



Operating Instruction Manual

## **RIF 1769-DPS**

**PROFIBUS-DP Slave for CompactLogix 1769  
and MicroLogix 1764 Controller Series**

Edition: 1

Language: English (EN)

### **Hilscher Gesellschaft für Systemautomation mbH**

Rheistraße 15  
D-65795 Hattersheim  
Germany

Tel. +49 (0) 6190 / 99070

Fax. +49 (0) 6190 / 990750

Sales: +49 (0) 6190 / 99070

Hotline and Support: +49 (0) 6190 / 990799

Sales email: [sales@hilscher.com](mailto:sales@hilscher.com)

Hotline and Support email: [hotline@hilscher.com](mailto:hotline@hilscher.com)

Web: [www.hilscher.com](http://www.hilscher.com)

# Contact

## Hilscher Europe

Contact	Germany	France
Address	Hilscher Gesellschaft für Systemautomation mbH Rheinstraße 15 D-65795 Hattersheim	Hilscher France s.a.r.l. 12 rue du 35ième Régiment d'Aviation Miniparc du Chêne FR-69500 Bron
Phone	+49 (0) 6190 9907-0	+33 (0) 472379840
Phone Sales	+49 (0) 6190 9907-90	+33 (0) 472379840
Phone Support	+49 (0) 6190 9907-99	+33 (0) 472379840
Fax	+49 (0) 6190 9907-50	+33 (0) 478268327
E-Mail	<a href="mailto:info@hilscher.com">info@hilscher.com</a>	<a href="mailto:info@hilscher.fr">info@hilscher.fr</a>
E-Mail Sales	<a href="mailto:sales@hilscher.com">sales@hilscher.com</a>	
E-Mail Support	<a href="mailto:hotline@hilscher.com">hotline@hilscher.com</a>	
Web	<a href="http://www.hilscher.com">www.hilscher.com</a>	<a href="http://www.hilscher.com">www.hilscher.com</a>

Contact	Italy	Switzerland
Address	Hilscher Italia s.r.l. Via Grandi, 25 IT-20090 Vimodrone (MI)	Hilscher Swiss GmbH Hubelmattstraße 29 CH-4500 Solothurn
Phone	+39 / 0225007068	+41 (0) 32 6236 633
Phone Sales	+39 / 0225007068	+41 (0) 32 6236 633
Phone Support	+39 / 0225007068	+49 (0) 6190 9907-99
Fax	+39 / 0225029973	+41 (0) 32 6236 632
E-Mail	<a href="mailto:info@hilscher.it">info@hilscher.it</a>	<a href="mailto:info@hilscher.ch">info@hilscher.ch</a>
E-Mail Sales	<a href="mailto:sales@hilscher.it">sales@hilscher.it</a>	<a href="mailto:sales@hilscher.com">sales@hilscher.com</a>
E-Mail Support	<a href="mailto:supporto@hilscher.it">supporto@hilscher.it</a>	<a href="mailto:hotline@hilscher.com">hotline@hilscher.com</a>
Web	<a href="http://www.hilscher.com">www.hilscher.com</a>	<a href="http://www.hilscher.com">www.hilscher.com</a>

## Hilscher North America

Contact	North America
Address	Hilscher North America, Inc. 2443 Warrenville Road, Suite 100 Lisle, Illinois 60532, USA
Phone	(+1) 630 505 5301
Phone Sales	(+1) 630 505 5301
Phone Support	(+1) 630 505 5301
Fax	(+1) 630 505 7532
E-Mail	<a href="mailto:info@hilscher.us">info@hilscher.us</a>
E-Mail Sales	<a href="mailto:info@hilscher.us">info@hilscher.us</a>
E-Mail Support	<a href="mailto:info@hilscher.us">info@hilscher.us</a>
Web	<a href="http://www.hilscher.com">www.hilscher.com</a>

## Worldwide: Distributors

Please visit our Homepage at

[www.hilscher.com](http://www.hilscher.com)

## List of Revisions

Index	Date	Version	Chapter	Revisions
1	05.08.2005	V1.000	all	created
2	19.09.2005	V1.001	all	Support of MicroLogix 1500 Controller Series

Although this program has been developed with great care and intensively tested, Hilscher Gesellschaft für Systemautomation mbH cannot guarantee the suitability of this program for any purpose not confirmed by us in writing.

Guarantee claims shall be limited to the right to require rectification. Liability for any damages which may have arisen from the use of this program or its documentation shall be limited to cases of intent.

We reserve the right to modify our products and their specifications at any time in as far as this contribute to technical progress. The version of the manual supplied with the program applies.

Microsoft and Windows are registered trademarks of Microsoft Corporation. Pentium is a registered trademark of Intel Corporation. Adobe and Acrobat are trademarks of Adobe Systems Incorporated. RSLogix and CompactLogix are the trademarks of Rockwell Automation. CIF and SYCON.net are trademarks of Hilscher Gesellschaft für Systemautomation mbH.

Values with a following 'hex' are in hexadecimal notation such as 1E hex = 30. Values without any following letter are in decimal notation.

# Table of Contents

1	INTRODUCTION.....	9
1.1	Intended Audience.....	9
1.2	General Information RIF 1769-DPS.....	9
1.3	Software Requirements .....	9
1.4	Hardware Requirements.....	9
1.5	Reference Manuals.....	11
1.6	Reference Systems .....	11
1.7	Programmable Controller Functionality .....	12
2	INSTALLATION, WIRING AND SYSTEM PLANNING .....	13
2.1	CompactLogix System.....	13
2.2	MicroLogix 1500 System .....	14
2.3	Consideration when planning the system: .....	15
3	PROFIBUS FUNCTIONALITY.....	16
3.1	DPV0 Services.....	16
3.1.1	Fail Safe Mode .....	16
3.1.2	Global Control.....	16
3.1.3	Sync and Freeze .....	16
3.1.4	Extended Device Diagnostics.....	16
3.1.5	Watchdog .....	17
3.2	DPV1 Services.....	17
3.2.1	Read Request.....	17
3.2.2	Write Request.....	17
3.2.3	Alarm Indication.....	17
3.3	Start/Stop Communication .....	17
4	CONFIGURATION AND START-UP.....	18
4.1	RSLogix 5000 .....	19
4.1.1	Module Selection.....	19
4.1.2	Module Properties 1 .....	21
4.1.3	Module Properties 2 .....	22
4.2	RSLogix 500 .....	23
4.2.1	Module Selection.....	23
4.2.2	Expansion General Configuration .....	25
4.2.3	Generic Extra Data Config .....	26
4.3	Slave Configuration .....	27
4.3.1	General.....	27
4.3.2	GSD File.....	27
4.3.3	Configuration by Master .....	27
4.3.4	Configuration by Controller Application.....	28

4.3.5	Explanation of settable configuration values:.....	29
4.3.5.1	Busaddress .....	29
4.3.5.2	Force User Configuration .....	29
4.3.5.3	Watchdog Timeout .....	29
4.3.5.4	Number of Valid Configuration Bytes .....	30
4.3.5.5	Module n Type / Module n Length .....	30
<b>5</b>	<b>COMMUNICATION .....</b>	<b>31</b>
5.1	IO Communication and IO Memory Map .....	31
5.1.1	IO Array Overview .....	31
5.1.1.1	Module Input Array .....	31
5.1.1.2	Module Output Array .....	32
5.1.2	Module Input Array .....	33
5.1.2.1	Device Status Registers .....	33
5.1.2.2	Firmware Revision .....	34
5.1.2.3	Slave Status Information .....	35
5.1.2.4	PROFIBUS Output Data .....	39
5.1.3	Module Output Array .....	40
5.1.3.1	Device Command Register .....	40
5.1.3.2	PROFIBUS Input Data .....	41
5.2	CIP Messaging .....	42
5.2.1	Using the MSG Instruction in RSLogix5000 .....	42
5.2.2	Supported PROFIBUS-DP Messages .....	45
5.2.3	Standard Messaging .....	46
5.2.3.1	DPS Diagnostic Request .....	46
5.2.4	DPV1 Messaging .....	48
5.2.4.1	DPV1 Class 1 Read Response .....	48
5.2.4.2	DPV1 Class 1 Write Response .....	50
5.2.4.3	DPV1 Class 1 Alarm Request .....	52
5.2.4.4	DPV1 Error Coding Scheme .....	54
5.2.5	Messaging Error Codes .....	55
5.2.5.1	CIP Messaging General .....	55
5.2.5.2	DPS Diagnostic Request .....	57
5.2.5.3	DPV1 Class 1 Read and Write .....	58
5.2.5.4	DPV1 Class 1 Alarm Request .....	58
<b>6</b>	<b>DIAGNOSTICS AND TROUBLESHOOTING .....</b>	<b>59</b>
6.1	Hardware Diagnostics (LED) .....	59
6.1.1	CompactLogix .....	59
6.1.2	MicroLogix 1500 .....	59
6.1.3	RIF 1769 LEDs .....	60
6.2	Troubleshooting .....	61
6.2.1	CompactLogix I/O LED .....	61
6.2.2	MicroLogix Fault LED .....	61
6.2.3	SYS and COM Status LEDs .....	61
6.2.4	Error Sources and Reasons .....	61
<b>7</b>	<b>RSLOGIX EXAMPLE PROGRAM .....</b>	<b>63</b>
7.1	CompactLogix I/O Example .....	63
7.2	CompactLogix Messaging Example .....	65

7.3	MicroLogix I/O Example.....	67
8	A-SPECIFICATIONS.....	71
8.1	RSLogix5000 User Defined Data Types.....	71
8.2	RSLogix500 User Defined Data Files.....	78
8.3	Firmware Upgrade using Compro.....	79
8.3.1	Step1: Running Compro.....	79
8.3.2	Step2: Selecting the Download Process.....	80
8.3.3	Step3:Compro Download Process.....	81
8.4	Product Specifications.....	82
9	LISTS.....	83
9.1	List of Figures.....	83
9.2	List of Tables.....	84



# 1 Introduction

## 1.1 Intended Audience

The intended audiences for this manual are the individuals responsible for designing, installing, programming, or troubleshooting control systems that use Rockwell CompactLogix programmable controllers and the Hilscher RIF 1769-DPS PROFIBUS-DP Slave module. You should have a basic understanding of electrical circuitry and familiarity with relay logic. If you do not, obtain the proper training before using this product.

## 1.2 General Information RIF 1769-DPS

The communication module RIF 1769-DPS is a slot extension module for a CompactLogix Controller which enables the controller to communicate as a Slave on a PROFIBUS network. The RIF 1769-DPS is a PROFIBUS-DP Slave. The configuration of the PROFIBUS system is done by two different methods explained in the configuration section of this document. No external configuration tool is required. The data exchange between controller and module is done via the I/O process data image using CompactLogix back plane technology.

