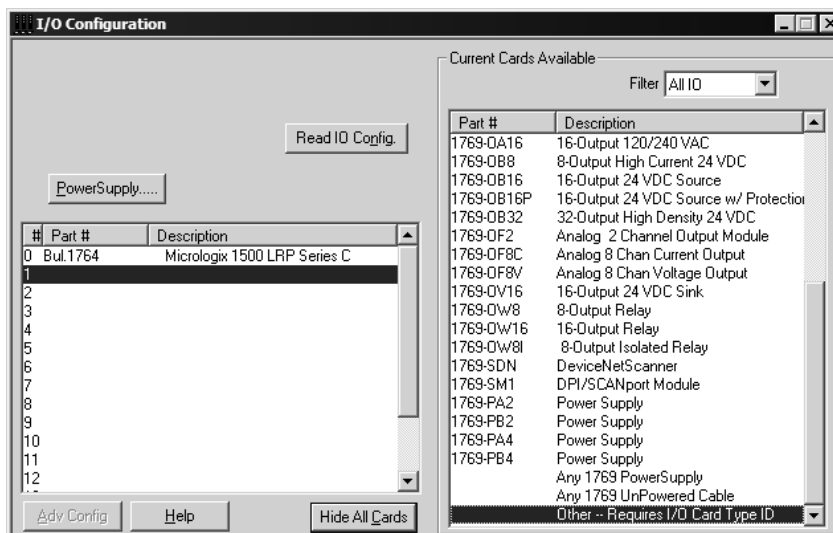
Contained in the sections below are the instructions for configuring the PS69-DPS module in a MicroLogix system using RSLogix500.

2.2.1 Module Selection

Create a new project in RSLogix500 using a MicroLogix1500 controller. Then the first step is to select the module and add it to your project. In the Controller Organization window, expand the Controller folder, and then select IO Configuration. Click the right mouse button to open a shortcut window, and then choose Open.



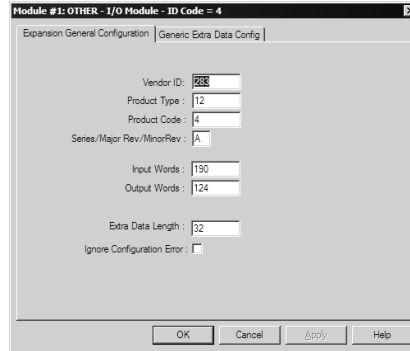
This action opens the following dialog box.



In the I/O Configuration dialog box, select the slot number in the left pane. In the right pane, select and double-click **Other - Requires I/O Card Type ID**.

2.2.2 Expansion General Configuration

Fill in the values on the Expansion General Configuration tab as shown in the following illustration.



Expansion General Configuration	Value
Vendor	283
Product Type	12
Product Code	4
Series/Major Rev/Minor Rev	A
Input Words	(68 + X) ... 190
Output Words	(2 + Y) ... 124
Extra Data Length	32

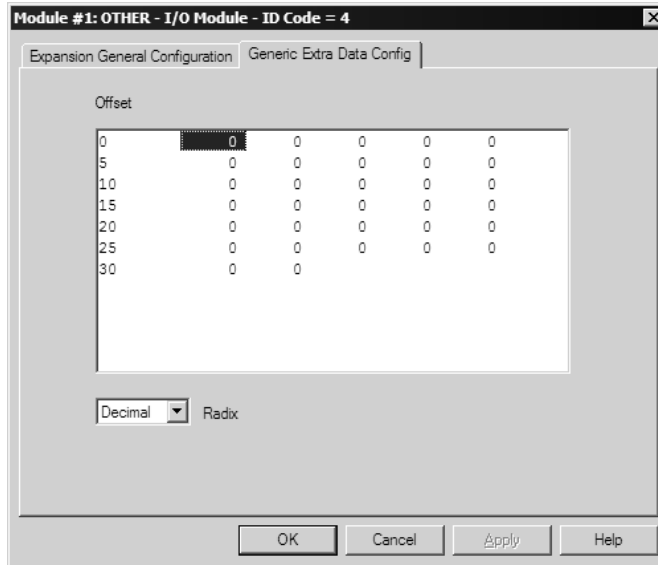
X = Number of Words configured for slave modules (PROFIBUS output data). The output size can range from 68 to 190 words.

Y = Number of Words configured for slave modules (PROFIBUS input data). The input size can range from 2 to 124 words.

- Vendor Name / Product Type / Product Code for the PS69-DPS module are 283,12,4.
- Input Size: This value must be large enough to accommodate 68 words (136 bytes) of status information required by the module, plus the number of words of PROFIBUS output data. You can increase the size of this area using the size of each Output module connected. The PROFIBUS Output data image starts with Word 68 (Byte 136).
- Output Size: This value must large enough to accommodate 2 Words (4 Bytes) of command information required by the module, which is 2 Words (4 bytes), plus the number of words of PROFIBUS input data. You can increase the size of this area using the size of each Input module connected. The PROFIBUS Input data image starts with byte 4.
- Configuration Size: The size for the configuration array must be always 32 Words.

Note: If the parameters do not correspond to the template values, then the controller cannot establish communication with the module.

2.2.3 Generic Extra Data Config



If you select the "**Generic Extra Data Config**" Tab you can enter a configuration for the module manually. For the first project it is not necessary to enter any configuration data here. If you don't enter any configuration data the slave will receive its configuration data from the Profibus master. For more information about the slave configuration see the following section "**Slave Configuration**". The meaning of each word of the configuration array can be found in section "**Configuration by Controller Application**".

Click OK to end the I/O configuration of the module.

2.3 Slave Configuration

2.3.1 General

The following section will detail the basics of configuring the PS69-DPS module. The PROFIBUS-DP Slave module does not require a configuration tool. There are two ways to configure the Slave module. These two methods are described in the following sections.

2.3.2 GSD File

A GSD file is kind of an electronic datasheet for a particular Slave device. The GSD-File for the PS69-DPS slave named "PSFT097A.gsd" is located on the DVD supplied with the module. You have to provide this file to the configuration tool for the network master. Refer to the Master's user manual of how to import GSD files.

2.3.3 Configuration by Master

The "Configuration by Master" is the easiest way to configure the Slave. The projects on the DVD contain examples which show the use of this method. During the network startup phase, the PROFIBUS Master sends the expected slave configuration over the network to compare it with the real configuration of the Slaves connected to the bus. The slave PS69-DPS automatically takes over the configuration which is sent by the master during its comparison of the configuration. This method is activated by default, since the parameter "Force User Config" in the configuration area is set to 0. The only setting required by the user is setting the Address rotary switches on the front of the module to the required network address.

Note: This is the easiest way to configure the Slave. But be aware that the master can send a new configuration to the slave at any time. This can cause inconsistency, if the new configuration does not match to the controller application. For more safety use the method "Configuration by Controller Application". With this method the slave module does not start any communication as long as the slave configuration and the master configuration don't match to each other.

2.3.4 Configuration by Controller Application

The second option to configure the Slave module is to let the controller application decide on the configuration. To do so the parameter "Force User Config" in the configuration array has to be set to 1. By setting this parameter and initialization of the other values the controller program can configure the slave. With this method the slave module will not start any network communication as long as the master and slave configuration don't match to each other. The following table shows the outline of the mapping of the configuration data image.

Word Offset	Configuration word	Data type	Low/High Byte	Description	Valid values
0	Local: 1:C.Data[0]	INT	LOW Byte	Busaddress	0 ... 125
			HIGH Byte	Force User Configuration	0 = ForceMasterConfig 1 = ForceUserConfig
1	Local: 1:C.Data[1]	INT		Reserved	
2	Local: 1:C.Data[2]	INT		Watchdog Time	0 ... FFFFh
3	Local: 1:C.Data[3]	INT		Number of valid config bytes (starting with Local:1:C.Data[8])	2 ... 48
4	Local: 1:C.Data[4]	INT		Reserved	
5	Local: 1:C.Data[5]	INT		Reserved	
6	Local: 1:C.Data[6]	INT		Reserved	
7	Local: 1:C.Data[7]	INT		Reserved	
8	Local: 1:C.Data[8]	INT	LOW Byte	Module 1 Type	see table "Module Types"
			HIGH Byte	Module 1 Length	see table "Module Types"
9	Local: 1:C.Data[9]	INT	LOW Byte	Module 2 Type	see table "Module Types"
			HIGH Byte	Module 2 Length	see table "Module Types"
...	
...	
31	Local: 1:C.Data[31]	INT	LOW Byte	Module 24 Type	see table "Module Types"
			HIGH Byte	Module 24 Length	see table "Module Types"

2.3.5 Explanation of settable configuration values

Busaddress

The valid PROFIBUS address range is from 0 to 125. The module has two rotating address switches to set the network address from 0 to 99.

With the rotating switches, however, you are not able to select bus addresses above 99. If you choose 0 on the address switches, then the module will take the address parameter from the configuration data array. Now you are able to set up bus addresses above 99.

Address Switches	Configuration	Address Parameter	Active Bus Address	Description
1 .. 99	XX		1 .. 99	Address switches are valid
0		0 .. 125	0 .. 125	Configuration parameter is valid
0		> 125	XX	Invalid (will cause an initialization error)

XX - Don't Care

Force User Configuration

If the parameter *ForceUserConfiguration* is set to 1, the slave will not start its network communication until the master and slave configurations match each other. If this value is set to 0, the slave accepts any valid configuration sent from the master.

Watchdog Timeout

The Slave module supervises its I/O exchange with the controller with a timeout. If the controller does not update the output data within this time, the Slave stops the cyclic data exchange to the master and goes into a safe state.

If the parameter *ForceUserConfiguration* is set to 0 then the module calculates automatically a timeout value by the RPI (Requested Packet Interval). The calculated watchdog results to 2 x the RPI (+/- 5ms). The smallest watchdog value is 15 ms. The module will round the watchdog to a multiples of 5 ms.

$$\text{WATCHDOG_TIME (ms)} = \text{MAX} (2 * \text{RPI} ; 15) (+/-5\text{ms})$$

If the parameter "ForceUserConfiguration" is set to 1 the module will take the watchdog value from the configuration array. Make sure that the watchdog is not smaller than the RPI. The module will also round the watchdog to the next multiple of 5ms.

Number of Valid Configuration Bytes

This parameter holds the number of valid configuration bytes that define PROFIBUS Input/Output modules.

Module n Type / Module n Length

The PS69-DPS PROFIBUS-DP Slave offers a flexible, modular composition of its I/O data. This means that parts of the input and output image can be viewed as single modules. The master can put the different modules from the PROFIBUS-DP Slave to different locations in its I/O area. The individual configured modules are mapped linearly in the I/O area of the Slave module. It is possible to configure up to 24 I/O modules. A module is defined by a Module Type and its Module Length:

Parameter	Data type	Valid values	Description
Module Type	SINT	0 = IN Byte	Input Byte without consistence
		1 = IN Word	Input Word without consistence
		2 = OUT Byte	Output Byte without consistence
		3 = OUT Word	Output Word without consistence
		4 = IN Byte con	Input Byte with consistence
		5 = IN Word con	Input Word with consistence
		6 = OUT Byte con	Output Byte with consistence
		7 = OUT Word con	Output Word with consistence
		8 = Blank space	Blank space
ModuleLength	SINT	0	1 Byte/Word
		1	2 Byte/Word
		2	3 Byte/Word
		3	4 Byte/Word
		4	8 Byte/Word
		5	12 Byte/Word
		6	16 Byte/Word
		7	20 Byte/Word
		8	32 Byte/Word
		9	64 Byte/Word

Note: Please notice the definition of Input/Output modules and do not confuse them with the input and output area of the module in the controller memory map. PROFIBUS gives a clear definition of Inputs/Outputs. Inputs and Outputs modules are always defined from viewpoint of the PROFIBUS master. If you configure an Output module you will see this in the input area of the communication module, because the input area of the controller memory map is the output area from point of view of a PROFIBUS master. The same applies to an Input module. If you define an Input module it is mapped in the output area of the controller memory map, because the output area is the input area from viewpoint of a PROFIBUS master.

The projects on the DVD can also be used as an example for a module configuration by the controller application. You only have to set the parameter "ForceUserConfiguration" from 0 to 1 in the configuration array. In chapter "RSLOGIX SAMPLE PROGRAM" later in this manual explains the predefined configuration parameter of the sample project.

