BASEstar™ Open DAS for Allen-Bradley Data Highway™ on OpenVMS™

Installation and User's Guide

Order Number: AA-Q5G1D-TE

April 2000

This manual describes how to install and use the BASEstar Open Device Access Software for Allen-Bradley Data Highway.

Revision/Update Information:	This is a revised document.
Operating System and Version:	OpenVMS/Alpha Version 6.1
Operating System and Version:	OpenVMS/VAX Version 6.1
Interface Software and Version:	BASEstar Open Version 3.1
Software Version:	Device Access Software for Allen-Bradley Data Highway, Version 3.1A

Compaq Computer Corporation
Houston, Texas

April 2000

© 1994 Compaq Computer Corporation.

COMPAQ, VAX and VMS Registered in U.S. Patent and Trademark Office.

BASEstar and OpenVMS are trademarks of Compaq Information Technologies Group, L.P. in the United States and/or other countries.

PLC, PLC-2, PLC-3 and PLC-5 are registered trademarks of Allen-Bradley Company Inc.

PLC-5/250, SLC, SLC 500, SLC 5/01, SLC 5/02, SLC 5/03, SLC 5/04, SLC 5/05, Data Highway, DH+ and Data Highway Plus are trademarks of Allen-Bradley Company, Inc.

All other products mentioned herein may be trademarks or registered trademarks of their respective companies.

Confidential computer software. Valid license from Compaq or authorized sublicensor required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentaion, and Technical data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Compaq shall not be liable for technical or editorial errors or omissions contained herein. The information in this document is subject to change without notice.

This document is available on CD–ROM.

This document was prepared using DECdocument, Version 3.3.

Contents

Preface vii

1 Overview

1.1 Description 1–1
1.2 Device Communications 1–1
1.3 Supported Functions and Devices 1–2

2 Installing and Configuring the DAS

2.1 Installation Requirements 2–1
2.1.1 Hardware 2–1
2.1.2 Software 2–3
2.1.3 Disk Space 2–4
2.2 Installing the DAS 2–4
2.2.1 Files Created During Installation 2–6
2.2.2 Installation Messages 2–6
2.3 Postinstallation Tasks 2–8
2.3.1 Editing the Configuration Files 2–8
2.3.1.1 Editing ODS Registrations 2–8
2.3.1.2 Editing BASEstar Open Protocol Profile Definitions 2–10
2.3.1.3 Editing VMD Definitions 2–11
2.3.1.4 Configuration Files 2–13
2.3.2 Configuring Ports 2–16
2.3.3 Configuring TELNET Access 2–17
2.3.4 Setting Up Plant-Floor Equipment 2–18
2.4 Tracing device communications 2–18
2.5 Failures During Product Use 2–19

3 Using the DAS

3.1 Accessing Allen-Bradley Functions 3–1
3.2 Supported Functions 3–1
3.2.1 Connect, Conclude and Abort Functions 3–2
3.2.2 Read and Write Variable Functions 3–2
3.2.3 Read and Write Data Point Functions 3–5
3.2.4 Receiving Indications 3–8
3.2.4.1 Write Indications 3–8
3.2.4.2 Abort Indications 3–11
3.3 Addressing Syntax and Supported Data Types 3–12
3.3.1 PLC-2 Addressing Syntax and Supported Data Types 3–12
3.3.2 PLC-3 Addressing Syntax and Supported Data Types 3–12
3.3.2.1 PLC-3 Data Table Addressing 3–13
3.3.2.2 PLC-3 Extended Addressing 3–17

3.3.3 PLC-5 Addressing Syntax and Supported Data Types 3–19
3.3.4 PLC-5/250 Addressing Syntax and Supported Data Types 3–29
3.3.5 SLC Addressing Syntax and Supported Data Types 3–44

A I/O Error Messages

B DAS Error Messages

Index

Examples

2–1 ODS Registration 2–8
2–2 BASEstar Open Protocol Profile Configuration 2–10
2–3 VMD Configuration 2–11
2–4 ODS Configuration File 2–13
2–5 BSTR Configuration File 2–14
3–1 UNNAMED_VARIABLE CREATE Command 3–4
3–2 UNNAMED_VARIABLE GET VALUE Output 3–4
3–3 UNNAMED_VARIABLE PUT VALUE Output 3–5
3–4 DEVICE_DATA_POINT CREATE Command 3–6
3–5 DATA_POINT GET VALUE Output 3–7
3–6 DATA_POINT PUT VALUE Output 3–7
3–7 Write Indication Configuration 3–8
3–8 AB.PAR Contents 3–9
3–9 Write Indication Example 3–10