## 1.3 Software Requirements

Follows are the software requirements for using the RIF 1769-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

## 1.4 Hardware Requirements

The following minimum hardware is required to use the 1769 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.

## 1.5 Reference Manuals

### CompactLogix System

Manual	Description	Note
1769-IN047C-EN-P	CompactLogix Controller Installation Instructions	Rockwell Automation
1769-UM007D-EN-P	CompactLogix System User Manual	Rockwell Automation
RIF1769 Booklet.pdf	Booklet (Hardware installation RIF 1769, Wiring, LED displays, and technical data)	Hilscher GmbH

Table 1 : Reference Manuals CompactLogix System

### MicroLogix 1500 System

Manual	Description	Note
1762-rm001_-en-p.pdf	MicroLogix™ 1200 and MicroLogix™ 1500 Programmable Controllers Reference Manual	Rockwell Automation
1764-in001_-mu-p.pdf	MicroLogix™ 1500 Programmable Controllers Base Units Installation Instruction	Rockwell Automation
1764-um001_-en-p.pdf	MicroLogix™ 1500 Programmable Controllers User Manual	Rockwell Automation
RIF1769 Booklet.pdf	Booklet (Hardware installation RIF 1769, Wiring, LED displays, and technical data)	Hilscher GmbH

Table 2 : Reference Manuals MicroLogix 1500 System

## 1.6 Reference Systems

The firmware of the communication module RIF 1769-DPS was developed and tested with following CompactLogix / MicroLogix Controller types and firmware revisions.

### CompactLogix System

RIF 1769-DPS	CompactLogix 1769-L20	CompactLogix 1769-L32E
Firmware V1.000	Firmware V13.18	Firmware V13.28

Table 3 : CompactLogix Reference System

### MicroLogix 1500 System

RIF 1769-DPS	MicroLogix 1500 (Processor 1764-LRP/A Rev2.0)
Firmware V1.001	Firmware: OS 1510; Series C ; Revision 9.0

Table 4 : MicroLogix Reference System

## 1.7 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

Table 5 : 1769-Programmable Controller Functionality

### MicroLogix 1500 System

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

Table 6 : 1764-Programmable Controller Functionality

yes = functionality supported

no = functionality not supported

## 2 Installation, Wiring and System Planning

This section describes how to install and wire the RIF 1769-DPS Slave module. The following table describes what this chapter contains and where to find specific information. When planning, installing and wiring your system please read the referenced manuals.

### 2.1 CompactLogix System

#### **RIF1769 Booklet.pdf**

This manual can be found on the CD delivered with the RIF 1769-DPS module and contains detailed information about:

- How to assemble the RIF 1769-DPS module into a CompactLogix system.
- PROFIBUS wiring
- Modules LED displays
- Modules technical data and specifications
- ...

#### **1769-IN047C-EN-P.PDF / 1769-UM007D-EN-P.PDF**

These manuals are available from Rockwell Automation and can be found on every RSLogix CD or on the Homepage of Rockwell Automation.

These manuals contain **important** information about:

- CompactLogix System planning
- CompactLogix Controller Installation Instructions
- CompactLogix System specifications
- ...

## 2.2 MicroLogix 1500 System

### RIF1769 Booklet.pdf

This manual can be found on the CD delivered with the RIF 1769-DPS module and contains detailed information about:

- How to assemble the RIF 1769-DPS module into a CompactLogix system.

---

**Note:** The mechanical assembly/installation into a CompactLogix or MicroLogix 1500 is the same. That's why this manual refers only to the CompactLogix system.

---

- PROFIBUS wiring
- Modules LED displays
- Modules technical data and specifications
- ...

### 1764-IN001\_-MU-P.pdf / 1764-UM001\_-EN-P.pdf

These manuals are available from Rockwell Automation and can be found on the Homepage of Rockwell Automation.

These manuals contain **important** information about:

- MicroLogix System planning
- MicroLogix Controller Installation Instructions
- MicroLogix System specifications
- ...

### MircroLogix \_1500\_Qualifier\_February\_2005.xls

This EXCEL spreadsheet in its actual version is available from Rockwell Automation and can be found on the Homepage of Rockwell Automation.

- This system qualifier helps you to caculate the total amount of power consumption of a MicroLogix 1500 system with Compact I/O expansion modules
- ...

## 2.3 Consideration when planning the system:

- The PROFIBUS Slave Module has address rotary switches to set an address from 0-99. Using the PLCs configuration data an address range from 0-125 is adjustable.
- The Slave is capable of automatic baud rate detection.
- A 1769-ECR (right end cap) or 1769-ECL (left end cap) is required to terminate the end of the Compact I/O bus.
- Each bank of Compact I/O must have its own power supply.
- A Compact I/O power supply has limits on the amount of +5V dc and +24V dc current it can supply to modules in its I/O bank. These limits depend on the catalog number (e.g. 1769-PA2) of the supply. A bank of modules must not exceed the current limits of the I/O bank power supply. Refer to the Compact 1769 Expansion I/O Power Supplies Installation Instructions.
- The PROFIBUS module has a distance rating of 6; therefore, the Slave module must be within 6 modules of the I/O bank's power supply.
- Consider the number of words of I/O data the host controller supports.
- When planning a MicroLogix 1500 system use the System Qualifier: [MircoLogix \\_1500\\_Qualifier\\_February\\_2005.xls](#) spreadsheet which can be found on the Homepage of Rockwell Automation. This system qualifier helps you to calculate the total amount power consumption of a MicroLogix 1500 system with Compact I/O expansion modules.

## 3 PROFIBUS Functionality

### 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.

#### 3.1.1 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".

#### 3.1.2 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.

#### 3.1.3 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.

#### 3.1.4 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 Number") 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.

### 3.1.5 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".

## 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.

### 3.2.1 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.

### 3.2.2 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.

### 3.2.3 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.

## 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.

## 4 Configuration and Start-Up

The following sections will describe the individual steps for configuration and start-up of the RIF 1769-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 RSLogix500 programming tool.
- Determine configuration method to be used by the Slave module during startup.
- Creating the data objects and the ladder diagram in RSLogix5000 or RSLogix500.

## 4.1 RSLogix 5000

This section contains instructions for configuring the RIF 1769-DPS module in a CompactLogix system using RSLogix5000.

**Note:** The simplest way to startup the module in a RSLogix5000 project is to use the “RIF\_1769\_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.

### 4.1.1 Module Selection

Create a new project in RSLogix5000 using a CompactLogix controller. Then the first step is to select the module and add it to your project. Right-click the mouse on the I/O configuration CompactBus Local of the controller project. Select **New Module** from the context menu as shown below.

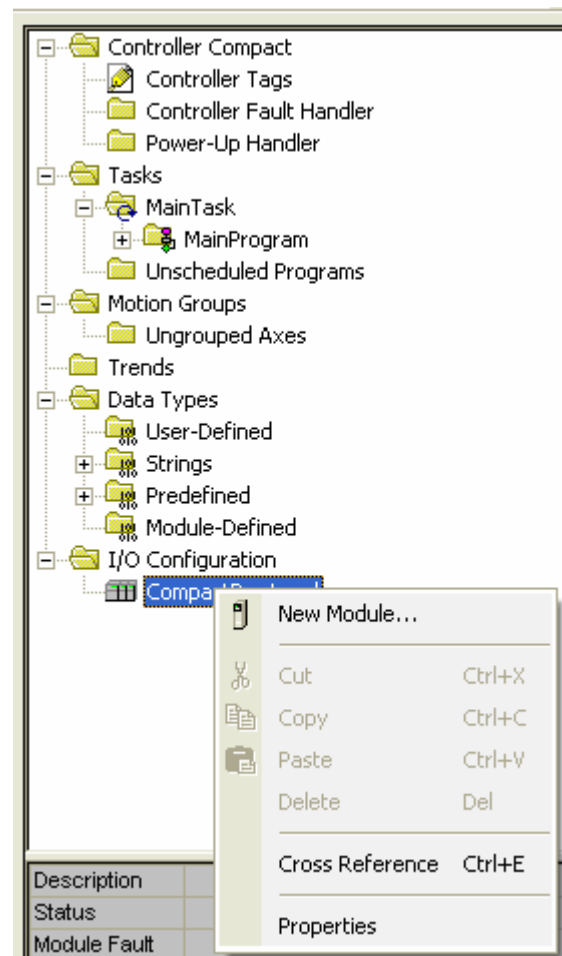


Figure 1 : Insert New Module

The following dialog box appears for the selection of the new Module.

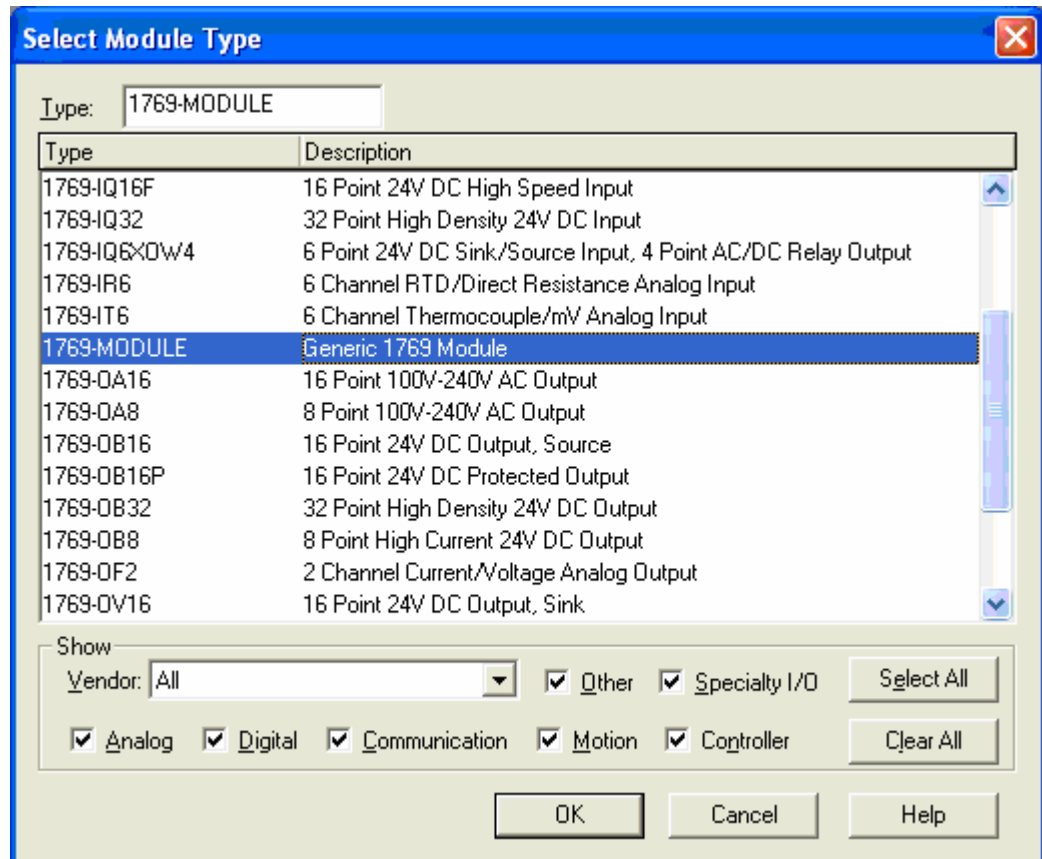


Figure 2 : Select Module Type

Select “**1769-MODULE Generic 1769 module**” from the select module type list and then OK.