3 RSLogix Example Program

In This Chapter

- ❖ CompactLogix I/O Example..... 50
- ❖ CompactLogix Messaging Example 52

The ProSoft Solutions DVD contains example Ladder Logic programs. These examples should be used as templates for starting your project. An explanation of each project is in the following sections. If you are using another type of CompactLogix Controller, change the ControllerType in RSLogix and then store it to your individual project. If you setup up a new controller project you can use the Copy and Paste functionality of RSLogix to transfer the user defined data types or ladder logic needed with the module PS69-DPS from the template projects to your own application

Sample Project	Controller Type	RSL Version	Description
AOIPS69DPS.L5X	1769-L32E	5000 V16	CompactLogix Add-On Instruction
PS69_DPS_L32E_V13.ACD	1769-L32E	5000 V13	Basic CompactLogix I/O example
PS69_DPS_messaging_L32E_v13.ACD	1769-L32E	5000 V13	Basic CompactLogix messaging example
PS69_DPS_ML15.rss	MicroLogix 1500	500 V6.2	Basic MicroLogix I/O example

3.1 CompactLogix I/O Example

This ladder logic program is a basic example for the setup of the PROFIBUS-DP Slave communications module "PS69-DPS" in RSLogix5000. This example can be used to start a project when using a CPU 1769-L32E. Basic PROFIBUS I/O data exchange is shown. Details on the Subroutines created and the User Defined Data Types are as follows.

- **MainRoutine:** The MainRoutine calls all of the following routines. This routine also contains a simple I/O transfer function block.
- **DPS_Update_Ext_Data:** The DPS_Update_Ext_Data routine serves as an example of how the user can map each of the different extended status information provided in the ExtStatusInfo array. This routine evaluates the content of the ExtStaSelect value and copies the information into the appropriate user defined data type.
- **SR_Copy_Input:** The SR_Copy_Input routine on every scan updates the DpsInputArray structure with the Input Data of the module.
- **SR_Copy_Output:** The SR_Copy_Output routine on every scan updates the DpsOutputArray structure with the Output Data of the module.

Numerous user defined data types have been created to make it easier to address different elements of the Input and Output array of the module. The two main structures are DpsInputArray and DpsOutputArray. Their definitions and the structures included in each are shown in the following tables.

The I/O example program can also be used as an example for the two methods of configuration "Configuration by Master" (ForceMasterConfig) and "Configuration by Controller Application" (ForceUserConfig). The configuration array of the sample project is pre-initialized with following values:

Configuration word	Data type	Low/High Byte	Description	Configured values	Explanation
0	INT	LOW Byte	Busaddress	2	This address will be active if the rotary switches of the module are adjusted to "00"
		HIGH Byte	Force User Configuration	0	0 = ForceMasterConfig The module will take over the configuration from the master.
1	INT		Reserved		
2	INT		Watchdog Timeout	C8h (200dec)	Watchdog 200 ms
3	INT		Number of valid config bytes (starting with Local:1:C.Data[8])	8	8 Bytes of the module definition array are valid
4	INT		Reserved		
5	INT		Reserved		
6	INT		Reserved		
7	INT		Reserved		
8	INT	LOW Byte	Module 1 Type	4	Module 1 Type: Input Byte with consistency
		HIGH Byte	Module 1 Length	3	Module 1 Length: 4 (byte)

Configuration word	Data type	Low/High Byte	Description	Configured values	Explanation
9	INT	LOW Byte	Module 2 Type	5	Module 2 Type: Input Word with consistency
		HIGH Byte	Module 2 Length	1	Module 2 Length: 2 (word)
10	INT	LOW Byte	Module 3 Type	6	Module 3 Type: Output Byte with consistency
		HIGH Byte	Module 3 Length	3	Module 3 Length: 4 (byte)
11	INT	LOW Byte	Module 4 Type	7	Module 4 Type: Output Word with consistency
		HIGH Byte	Module 4 Length	1	Module 4 Length: 2 (word)
...			...		
...			...		
31	INT	LOW Byte	Module 24 Type	0	
		HIGH Byte	Module 24 Length	0	

The parameter "ForceUserConfiguration" is initialized with 0. If you want to activate the pre-defined modules set this parameter to 1. When this parameter is active the master configuration has to match exactly with this configuration otherwise the slave will not start the communication with the master.

The parameter busaddress is independent from the parameter "ForceUserConfiguration". This parameter will be active, if the rotary address switches of the module are set to "00".

3.2 CompactLogix Messaging Example

This ladder logic program is a CIP messaging example for the setup of the PROFIBUS-DP Slave communications module "PS69-DPS" in RSLogix5000. This example can be used to start a project when using a CPU 1769-L32, which supports CIP messaging. Basic PROFIBUS I/O data exchange and all messaging function examples are shown. Details on the Subroutines created and the User Defined Data Types are as follows.

- **MainRoutine:** The MainRoutine calls all of the following routines based on conditions such as doing a diagnostic request or to check the progress of each individual DPV1 function issued by the Master. This routine also contains a simple I/O transfer function block.
- **Diagnostic_Req:** This subroutine Diagnostic_Req assembles a Diagnostic Request message which will be sent to the Master. A CIP Generic Message is used to send this message.
- **Diagnostic_Req_Progress:** This subroutine Diagnostic_Req_Progress checks the status of the diagnostics request message sent to the Master. When the Diagnostic request is done it will increment a status counter to check how many requests have been send successfully and how many failed.
- **DPV1C1_Alarm_Req:** This subroutine DPV1C1_Alarm_Req assembles a DPV1 Alarm message which will be sent to the Master. A CIP Generic Messages is used to send this message.
- **DPV1C1_Alarm_Req_Progress:** This subroutine DPV1C1_Alarm_Req_Progress checks the status of the Alarm request message sent to the Master. When the Alarm request is done it will increment a status counter to record how many requests have been send successfully and how many have been failed.
- **DPV1C1_Progress:** This subroutine DPV1C1_Progress checks the inprogress bit of each service and sends the appropriate response. CIP Generic Messages are used to send the response message from the Slave to the Master.
- **DPV1C1_Read_Resp:** This subroutine DPV1C1_Read_Resp assembles the DPV1 Read response message. A CIP Generic Message is used to send this message. This routine will return immediately if there is still a DPV1 Read Response in progress.
- **DPV1C1_Read_Resp_Progress:** This subroutine DPV1C1_Read_Resp_Progress checks the status of the DPV1 Read Response sent to the Master. When the Read response request is done it will increment a status counter to record how many response requests have been sent successfully and how many failed.
- **DPV1C1_Write_Resp:** This subroutine "DPV1C1_Write_Resp" assembles the DPV1 Write response message. A CIP Generic Message is used to send this message. This routine will return immediately if there is still a DPV1 Write Response in progress.

- **DPV1C1_Write_Resp_Progress:** This subroutine DPV1C1_Write_Resp_Progress checks the status of the DPV1 Write Response message sent to the Master. When the Write response request is done it will increment a status counter to check how many response requests have been sent successfully and how many failed. If the Write Response message was successful the first Rung will copy the Write data to a local buffer which can be transferred with a CIP response message.
- **SR_Copy_Input:** The SR_Copy_Input routine on every scan updates the DpsInputArray structure with the Input Data of the module.
- **SR_Copy_Output:** The SR_Copy_Output routine on every scan updates the DpsOutputArray structure with the Output Data of the module.
- **SR_Main_Init:** Initializes several variables used by different routines.

Numerous user defined data types have been created to make it easier to address different elements of the Input and Output array of the module. The two main structures are DpsInputArray and DpsOutputArray.

4 Diagnostics and Troubleshooting

In This Chapter

- ❖ Hardware Diagnostics (LED) 56
- ❖ Troubleshooting..... 58

This section describes the possible diagnostics and troubleshooting procedures for the PS69-DPS Slave module.

4.1 Hardware Diagnostics (LED)

The following section contains the LED diagnostic indications and their meaning for both the CPU in use and the PS69-DPS module.

4.1.1 CompactLogix

The following table shows the possible LED indications of the CompactLogix CPU module.

Indicator	Color/Status	Description
RUN	Off	No task(s) running; controller in Program mode
	Green	One or more tasks are running; controller is in the Run mode
FORCE	Off	No forces enabled
	Amber	Forces enabled
	Amber Flashing	One or more input or output addresses have been forced to an On or Off state, but the forces have not been enabled.
OK	Off	No power applied
	Green	Controller OK
	Red flashing	Recoverable controller fault
	Red	Non-recoverable controller fault: Cycle power. The OK LED should change to flashing red. If LED remains solid red, replace the controller.
I/O	Off	No activity; no I/O or communications configured
	Green	Communicating to all devices
	Green flashing	One or more devices not responding
	Red flashing	Not communicating to any devices controller faulted

4.1.2 MicroLogix 1500

To identify problems via possible LED indications of a MicroLogix1500 controller refer to the MicroLogix1500 User Manuals chapter "Troubleshooting Your System". Here you will find a detailed description of fault indications and possible reasons.

4.1.3 PS69 LEDs

The LEDs as shown on the front panel will be used to indicate status information of the PS69-DPS Slave module. Each LED has a specific function during Run time, configuration download, and error indications. The following table shows the reaction of each during these states for the Slave.

LED	Color	State	Description
SYS	Yellow	Flashing cyclic at 1Hz	Device is in boot loader mode and is waiting for firmware download.
	Yellow	Flashing cyclic at 5Hz	Firmware download is in progress.
	Yellow	Flashing irregular (*)	Hardware or runtime error detected.
	Green	Static On	Slave in cyclic data exchange with DP Master.
	Green	Flashing cyclic at 5Hz	Slave has no cyclic data exchange with DP Master.
	Green	Flashing irregular (*)	Power Up: Configuration missing or faulty, device needs commissioning. Runtime: Host Watchdog timeout
	Off	Off	Device has no power supply or hardware defect or PLC holds the module in reset.
COM	Green	On	Slave has received parameter data/configuration data from the DP Master and has reached the state of data exchange.
	Red	On	unused
	Off	Off	Slave has not reached the state data exchange.

(*) 3 times fast at 5 Hz, 8 times between 0,5Hz and 1Hz

4.2 Troubleshooting

Troubleshooting of the system is done by examining the LEDs on the front panel of the CPU and the LEDs on the front of the module. The following sections contain some troubleshooting ideas.

4.2.1 CompactLogix I/O LED

Communication between the module and controller is displayed via the I/O LED of the Controller. The proper communication state is reached, if the I/O LED of the CompactLogix Controller is static Green. If this LED is flashing or off, no communication has been established between controller and the Slave Module.

4.2.2 MicroLogix Fault LED

When the Fault LED of the MicroLogix Controller is off, the module is not in a fault state.

If there is a problem with the expansion module, the Fault LED will be flashing red. To diagnose the error, go online with your RSLogix500 project and open up the processor status dialog box. Click the Error tab to view the fault reason.

4.2.3 SYS and COM Status LEDs

This PS69-DPS module has two bicolor status LEDs. They inform the user about the communication state of the module.

- The **SYS**-LED shows the common system status of the card. It may flash yellow or green.
- The **COM**-LED displays the status of the PROFIBUS communication. It can be solid green or off.

If the SYS-LED is solid Green and the COM-LED solid green, the card is in cyclic data exchange with the Master and the communication is running with outfault.

4.2.4 Error Sources and Reasons

This section describes typical problems, error sources and questions that come up while commissioning the PROFIBUS-DP Slave module PS69-DPS. The following table summarizes the typical error sources and gives a hint of possible reasons for the problem.