Figures

1–1 DAS Communications 1–2
2–1 Allen-Bradley 15 Pin Device Connector Layout 2–2
2–2 Allen-Bradley 25 Pin Device Connector Layout 2–2
2–3 Allen-Bradley 9 Pin Device Connector Layout 2–3
3–1 General PLC-2 Address Format 3–12
3–2 PLC-3 I/O Data Table Address Format 3–13
3–3 PLC-3 Timer and Counter Data Table Address Format 3–14
3–4 PLC-3 Timer Structure 3–14
3–5 PLC-3 Counter Structure 3–15
3–6 PLC-3 ASCII, Binary, Decimal, Floating-Point, High-Order Integer, Integer and Status Data Table Address Format 3–16
3–7 PLC-3 Pointer Data Table Address Format 3–17
3–8 PLC-3 Pointer Structure 3–17
3–9 PLC-3 Extended Address Format 3–18
3–10 PLC-5 I/O Data Table Address Format 3–19
3–11 PLC-5 Status Data Table Address Format 3–20
3–12 PLC-5 ASCII, Binary, Decimal, Floating-Point and Integer Data Table Address Format 3–21

3–13	PLC-5 Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control Data Table Address Format	3–22
3–14	PLC-5 Timer Structure	3–22
3–15	PLC-5 Counter Structure	3–23
3–16	PLC-5 Control Structure	3–23
3–17	PLC-5 SFC Status Structure	3–23
3–18	PLC-5 Block-Transfer Control Structure	3–24
3–19	PLC-5 Token Data Structure	3–24
3–20	PLC-5 ASCII String Structure	3–24
3–21	PLC-5 PID Control Structure	3–25
3–22	PLC-5 Message Control Structure	3–26
3–23	PLC-5/250 Binary, Floating-Point, Long Integer and Integer Address Format	3–30
3–24	PLC-5/250 Timer, Counter, Control, ASCII String, PID Control and Message Control Address Format	3–31
3–25	PLC-5/250 Timer Structure	3–31
3–26	PLC-5/250 Counter Structure	3–32
3–27	PLC-5/250 Control Structure	3–32
3–28	PLC-5/250 ASCII String Structure	3–32
3–29	PLC-5/250 PID Control Structure	3–33
3–30	PLC-5/250 Message Control Structure	3–34
3–31	PLC-5/250 Status Address Format	3–36
3–32	PLC-5/250 I/O Address Format	3–37
3–33	PLC-5/250 Block-Transfer Data Address Format	3–38
3–34	PLC-5/250 Block-Transfer Read/Write Control Address Format	3–39
3–35	PLC-5/250 Block-Transfer Read/Write Control Structure	3–39
3–36	PLC-5/250 Adapter Status Address Format	3–40
3–37	PLC-5/250 Adapter Status Structure	3–41
3–38	PLC-5/250 Internal Storage Address Format	3–42
3–39	PLC-5/250 Shared Data Address Format	3–43
3–40	SLC I/O Data Table Address Format	3–44
3–41	SLC Status Data Table Address Format	3–45
3–42	SLC ASCII, Binary, Floating-Point and Integer Data Table Address Format	3–46
3–43	SLC Timer, Counter, Control and ASCII String Data Table Address Format	3–47
3–44	SLC Timer Structure	3–47
3–45	SLC Counter Structure	3–48
3–46	SLC Control Structure	3–48
3–47	SLC ASCII String Structure	3–48

Tables

2–1	Disk Space Requirements	2–4
2–2	Files Created During Installation	2–6
2–3	REGISTER Attributes	2–9
2–4	VMD CREATE	2–12
2–5	KE/KF/KF-2/KF-3 Switch Settings	2–18
3–1	Connect, Conclude and Abort Interface Access	3–2
3–2	Read and Write Variable Types	3–3
3–3	Read and Write Variable Interface Access	3–3
3–4	Read and Write Data Point Interface Access	3–5
3–5	Indication Interface Access	3–8
3–6	Supported PLC Write Functions	3–11
3–7	Example PLC-2 Data Types	3–12
3–8	Example PLC-3 I/O Data Table Addresses	3–14
3–9	Example PLC-3 Timer and Counter Data Table Addresses	3–15
3–10	Example PLC-3 ASCII, Binary, Decimal, Floating-Point, High-Order Integer, Integer and Status Data Table Address Format	3–16
3–11	Example PLC-3 Pointer Data Table Addresses	3–17
3–12	Example PLC-3 Extended Addresses	3–18
3–13	Example PLC-5 I/O Data Table Addresses	3–20
3–14	Example PLC-5 Status Data Table Addresses	3–20
3–15	Example PLC-5 ASCII, Binary, Decimal, Floating-Point and Integer Data Table Address Format	3–21
3–16	Example PLC-5 Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control Data Table Address Formats	3–26
3–17	Example PLC-5/250 Binary, Floating-Point, Long Integer and Integer Address Format	3–30
3–18	Example PLC-5/250 Timer, Counter, Control, ASCII String, PID Control and Message Control Address Formats	3–34
3–19	Example PLC-5/250 Status Addresses	3–36
3–20	Example PLC-5/250 I/O Addresses	3–37
3–21	Example PLC-5/250 Block-Transfer Data Addresses	3–38
3–22	Example PLC-5/250 Block-Transfer Read/Write Control Addresses	3–40
3–23	Example PLC-5/250 Adapter Status Addresses	3–41
3–24	Example PLC-5/250 Internal Storage Addresses	3–42
3–25	Example PLC-5/250 Shared Data Addresses	3–43
3–26	Example SLC I/O Data Table Addresses	3–45
3–27	Example SLC Status Data Table Addresses	3–45
3–28	Example SLC ASCII, Binary, Floating-Point and Integer Data Table Address Format	3–46
3–29	Example SLC Timer, Counter, Control and ASCII String Data Table Address Formats	3–48