## 4.1.2 Module Properties 1

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

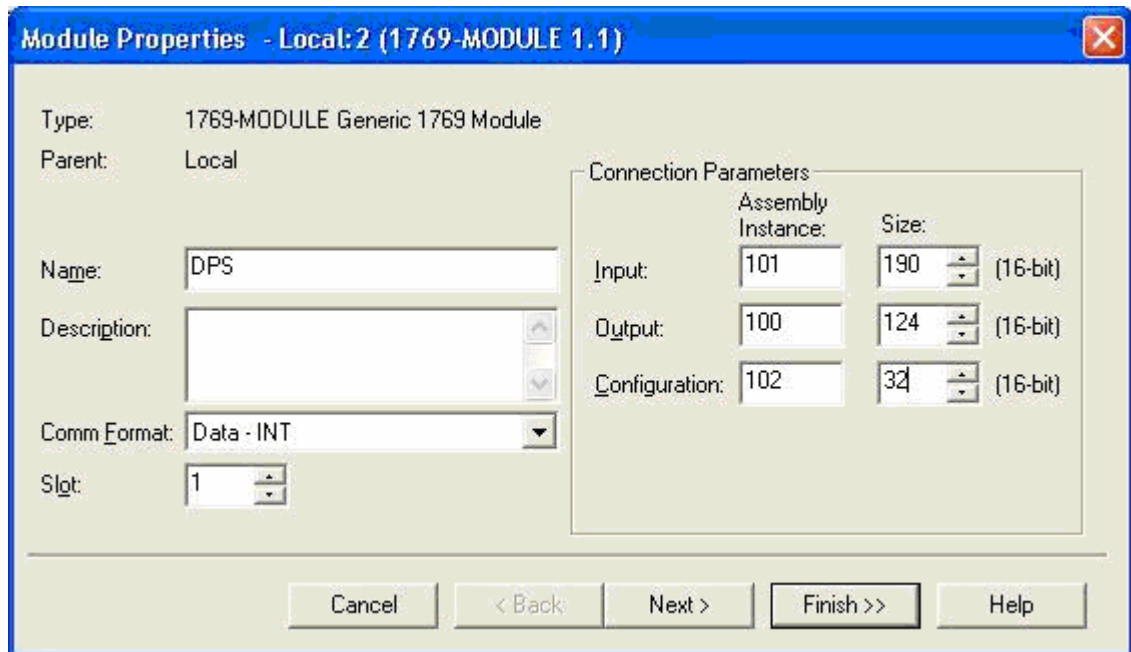


Figure 3 : Module Properties 1

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

Table 7 : Connection Parameters

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.

### 4.1.3 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 1769-DPS module supports all possible RPI values.

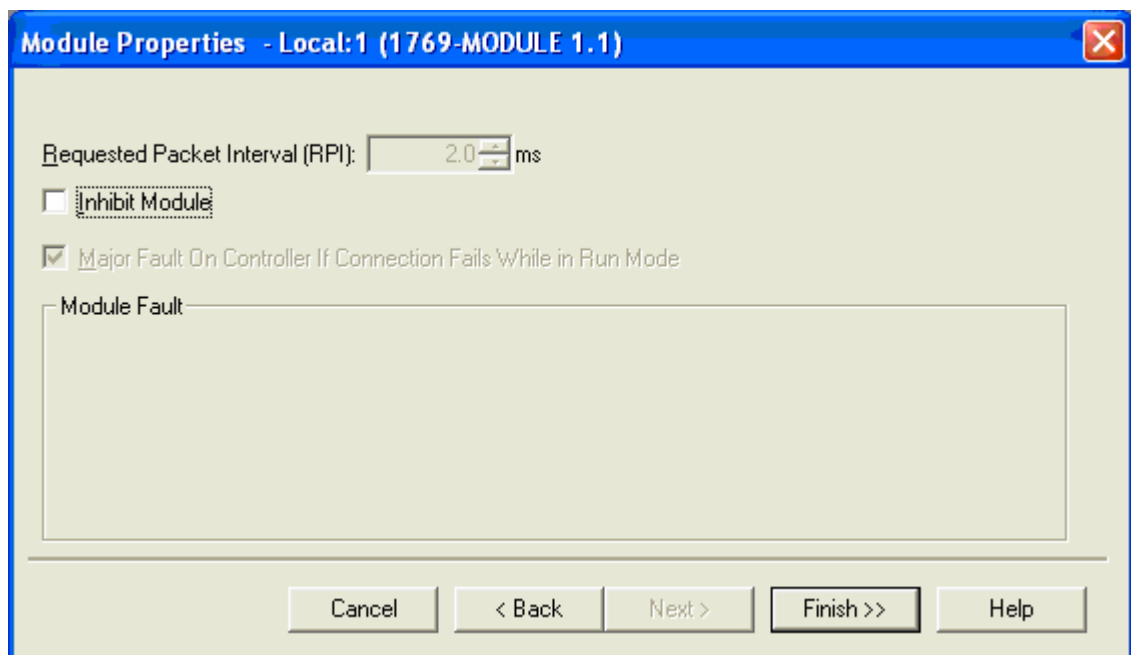


Figure 4 : Module Properties 2

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

## 4.2 RSLogix 500

Contained in the sections below are the instructions for configuring the RIF 1769-DPS module in a MicroLogix system using RSLogix500.

**Note:** The simplest way to startup the module in a RSLogix500 project is to use the “RIF\_1769\_DPS\_ML15.RSS” 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.

### 4.2.1 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. Right-click the mouse on the I/O configuration of the controller project. Select **Open** form the context menu as shown below.

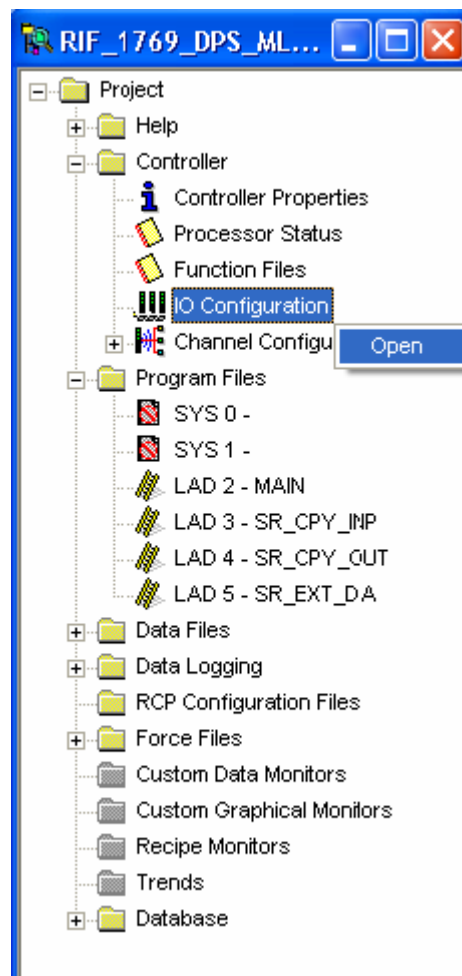


Figure 5: Open I/O Configuration

The following dialog box appears for the selection of the new Module.

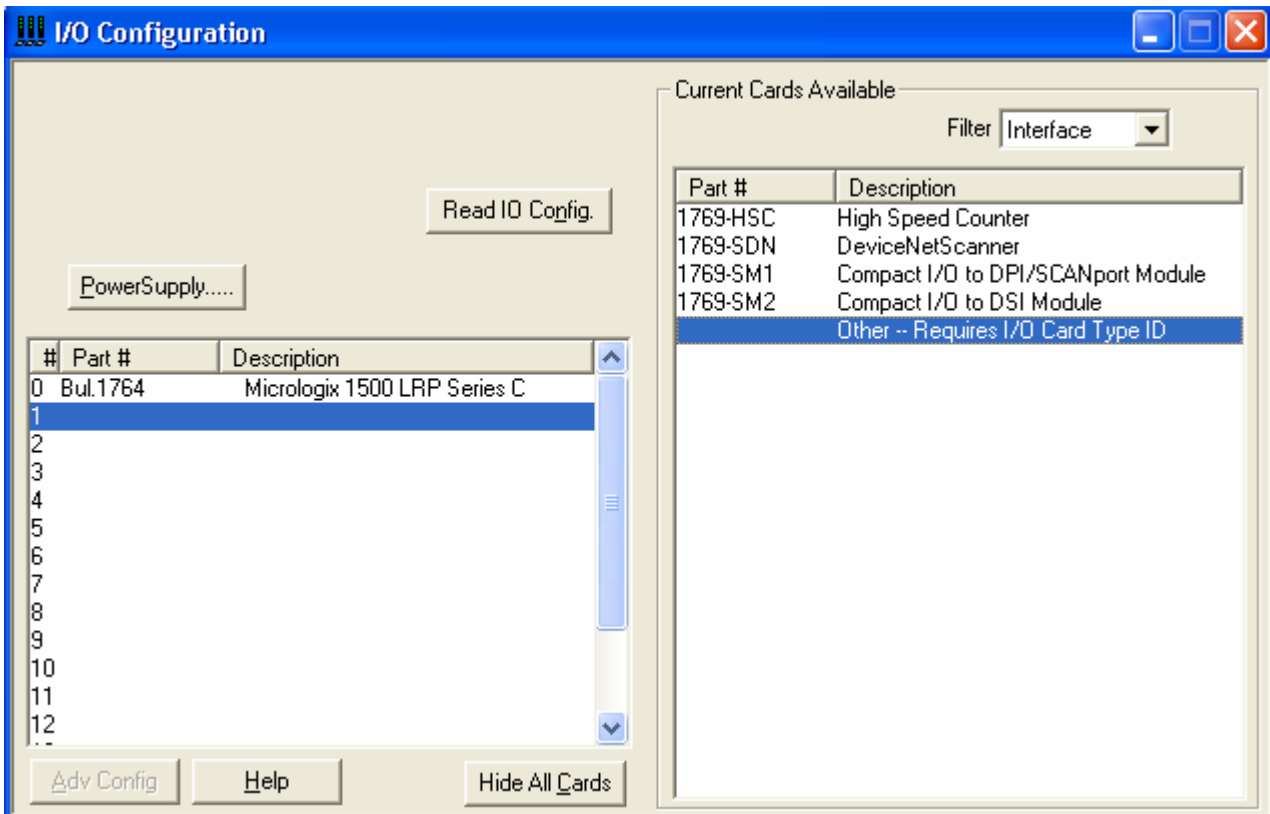


Figure 6 : Select Module Type

On the left side of the dialog set the focus to the slot number where the module is installed. Set the focus to **“Other – Requires I/O Card Type ID”** from the select module type list and then double click on it.