Behavior	Significance	Typical Reason	Help
CompactLogix I/O LED is flashing Green	No communication with the PS69 module (or other modules)	Modules slot number in RSLogix program does not match with the physical slot of the module Configured Input / Output / Configuration array size is wrong	Check modules slot number in RSLogix project Compare configured Input / Output size with required values

Behavior	Significance	Typical Reason	Help
MicroLogix Fault LED is flashing Red	No communication with the PS69 module (or other modules)	Modules slot number in RSLogix program does not match with the physical slot of the module Configured Vendor ID / Module ID / Input / Output / Configuration array size is wrong	Check modules slot number in RSLogix project Compare configured Input / Output size / Vendor ID / Module ID with required values
PS69-DPS COM LED is off SYS LED Flashing irregular green	Configuration missing or faulty	No configuration or faulty stored	Check initialization values of the configuration array Check the value "LastError" in SlaveStatusField to determine the error reason
	Watchdog expired	Watchdog value in configuration is smaller than RPI (Requested PackedIntervall)	Increase Watchdog value in configuration array Module has to be reset
PS69-DPS COM LED is off and SYS LED flashing cyclic fast green	Application is not ready	PLC is not in RUN Mode. PLC application has set the NRDY bit. PLC has no I/O communication with the module	Bring PLC into RUN Mode. Check that the PLC application has deleted the NRDY bit. Check PLC's I/O LED
	Master and Slave Configuration mismatch	The configuration of the master which wants to communicate with the module don't match to the configuration of the slave module	Use the ExtStainfo in SlaveStatusField to compare what the expected configuration is from the master and the modules configuration
	Network problem	No physical network connection No master present who wants to communicate	Check if the slave module is properly connected to the PROFIBUS Network Check if bus activity can be detected in "IrqCounter" in SlaveStatusField Check if a master is present who wants to communicate to the module and check if the slave address is correct
Master output data can not be found in RSLogix program	Input array mismatch	Configured input size in RSLogix too small	Check if the configured input size in RSLogix covers the mandatory size of 136 byte status data plus the size of the outputs configured.
Inputs are not transferred to Master although PROFIBUS is running	Output array mismatch	Configured output size in RSLogix too small	Check if the configured output size in RSLogix covers the mandatory size of 4 byte status data plus the configured PROFIBUS input data

4.2.5 Cable

- Check that the cable is wired correctly (page 99).
- Check to confirm that the bus termination resistors are switched on at the beginning and the end of the cable and switched off at all other connectors in between.

5 Reference

In This Chapter

❖ Specifications	62
❖ RSLogix5000 User Defined Data Types.....	66
❖ PROFIBUS Functionality	73
❖ Communication	75
❖ Constructing a Bus Cable for PROFIBUS DP	99

5.1 Specifications

The PS69-DPS module expands the functionality of Rockwell Automation's CompactLogix or MicroLogix to include PROFIBUS DP V0/V1. The PS69-DPS is a more cost-effective option offering more features than the PS69-PDPS, and supports both I/O control and messaging. Explicit ladder logic CIP message blocks provide slave status diagnostic data and acyclic messaging.

The PS69-DPS interface appears to the CompactLogix or MicroLogix controller as a standard I/O module allowing it to be configured via RSLogix5000, or configuration can be transferred from the Master to the PS69-DPS. For third party configuration a GSD file is supplied. The slave interface possesses a diagnostic interface and has rotary switches for setting of the bus address. Complete program examples for simple and quick start-up are available. Each module is equipped with LEDs to display communication and device status.

5.1.1 General Specifications

- Single Slot - 1769 backplane compatible
- The module is recognized as an Input/Output module and has access to processor memory for data transfer between processor and module
- Ladder Logic is used for data transfer between module and processor. Sample ladder file included.
- Configuration data obtained from configuration text file downloaded to module. Sample configuration file included.
- Supports all CompactLogix processors: L20/L30/L31/L32/L35 and L43 (L43 supported with RSLogix 5000 v16)
- Also supports MicroLogix 1500 LRP

5.1.2 Hardware Specifications

Specification	Description
Processor	EC1-160P with integrated ASPC2
PROFIBUS Interface	RS-485, max. 12 MBaud, potential free, according EN 50170
Diagnostic Interface	RS232, PS/2 Mini DIN female connector, 9600 Baud, non isolated
Power Supply	+5 V \pm 5 % / 260 mA
Max. Distance Rating	max. 6 modules to the power supply module
Dimensions	Standard 1769 Single-slot module
Burst	EN 61000-4-4, 2 kV, 5 kHz
Surge	EN 61000-4-5, 2 kV common mode, 1 kV differential mode
ESD	EN 61000-4-2, 4 kV contact, 8 kV air, 4 kV indirect
Radiated/ Conducted Immunity	EN 61000-4-3, 10 V/m, 30...1000 MHz, 80% AM, 1 kHz sinewave EN 61000-4-6, 10 V, 0,15...30 MHz
Radiated/ Conducted Emission	EN 55011 Class A
Vibration/Shock	IEC 600068-2-6, 10-150 Hz, \pm 0,75 mm, \pm 1 g, 1 Octave/min IEC 600068-2-27, 15 g, 11 ms
Operating Temp.	0 to 60°C (32 to 140°F)
Relative Humidity	5 to 95% (non-condensing)
Agency Certification: UL/CE	C-UL certified, UL 508 listed, CE
PROFIBUS conformance	certified

5.1.3 Functional Specifications

Specification	Description
Input data	max. 244 Bytes
Output data	max. 244 Bytes
DPV1 services	Read / Write class 1, Alarm
GSD file	PSFT097A.gsd

Up to 32 PROFIBUS devices can be connected to one bus segment. If several bus segments are linked to each other with repeaters, there can be up to 127 devices on the network.

The maximum length of a bus segment depends on the baud rate used. Only PROFIBUS certified cable, preferably Belden 3079A cable (or equivalent), should be used.

Baud rate in kBit/s	Max. Distance
9.6	1,200 m / 4000 ft
19.2	1,200 m / 4000 ft
93.75	1,200 m / 4000 ft
187.5	1,000 m / 3280 ft
500	400 m / 1300 ft
1,500	200 m / 650 ft
3,000	100 m / 325 ft
6,000	100 m / 325 ft
12,000	100 m / 325 ft

Parameter	Value
Impedance	135 to 65 Ohm
Capacity	< 30 pF/m
Loop resistance	110 Ohm/km
Wire gauge	0.64 mm

Product Comparison

	MVI69-PDPS	PS69-DPS
Cross-platform similarity	X	X
PROSOFT.fdt network configuration	X	X
Cyclic I/O (words)	200 in/ 200 out	244 in/ 244 out
FDT communication support (serial only)	X	X
PROFIBUS PNO Certified		X
PROFIBUS DPV1 Support (Acyclic messages, Acyclic Read/Write Data)		X
Price/Performance for OEM & Machine Builders		X
RSLogix 5000 configuration		X
CIP Messaging (better performance, ladder communication)		X
Lower Cost		X

5.1.4 PROFIBUS Interface

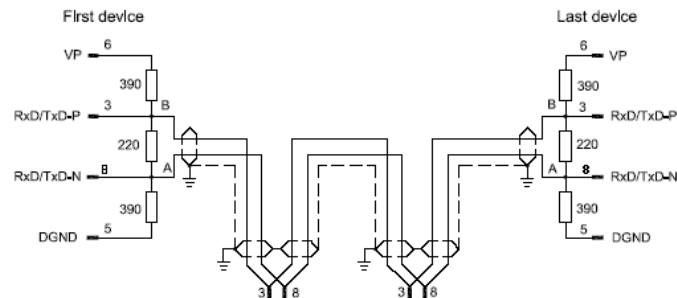
Isolated RS-485 interface per EN 50170.

3	RxD/TxD-P	Receive/Send Data-P respectively connection B plug
5	DGND	Reference potential
6	VP	Positive power supply
8	RxD/TxD-N	Receive/Send Data-N respectively connection A plug

Please ensure that termination resistors are available at both ends of the cable. If special PROFIBUS connectors are being used, these resistors are often found inside the connector and must be switched on. For baud rates above 1.5 MBaud use only PROFIBUS connectors, which also include additional inductance. It is not permitted to have T stubs at high baud rates.

Use only a special cable which is approved for PROFIBUS-DP. Make a solid connection from the cable shield to ground at every device and make sure that there is no potential difference between the grounds at the devices.

If the PS69 is linked with only one other device on the bus, both devices must be at the ends of the bus line. The reason is that these devices must deliver the power supply for the termination resistors. Otherwise the Master can be connected at any desired position.



5.2 RSLogix5000 User Defined Data Types

Contained in this appendix are all the user defined data types created and used in the example programs.

5.2.1 Input: *DPS_INPUT_ARRAY*

Name	Data Type	Description
DevStaReg	DPS_DEV_STATUS_REGISTER	Device Status
FwRev	DPS_FW_REVISION	Firmware Revision
StaField	DPS_STATUS_FIELD	DPS Status Registers
PBOutputData	INT[32]	PROFIBUS Output Data

5.2.2 Input: *DPS_DEV_STATUS_REGISTER*

Name	Data Type	Description
Reserved0	BOOL	Reserved
Reserved1	BOOL	Reserved
Reserved2	BOOL	Reserved
Reserved3	BOOL	Reserved
Reserved4	BOOL	Reserved
Com	BOOL	Communication
Run	BOOL	Running
Rdy	BOOL	Ready
Reserved5	SINT	Reserved
Reserved6	SINT	Reserved
Reserved7	SINT	Reserved

5.2.3 Input: *DPS_FW_REVISION*

Name	Data Type	Description
FwMinor	SINT	Firmware Minor Revision
FwMajor	SINT	Firmware Major Revision
Reserved	INT	Reserved

5.2.4 Input: *DPS_STATUS_FIELD*

Name	Data Type	Description
ExtStaSelect	INT	Extended Status Select
ExtStaLen	INT	Extended Status Length
Baudrate	INT	Slave Baud rate
Busaddress	SINT	Slave Bus Address
UserFlags	SINT	User Flags
Ident	INT	Slave Ident Number
TaskState	INT	Slave Task State
InputDataLen	INT	Slave Input Data Length
OutputDataLen	INT	Slave Output Data Length
ErrorCount	INT	Slave Error Count
LastError	SINT	Slave Last Error
Pad	SINT	Reserved
WatchdogTime	INT	Slave Watchdog Time
IrqCounter	INT	Slave Interrupt Counter
C1Ind	DPS_DPV1C1_RW_INDICATION	DPV1 Class 1 Indication Registers
ExtStatusInfo	SINT[96]	Extended Status Information

5.2.5 Output: *DPS_OUTPUT_ARRAY*

Name	Data Type	Description
DevCmdReg	DPS_DEV_COMMAND_REGISTER	Device Command Register
PBInputData	INT[32]	PROFIBUS Input Data for Master

5.2.6 Output: *DPS_DEV_COMMAND_REGISTER*

Name	Data Type	Description
Reserved0	BOOL	Reserved
Reserved1	BOOL	Reserved
Reserved2	BOOL	Reserved
Reserved3	BOOL	Reserved
Reserved4	BOOL	Reserved
NRdy	BOOL	Application Not Ready
Init	BOOL	Init (Warm boot)
Reset	BOOL	Reset (Cold boot)
Reserved5	SINT	Reserved
Reserved6	SINT	Reserved
Reserved7	SINT	Reserved

5.2.7 APP_CONSTANT_PATTERN

Name	Data Type	Description
UserExtDiagData	SINT[32]	Constants for Extended Diagnostics.
UserAlarmData	SINT[32]	Constants for Alarm Data.

5.2.8 APP_DPV1_PROG_CONTROL

Name	Data Type	Description
MainInitDone	BOOL	Main Initialization Complete
Dpv1ReadResplnProgress	BOOL	DPV1 Read Response in Progress
Dpv1WriteResplnProgress	BOOL	DPV1 Write Response in Progress
Dpv1AlarmReqSend	BOOL	DPV1 Alarm Request Flag
Dpv1AlarmReqInProgress	BOOL	DPV1 Alarm Request in Progress
DpsDiagReqSend	BOOL	DPS Diagnostics Request Flag
DpsDiagReqInProgress	BOOL	DPS Diagnostic Request in Progress
Reserved0	BOOL	Reserved
Reserved1	BOOL	Reserved
Reserved2	BOOL	Reserved
Reserved3	BOOL	Reserved
Reserved4	BOOL	Reserved
Reserved5	BOOL	Reserved
Reserved6	BOOL	Reserved
Reserved7	BOOL	Reserved
Reserved8	BOOL	Reserved
DPV1RWIndCnt	BOOL	DPV1 R/W Indication Counter
MainInitDone	SINT	Main Initialization Complete

5.2.9 APP_DPV1_STAT_COUNTER

Name	Data Type	Description
NumReadWrite	DINT	Number of DPV1 Read Write Response Send
ReadRespPos	DINT	Number of DPV1 Read Response Successful
ReadRespNeg	DINT	Number of DPV1 Read Response Failed
WriteRespPos	DINT	Number of DPV1 Write Response Successful
WriteRespNeg	DINT	Number of DPV1 Write Response Failed
NumAlarmRequest	DINT	Number of DPV1 Alarm Requests send
AlarmRequestPos	DINT	Number of DPV1 Alarm Requests Successful
AlarmRequestNeg	DINT	Number of DPV1 Alarm Requests Failed
NumDiagReq	DINT	Number of Diagnostic Report Send
DiagReqPos	DINT	Number of Diagnostic Report Successful
DiagReqNeg	DINT	Number of Diagnostic Report Failed