Preface

This document describes how to install and use the BASEstar Open DAS for Allen-Bradley Data Highway.

Intended Audience

This document is intended for system managers who must set up and maintain the following:

- BASEstar Open software
- BASEstar Open DAS for Allen-Bradley Data Highway

This document is also intended for application programmers who develop plant-floor management software layered on BASEstar Open software.

Readers of this document should have knowledge of:

- OpenVMS operations and administration
- OpenVMS application software
- BASEstar Open software
- Site-specific installation requirements

In addition, knowledge of the Allen-Bradley PLCs is required.

Document Structure

This document is organized as follows:

- Chapter 1 provides an overview of the DAS for Allen-Bradley Data Highway.
- Chapter 2 provides the information you need to install and configure the DAS for Allen-Bradley Data Highway.
- Chapter 3 provides information about the supported functions for Allen-Bradley PLCs, and how to access the functions.

Associated Documents

For information on installing and configuring a BASEstar Open system, refer to the following documents:

- *BASEstar Open for OpenVMS Release Notes*
- *BASEstar Open for OpenVMS Installation Guide*
- *BASEstar Open for OpenVMS Management Guide*

For information on installing and configuring a BASEstar Open system, refer to the following documents:

- *BASEstar Open for OpenVMS Release Notes*

- *BASEstar Open for OpenVMS Installation Guide*
- *BASEstar Open for OpenVMS Management Guide*

For information on using BASEstar Open software functions, refer to the following documents:

- *BASEstar Open for OpenVMS Introduction*
- *BASEstar Open for OpenVMS Reference Guide (Model)*
- *BASEstar Open for OpenVMS Command Language Interface*
- *BASEstar Open for OpenVMS Error Messages Guide*

For more information on developing software to integrate manufacturing applications and equipment, refer to the following documents:

- *BASEstar Open for OpenVMS Application Programming Interface*

For information on installing and using OMNI functions, refer to the following documents:

- *Compaq OMNI for OpenVMS Installation Guide*
- *Compaq OMNI for OpenVMS Application Programmer's Interface*
- *Compaq OMNI for OpenVMS Guide to Using Omni Directory Services*
- *Compaq OMNI for OpenVMS Guide to Using OmniView*

For information on Allen-Bradley PLC communications and addressing, refer to the following documents:

- *Allen-Bradley Data Highway/Data Highway Plus/DH-485 Communication Protocol and Command Set*
- *Allen-Bradley PLC-2 Family of Programmable Controllers Addressing Reference*
- *Allen-Bradley PLC-3 Family of Programmable Controllers Addressing Reference*
- *SLC 500 Family of Programmable Controllers Addressing Reference*
- *Allen-Bradley 1785 PLC-5 Programmable Controllers Addressing Reference*
- *Allen-Bradley Pyramid Integrator System Addressing Reference*

Conventions

This document uses the following conventions:

Boldface	Highlights user input within textual descriptions.
Return	Press the key labeled Return. Unless otherwise specified, press Return after entering a command or responding to a prompt.
Enter	Type the words or symbols described and press Return.

1 Overview

This chapter provides an overview of the BASEstar Open DAS for Allen-Bradley Data Highway. It also briefly describes Allen-Bradley PLC communications, and the supported functions for the DAS for Allen-Bradley Data Highway.

1.1 Description

The DAS for Allen-Bradley Data Highway allows you to access Allen-Bradley PLCs using BASEstar Open Device Services, BASEstar Open Data Services or OMNI Services.