## 4.2.2 Expansion General Configuration

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

Module #1: OTHER - I/O Module - ID Code = 3

Expansion General Configuration | Generic Extra Data Config

Vendor ID: 283

Product Type: 12

Product Code: 4

Series/Major Rev/Minor Rev: A

Input Words: 190

Output Words: 124

Extra Data Length: 32

Ignore Configuration Error:

OK Abbrechen Übernehmen Hilfe

Figure 7 : Expansion General Configuration

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

Table 8 : Connection Parameters

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

- Vendor Name / Product Type / Product Code for the RIF 1769-DPS module are 283,12,4.
- 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) and 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 data image 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 contain the command information required by the module, which is 2 Words (4 bytes), and 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.

### 4.2.3 Generic Extra Data Config

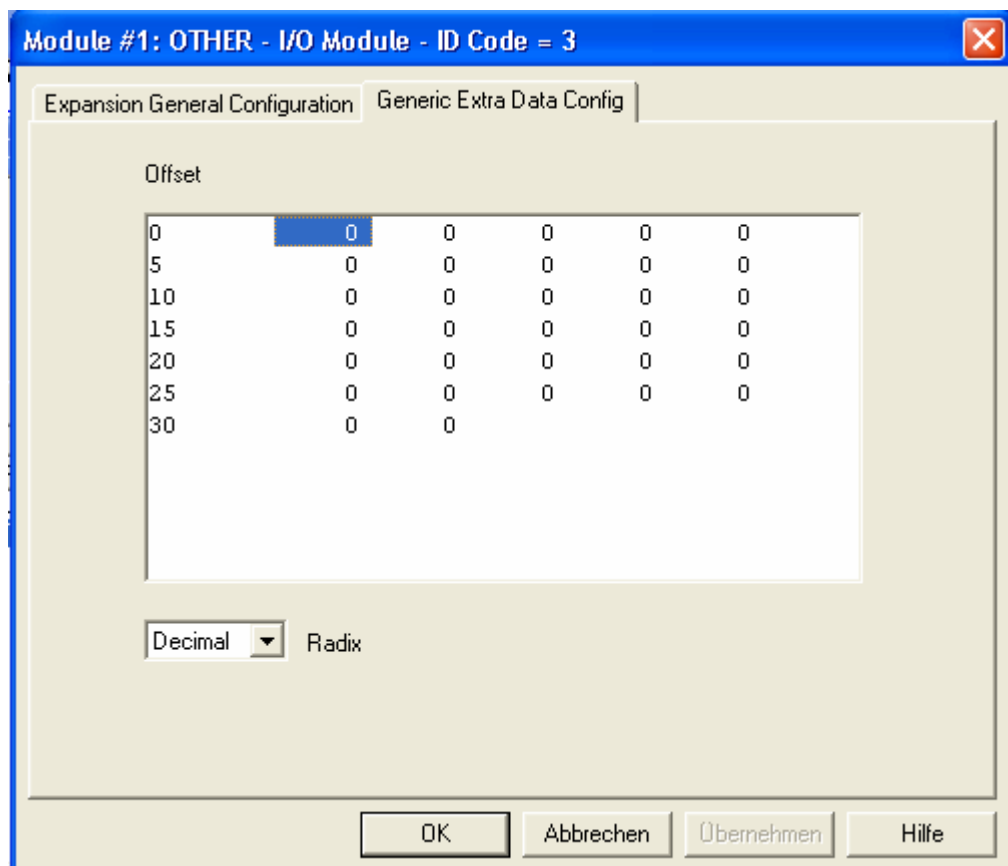


Figure 8 : Generic Extra Config

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**”.

Select **OK** to end the I/O configuration of the module.

## 4.3 Slave Configuration

### 4.3.1 General

The following section will detail the basics of configuring the RIF 1769-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.

### 4.3.2 GSD File

A GSD file is kind of an electronic datasheet for a particular Slave device. The GSD-File for the 1769-DPS slave named "HIL\_097A.GSD" is located on the CD 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.

### 4.3.3 Configuration by Master

The "Configuration by Master" is the easiest way to configure the Slave. The project "RIF\_1769\_DPS\_L32E.ACD" (for CompactLogix) and "RIF\_1769\_DPS\_ML15.RSS" (for MicroLogix) is an example which shows 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 RIF 1769-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.

---

### 4.3.4 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 table below 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'

Table 9 : DP Slave Configuration Data

## 4.3.5 Explanation of settable configuration values:

### 4.3.5.1 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 setup 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 initialisation error)

Table 10 : HW and SW Address Combinations

XX - Don't Care

### 4.3.5.2 Force User Configuration

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

### 4.3.5.3 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 Intervall). 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 multiples of 5ms.

#### 4.3.5.4 Number of Valid Configuration Bytes

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

#### 4.3.5.5 Module n Type / Module n Length

The 1769-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 1 = IN Word 2 = OUT Byte 3 = OUT Word 4 = IN Byte con 5 = IN Word con 6 = OUT Byte con 7 = OUT Word con 8 = Blank space	Input Byte without consistence Input Word without consistence Output Byte without consistence Output Word without consistence Input Byte with consistence Input Word with consistence Output Byte with consistence Output Word with consistence Blank space
ModuleLength	SINT	0 1 2 3 4 5 6 7 8 9	1 Byte/Word 2 Byte/Word 3 Byte/Word 4 Byte/Word 8 Byte/Word 12 Byte/Word 16 Byte/Word 20 Byte/Word 32 Byte/Word 64 Byte/Word

Table 11 : Coding of Module Types

**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 “RIF\_1769\_DPS\_L32E.ACD” / “RIF\_1769\_DPS\_ML15.RSS” 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 pre defined configuration parameter of the sample project.

## 5 Communication

### 5.1 IO Communication and IO Memory Map

Contained in the following sections are the I/O memory mappings for the RIF 1769-DPS interface. The I/O area will be used for communication of status and command information as well as cyclic I/O data.

#### 5.1.1 IO Array Overview

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

##### 5.1.1.1 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

Table 12 : Input Register Summary

### 5.1.1.2 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

Table 13 : Output Register Summary

## 5.1.2 Module Input Array

### 5.1.2.1 Device Status Registers

The RIF 1769-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

Table 14 : Device State Register

#### **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

Table 15 : Module Status Bits

- **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

### 5.1.2.2 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 table below.

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

Table 16 : Firmware Field

Example:

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

Because Hilscher use a different internal firmware numbering scheme than Major/Minor version the following method is used to utilize this information to support requirements for a Major/Minor revision of the CompactLogix controller. Details are provided in the table below. Because the first release of the modules internal firmware will start with at least V01.000 the first firmware version in Major Minor scheme will be at least 10.00.

Hilscher FW Revision	FW Major	FW Minor
V01.000	10	00
V01.001	10	01

Table 17 : Firmware Major/Minor mapping

### 5.1.2.3 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
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

Table 18 : Slave Status Information

**(\*)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:**

<b>Value (hex) (x =don't care)</b>	<b>Meaning</b>	<b>Description</b>
xxx1	Task is During initialization	If this state stays for some seconds, the configuration parameters may be invalid.
xx1x	Task running	The initialization happened 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.

Table 19 : Task State

### **LastError:**

<b>Value</b>	<b>Meaning</b>	<b>Description</b>
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 our hotline.

Table 20 : Last Error

### 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 table below 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

Table 21 : DPV1 Status Registers

### 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

Table 22 : DPV1 Read/Write Indication Status Bits

**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 1769-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	"RIF 1769"
FwVersion	SINT[8]	Firmware Version	"V01.000 "
FwDate	SINT[8]	Firmware Date	"25.07.05 "

Table 23 : Extended Status Information Firmware

**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

Table 24 : Extended Status Information Slave Configuration

**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

Table 25 : Extended Status Information Master Configuration

**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

Table 26 : Extended Status Information Parameter Data

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

Structure member	Data type	Description
StaReqUsr	DINT	Status Request from User
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

Table 27 : Extended Status Information DPV1-C1-Diag

**5.1.2.4 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.

## 5.1.3 Module Output Array

### 5.1.3.1 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

Table 28 : Device Command Register

#### **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

Table 29 : Module Command Bits

#### **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

Table 30 : ExtendedStatusSelect Values

**5.1.3.2 PROFIBUS Input Data**

The remainder of the output area is used for the PROFIBUS Input data to be 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 4<sup>th</sup> Byte (based on Start Index 0) in the modules output data area.

## 5.2 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 RIF 1769-DPS module not all of the MicroLogix 1500 controller support CIP messaging or CIP messaging for generic Compact I/O modules. That’s why CIP messaging, means PROFIBUS DPV1 services, are not yet supported with a MicroLogix System and a RIF 1769-DPS module.