5.2.10 DPS_DIAGNOSTIC_CONFIRM

Name	Data Type	Description
Reserved1	INT	Reserved
Reserved2	INT	Reserved
Answer	SINT	Answer of Diag Req
Failure	SINT	Failure
Reserved4	INT	Reserved
Reserved5	INT	Reserved
Reserved6	INT	Reserved
Reserved7	SINT	Reserved
ExtDiagDataCnt	SINT	Extended Diagnostics Data Count
Mode	SINT	Mode
Function	SINT	Function

5.2.11 DPS_DIAGNOSTIC_REQUEST

Name	Data Type	Description
Reserved1	INT	Reserved
Reserved2	INT	Reserved
Reserved3	INT	Reserved
Command	SINT	Command for Diag Req = 24
Reserved4	SINT	Reserved
Reserved5	INT	Reserved
Reserved6	INT	Reserved
Reserved7	SINT	Reserved
ExtDiagDataCnt	SINT	Number of ext. diag data
Mode	SINT	Diag mode: (0 = default, 1 = suppress Ext.DiagBit)
Function	SINT	Send diag once (fix 18)
DiagData	SINT[32]	Extended diag data (format see PROFIBUS Norm)

5.2.12 DPS_DPV1C1_ALARM_CONFIRM

Name	Data Type	Description
Reserved1	INT	Reserved
Reserved2	INT	Reserved
Answer	SINT	DPV1 Alarm Answer Flag
Failure	SINT	DPV1 Alarm Failure
Reserved3	INT	Reserved
Reserved5	SINT	Reserved
Reserved6	SINT	Reserved
SlotNumber	INT	DPV1 Alarm Slot Number
SequenceNumber	SINT	DPV1 Alarm Sequence Number
DataCnt	SINT	DPV1 Alarm Data Count
AlarmType	SINT	DPV1 Alarm Type
Specifier	SINT	DPV1 Alarm Specifier

5.2.13 DPS_DPV1C1_ALARM_REQUEST

Name	Data Type	Description
Reserved1	INT	Reserved
Reserved2	INT	Reserved
Reserved3	INT	Reserved
Command	SINT	DPV1 Alarm Command
Reserved4	SINT	Reserved
Reserved5	SINT	Reserved
Reserved6	SINT	Reserved
SlotNumber	INT	DPV1 Alarm Slot Number
SequenceNumber	SINT	DPV1 Alarm Sequence Number
DataCnt	SINT	DPV1 Alarm Data Count
AlarmType	SINT	DPV1 Alarm Type
Specifier	SINT	DPV1 Alarm Specifier
AlarmData	SINT[28]	DPV1 Alarm Data Array

5.2.14 DPS_DP1C1_RW_INDICATION

Name	Data Type	Description
ReadReq	BOOL	Indicates a Read request
WriteReq	BOOL	Indicates a Write request
Reserved2	BOOL	Reserved
Reserved3	BOOL	Reserved
Reserved4	BOOL	Reserved
Reserved5	BOOL	Reserved
Reserved6	BOOL	Reserved
Reserved7	BOOL	Reserved
RwCnt	SINT	ReadWrite indication counter
MaAdr	SINT	Address of requesting master
Slot	SINT	Requested Slot Number
Index	SINT	Requested Index
DataLen	SINT	Requested Data Length
Reserved8	SINT	Reserved
Reserved9	SINT	Reserved

5.2.15 DPS_DP1C1_RW_RESP_CONFIRM

Name	Data Type	Description
Reserved1	INT	Reserved
Reserved2	INT	Reserved
Answer	SINT	DPV1 R/W Answer
Failure	SINT	DPV1 R/W Failure
Reserved3	INT	Reserved
RwResp	SINT	Read Resp (=1) or Write Resp (=2)
Reserved4	SINT	Reserved
MaAdr	SINT	Reply of requesting Master Address
Slot	SINT	Reply of requested Slot Number
Index	SINT	Reply of requested Index
DataLen	SINT	Number of requested data
ErrCode1	SINT	Reply of Error code 1 according to DPV1
ErrCode2	SINT	Reply of Error code 2 according to DPV1
RWRespData	SINT[240]	DPV1 Write data

5.2.16 DPS_DPV1C1_RW_RESP_REQUEST

Name	Data Type	Description
Reserved1	INT	Reserved
Reserved2	INT	Reserved
Reserved3	INT	Reserved
Command	SINT	DPV1 Read Write Resp. Request (=17)
Reserved4	SINT	Reserved
RwResp	SINT	Read Resp (=1) or Write Resp (=2)
Reserved5	SINT	Reserved
MaAdr	SINT	Reply of requesting Master Address
Slot	SINT	Reply of requested Slot Number
Index	SINT	Reply of requested Index
DataLen	SINT	Reply of requested number of data
ErrCode1	SINT	Error code 1 according to DPV1, if occurs
ErrCode2	SINT	Error code 2 according to DPV1, if occurs
RWRespData	SINT[240]	DPV1 Read response data

5.3 PROFIBUS Functionality

5.3.1 DPV0 Services

DPV0 services in PROFIBUS refer to the cyclic data exchange mechanism between a class 1 master and a network slave. PROFIBUS-DP defines two types of masters. The class 1 master handles data communication with slaves assigned to it. A class 2 master should only be used for commissioning purposes. In a PROFIBUS telegram, class 1 masters and slaves transmit up to 244 bytes per telegram. Valid station addresses on PROFIBUS range from 0 to 125.

Fail Safe Mode

For safety reasons, the PROFIBUS master informs connected slaves of its current control status at certain intervals using a "Global Control" telegram. If the master goes to Clear Mode, the Fail Safe enabled slaves will switch to a Fail Safe state. Slaves capable of the Fail Safe state can be configured to either to hold the last state of the outputs or set its outputs to "0". Slaves that do not support the Fail Safe state set their outputs to "0".

Global Control

With the Global Control telegram, the master can send unsolicited commands like Sync/Unsync, Freeze/Unfreeze and Clear Data to a slave or a group of slaves for synchronization purposes. Group membership is defined during network start-up and can be set in the master configuration tool.

Sync and Freeze

Sync and Freeze are optional commands and slaves do not need to support them. However, they must be able to process the Global Control telegram. With a Freeze command, the master prompts a slave or a group of slaves to "freeze" their inputs to the current state. A Sync telegram causes the current output data to latch at their current state until the next Sync telegram arrives. Unfreeze and Unsync cancel each corresponding state.

Extended Device Diagnostics

Using diagnostic telegrams, the slave informs the network master of its current state in a high-priority telegram. The first 6 bytes of the diagnostic telegram are comprised of information such as its identity code ("Ident Code") or correct/incorrect configuration. The remaining bytes of this telegram are referred to as Extended Device Diagnostics and they contain information that is specific to the particular slave.

Watchdog

Using the Watchdog functionality a network slave is able to monitor bus traffic in order to ensure that the network master is still active and process data sent and received are still being updated. The Watchdog time is configured in the master config tool and is transmitted from the master to the slave during the network start-up phase. If the Watchdog time out has been reached the slaves go to their Fail Safe state (if supported) or set their outputs to "0".

5.3.2 DPV1 Services

As an addition to cyclic DPV0 services, non-cyclic services called Read, Write and Alarm were added to PROFIBUS. These services are referred to as DPV1. With DPV1, it is possible to address individual modules within the slave. In addition, DPV1 services allow transferring non-time critical data to slaves who require a large amount of configuration data or slaves that have to change measurement ranges during runtime. DPV1 data exchange takes place after cyclic data exchange in a PROFIBUS network cycle.

Read Request

With a Read Request telegram, the class 1 master can read data addressed by slot and index within the data range of a slave device. This may take several DPV0 cycles. If the master discovers a timeout, it aborts both DPV1 and DPV0 communication with the slave. Then the communication to the slave has to be re-established. The master initiates the Read Request service.

Write Request

With a Write Request telegram, the class 1 master can write data addressed by slot and index into the data range of a slave device. The timeout handling is identical to the Read Request. The master initiates the Write Request service.

Alarm Indication

DPV1 Alarm handling is an addition to the Device Diagnostic function in PROFIBUS. Alarms are reported to the master as device specific diagnostic information. Therefore, the slave initiates an Alarm Indication. Other than Device Diagnostic messages, Alarms have to be acknowledged by the Master.

5.3.3 Start/Stop Communication

Start/Stop communication with one bit: With the "NRDY" (NotReady) Bit the user program can start or stop communication with the PROFIBUS-DP system. When this Bit is set from the user program, the communication between the slave and the master is stopped and will activate diagnostic request which is reported to the PROFIBUS master during runtime. The cyclic data exchange will be suspended and the module switches into a diagnostic mode and reports static diagnosis to the master. This control bit allows the user program to make a controlled start of the communication with the PROFIBUS system.

5.4 Communication

5.4.1 IO Communication and IO Memory Map

Contained in the following sections are the I/O memory mappings for the PS69-DPS interface. The I/O area will be used for communication of status and command information as well as cyclic I/O data.

IO Array Overview

We use the term "Byte" for 8-bit values; we use the term "Word" for 16-bit values.

Module Input Array

Below is a summary of the register layout of the input area of the PROFIBUS Slave module. The offset values are defined as byte.

Offset	Register Type	Name
0	Device Status Register	Status Bits
1	Reserved	Reserved
2	Reserved	Reserved
3	Reserved	Reserved
4	Firmware Revision	Minor Version
5	Firmware Revision	Major Version
6-7	Reserved	Reserved
8-9	Slave Status Information	ExtStaSelect
10-11	Slave Status Information	ExtStaLen
12-13	Slave Status Information	Baudrate
14	Slave Status Information	Busaddress
15	Slave Status Information	UserFlags
16-17	Slave Status Information	Ident
18-19	Slave Status Information	TaskState
20-21	Slave Status Information	InputDataLen
22-23	Slave Status Information	OutputDataLen
24-25	Slave Status Information	ErrorCount
26	Slave Status Information	LastError
27	Slave Status Information	Reserved
28-29	Slave Status Information	WatchdogTime
30-31	Slave Status Information	IrqCounter
32-37	Slave Status Information	Dpv1StatusRegister
38-39	Slave Status Information	Reserved
40-135	Slave Status Information	ExtStatusInfo[96]
136-379	PROFIBUS Output Area	PBOutputData

Module Output Array

Below is a summary of the register layout of the output area of the PROFIBUS Slave module. The offset values are defined as byte

Offset	Register Type	Name
0	Device Command Register	Command Bits
1	Device Command Register	Reserved
2	Device Command Register	Reserved
3	Device Command Register	ExtStaSelect
4-248	PROFIBUS Input Area	PBInputData

Module Input Array

Device Status Registers

The PS69-DPS module uses the first 4 bytes of the CPUs input area to transfer Device Status Register information. The Device State Register contains information indicating the modules communication status and command status. The PLCs input area mapping of this information is shown below.

Byte Offset	Structure Member	Data Type	Description
0	MSB	SINT	Module Status Bits
1	Reserved	SINT	Reserved
2	Reserved	SINT	Reserved
3	Reserved	SINT	Reserved

MSB := Module Status Bits

Bit Offset	Structure Member	Data Type	Description
0	Reserved	BOOL	Reserved
1	Reserved	BOOL	Reserved
2	Reserved	BOOL	Reserved
3	Reserved	BOOL	Reserved
4	Reserved	BOOL	Reserved
5	COM	BOOL	Communication
6	RUN	BOOL	Run
7	RDY	BOOL	Ready

▪ **RDY (Ready)**

When this bit is set, the module is operational. The RDY-Bit should always be set by the module. If this bit is not set a system error has occurred and the communication between controller and module is not possible.

▪ **RUN (Run)**

When the RUN-bit is set, the module is ready for communication. Otherwise an initialization error or incorrect Parameterization occurs.

▪ **COM (Communication)**

When this bit is set, the communication is started and the module is engaged in cyclic data exchange with the Master.

These three bits are the most important bits that the controller application can use to monitor the communication and operating status of the module

Firmware Revision

This data field, which is part of the input image of the PROFIBUS Slave module, will contain the current firmware revision. The Minor revision indication will be in the low byte and the Major revision will be in the high byte. The Firmware Field is placed in the Input area as shown in the following table.

Byte Offset	Structure Member	Data Type	Description
4	FwMinor	SINT	Firmware Minor Revision
5	FwMajor	SINT	Firmware Major Revision
6-7	Reserved	INT	Reserved

Example:

If FwMajor = 10 and FwMinor = 1 then the firmware revision is 10.1.