Device Services is the device configuration and data collection component of BASEstar Open for OpenVMS, allowing the user to:

- Configure device and variable definitions
- Read and write variable values
- Configure polled data collection

Data Services is the data management component of BASEstar Open for OpenVMS that provides device independent access to shop floor data allowing the user to:

- Configure data points
- Read and write data point values
- Generate events based on data changes

OMNI Services provides the Applications Programming Interface (API) used by the BASEstar Open Device/Data Services to communicate with shop floor devices. OMNI uses the Manufacturing Messaging Specification (MMS) (ISO-9506) model to communicate with shop floor devices. OMNI also has the capability to support non-MMS devices such as Allen-Bradley PLCs through protocol specific Device Access Software, such as the DAS for Allen-Bradley Data Highway.

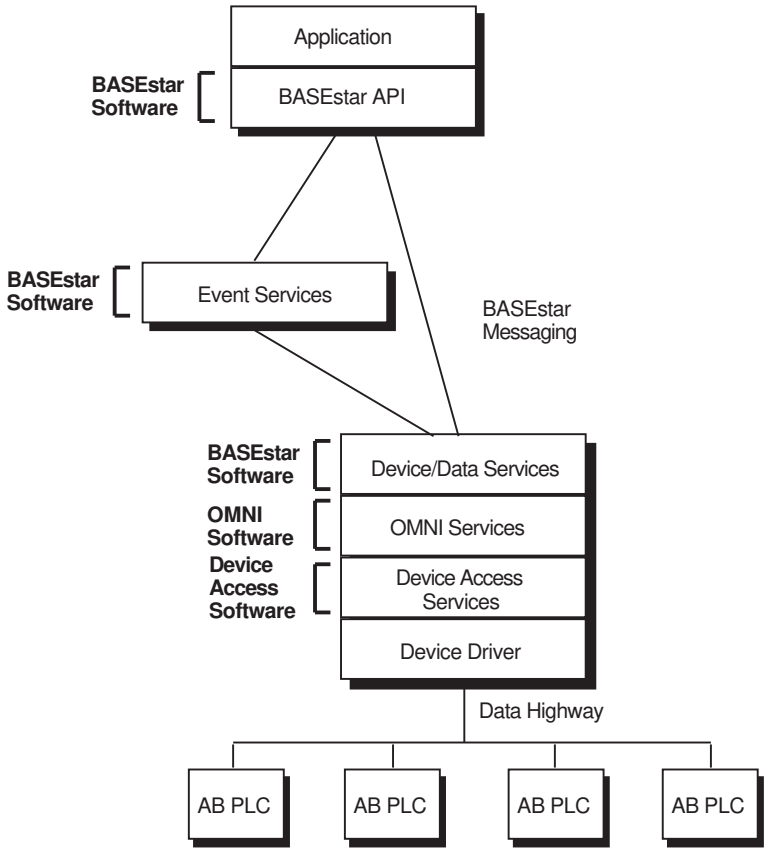
Using the DAS for Allen-Bradley Data Highway, applications can perform a variety of device access functions for the Allen-Bradley PLCs.

1.2 Device Communications

The DAS for Allen-Bradley Data Highway consists of software that provides device-specific communications for users of the OMNI API. BASEstar Open Device/Data Services use the OMNI API to provide device communications to BASEstar Open users.

The DAS software communicates the requests and data directly with an OpenVMS device driver to send data to and receive data from plant-floor devices. Figure 1–1 shows how the DAS for Allen-Bradley Data Highway facilitates communications between BASEstar Open Device/Data Services and the device.

Figure 1–1 DAS Communications



1.3 Supported Functions and Devices

You can perform only device access funtions that are supported by the DAS.

The DAS for Allen-Bradley Data Highway supports the following OMNI Services functions for Allen-Bradley PLCs:

- Initiate a connection to a device (omni_connect)
- Conclude a connection with a device (omni_conclude)
- Abort a connection with a device (omni_abort)
- Get a value from a device's memory (omni_get_value)
- Put a value to a device's memory (omni_put_value)
- Receive a write indication from a device (omni_get_indications)
- Receive an abort from a device (omni_get_indications)

The DAS for Allen-Bradley Data Highway supports the following Allen-Bradley PLCs:

- Allen-Bradley PLC-2, PLC-2/5, PLC-2/15, PLC-2/16, PLC-2/17, PLC-2/20, PLC-2/30
- Allen-Bradley PLC-3 Family

- Allen-Bradley PLC-5/10, PLC-5/11, PLC-5/12, PLC-5/15, PLC-5/20, PLC-5/25, PLC-5/30, PLC-5/40, PLC-5/40L, PLC-5/60, PLC-5/60L, PLC-5/80, PLC-5/250
- Allen-Bradley SLC 500, SLC 5/01, SLC 5/02, SLC 5/03, SLC 5/04

For more information about the supported functions for these PLCs, refer to Chapter 3 of this document.