---

### 5.2.1 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.

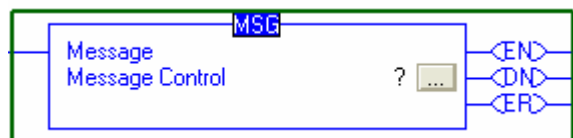


Figure 9 : “MSG” Instruction

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

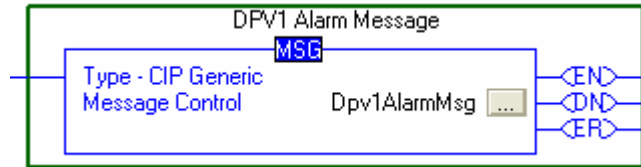



Figure 10 : "MSG" Instruction with Dpv1AlarmMsg

**Step3: Message Configuration**

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

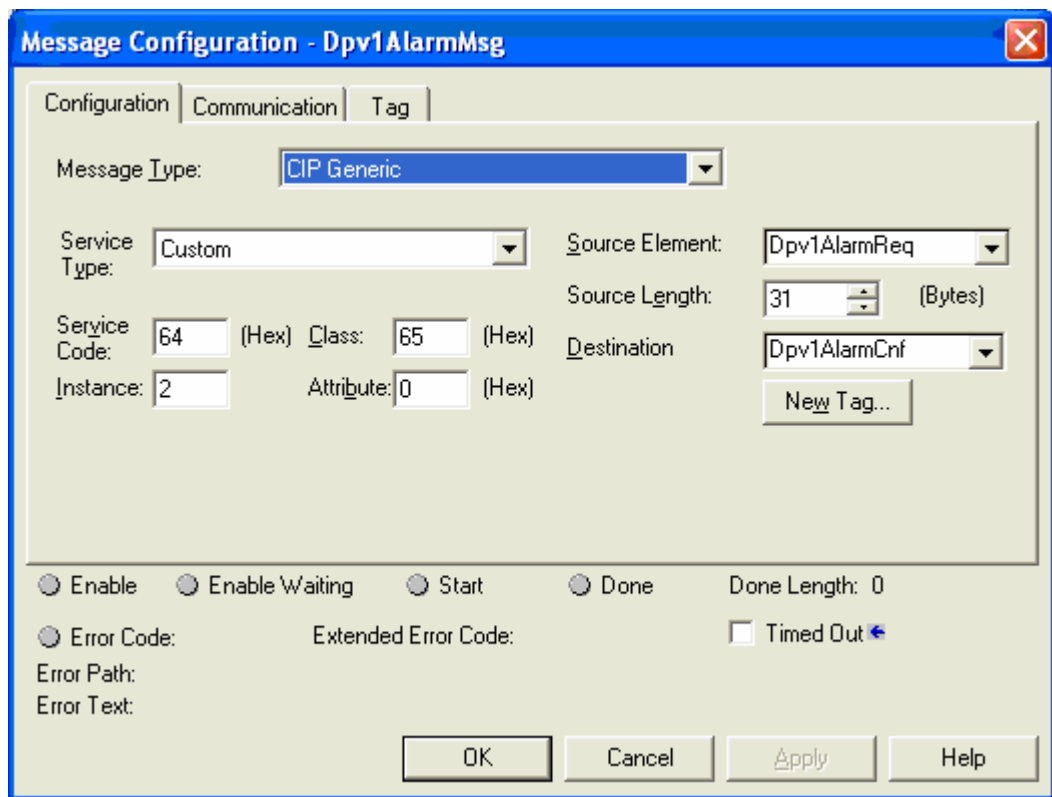


Figure 11 : Message Configuration - Configuration Tab

**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.

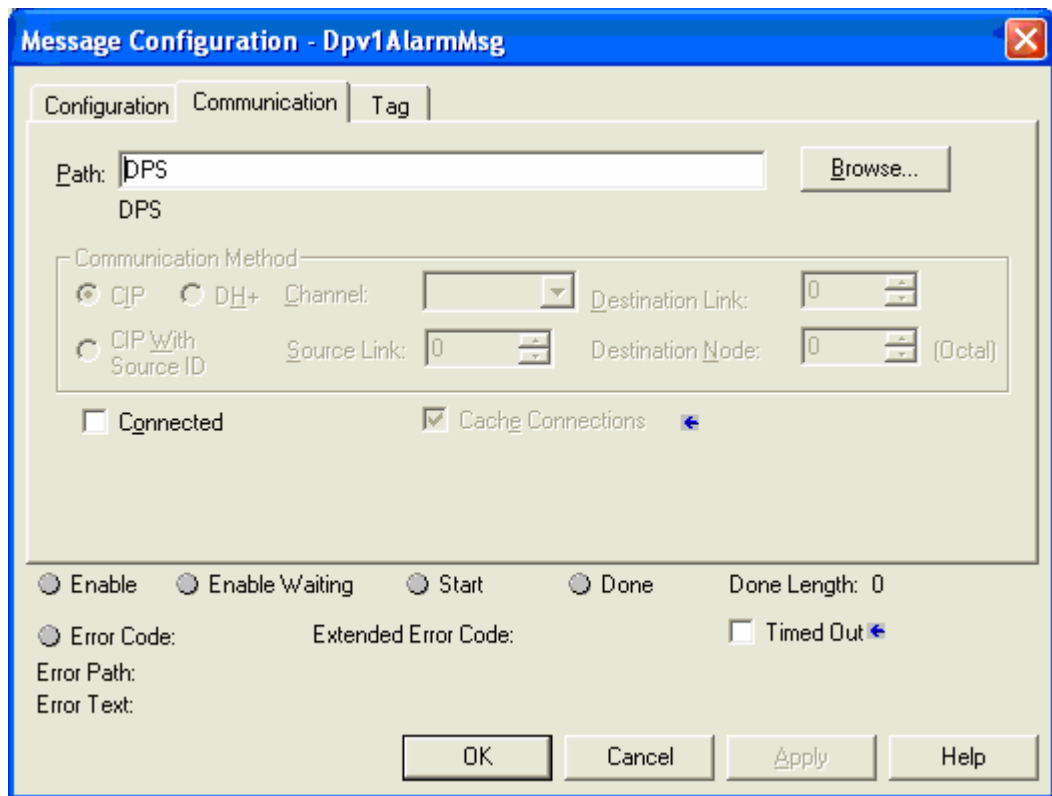


Figure 12 : Message Configuration - Communication Tab

The Path in the dialog above must point to the 1769-DPS Module. Use the Browse button to select the path.

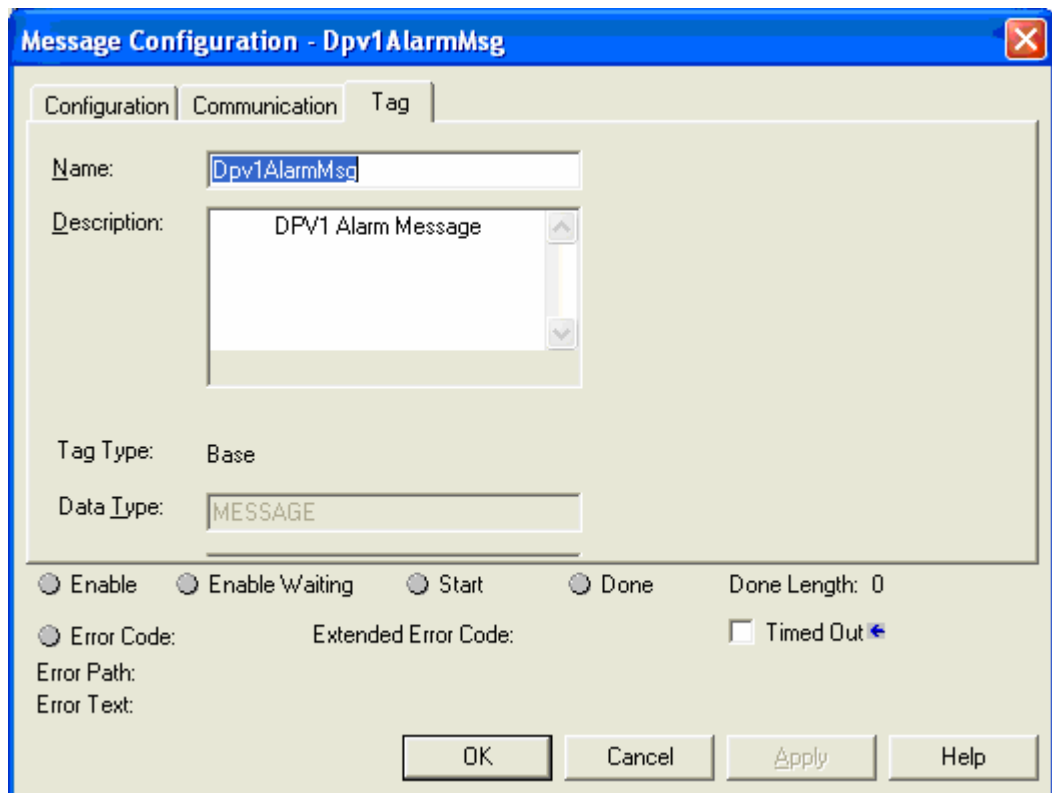


Figure 13 : Message Configuration - Tag Tab

**Step4: 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 RIF\_1769\_DPS\_messaging.ACD.

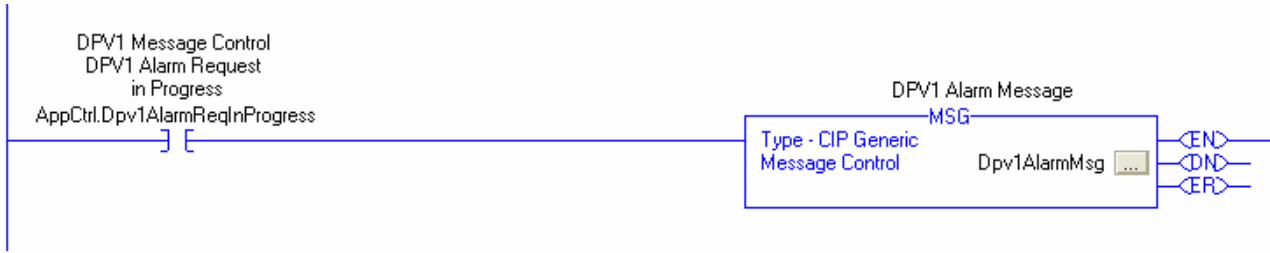


Figure 14 : Example MSG Logic

**5.2.2 Supported PROFIBUS-DP Messages**

The section shall define the message functions supported by the CompactLogix Slave module. Below is a summary of the functions that are supported.

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.

Table 31 : Supported PROFIBUS Messages

**Note:** Contained with in the “RIF\_1796\_DPS\_messaging\_L32E.ACD” project is an example for each of these services.

## 5.2.3 Standard Messaging

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

### 5.2.3.1 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 (*)

Table 32 : DPS Diagnostic Request

(\*) 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

Table 33 : 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

Table 34 : CIP Message Parameters for DPS Diagnostic Request

## 5.2.4 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 1769-DPS slave.

### 5.2.4.1 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\_DPVC1\_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.

Table 35 : DPV1 Class 1 Read 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 ErrorCode1 and ErrorCode1 will be transferred to the master. The ErrorCode must be PROFIBUS conform. For the proper format refer to the PROFIBUS norm..

**DPS\_DPV1C1\_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.

Table 36 : DPV1 Class 1 Read 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	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

Table 37 : CIP Message Parameters for DPV1 Class 1 Read Response

### 5.2.4.2 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\_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 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

Table 38 : DPV1 Class 1 Write 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 Errorcode1 and Errorcode1 which will be transferred to the master. But the Errorcode must conform to the PROFIBUS specification.

**DPS\_DPVC1\_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 send.

Table 39 : DPV1 Class 1 Write 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	Dpv1RWRespReq	Reference to a Tag of type DPS_DPVC1_RW_RESP_REQUEST
Destination	Dpv1RWRespCnf	Reference to a Tag of type DPS_DPVC1_RW_RESP_CONFIRM
Source Length	16	Corresponds to the constant size of DPS_DPVC1_RW_REQUEST structure

Table 40 : CIP Message Parameters for DPV1 Class 1 Write Response

### 5.2.4.3 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\_DPVC1\_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.