Slave Status Information

A 128 byte state field is transferred to the user program via the input data area image. It contains information about the slave modules status and begins with byte 8 of the input region. The status information encompasses 32 bytes of static information and 96 bytes reserved for the extended status field. The content of the extended status is controlled by the command "ExtStaSelect" byte in the Device Command Register in the user program (byte offset 8-9).

Byte Offset	Structure member	Data type	Description	Valid Values
8-9	ExtStaSelect	INT	Shows which extended status information are currently transmitted in the field "Extended Status Information"	0 = No extended status information 1 = Firmware version 2 = Slave configuration 3 = Master configuration 4 = Parameter data 6 = DPV1 C1 Diag
10-11	ExtStaLen	INT	Number of valid bytes in the region "Extended Status Information"	Depends on the selected extended status 0 = 0 Byte 1 = 32 Byte 2 = 49 Byte 3 = 49 Byte 4 = 33 Byte 6 = 80 Byte
12-13	Baudrate	INT	Baud Rate on PROFIBUS	12000 = 12 MBaud 6000 = 6 MBaud 3000 = 3 MBaud 1500 = 1,5 MBaud 500 = 500 kBaud 187 = 187,5 kBaud 93 = 93,75 kBaud 9 = 9,6 kBaud 0 = not detected
14	Busaddress	SINT	Bus Address of the Slave	0 ... 125
15	UserFlags	SINT	User Fags	D0 = Parameter data changed D1 = Configuration data changed D2 ... D7 Don't care
16-17	Ident	INT	Slave ident number	097Ah
18-19	TaskState	INT	Slave status	See following table
20-21	InputDataLen	INT	Length of input data(*)	0 ... 244
22-23	OutputDataLen	INT	Length of output data(*)	0 ... 244
24-25	ErrorCount	INT	Error counter	0 ... FFFFh
26	LastError	SINT	Last error	See following table
27	Pad	SINT	Reserved	Reserved
28-29	WatchdogTime	INT	Current watchdog time	5 ... 65535 ms
30-31	IrqCounter	INT	Indication of bus activity	0 ... 0xFFFF

Byte Offset	Structure member	Data type	Description	Valid Values
32-37	Dpv1StatReg	INT	DPV1 Status Register	See following Table
38-39	Reserved	SINT	Reserved	Reserved
40-135	ExtStatusInfo	SINT[96]	Extended Status Information	See following section

(*)**Note:** The status information "InputDataLen" and "OutputDataLen" are related to the definition of inputs and outputs from point of view of PROFIBUS. There is a clear definition of inputs and outputs by PROFIBUS. They are always defined from point of view of a PROFIBUS-Master. Do not confuse them with the input and output area of the communication module. Example: If in status 'OutputDataLen' is indicated a value of 4 Bytes, then it is related to the input area of the communication module, because the input area of the communication module are outputs from point of view of a PROFIBUS-Master. The same relation applies to the status 'InputDataLen' and the output area of the communication module.

TaskState:

Value (hex) (x =don't care)	Meaning	Description
xxx1	Task is During initialization	If the module remains in this state for more than a few seconds, the configuration parameters may be invalid.
xx1x	Task running	The module initialized without error, generally the task is able to run communication on the bus.
x1xx	Diagnostic	Slave diagnostic telegrams will be sent at the moment on the bus. Reasons could be the user program (NRDYbit is set) or the DP master orders this.
1xxx	Data exchange	The data exchange mode is active. The user-data will be transferred on the bus between the master and the slave actually.

LastError:

Value	Meaning	Description
52	Invalid bus address	Valid addresses are between 0 and 125
54	Invalid 'Module Type'	The configured code of the 'Module Type' parameter is invalid. If this error happens after a configuration by the controller application check the configured "ModuleTypes" also the value "Number of valid config bytes".
55	Invalid 'ModuleLength'	The configured code for a parameter "ModuleLength" is not defined.
61	No address-switches available on the hardware	Please contact your distributor
70	I/O-data too long	The maximum size of I/O-data has been exceeded. Please check the length of all modules.
71	SPC3/ASPC2 initialization error	The SPC3/ASPC2 returns an error during initialization. Please contact ProSoft Technical Support.

DPV1 Status Registers

The controller application program will use the DPV1 status registers as an indication that the network Master has sent an unsolicited DPV1 Read/Write request. The first will contain two bits which indicate if a read or write needs to be processed. If this register contains a non-zero value, the Slave's user program must create an appropriate response to this request by using a CIP MSG command (shown in messaging section). The following table contains the mapping of these registers.

Byte Offset	Structure Member	Data Type	Data Type	Description
32	RWInd	SINT	Read/Write Indication	A Read/Write Request has been received
33	RWIndCnt	SINT	Read Write Indication Counter	Increments on every new DPV1 request
34	MaAdr	SINT	Master Address	Address of Requesting Master
35	Slot	SINT	Slot number	Requested Slot Number
36	Index	SINT	Index	Requested Index
37	DataLen	SINT	Date Length	Requested Data Length

RWInd := DPV1 Read/Write Indication Status Bits

Bit Offset	Structure Member	Data Type	Description
0	ReadReq	BOOL	1 = Indicates a Read Request
1	WriteReq	BOOL	1 = Indicates a Write Request
2	Reserved	BOOL	Reserved
3	Reserved	BOOL	Reserved
4	Reserved	BOOL	Reserved
5	Reserved	BOOL	Reserved
6	Reserved	BOOL	Reserved
7	Reserved	BOOL	Reserved

Note: Every DPV1 read or write request must be acknowledged by the PLC application program. Otherwise the PROFIBUS master shuts down communication for both channels, V0 (cyclic IO data) and V1 (non-cyclic messages). This can cause unexpected lost of data between the master and the PS69-DPS slave.

Extended Status Information

Via the extended status area the Slave module is in the position to transfer 96 Byte extended status information to the controller application. The information transferred depends on the parameter "ExtStaSelect" in the "Device Command Register". This can be controlled by the application program. If the controller application selects a specific extended status, it will be acknowledged by the Slave module in the status region in "ExtStaSelect". If the slave adapter does not acknowledge this selection, the extended information is invalid. The number of bytes within the extended status area which are valid depends on the selected status. The number of valid bytes will be shown in the status area in "ExtStaLen".

Ext. Status 0 - (Length 0 Byte):

No extended status information transferred.

Ext Status 1 - Firmware (Length 32 Byte)

Structure member	Data type	Description	Example
FwName	SINT[8]	Firmware Name	"DPS "
FwType	SINT[8]	Firmware Type	"RIF1769"
FwVersion	SINT[8]	Firmware Version	"V01.000 "
FwDate	SINT[8]	Firmware Date	"25.07.05 "

Ext. Status 2 - Slave Configuration (Length 49 Byte)

Structure member	Data type	Description
CfgLength	SINT	Number of valid configuration bytes
CfgByte1	SINT	Configuration byte 1
CfgByte2	SINT	Configuration byte 2
CfgByte3	SINT	Configuration byte 3
CfgByte4	SINT	Configuration byte 4
....
CfgByte48	SINT	Configuration byte 48

Ext. Status 3 - Master Configuration (Length 49 Byte)

Structure member	Data type	Description
CfgLength	SINT	Number of valid configuration bytes
CfgByte1	SINT	Configuration byte 1
CfgByte2	SINT	Configuration byte 2
CfgByte3	SINT	Configuration byte 3
CfgByte4	SINT	Configuration byte 4
....
CfgByte48	SINT	Configuration byte 48

Ext. Status 4 - Parameter Data (Length 33 Byte)

Structure member	Data type	Description
PrmLength	SINT	Number of valid parameter bytes
PrmByte1	SINT	Parameter byte 1
PrmByte2	SINT	Parameter byte 2
PrmByte3	SINT	Parameter byte 3
PrmByte4	SINT	Parameter byte 4
....
PrmByte32	SINT	Parameter byte 32

Ext. Status 6 - DPV1-C1-Diag (Length 80 Byte)

Structure member	Data type	Description
StaReqUsr	DINT	Status Request from User

Structure member	Data type	Description
StaMsgSen	DINT	Status Messages Sent
NegStaCnf	DINT	Negative Status Confirmations to User
DiagReqUsr	DINT	Diagnostic Requests from User
DiagMsgSen	DINT	Diagnostic Messages Sent
NegDiagCnf	DINT	Negative Diag Confirmations to User
AlaReqUsr	DINT	Alarm Request from User
AlaMsgSen	DINT	Alarm Messages Sent
PosAlaCnf	DINT	Positive Alarm Confirmations to User
NegAlaCnf	DINT	Negative Alarm Confirmations to User
Requests	DINT	Requests
ImmNegCnf	DINT	Immediate Negative Confirmations
RW_Ind	DINT	R/W Indications to User
PosRWResp	DINT	Positive R/W Responses from User
NegRWResp	DINT	Negative R/W Responses from User
AlaAckInd	DINT	Alarm Ack Indications
AlaAckResp	DINT	Alarm Ack Responses
AlaAckErr	DINT	Alarm Ack Errors
ErrRespUsr	DINT	Erroneous Responses from User
UnxRespUsr	DINT	Unexpected Responses from User

PROFIBUS Output Data

The remainder of the PLCs input area is used for the PROFIBUS output data from the Master. The PROFIBUS output information is transferred from the module to the controller. Output data from the PROFIBUS system always starts with Byte 136 (based on Start Index 0) in the input image. The maximum number of output data of a PROFIBUS Slave is 244 Byte.

Module Output Array

Device Command Register

The Device Command Register is transferred from the controller to the module via the output data image. The Command register always lies in the first 4 Bytes of the output region. Follows is the mapping for the Device Command Register.

Byte Offset	Structure Member	Data Type	Description
0	MCB	SINT	Module Command Bits
1	Reserved	SINT	Reserved
2	Reserved	SINT	Reserved
3	ExtStaSelect	SINT	Extended Status Information Select

MCB := Module Command Bits

Bit Offset	Structure Member	Data Type	Description
0	Reserved	BOOL	Reserved
1	Reserved	BOOL	Reserved
2	Reserved	BOOL	Reserved
3	Reserved	BOOL	Reserved
4	Reserved	BOOL	Reserved
5	NRDY	BOOL	Application not ready
6	INIT	BOOL	Init
7	RST	BOOL	Reset

NRDY := Not Ready

With this bit, the user program can start or stop communication with the PROFIBUS system. When this bit is set from the user program, the communication between the module and connected network Master is stopped. This control bit allows the user program to make a controlled start of the communication with the PROFIBUS Master.

INIT := Init

With this Bit, the user program can execute a Reset (Warm Start) of the module. This function is not implemented.

RST := Reset

The user program can use this bit to execute a Reset (Cold Start) of the module.

Attention: Using the Reset command will cause an immediate interruption in bus communication. The connection to the network Master will be closed.

ExtStaSelect := Extended Status Select

The user program can use this byte to select the extended status information they would like to see appear in the ExtStatusInfo Input area. See the previous section on extended status information details of the structure that is returned.

Value	Meaning	Description
1	Firmware	Returns the Firmware Version structure to the Extended Status Information
2	Slave Configuration	Returns the Slave configuration structure to the Extended Status
3	Master Configuration	Returns the Master Configuration structure to the Extended Status
4	Parameter Data	Returns the Parameter Data structure to the Extended Status Information
5		Reserved
6	DPV1-C1-Diag	Returns DPV1 Class 1 diagnostic structure to the Extended Status Information
7 and higher		Reserved

PROFIBUS Input Data

The remainder of the output area is used for the PROFIBUS Input data to the sent to the network Master. The input information is transferred from the controller to the module. INPUT data from the PROFIBUS system always starts at the 4Th Byte (based on Start Index 0) in the modules output data area.

5.4.2 CIP Messaging

PROFIBUS-DP supports acyclic services through messages. These PROFIBUS-DP services are supported by the RSLogix5000 programming tool by means of CIP messages using the "MSG" instruction. The outline and usage of these commands for the PROFIBUS-DP Slave are explained with in this section.

Note: At the time of this release of the PS69-DPS module, the MicroLogix 1500 processor, as well as some CompactLogix processors, do not support CIP messaging or CIP messaging for generic Compact I/O modules. CIP messaging for PROFIBUS DPV1 services, are not yet supported with a MicroLogix System and a PS69-DPS module.

Using the MSG Instruction in RSLogix5000

CIP messages are carried out by the use of the "MSG" function block in RSLogix5000. The "MSG" function block can be found under the Input/Output Instructions tab within the RSLogix Instruction Set. The MSG instruction asynchronously reads or writes a block of data to another module on a network. The following is an example of how this instruction is assembled using the acyclic PROFIBUS-DP service DPV1 Class 1 Alarm Request command.