2
Installing and Configuring the DAS

This chapter provides the information you need to install the DAS for Allen-Bradley Data Highway and to configure your system.

2.1 Installation Requirements

Review the following hardware and software requirements to ensure that your system is prepared for the DAS for Allen-Bradley Data Highway installation.

Note

Back up the disks on your system before installing this software. This will provide a method to restore your system in the event of an installation problem. The procedure for backing up disks is described in the *OpenVMS System Management Utilities Reference Manual*.

2.1.1 Hardware

The hardware requirements for the DAS for Allen-Bradley Data Highway are the same as those for the BASEstar Open software. For specific hardware requirements, refer to the *BASEstar Open for OpenVMS Installation Guide* .

Additional hardware required for configuring the DAS for Allen-Bradley Data Highway includes:

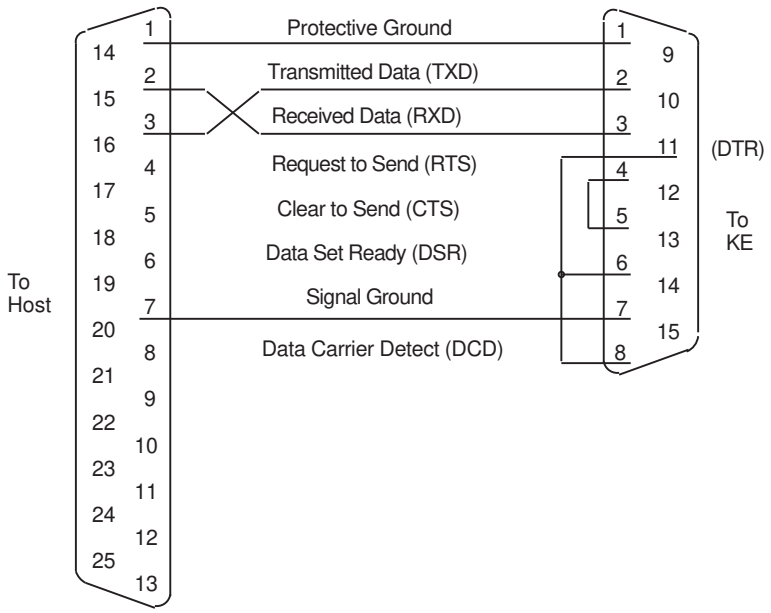
- Any supported Allen-Bradley PLC
- Allen-Bradley KE, KF, KF2 or KF3 communications card
- A cable from the OpenVMS system to the communications card

The cable used to connect to the communications card depends on the communication option being used. The following figures describe the pinouts for 15 pin (KE card), 25 pin (KF communications) and 9 pin (SLC DF1 communications) connectors.

The cable to the KE communications card must be terminated with a 15 pin male connector as shown in Figure 2–1.

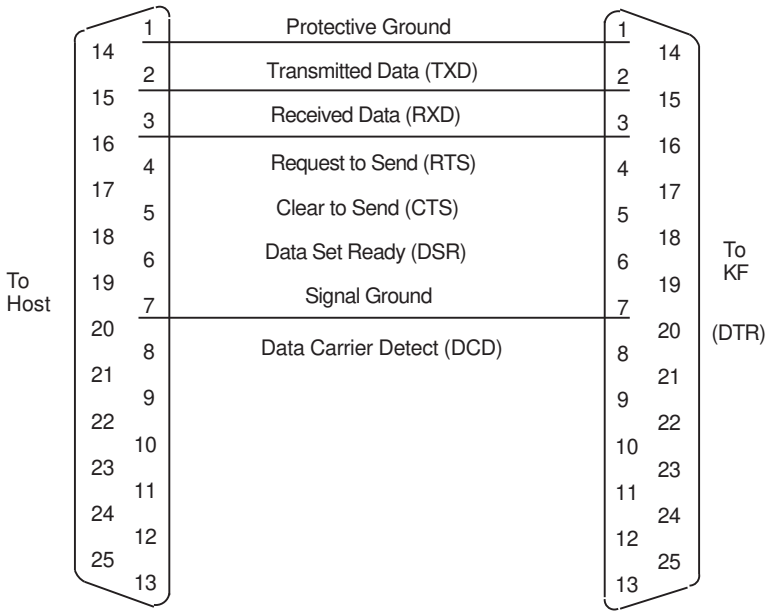
Installing and Configuring the DAS
2.1 Installation Requirements