Table 41 : DPV1 Class 1 Alarm Request

**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

Table 42 : DPV1 Class 1 Alarm 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	Dpv1AlarmReq	Reference to a Tag of type DPS_DPVC1_ALARM_REQUEST
Destination	Dpv1AlarmCnf	Reference to a Tag of type DPS_DPVC1_ALARM_CONFIRM
Source Length	16 + n	n = Number user specific Alarm Data (0 .. 28)

Table 43 : CIP Message Parameters for DPV1 Class 1 Alarm Request

### 5.2.4.4 DPV1 Error Coding Scheme

Error Codes according to the DPV1 specification:

#### Error Code 1

B7	B6	B5	B4	B3	B2	B1	B0
Error Class (see below)				Error Code (see below)			

Table 44 : DPV1 Error Code 1

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
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	

Table 45 : DPV1 Error Code 1 Explanation

#### Error Code 2

Error code 2 is application specific.

## 5.2.5 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.

### 5.2.5.1 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
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 check the consistency of all length parameter within the requested command.
15 hex	0000 hex	Configuration data size too large	Too much data transferred with the CIP Message	Check if the overall length of the requested command send with the CIP message and the consistency of all length parameter within the requested command is correct.
16 hex	0000 hex	Object not supported	The requested object doesn't exist within the module.	

CIP Status	Extended Status	Meaning	Cause	Help
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 answer with proper Slot, Index etc. from Dpv1StatusRegister

Table 46 : CIP Message Error Codes

### 5.2.5.2 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.
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

Table 47 : Error Codes DPS Diagnostic Request

### 5.2.5.3 DPV1 Class 1 Read and Write

Failure	Error source
0	No error

Table 48 : Error Codes DPV1 Class 1 Read and Write

### 5.2.5.4 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

Table 49 : Error Codes DPV1 Class 1 Alarm Request

## 6 Diagnostics and Troubleshooting

This section details the possible diagnostics and troubleshooting procedures for the RIF 1769-DPS Slave module.

### 6.1 Hardware Diagnostics (LED)

The following section contains the LED diagnostic indications and their meaning for both the CPU in use and the RIF 1769-DPS module.

#### 6.1.1 CompactLogix

The table below 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

Table 50 : CompactLogix CPU LEDs

#### 6.1.2 MicroLogix 1500

To identify problems via possible LED indications of a MicroLogix1500 controller refer to the MicroLogix1500 User Manuals “1764-um001\_-en-p.pdf” chapter “C-Troubleshooting Your System” and “1762-um001\_-en-p.pdf” appendix C “Troubleshooting Your System“. Here you will find a detailed description of fault indications and possible reasons.

### 6.1.3 RIF 1769 LEDs

The LEDs as shown on the front panel will be used to indicate status information of the RIF 1769-DPS Slave module. Each LED has a specific function during Run time, configuration download, and error indications. The table below shows the reaction of each during these states for the Slave.

LED	Color	State	Description
<b>SYS</b>			
	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.
<b>COM</b>			
	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			

Table 51 : LED Diagnostic Indications

## 6.2 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.

### 6.2.1 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.

### 6.2.2 MicroLogix Fault LED

The faultless communication state is reached, if the Fault LED of the MicroLogix Controller is in off state.

If there is a problem with the expansion module the Fault LED is flashing red. Then go online with your RSLogix500 project and open up the processor status dialog and check the error tab for the fault reason.

### 6.2.3 SYS and COM Status LEDs

This RIF1769-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 static green or off. The meaning of the LEDs is described in the booklet of the System Software CD. If the SYS-LED is solid Green and the COM-LED static green, the card is in cyclic data exchange with the Master and the communication is running with out fault.

### 6.2.4 Error Sources and Reasons

This section describes typical problems, error sources and questions that come up while commissioning the PROFIBUS-DP Slave module RIF 1769-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 RIF 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
MicroLogix Fault LED is flashing Red	No communication with the RIF 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

Behavior	Significance	Typical Reason	Help
RIF 1769-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
RIF 1769-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

Table 52 : Troubleshooting

## 7 RSLogix Example Program

Provided on the installation CD are two example Ladder Logic programs RIF\_1769\_DPS\_L32E.ACD and RIF\_1769\_DPS\_Messaging\_L32E.ACD. These two 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 RIF 1769-DPS from the template projects to your own application

Sample Project	Controller Type	RSL5K Version	Description
RIF_1769_DPS_L32E.ACD	1769-L32E	V13	Basic CompactLogix I/O example
RIF_1769_DPS_Messaging_L32E.ACD	1769-L32E	V13	Basic CompactLogix messaging example

Table 53 : CompactLogix Sample Projects

Sample Project	Controller Type	RSL5H Version	Description
RIF_1769_DPS_ML15.RSS	1764-LRP	V6.30	Basic MicroLogix1500 I/O example

Table 54: MicroLogix1500 Sample Projects

### 7.1 CompactLogix I/O Example

This ladder logic program is a basic example for the setup of the PROFIBUS-DP Slave communications module "RIF 1769-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 the sample project is pre-initialized with following values:

Configuration word	Data type	Low/High Byte	Description	Configured values	Explanation
Local: 1:C.Data[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.
Local: 1:C.Data[1]	INT		Reserved		
Local: 1:C.Data[2]	INT		Watchdog Timeout	200	Watchdog 200 ms
Local: 1:C.Data[3]	INT		Number of valid config bytes (starting with Local:1:C.Data[8])	8	8 Bytes of the module definition array are valid
Local: 1:C.Data[4]	INT		Reserved		
Local: 1:C.Data[5]	INT		Reserved		
Local: 1:C.Data[6]	INT		Reserved		
Local: 1:C.Data[7]	INT		Reserved		
Local: 1:C.Data[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)
Local: 1:C.Data[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)
Local: 1:C.Data[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)
Local: 1:C.Data[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)
...			...		
...			...		
Local: 1:C.Data[31]	INT	LOW Byte	Module 24 Type	0	
		HIGH Byte	Module 24 Length	0	

Table 55 : I/O Example Pre Configuration

The parameter “ForceUserConfiguration” is initialized with 0. If you want to activate the pre-definded 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”.

## 7.2 CompactLogix Messaging Example

This ladder logic program is a CIP messaging example for the setup of the PROFIBUS-DP Slave communications module "RIF 1769-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 like 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 sent 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 check how many requests have been sent 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 check 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 are 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. Their definitions and the structures included in each are shown in the following tables.

## 7.3 MicroLogix I/O Example

This ladder logic program is a basic example for the setup of the PROFIBUS-DP Slave communications module "RIF 1769-DPS" in RSLogix500. This example can be used to start a project when using a CPU 1764-MicroLogix1500-LRP. Basic PROFIBUS I/O data exchange is shown. Details on the Subroutines created and the User Defined Data Types are as follows.

- **LAD 2 - MAIN:** The MainRoutine calls all of the following routines. This routine also contains a simple I/O transfer function block.
- **LAD 3 - SR\_CPY\_INP:** The SR\_Copy\_Input routine on every scan updates the DPS\_INUPT file with the Input Data of the module.
- **LAD 4 - SR\_CPY\_OUT:** The SR\_Copy\_Output routine on every scan updates the Output Data of the module with the data from local DPS\_OUTPUT file.
- **LAD 5 - SR\_EXT\_DIA:** 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 file. This routine looks at the ExtStaSelect value and copies the information into the appropriate user defined data type.

Numerous user defined data files 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 DPS\_INPUT and DPS\_OUTPUT. 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 the sample project is pre-initialized with following values:

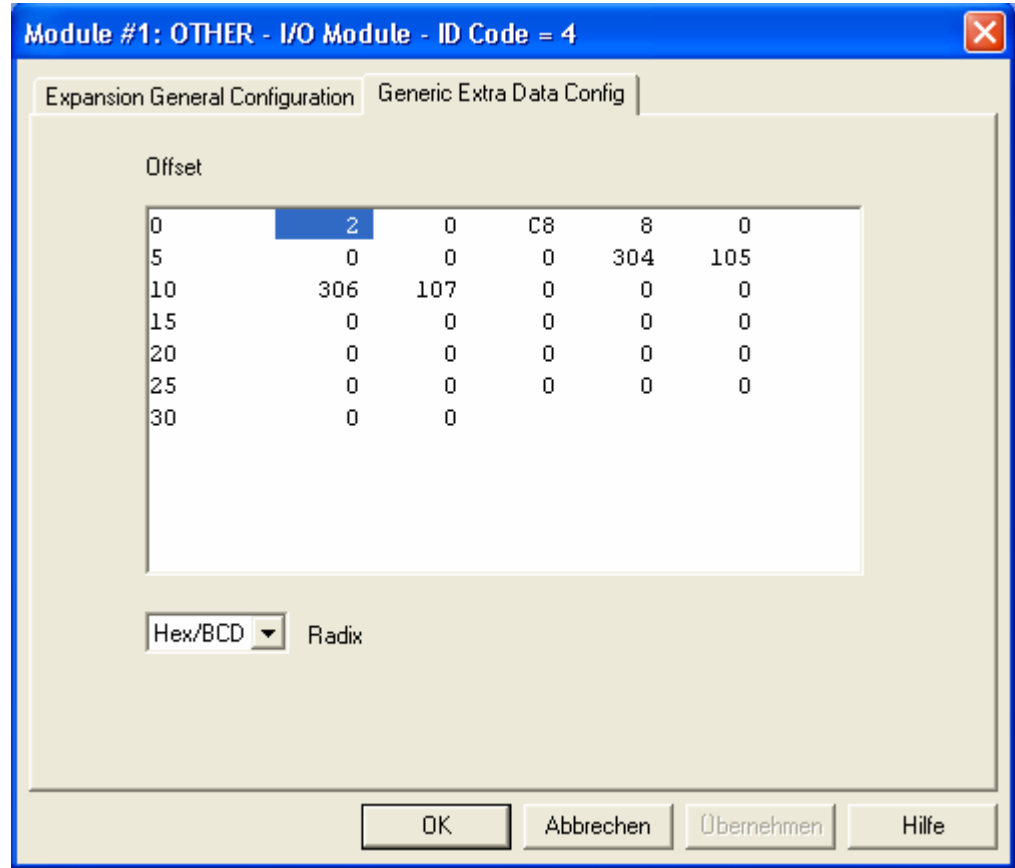


Figure 15 : MicroLogix ExtraDataConfig sample for RIF 169-DPS

Config 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)
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	

*Table 56 : I/O Example Pre Configuration*

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".