Step1: Create New Controller Tag

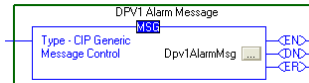
Double click on the Controller Tags tree selection under Controller CompactLogix. The Controller Tags dialog box will appear. Select the Edit Tags tab. Add a new tag called Dpv1AlarmMsg and make its Type equal to MESSAGE.

Step2: Insert the "MSG" instruction


From the language element tool bar in RSLogix select the Input/Output tab and click on the "MSG" button. The instruction will be inserted into your ladder logic as shown in the figure below.

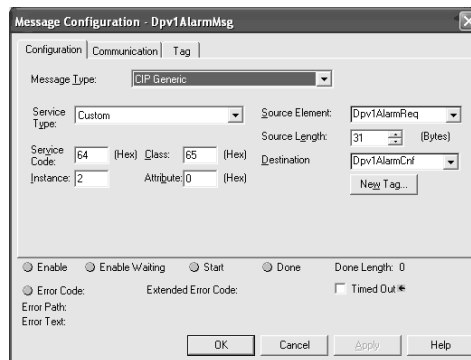


Select the ? And enter the MESSAGE type created Dpv1AlarmMsg as shown below.

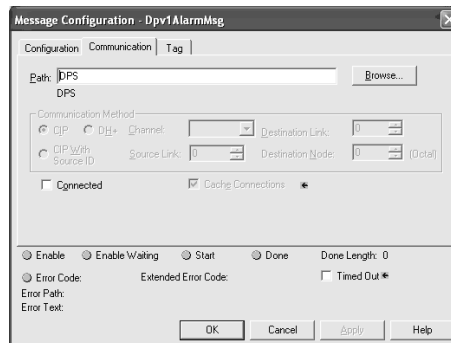


Step3: Message Configuration

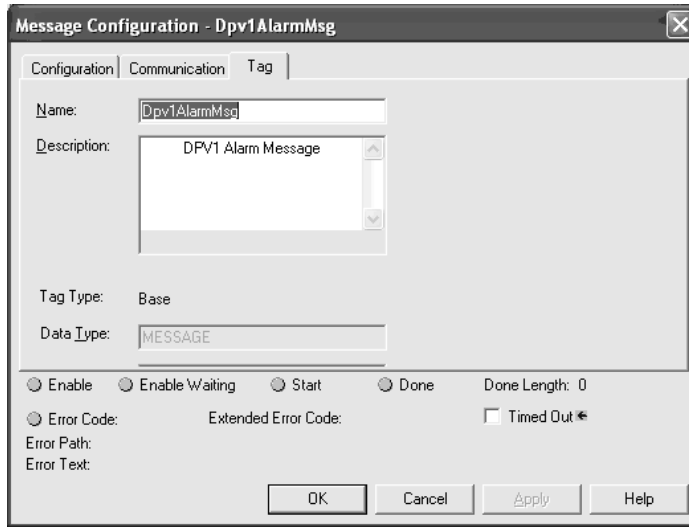
Select the button , which will open the Message Configuration dialog box. The configuration dialog will allow the user to fill in the appropriate information needed to execute the Dpv1AlarmMsg. The entries should be as follows.



Note: The user must create two user defined data types to send and receive the information for this command message. In this example Dpv1AlarmReq and Dpv1AlarmCnf were created to hold the command specific information.

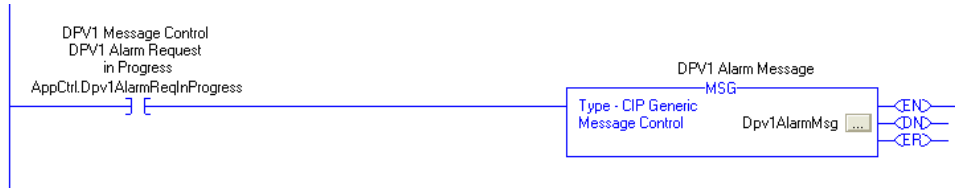


The Path in the dialog above must point to the PS69-DPS Module. Use the Browse button to select the path.



Step 4: Add Logic to Execute MSG Instruction

With the "MSG" instruction now configured the user can add the required logic needed to execute the instruction. The example below shows the "MSG" instruction used in the example logic in PS69_DPS_messaging.ACD.



Supported PROFIBUS-DP Messages

The CompactLogix Slave module supports the following message functions.

Service	Cmd Code	Group	Description
DPS Diagnostic Request	24		This service enables the user to send a single diagnostics request to a Master.
DPV1 Class 1 Read Response	17	DPV1	With this service, the Slave module can respond to a DPV1 Read Request from the PROFIBUS Master. This service works by utilizing the Master address, Data size, Slot and Index indicated within the DPV1 Status Registers.
DPV1 Class 1 Write Response	17	DPV1	With this service, the Slave module can respond to a DPV1 Write Request from the PROFIBUS Master. . This service works by utilizing the Master address, Data size, Slot and Index indicated within the DPV1 Status Registers.
DPV1 Class 1 Alarm Request	18	DPV1	This service is used to send a DPV1 Alarm Request message to a PROFIBUS Master.

Note: The sample project includes an example for each of these services.

Standard Messaging

The sections below contain the descriptions of the Standard Message supported by the PROFIBUS Slave module.

DPS Diagnostic Request

The Diagnostic Request command can be used by the controller user application to generate a single diagnostic request to a Master. The MSG instruction Request/Confirmation format is as follows.

DPS_DIAGNOSTIC_REQUEST

Parameter	Data Type	Value	Description
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Reserved3	INT	0	Reserved
Command	SINT	24	Command for Service Diagnostic Request
Reserved4	SINT	0	Reserved
Reserved5	INT	0	Reserved
Reserved6	INT	0	Reserved
Reserved7	SINT	0	Reserved
ExtDiagDataCnt	SINT	0.. 32	Number of extended diagnostic bytes to send.
Mode	SINT	1 or 0	Bit 0 = 1: don't set the Ext_Diag_Data bit in the standard diagnostic data even if user diagnostic data are present. Bit 1 ... 7: reserved
Function	SINT	18	DPS_FUNC_SINGLE_DIAG (send diagnostic request once)
Data[0 .. 31]	SINT[32]	0-255	Data for user specific extended diagnostic. The user can enter up to 32 bytes (*)

(*) For the proper format of ext. diag data refer to the PROFIBUS Norm. If the ext. diag data are not well formatted the module will reject the diagnostic request.

DPS_DIAGNOSTIC_CONFIRM

Parameter	Data Type	Value	Description
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Answer	SINT	24	Answer DPS Diag
Failure	SINT	e	Error, status (see following section)
Reserved3	INT	0	Reserved
Reserved4	INT	0	Reserved
Reserved5	INT	0	Reserved
Reserved6	SINT	0	Reserved
ExtDiagDataCnt	SINT	0	Always 0 in answer
Mode	SINT	0	Always 0 in answer
Function	SINT	18	DPS_FUNC_SINGLE_DIAG

DPS Diagnostic Confirmation

CIP MSG Parameterization

Parameter	Value	Remarks
Message Type	CIP Generic	
Service Type	Custom	
Service Code	64 hex	Service Code "Bridge Message"
Class	65 hex	CIP Object "CIP_MSG_BRIDGE"
Instance	1	
Attribute	0	
Source Element	DiagReq	Reference to a Tag of type DPS_DIAGNOSTIC_REQUEST
Destination	DiagCnf	Reference to a Tag of type DPS_DIAGNOSTIC_CONFIRM
Source Length	16 ... 48	Corresponds to the size of the DPS_DIAGNOSTIC_REQUEST structure

DPV1 Messaging

This section describes DPV1 messaging functions supported by the PROFIBUS Slave module.

Note: Every DPV1 read or write request has to be acknowledged by the PLC application program. Otherwise the PROFIBUS master shuts down communication for both channels, V0 (cyclic IO data) and V1 (non-cyclic messages). This can cause unexpected lost of data between the master and the PS69-DPS slave.

DPV1 Class 1 Read Response

The DPV1 Class 1 Read Response message is used by the Slave to reply to a Master DPV1 Read Request. The MSG instruction Request/Confirmation format is as follows.

DPS_DP1C1_RW_RESP_REQUEST

Parameter	Data Type	Value	Description
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Reserved3	INT	0	Reserved
Command	SINT	17	Command for DPV1 Class 1 Read Response
Reserved4	SINT	0	Reserved
RwResp	SINT	1	1 = Read Response Request
MaAdr	SINT	0.. 125	Bus Address of Master which sent the request. This value is obtained from the DPV1 Status Register.
Slot	INT	0.. 254	Slot Number. This value is obtained from the DPV1 Status Register.
Index	SINT	0.. 254	Index. This value is obtained from the DPV1 Status Register.
DataLen	SINT	1.. 240 (x)	Length of the data block to be read. This value is obtained from the DPV1 Status Register.
ErrCode1	SINT	E1	E1 = 0 no error occurred (*) E1 <> 0 Error code 1 according to DPV1
ErrCode2	SINT	E2	E2 = 0 no error occurred (*) E2 <> 0 Error code 2 according to DPV1
Data[1..x-1]	SINT[1..240]	0-255	DPV1 Read data to be sent to Master in response.

(*) If the module is not able to process the requested service for example the master is requesting a slot or index that is not supported for whatever reason then the user can set the ErrorCodes and ErrorCodes will be transferred to the master. The ErrorCodes must be PROFIBUS conform. For the proper format refer to the PROFIBUS norm..

DPS_DP1C1_RW_RESP_CONFIRM

Parameter	Data Type	Value	Description
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Answer	SINT	17	Command for DPV1 Class 1 Read Response
Failure	SINT	0	no error
Reserved4	INT	0	Reserved
RwResp	SINT	1	1 = Read Response Request. Reply from request
MaAdr	SINT	0.. 125	Bus Address of Master which sent the request. Reply from request.
Slot	INT	0.. 254	Slot Number. Reply from request.
Index	SINT	0.. 254	Index. Reply from request.
DataLen	SINT	1.. 240	Length of the data block to be read. Reply from request.
ErrCode1	SINT	E1	DPV1 Error code 1. Reply from request.
ErrCode2	SINT	E2	DPV1 Error code 2 Reply from request.

CIP MSG Parameterization

Parameter	Value	Remarks
Message Type	CIP Generic	
Service Type	Custom	
Service Code	64 hex	Service Code "Bridge Message"
Class	65 hex	CIP Object "CIP_MSG_BRIDGE"
Instance	1	
Attribute	0	
Source Element	Dpv1RWRespReq	Reference to a Tag of type DPS_DPV1C1_RW_RESP_REQUEST
Destination	Dpv1RWRespCnf	Reference to a Tag of type DPS_DPV1C1_RW_RESP_CONFIRM
Source Length	16 + n	Corresponds to the constant size of the DPS_DPV1C1_RW_REQUEST structure plus number of requested data

DPV1 Class 1 Write Response

The DPV1 Class 1 Write Response is used by the Slave to reply to a Master DPV1 write request. The MSG instruction Request/Confirmation format is as follows.

DPS_DPV1C1_RW_RESP_REQUEST

Parameter	Data Type	Value	Description
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Reserved3	INT	0	Reserved
Command	SINT	17	Command for DPV1 Class 1 Write Response
Reserved4	SINT	0	Reserved
RwResp	SINT	2	2 = Write Response Request
MaAdr	SINT	0.. 125	Bus Address of Master which send the request. This value is obtained from the DPV1 Status Register.
Slot	INT	0.. 254	Slot Number. This value is obtained from the DPV1 Status Register.
Index	SINT	0.. 254	Index. This value is obtained from the DPV1 Status Register.
DataLen	SINT	1.. 240	Length of the data block to be written. This value is obtained from the DPV1 Status Register.
ErrCode1	SINT	E1	E1 = 0 no error occurred (*) E1 <> 0 Error code 1 according to DPV1
ErrCode2	SINT	E2	E2 = 0 no error occurred (*) E2 <> 0 Error code 2 according to DPV1

(*) If the module is not able to process the requested service for example the master is requesting a slot or index that is not supported for whatever reason then the user can set the ErrorCodes which will be transferred to the master. But the ErrorCodes must conform to the PROFIBUS specification.