Figure 2–1 Allen-Bradley 15 Pin Device Connector Layout



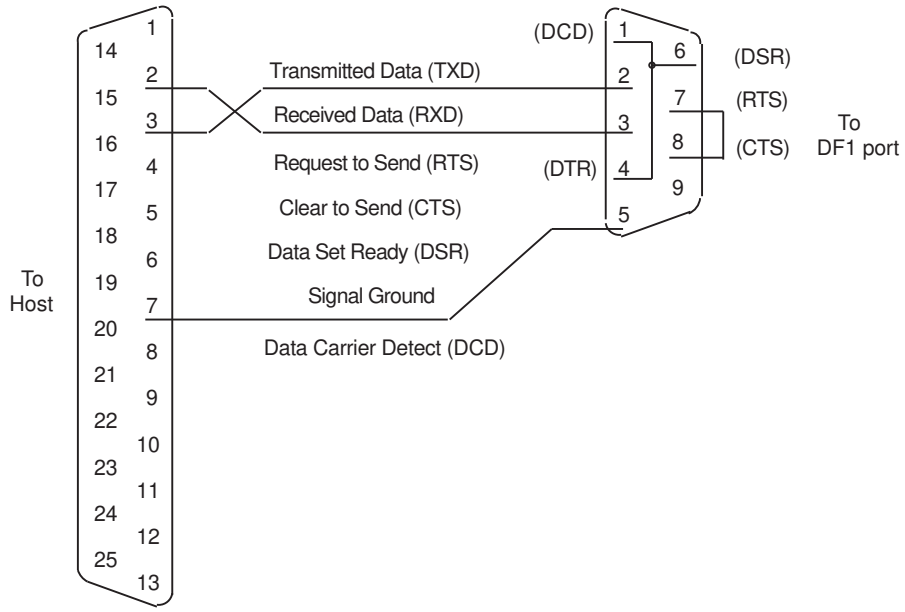
The cable to the KF communications card must be terminated with a 25 pin male RS232-C connector as shown in Figure 2–2.

Figure 2–2 Allen-Bradley 25 Pin Device Connector Layout



The cable to the SLC 5/03 or SLC 5/04 DF1 port must be terminated with a 9 pin male RS232 connector as shown in Figure 2–3.

Figure 2–3 Allen-Bradley 9 Pin Device Connector Layout



2.1.2 Software

The following software must be installed prior to installing DAS for Allen-Bradley Data Highway:

- OpenVMS/Alpha Version 6.1
- OpenVMS/VAX Version 6.1
- BASEstar Open for OpenVMS Version 3.1

For more information on installing BASEstar Open software, refer to the *BASEstar Open for OpenVMS Installation Guide*.

Note

Before using this product on a system, you must first register a License Product Authorization Key (License PAK) using the License Management Facility (LMF). For more information about the License Management Utility, refer to the *License Management Utility Manual* for OpenVMS.

2.1.3 Disk Space

Table 2–1 lists the disk space required to install the DAS for Allen-Bradley Data Highway. The space requirements are approximations; actual sizes may vary depending on your system environment and configuration.

Table 2–1 Disk Space Requirements

Usage	Approximate Space Requirements (Blocks)
Peak (during installation)	1250 (Alpha)
	3000 (VAX)
Net (after installation)	600 (Alpha)
	1500 (VAX)

2.2 Installing the DAS

When your system meets all hardware and software requirements, you can install the DAS for Allen-Bradley Data Highway. The installation takes from 1 to 5 minutes, depending on your system load and configuration.

Install the DAS for Allen-Bradley Data Highway by using the following steps:

1. Log in to a privileged system manager's account.
2. Set the default directory to SYS$UPDATE:

   ```
   $  SET DEFAULT SYS$UPDATE
   ```

3. Invoke VMSINSTAL:

   ```
   $  @SYS$UPDATE:VMSINSTAL ABDVAA031 ddcu:
   ```

   where:

   - **ABDVAA** argument is the kit name
   - **VV** in the name means this is an OpenVMS/VAX kit
   - **VA** in the name means this is an OpenVMS/Alpha kit
   - **031** portion of the name is the version number
   - **ddcu** argument represents the name of the device on which the installation media is mounted, where:
     - **dd** is the device code
     - **c** is the controller designation
     - **u** is the unit number