## 8 A-Specifications

### 8.1 RSLogix5000 User Defined Data Types

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

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

Table 57 : Input - DPS\_INPUT\_ARRAY

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

Table 58 : Input - DPS\_DEV\_STATUS\_REGISTER

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

Table 59 : Input - DPS\_FW\_REVISION

Name	Data Type	Description
ExtStaSelect	INT	Extended Status Select
ExtStaLen	INT	Extended Status Length
Baudrate	INT	Slave Baudrate
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

Table 60 : Input - DPS\_STATUS\_FIELD

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

Table 61 : Output - DPS\_OUTPUT\_ARRAY

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

Table 62 : Output - DPS\_DEV\_COMMAND\_REGISTER

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

Table 63 : APP\_CONSTANT\_PATTERN

Name	Data Type	Description
MainInitDone	BOOL	Main Initialization Complete
Dpv1ReadRespInProgress	BOOL	DPV1 Read Response in Progress
Dpv1WriteRespInProgress	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

Table 64 : APP\_DPV1\_PROG\_CONTROL

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

Table 65 : APP\_DP1\_STAT\_COUNTER

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

Table 66 : DPS\_DIAGNOSTIC\_CONFIRM

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)

Table 67 : DPS\_DIAGNOSTIC\_REQUEST

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

Table 68 : DPS\_DPV1C1\_ALARM\_CONFIRM

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

Table 69 : DPS\_DPVC1\_ALARM\_REQUEST

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

Table 70 : DPS\_DPVC1\_RW\_INDICATION

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

Table 71 : DPS\_DP1C1\_RW\_RESP\_CONFIRM

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

Table 72 : DPS\_DP1C1\_RW\_RESP\_REQUEST

## 8.2 RSLogix500 User Defined Data Files

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

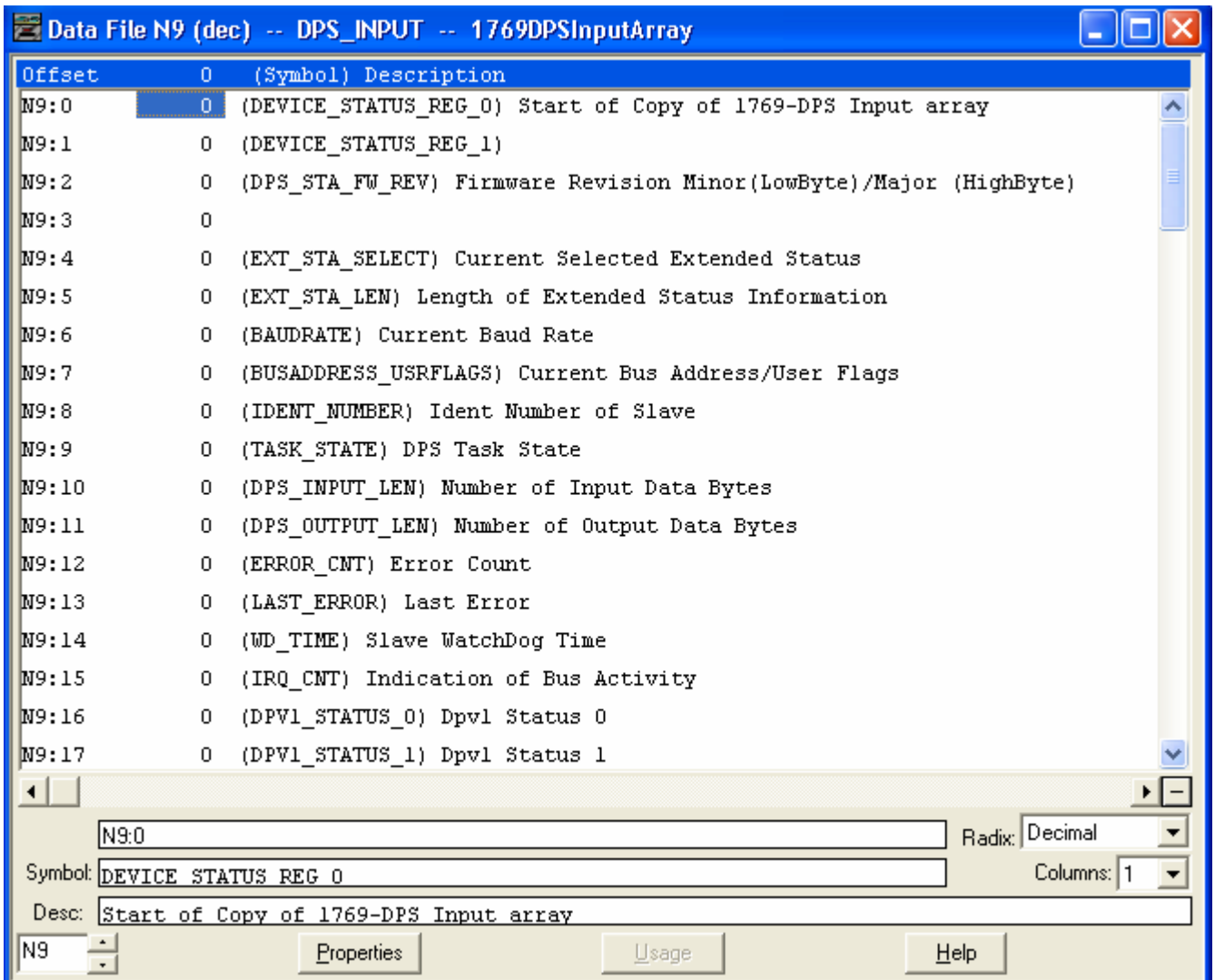


Figure 16 : N9 DPS\_INPUT Local copy of DPS Input array with Symbols and Description

File Name	Data Type	Array Size	Description
N9 DPS_INPUT	INT	68 ... 190	Local Copy of the 1769-DPS Input Array
N10 DPS_OUTPUT	INT	2 ... 124	Local Copy of the 1769-DPS Output Array
N11 FW_REV	INT	8	Ext. Status: Firmware Version
N12 SLV_CFG	INT	25	Ext. Status: SlaveConfiguration
N13 MAS_CFG	INT	25	Ext. Status: MasterConfiguration
N14 SLV_PRM	INT	16	Ext. Status: Slave Parameter Data

Table 73 : RSLogix500 User Defined Data Files

## 8.3 Firmware Upgrade using Compro

The modules firmware can be upgraded by using the Compro command line utility. This section contains the steps required to upgrade the modules firmware. To upgrade the firmware you need a serial diagnostic cable to connect your PC with the module. The Article Name is "CAB\_SRV\_MD8". Please contact your distributor to order this cable.

Connect the D-SUB Connector of the cable with a free COM port on your PC and the MINI-DIN-8 connector to the diagnostic interface in the front of the module.

### 8.3.1 Step1: Running Compro

The Compro utility can be found on the installation disk for this product. Please copy this utility to your local hard drive. In the same directory, copy the firmware file 1769DPS.E36 which will be the latest firmware you need to upgrade your module. The Compro utility requires the use of a communications port on your computer. Execute the Compro.exe utility by typing the follow at the command prompt:

Compro /S:1 <ENTER> for Com1

Compro /S:2 <ENTER> fro Com2

The following screen should appear.

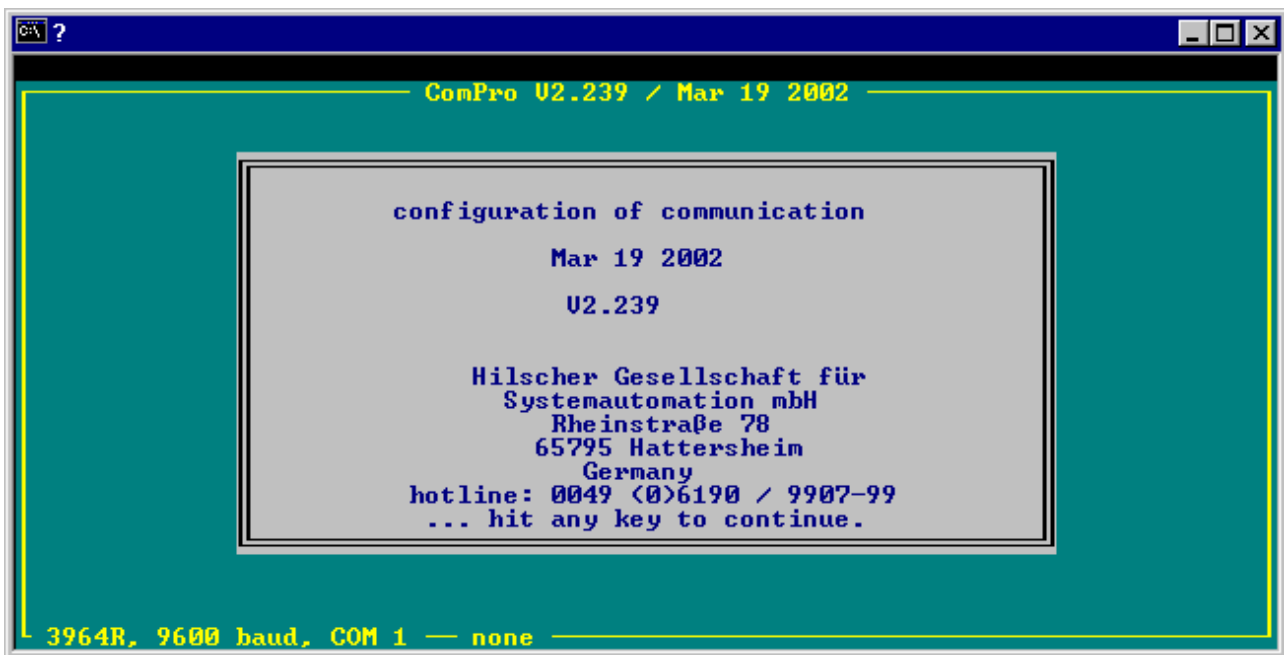


Figure 17 : Initial Compro Screen

Hit the <ENTER> key twice.

### 8.3.2 Step2: Selecting the Download Process

Using the arrow keys select Online>Software>Firmware Load then <ENTER>. Your screen should show the following.