DPS_DPV1C1_RW_RESP_CONFIRM

Parameter	Data Type	Value	Meaning
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Answer	SINT	17	Command for DPV1 Class 1 Read Response
Failure	SINT	0	no error
Reserved4	INT	0	Reserved
RwResp	SINT	2	2 = Write Resp Request. Reply from resp. request
MaAdr	SINT	0.. 125	Bus Address of Master which sent the request. Reply from resp. request
Slot	INT	0.. 254	Slot Number. Reply from resp. request
Index	SINT	0.. 254	Index. Reply from resp. request
DataLen	SINT	1.. 240 (x)	Length of the data block to be written.
ErrCode1	SINT	E1	DPV1 Error code 1 Reply from resp. request
ErrCode2	SINT	E2	DPV1 Error code 2 Reply from resp. request
Data[1..x]	SINT[1..240]	0-255	DPV1 Write data the Master has sent.

CIP MSG Parameterization

Parameter	Value	Remarks
Message Type	CIP Generic	
Service Type	Custom	
Service Code	64 hex	Service Code "Bridge Message"
Class	65 hex	CIP Object "CIP_MSG_BRIDGE"
Instance	1	
Attribute	0	
Source Element	Dpv1RWRespReq	Reference to a Tag of type DPS_DPV1C1_RW_RESP_REQUEST
Destination	Dpv1RWRespCnf	Reference to a Tag of type DPS_DPV1C1_RW_RESP_CONFIRM
Source Length	16	Corresponds to the constant size of DPS_DPV1C1_RW_REQUEST structure

DPV1 Class 1 Alarm Request

The DPV1 Class 1 Alarm Request is used to indicate a DPV1 Alarm to the connected Master. The MSG instruction Request/Confirmation format is as follows.

DPS_DPV1C1_ALARM_REQUEST

Parameter	Data Type	Value	Description
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Reserved3	INT	0	Reserved
Command	SINT	18	Command for Service DPV1 Class 1 Alarm Request
Reserved4	SINT	0	Reserved
Reserved5	INT	0	Reserved
SlotNumber	INT	0..254	Alarm Slot Number
SequenceNumber	SINT	0...31	Alarm Sequence Number
DataCnt	SINT	0...28 (x)	Number of User Specific Alarm Data
AlarmType	SINT	1-6, 32-126	Alarm Type:Diag,-Process,-Pull,-Plug,-Status,-Update, Manufacturer specific
Specifier	SINT	0...7	Alarm Specifier Bit 0...1: Alarm Specifier Bit 2: Add Ack bit
Data[0..x]	SINT[28]	0...255	User Specific Alarm Data.

DPS_B_ACYC_C1_ALARM_CONFIRM

Parameter	Data Type	Value	Meaning
Reserved1	INT	0	Reserved
Reserved2	INT	0	Reserved
Answer	SINT	18	Answer DPS_B_ACYC_C1_ALARM
Failure	SINT	e	Error, status (see following section)
Reserved3	INT	0	Reserved
Reserved4	INT	0	Reserved
SlotNumber	INT	0..254	Alarm Slot Number, Reply from request
SequenceNumber	SINT	0...31	Alarm Sequence Number, Reply from request
DataCnt	SINT	0	not used
DataType	SINT	1-6, 32-126	Alarm Type, Reply from request
Specifier	SINT	0..7	Alarm Specifier, Reply from request

CIP MSG Parameterization

Parameter	Value	Remarks
Message Type	CIP Generic	
Service Type	Custom	
Service Code	64 hex	Service Code "Bridge Message"
Class	65 hex	CIP Object "CIP_MSG_BRIDGE"
Instance	1	
Attribute	0	
Source Element	Dpv1AlarmReq	Reference to a Tag of type DPS_DPV1C1_ALARM_REQUEST
Destination	Dpv1AlarmCnf	Reference to a Tag of type DPS_DPV1C1_ALARM_CONFIRM
Source Length	16 + n	n = Number user specific Alarm Data (0 .. 28)

DPV1 Error Coding Scheme

Error Codes according to the DPV1 specification:

Error Code 1

B7	B6	B5	B4	B3	B2	B1	B0
Error Code (see below)							
Error Class (see below)							
Error Class	Meaning	Error Code					
0 to 9	Reserved						
10	Application	0 = Read Error 1 = Write Error 2 = Module Failure 3 to 7 = Reserved 8 = Version Conflict 9 = Feature not Supported 10 to 15 = User Specific					
11	Access	0 = Invalid Index 1 = Write Length Error 2 = Invalid Slot 3 = Type Conflict 4 = Invalid Area 5 = State Conflict 6 = Access Denied 7 = Invalid Range 8 = Invalid Parameter 9 = Invalid Type 10 to 15 = User Specific					

Error Class	Meaning	Error Code
12	Resource	0 = Read Constrain Conflict 1 = Write Constrain Conflict 2 = Resource Busy 3 = Resource Unavailable 4 to 7 = Reserved 8 to 15 = User Specific
13 to 15	User Specific	

Error Code 2

Error code 2 is application specific.

Messaging Error Codes

The section includes all errors codes and conditions that can occur when using the CIP messaging commands outlined in the previous sections.

- Your application should be constructed in a manner in which it catches the two possible error cases listed below:
- CIP Message instruction failed itself
- The requested command returns an error in its request confirmation

Only if both possibilities are without any error has the requested command been successful.

CIP Messaging General

Applicable are the generally known error codes for CIP Messages such as "Service Not Supported". In this case, the parameters of the CIP Message must be checked (Service Code, Class, Instance). All CIP error codes that are returned by the module and their cause are described in the following table.

Note: Some CIP error codes are public and can be generated also by the Controller. Make sure the error was not generated by the controller.

CIP Status	Extended Status	Meaning	Cause	Help
02 hex	00CA hex	Resources unavailable Out of segments	System has no segments left to execute the command	
02 hex	03E8 hex	Resources unavailable Out of CIP com buffer	System has no CIP communication buffer left to execute the command	Check the number of parallel CIP messages send to the module. The module can process 5 CIP messages in parallel. Note that RSLinx can already consume 2 of this CIP com buffers if the online browser is active.
02 hex	0519 hex	Resources unavailable Out of command buffer	System has no command buffer left to execute the command	Call support

CIP Status	Extended Status	Meaning	Cause	Help
08 hex	0000 hex	Service not supported	The service code of the requested object is not supported	Check parameter of the CIP Message
14 hex	0000 hex	Attribute not supported	The attribute of the requested object is not supported	Check parameter of the CIP Message
13 hex	0000 hex	Insufficient data	Too little data was transferred with the CIP Message	Check the "Source Length" parameter in the parameter dialog of the CIP Message, and the consistency of all length parameters within the requested command.
15 hex	0000 hex	Configuration data size too large	Too much data transferred with the CIP Message	Check the overall length of the requested command sent with the CIP message, and the consistency of all length parameters within the requested command is correct.
16 hex	0000 hex	Object not supported	The requested object does not exist within the module.	
FE hex	0000 hex	Message Timeout	No answer message was received.	
FF hex	0514 hex	General Error Non specified error occurred		Call support
FF hex	0517 hex	General Error Unknown command / Invalid Parameter	The values in Requested Command is unknown or the parameter of the requested command are invalid	The value Req.Command must be initialized. For Read/Write Response request, check if you entered the proper Slot, Index and so on. from Dpv1StatusRegister

DPS Diagnostic Request

Failure	Error source
0	TASK_F_OK No error
115	DPS_ERR_DIAG_TOO_LONG Status data exceeds the length of the diagnostic buffer.
116	DPS_ERR_NO_FREE_DIAG_BUFFER No diagnostic buffer available at the moment. This Error will be temporary.
129	DPS_ERR_DIAG_DATA_ILLEG_LEN Mismatch between length of diagnostic block and length at msg.data_cnt.

Failure	Error source
130	DPS_ERR_DIAG_DEV_DP_DISABLED Device related diagnosis requested but DP mode currently not active
131	DPS_ERR_DIAG_DEV_ILLEG_LEN Device related diagnostic data of illegal length
132	DPS_ERR_DIAG_ID_ILLEG_LEN Id related diagnosis data of illegal length
133	DPS_ERR_DIAG_CHAN_ILLEG_ID Channel related data refer to unknown id byte
134	DPS_ERR_DIAG_REV_TOO_MANY More than one revision number in diag data
152	TASK_F_MESSAGECOMMAND Unknown command at msg.b
165	TASK_F_DATA_CNT Mismatch between length at msg.ln and length at msg.data_cnt
167	TASK_F_FUNCTION Unknown function code at msg.function
200	TASK_F_NOT_INITIALIZED Task not initialized

DPV1 Class 1 Read and Write

Failure	Error source
0	No error

DPV1 Class 1 Alarm Request

Failure	Description
0	TASK_F_OK No error
115	DPS_ERR_DIAG_TOO_LONG Status data exceeds the length of the diagnostic buffer.
116	DPS_ERR_NO_FREE_DIAG_BUFFER No diagnostic buffer available at the moment This Error will be temporary.
119	DPS_ERR_ALARM_DPV1_C1_DEACTIVATED DPV1 class 1 services are disabled
120	DPS_ERR_ALARM_OVERFLOW Maximum number of active alarms exceeded
121	DPS_ERR_ALARM_DISABLED Alarm is disabled
123	DPS_ERR_ALARM_ILLEG_LEN User specific alarm data of illegal length
125	DPS_ERR_ALARM_ILLEG_SEQU Sequence number out of range or already in use
152	TASK_F_MESSAGECOMMAND Unknown command in "Command" Field
165	TASK_F_DATA_CNT Mismatch between length in "length" field and length of message data
200	TASK_F_NOT_INITIALIZED Task not initialized

5.5 Constructing a Bus Cable for PROFIBUS DP

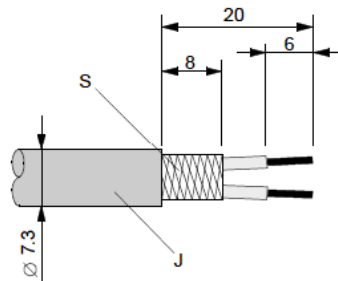
The bus cable for connecting PROFIBUS DP devices must be constructed by the user. A special PROFIBUS cable (twisted pair) is required here. This standard cable is available from various manufacturers and is a Belden part number 3079A.

If you plan to construct your own bus cable, the following part numbers are provided for your convenience.

- PROFIBUS connector: Siemens part number 6ES7972-0BA40-0XA0
- PROFIBUS cable: Belden part number 3079A.

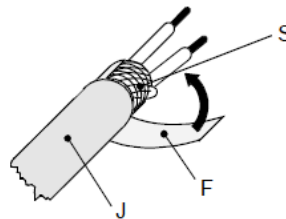
To construct the cable, proceed as follows:

- 1 Cut the cable to the required length.
- 2 Prepare the cable ends as shown in the illustration (dimensions in mm):



- J** PVC Jacket
- S** Braided shielding

- 3 Remove the PVC jacket J to the indicated length.
- 4 Wrap the provided copper shielding F around the shield braiding S:

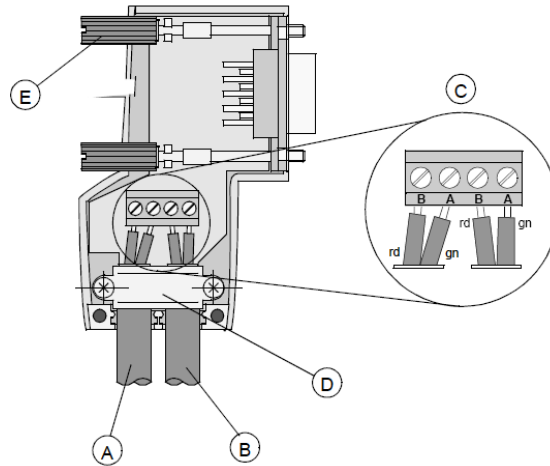


- J** PVC jacket
 - S** Braided shielding
 - F** Copper foil shielding
- Additional foil can be obtained from 3M.

- 5 Plug the leads of the corresponding cable(s) into the terminals as shown:
 - Green leads in terminal A
 - Red lead in terminal B

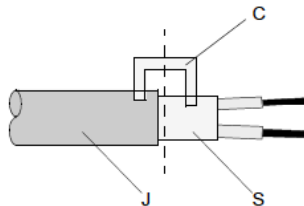
- **Note:** Do **not** tighten the corresponding screws yet.

Connection terminal assignment on the PROFIBUS DP:



- A** Incoming cable
- B** Outgoing cable
- C** Connection terminals (only once (B,A))
- D** Cable cleat for relieving tension
- E** Bus connector screws

- 6** Attach the cables with the provided cable cleat to create a robust shielded connection and to relieve any tension as shown:



- J** PVC Jacket
- S** Braided shielding with foil shielding
- C** Cable cleat

- **Note:** Half of the cable jacket must lie under the cable cleat!