VMSINSTAL prompts you for information during the installation. Note that DECnet software does not need to be running to perform the installation procedure. The following is an example of the output from the installation on OpenVMS/Alpha:

```
        Beginning installation of ABDVAA V3.1 at 12:36
%VMSINSTAL-I-RESTORE, Restoring product save set A ...
%VMSINSTAL-I-RELMOVED, Product's release notes have been moved to SYS$HELP.
   Copyright 1994 Compaq Computer Corportation
```

```
   Confidential computer software.  Valid license from Compaq or authorized
   sublicensor required for possession, use or copying.  Consistent with FAR
   12.211 and 12.212, Commercial Computer Software, Computer Software
   Documentation, and Technical Data for Commercial Items are licensed to the
   U.S. Government under vendor's standard commercial license.

   BASEstar Open DAS for Allen-Bradley Data Highway installation procedure.

   Checking for a valid license...


        Product:      ABDAS-V
        Producer:     DEC
        Version:      1.0
        Release Date: 01-JUN-1994

* Does this product have an authorization key registered and loaded? y

   Now checking OpenVMS version...

   Now checking that OMNI is installed...

   Now checking OMNI version...

   Now checking OMNI logicals...

   Now checking the OMNI integrator registration utility...

* Do you want BASEstar Open support [YES]?

   Now checking disk space...

* Do you want to purge files replaced by this installation [YES]?
* Do you want to run the IVP after the installation [YES]?

   ---------------------------------NOTE-----------------------------------

   During installation the file ODS_AB_CONFIG.TEMPLATE is copied to the
   ODS:[SCRIPTS] directory.  Copy and edit this file to create site specific
   /PATH and /LINE directory definitions for devices at your installation.

   ---------------------------------NOTE-----------------------------------

   During installation the file BSTR_AB_CONFIG.TEMPLATE is copied to the
   BSTR$KIT:[ETC] directory.  Copy and edit this file to create site specific
   VMD definitions.

   Now modifying the ODS schema to add Allen-Bradley specific
   object classes and attributes...

Checking ODS Attribute Type Schema definitions ...

Checking ODS Object Class Schema definitions ...

Querying ODS Schema definitions ...

Valid Schema definitions found in the attribute schema database as:
OC ALIASED_NAME CN C L S O OU P_ADDR APC OWNER APT AEQ MAPV SY OBJ_ID BER PSAP S
SAP TSAP N_ADDR AS_LIST DESCR VMD DOM DEL VN APPSN APROF VERSION NLEVEL MAX_CALL
ED MAX_CALLING MAX_SEG CBB_LIST SS_LIST VENDOR MODEL REV INUM DN SHARE CONT_FILE
 CAP_FILE PN REUSE ARG MON DOM_LIST VAR ATYPE ADDR_T ADDR_V PATH LINE STATION LI
NE_NAME PORT ENQ_RETRIES NAK_RETRIES TIMEOUT SPEED PARITY MODEM TRACE KE_STATION
 KE_TIMEOUT

Valid Schema definitions found in the object class schema database as:
TOP ALIAS C L O OU AP AE DSA DEVICE TAE OAE AS VMD DOMAIN PI NVAR UNVAR DH_LINE
DH_PATH
```

```
Normal Completion.
%VMSINSTAL-I-MOVEFILES, Files will now be moved to their target directories...
   Now registering BASEstar Open DAS for Allen-Bradley Data Highway with OMNI...

   Copyright 1994 Compaq Computer Corportation

   Executing the Installation Verification Procedure.

   BASEstar Open DAS for Allen-Bradley Data Highway installation procedure
   has succeeded.

        Installation of ABDVAA V3.1 completed at 14:33
    Adding history entry in VMI$ROOT:[SYSUPD]VMSINSTAL.HISTORY
    Creating installation data file: VMI$ROOT:[SYSUPD]ABDVAA031.VMI_DATA

        VMSINSTAL procedure done at 14:34
```

2.2.1 Files Created During Installation

Table 2–2 lists the files created by the DAS for Allen-Bradley Data Highway installation procedure, and the directories in which those files are placed.

Table 2–2 Files Created During Installation

Directory	Filename
ODS:[SCRIPTS]	ODS_AB_CONFIG.TEMPLATE
BSTR$KIT:[ETC]	BSTR_AB_CONFIG.TEMPLATE

2.2.2 Installation Messages

You may see VMSINSTAL messages during the installation procedure. The following messages are specific to the DAS for Allen-Bradley Data Highway installation:

BADOMNI, The version of OMNI required is V3.1 or later.,

Explanation: Error. This version of the DAS is designed to work with OMNI V3.1 or later.

User Action: Install OMNI V3.1 or later and then reinstall the DAS.

BADVMS (Alpha), BASEstar Open DAS for Allen-Bradley Data Highway must be installed under OpenVMS V6.1.,

Explanation: Error. The DAS is being installed on a version of OpenVMS that is less than OpenVMS V6.1.

User Action: Upgrade OpenVMS to the required minimum version and then install the DAS.

BADVMS (VAX), BASEstar Open DAS for Allen-Bradley Data Highway must be installed under OpenVMS V6.1.,

Explanation: Error. The DAS is being installed on a version of OpenVMS that is less than OpenVMS V6.1.