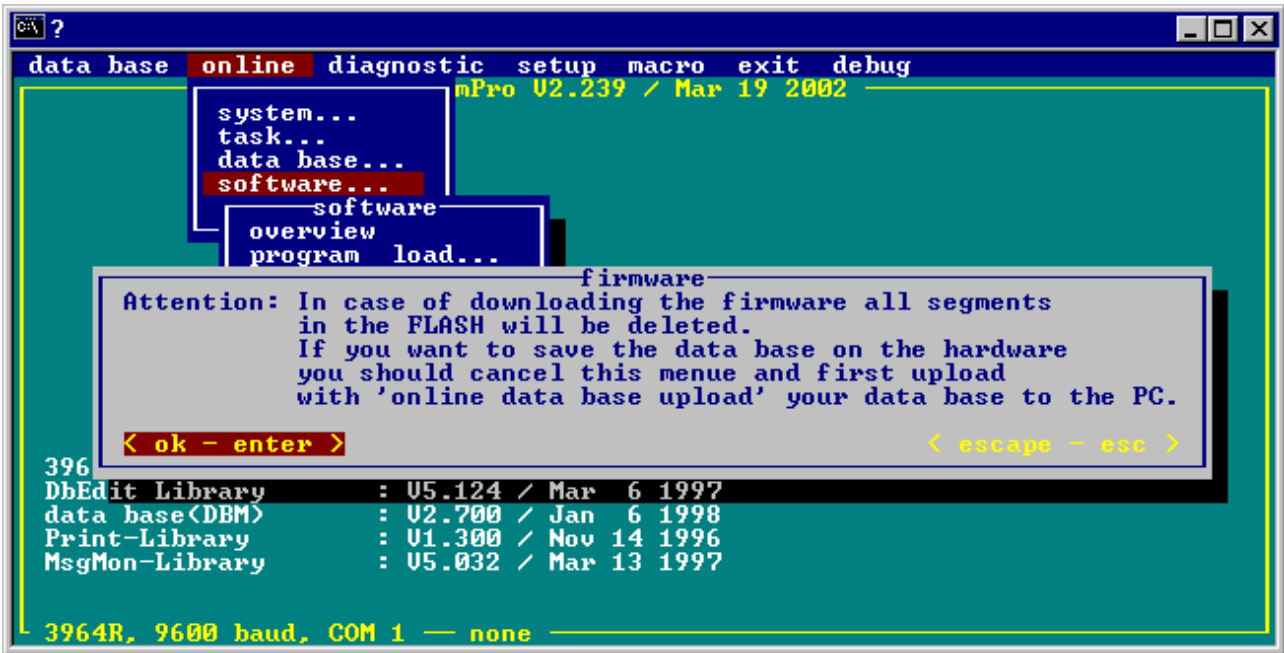


Figure 18 : Download 1

Hit <ENTER> and select the firmware as shown.

**Note:** It is not possible to download the Firmware while the Controller is in RUN mode.

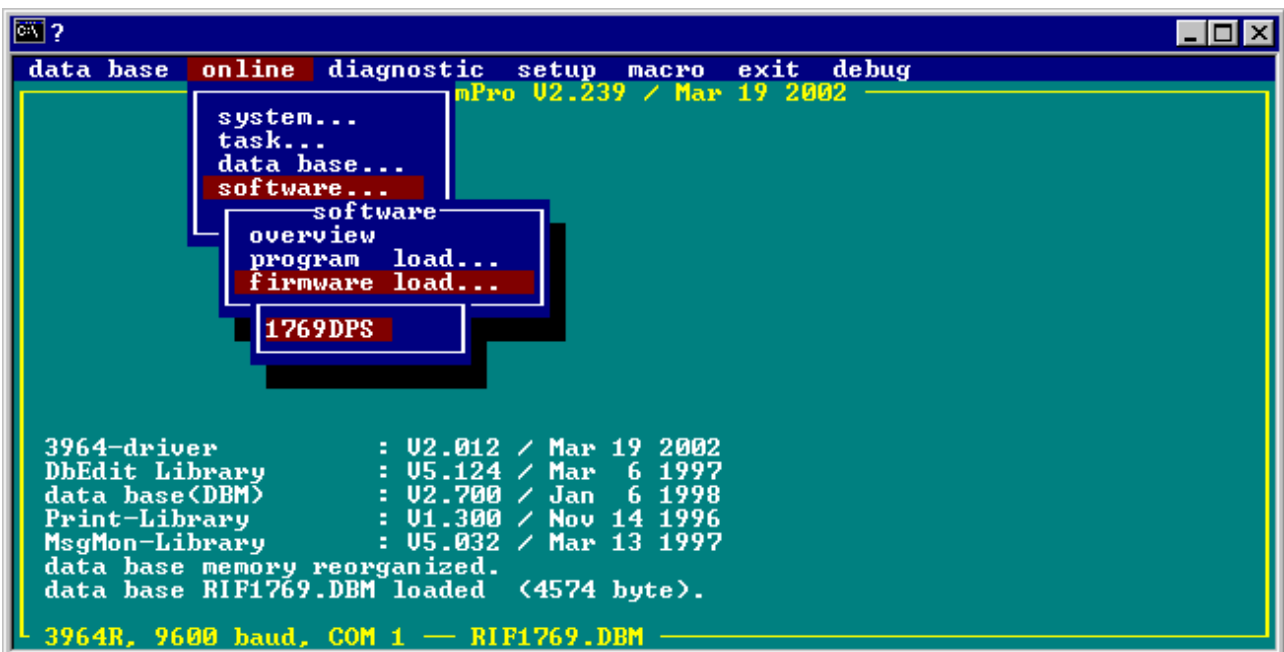


Figure 19 : Download 2

### 8.3.3 Step3:Compro Download Process

Once the download process has begun you should see the following screen.

```

COMPRO ?
data base  online  diagnostic  setup  macro  exit  debug
mPro U2.239 / Mar 19 2002
system...
task...
data base...
software...
software
overview
program load...
firmware load...

3964-driver
DbEdit Library
data base(DBM) : U2.700 / Jan 6 1998
Print-Library : U1.300 / Nov 14 1996
MsgMon-Library : U5.032 / Mar 13 1997
data base memory reorganized.
data base RIF1769.DBM loaded <4574 byte>.
RCS message : 100
can't find error description

3964R, 9600 baud, COM 1 — RIF1769.DBM
sending data rest: 323720 bytes

```

Figure 20 : Download 3

Please wait until the process is complete. You should see the number of bytes being downloaded decrease. The screen should return to a normal Compro screen and the red download window should disappear. Exit the program (go to Exit and hit <ENTER>) and reboot the module to ensure proper operation. After the new firmware is downloaded you must send the Master configuration again.

## 8.4 Product Specifications

For all technical data and electrical/environmental specifications of the module RIF 1769-DPS refer to the manual **RIF1769 Booklet.pdf** which can be found also on the CD delivered with the RIF 1769-DPS module.

## 9 Lists

### 9.1 List of Figures

Figure 1 : Insert New Module	19
Figure 2 : Select Module Type	20
Figure 3 : Module Properties 1	21
Figure 4 : Module Properties 2	22
Figure 5: Open I/O Configuration	23
Figure 6 : Select Module Type	24
Figure 7 : Expansion General Configuration	25
Figure 8 : Generic Extra Config	26
Figure 9 : "MSG" Instruction	42
Figure 10 : "MSG" Instruction with Dpv1AlarmMsg	43
Figure 11 : Message Configuration - Configuration Tab	43
Figure 12 : Message Configuration - Communication Tab	44
Figure 13 : Message Configuration - Tag Tab	44
Figure 14 : Example MSG Logic	45
Figure 15 : MicroLogix ExtraDataConfig sample for RIF 169-DPS	68
Figure 16 : N9 DPS_INPUT Local copy of DPS Input array with Symbols and Description	78
Figure 17 : Initial Compro Screen	79
Figure 18 : Download 1	80
Figure 19 : Download 2	80
Figure 20 : Download 3	81

## 9.2 List of Tables

Table 1 : Reference Manuals CompactLogix System	11
Table 2 : Reference Manuals MicroLogix 1500 System	11
Table 3 : CompactLogix Reference System	11
Table 4 : MicroLogix Reference System	11
Table 5 : 1769-Programmable Controller Functionality	12
Table 6 : 1764-Programmable Controller Functionality	12
Table 7 : Connection Parameters	21
Table 8 : Connection Parameters	25
Table 9 : DP Slave Configuration Data	28
Table 10 : HW and SW Address Combinations	29
Table 11 : Coding of Module Types	30
Table 12 : Input Register Summary	31
Table 13 : Output Register Summary	32
Table 14 : Device State Register	33
Table 15 : Module Status Bits	33
Table 16 : Firmware Field	34
Table 17 : Firmware Major/Minor mapping	34
Table 18 : Slave Status Information	35
Table 19 : Task State	36
Table 20 : Last Error	36
Table 21 : DPV1 Status Registers	37
Table 22 : DPV1 Read/Write Indication Status Bits	37
Table 23 : Extended Status Information Firmware	38
Table 24 : Extended Status Information Slave Configuration	38
Table 25 : Extended Status Information Master Configuration	38
Table 26 : Extended Status Information Parameter Data	39
Table 27 : Extended Status Information DPV1-C1-Diag	39
Table 28 : Device Command Register	40
Table 29 : Module Command Bits	40
Table 30 : ExtendedStatusSelect Values	41
Table 31 : Supported PROFIBUS Messages	45
Table 32 : DPS Diagnostic Request	46
Table 33 : DPS Diagnostic Confirmation	47
Table 34 : CIP Message Parameters for DPS Diagnostic Request	47
Table 35 : DPV1 Class 1 Read Response	48
Table 36 : DPV1 Class 1 Read Confirmation	49
Table 37 : CIP Message Parameters for DPV1 Class 1 Read Response	49
Table 38 : DPV1 Class 1 Write Response	50
Table 39 : DPV1 Class 1 Write Confirmation	51
Table 40 : CIP Message Parameters for DPV1 Class 1Write Response	51
Table 41 : DPV1 Class 1 Alarm Request	52
Table 42 : DPV1 Class 1 Alarm Confirmation	53
Table 43 : CIP Message Parameters for DPV1 Class 1 Alarm Request	53
Table 44 : DPV1 Error Code 1	54
Table 45 : DPV1 Error Code 1 Explanation	54
Table 46 : CIP Message Error Codes	56
Table 47 : Error Codes DPS Diagnostic Request	57
Table 48 : Error Codes DPV1 Class 1 Read and Write	58
Table 49 : Error Codes DPV1 Class 1 Alarm Request	58
Table 50 : CompactLogix CPU LEDs	59

Table 51 : LED Diagnostic Indications	60
Table 52 : Troubleshooting	62
Table 53 : CompactLogix Sample Projects	63
Table 54: MicroLogix1500 Sample Projects	63
Table 55 : I/O Example Pre Configuration	64
Table 56 : I/O Example Pre Configuration	69
Table 57 : Input - DPS_INPUT_ARRAY	71
Table 58 : Input - DPS_DEV_STATUS_REGISTER	71
Table 59 : Input - DPS_FW_REVISION	71
Table 60 : Input - DPS_STATUS_FIELD	72
Table 61 : Output - DPS_OUTPUT_ARRAY	72
Table 62 : Output - DPS_DEV_COMMAND_REGISTER	72
Table 63 : APP_CONSTANT_PATTERN	72
Table 64 : APP_DPV1_PROG_CONTROL	73
Table 65 : APP_DPV1_STAT_COUNTER	74
Table 66 : DPS_DIAGNOSTIC_CONFIRM	74
Table 67 : DPS_DIAGNOSTIC_REQUEST	75
Table 68 : DPS_DPV1C1_ALARM_CONFIRM	75
Table 69 : DPS_DPV1C1_ALARM_REQUEST	76
Table 70 : DPS_DPV1C1_RW_INDICATION	76
Table 71 : DPS_DPV1C1_RW_RESP_CONFIRM	77
Table 72 : DPS_DPV1C1_RW_RESP_REQUEST	77
Table 73 : RSLogix500 User Defined Data Files	78