Pay attention to the cable cleat installation instructions.

- 7** Fasten the individual wires of the PROFIBUS cable to the terminals
- 8** Close the connector housing.

- **Note:** The shielding of both cables is connected internally with the metal housing of the connector.

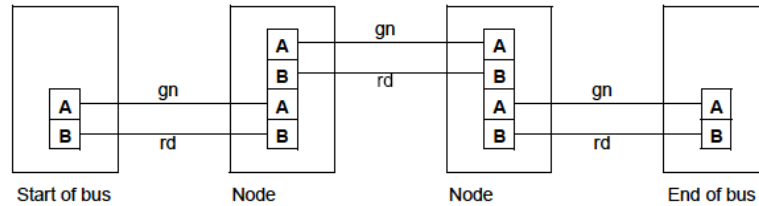
- 9** Complete the Central Shielding Measures (below) and grounding operations for the shielding before you connect the cable connector to the module.
- 10** Plug the PROFIBUS DP connector into the module and secure it with the screws.

Bus Begin and Bus End

The PROFIBUS connector with termination is required at the beginning and the end of the bus. These connectors emulate the line impedance.

It is recommended that at least one connector with diagnostics interface is used.

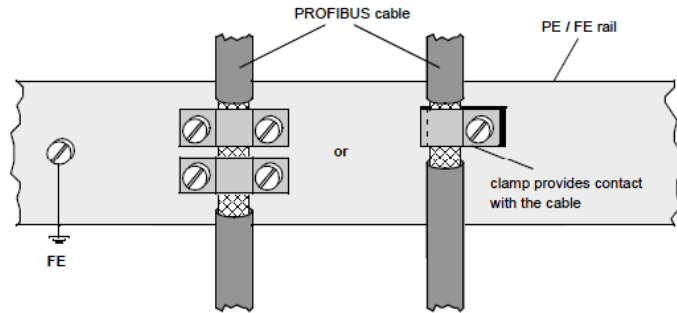
Wiring diagram for a PROFIBUS DP cable



Grounding and Shielding for Systems with Equipotential Bonding

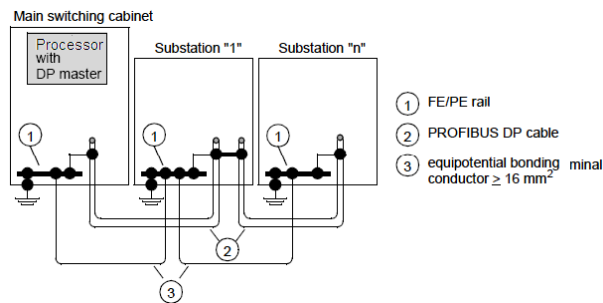
Each cable shield should be galvanically grounded with the earth using FE/PE grounding clamps immediately after the cable has been connected to the cabinet.

This example indicates the shielding connection from the PROFIBUS cable to the FE/PE rail.



Note: An equalization current can flow across a shield connected at both ends because of fluctuations in ground potential. To prevent this, it is imperative that there is potential equalization between all the attached installation components and devices.

This example indicates the system components and devices in a system with equipotential bonding.

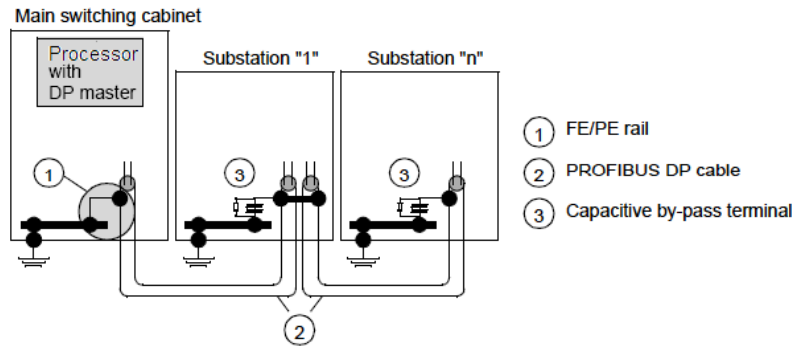


Grounding and Shielding for Systems without Equipotential Bonding

Note: Grounding and shielding is to be carried out the same as for systems **with** equipotential bonding.

If this is not possible because of system or construction specific reasons however, use distributed ground with a capacitive coupling of high frequency interference signals.

This representation shows distributed grounding with capacitive coupling.



6 Support, Service & Warranty

In This Chapter

- ❖ Contacting Technical Support 103
- ❖ Return Material Authorization (RMA) Policies and Conditions..... 105

Contacting Technical Support

ProSoft Technology, Inc. (ProSoft) is committed to providing the most efficient and effective support possible. Before calling, please gather the following information to assist in expediting this process:

- 1 Product Version Number
- 2 System architecture
- 3 Network details

If the issue is hardware related, we will also need information regarding:

- 1 Module configuration and associated ladder files, if any
- 2 Module operation and any unusual behavior
- 3 Configuration/Debug status information
- 4 LED patterns
- 5 Details about the serial, Ethernet or fieldbus devices interfaced to the module, if any.

Note: For technical support calls within the United States, an after-hours answering system allows 24-hour/7-days-a-week pager access to one of our qualified Technical and/or Application Support Engineers.

Internet	Web Site: www.prosoft-technology.com/support E-mail address: support@prosoft-technology.com
Asia Pacific (location in Malaysia)	Tel: +603.7724.2080, E-mail: asiapc@prosoft-technology.com Languages spoken include: Chinese, English
Asia Pacific (location in China)	Tel: +86.21.5187.7337 x888, E-mail: asiapc@prosoft-technology.com Languages spoken include: Chinese, English
Europe (location in Toulouse, France)	Tel: +33 (0) 5.34.36.87.20, E-mail: support.EMEA@prosoft-technology.com Languages spoken include: French, English
Europe (location in Dubai, UAE)	Tel: +971-4-214-6911, E-mail: mea@prosoft-technology.com Languages spoken include: English, Hindi

North America (location in California)	Tel: +1.661.716.5100, E-mail: support@prosoft-technology.com Languages spoken include: English, Spanish
Latin America (Oficina Regional)	Tel: +1-281-2989109, E-Mail: latinam@prosoft-technology.com Languages spoken include: Spanish, English
Latin America (location in Puebla, Mexico)	Tel: +52-222-3-99-6565, E-mail: soporte@prosoft-technology.com Languages spoken include: Spanish
Brasil (location in Sao Paulo)	Tel: +55-11-5083-3776, E-mail: brasil@prosoft-technology.com Languages spoken include: Portuguese, English

6.1 Warranty Information

For complete details regarding ProSoft Technology's TERMS & CONDITIONS OF SALE, WARRANTY, SUPPORT, SERVICE AND RETURN MATERIAL AUTHORIZATION INSTRUCTIONS, please see the documents on the Product DVD or at www.prosoft-technology/warranty.

Documentation is subject to change without notice.

Index

A

Adapter (PROFIBUS-DP-Slave) • 35
Adding Multiple Modules (Optional) • 25
Alarm Indication • 74
APP_CONSTANT_PATTERN • 68
APP_DPV1_PROG_CONTROL • 68
APP_DPV1_STAT_COUNTER • 68

B

Busaddress • 47

C

Cable • 60
CIP Messaging • 84
CIP Messaging General • 95
Communication • 75
CompactLogix • 56
CompactLogix I/O Example • 50
CompactLogix I/O LED • 58
CompactLogix Messaging Example • 52
Configuration and Start-Up • 37
Configuration by Controller Application • 46
Configuration by Master • 45
Configuring the RSLinx Driver for the PC COM Port • 33
Connecting Your PC to the Processor • 16
Constructing a Bus Cable for PROFIBUS DP • 60, 99
Contacting Technical Support • 103
Create a new RSLogix5000 project • 17
Create the Module • 18

D

Device Command Register • 83
Device Status Registers • 76
Diagnostics and Troubleshooting • 7, 55
Downloading the Sample Program to the Processor • 32
DPS Diagnostic Confirmation • 88
DPS Diagnostic Request • 87, 96
DPS_DIAGNOSTIC_CONFIRM • 69
DPS_DIAGNOSTIC_REQUEST • 69
DPS_DPV1C1_ALARM_CONFIRM • 70
DPS_DPV1C1_ALARM_REQUEST • 70
DPS_DPV1C1_RW_INDICATION • 71
DPS_DPV1C1_RW_RESP_CONFIRM • 71
DPS_DPV1C1_RW_RESP_REQUEST • 72
DPV0 Services • 73
DPV1 Class 1 Alarm Request • 92, 98
DPV1 Class 1 Read and Write • 97
DPV1 Class 1 Read Response • 89
DPV1 Class 1 Write Response • 91

DPV1 Error Coding Scheme • 94
DPV1 Messaging • 89
DPV1 Services • 74
DPV1 Status Registers • 80

E

Error Sources and Reasons • 58
Expansion General Configuration • 43
Explanation of settable configuration values • 47
Ext Status 1 - Firmware (Length 32 Byte) • 81
Ext. Status 0 - (Length 0 Byte): • 81
Ext. Status 2 - Slave Configuration (Length 49 Byte) • 81
Ext. Status 3 - Master Configuration (Length 49 Byte) • 81
Ext. Status 4 - Parameter Data (Length 33 Byte) • 81
Ext. Status 6 - DPV1-C1-Diag (Length 80 Byte) • 81
Extended Device Diagnostics • 73
Extended Status Information • 80
ExtStaSelect
= Extended Status Select • 84

F

Fail Safe Mode • 73
Firmware Revision • 77
Force User Configuration • 47
Functional Specifications • 64

G

General • 45
General Specifications • 62
Generic Extra Data Config • 44
Global Control • 73
GSD File • 45
Guide to the PS69-DPS User Manual • 7

H

Hardware Diagnostics (LED) • 56
Hardware Requirements • 10
Hardware Specifications • 63
How to Contact Us • 2

I

Import the Ladder Rung • 20
INIT
= Init • 83
Input
DPS_DEV_STATUS_REGISTER • 66
DPS_FW_REVISION • 66
DPS_INPUT_ARRAY • 66
DPS_STATUS_FIELD • 67
Installing the Module in the Rack • 13
IO Array Overview • 75
IO Communication and IO Memory Map • 75

L

LastError: • 79

M

MCB
= Module Command Bits • 83
Messaging Error Codes • 95
MicroLogix 1500 • 56
MicroLogix Fault LED • 58
Module Input Array • 75, 76
Module n Type / Module n Length • 48
Module Output Array • 76, 83
Module Properties 1 • 40
Module Properties 2 • 41
Module Selection • 38, 42
MSB
= Module Status Bits • 77

N

NRDY
= Not Ready • 83
Number of Valid Configuration Bytes • 47

O

Output
DPS_DEV_COMMAND_REGISTER • 67
DPS_OUTPUT_ARRAY • 67

P

Package Contents • 12
Pinouts • 37, 99
PROFIBUS Functionality • 73
PROFIBUS Input Data • 84
PROFIBUS Interface • 65
PROFIBUS Output Data • 82
Programmable Controller Functionality • 11
ProSoft Technology® Product Documentation • 2
PS69 LEDs • 57
PS69-DPS Sample Add-On Instruction Import
Procedure • 17

R

Read Request • 74
Reference • 7, 61
Reference Systems • 10
RSLogix 500 • 42
RSLogix 5000 • 38
RSLogix Example Program • 49
RSLogix5000 User Defined Data Types • 66
RST
= Reset • 83
RWInd
= DPV1 Read/Write Indication Status Bits • 80

S

Slave Configuration • 45
Slave Status Information • 78
Software Requirements • 10
Specifications • 7, 62
Standard Messaging • 87

Start Here • 7, 9
Start/Stop Communication • 74
Step 4
Add Logic to Execute MSG Instruction • 86
Step1
Create New Controller Tag • 84
Step2
Insert the • 85
Step3
Message Configuration • 85
Support, Service & Warranty • 7, 103
Supported PROFIBUS-DP Messages • 87
Sync and Freeze • 73
SYS and COM Status LEDs • 58

T

TaskState: • 79
Troubleshooting • 58

U

Using the MSG Instruction in RSLogix5000 • 84

W

Warranty Information • 105
Watchdog • 74
Watchdog Timeout • 47
Write Request • 74

Y

Your Feedback Please • 2