User Action: Upgrade OpenVMS to the required minimum version and then install the DAS.

NETBLOCKS (VAX), BASEstar Open DAS for Allen-Bradley Data Highway requires 1500 blocks after installation.,

Explanation: Error. There isn't enough space on the system disk to install the DAS.

User Action: Delete unneeded files and reattempt the installation.

NETBLOCKS (ALPHA), BASEstar Open DAS for Allen-Bradley Data Highway requires 600 blocks after installation.,

Explanation: Error. There isn't enough space on the system disk to install the DAS.

User Action: Delete unneeded files and reattempt the installation.

NOETC, Unable to find BSTR$KIT:[ETC] directory.,

Explanation: Error. The directory to place the BASEstar Open configuration template for Allen-Bradley devices was not found.

User Action: Reinstall BASEstar Open to set up the BSTR$KIT:[ETC] directory.

NOLICENSE, No license found for this product - IVP will not be run.,

Explanation: Informational. A valid license was not found. The installation will continue, but the IVP will not be run.

User Action: Register and load a valid license for this product before attempting to use the DAS.

NOLOAD, License for this product not loaded - IVP will not be run.,

Explanation: Informational. The license for this product has not been loaded by the License Management Utility. The installation willl proceed, but the IVP will not be run.

User Action: Load the license using the License Management Utility before attempting to use the DAS.

NOOMNI, OMNI must be installed before BASEstar Open DAS for Allen-Bradley Data Highway.,

Explanation: Error. The installation procedure could not verify that OMNI was installed on the system. OMNI installation is a prerequisite to the installation of this DAS.

User Action: Install OMNI V3.1 and then reinstall the DAS.

NOINTREG, Unable to find the integrator registration utility.,

Explanation: Error. OMNI provides a registration utility for DASes to use to register their presence with OMNI. This registration utility could not be found.

User Action: Reinstall OMNI to provide the required registration utility.

NOODS, Unable to find the directory for OMNI Directory Services.,

Explanation: Error. The installation procedure creates classes and attributes in the OMNI Directory Services database and this database could not be found.

User Action: Reinstall OMNI to set up the OMNI Directory Services directory.

NOSCRIPT, Unable to find ODS:[SCRIPTS] directory.,

Explanation: Error. The directory to place the ODS configuration template for Allen-Bradley devices was not found.

User Action: Reinstall OMNI to set up the OMNI Directory Services directory.

ODSINI, OMNI logicals have not been defined, attempting to define logicals.,

Explanation: Information. The DAS installation requires that certain OMNI logicals be defined and these logical names were not found. The installation procedure attempts to define the logical names by executing SYS$STARTUP:OMNI_LOGICALS.COM.

User Action: None.

2.3 Postinstallation Tasks

Before using the DAS for Allen-Bradley Data Highway you must perform the following postinstallation tasks:

- Edit the configuration files
- Configure ports
- Set up plant-floor equipment

2.3.1 Editing the Configuration Files

Configuration files, ODS_AB_CONFIG.TEMPLATE and BSTR_AB_CONFIG.TEMPLATE, are supplied with the DAS for Allen-Bradley Data Highway. The ODS configuration file contains OMNI Directory Services (ODS) registration commands for Allen-Bradley specific classes and attributes. The BASEstar Open configuration file contains definitions for sample Allen-Bradley VMDs and protocol profile(s).

Copy the template files and edit them to include site-specific directory, VMD and protocol profile definitions. When you complete your edits, execute the files to create the ODS and BASEstar Open Device Services definitions.

The following sections provide examples of ODS and BASEstar Open definitions from the configuration files, and indicate which attributes can be modified. For more information about creating VMD and protocol profile definitions, refer to the *BASEstar Open for OpenVMS Command Language Interface*. For more information about registering ODS definitions, refer to the *Compaq OMNI for OpenVMS Guide to Using Omni Directory Services*.

2.3.1.1 Editing ODS Registrations

ODS registrations allow the DAS software to locate a particular PLC on an Allen-Bradley data highway network. Example 2–1 shows the configuration file commands that create the ODS registrations.

Example 2–1 ODS Registration

```
REGISTER DIRECTORY NAME "/LINE=DATA_HIGHWAY" ATTRIBUTES "OC=DH_LINE
/PORT=LTA201:/ENQ_RETRIES=10/NAK_RETRIES=3/KE_TIMEOUT=128/SPEED=9600
/PARITY=none/MODEM=0/TRACE=0/KE_STATION=12/TIMEOUT=3"
```

(continued on next page)

Example 2–1 (Cont.) ODS Registration

```
REGISTER DIRECTORY NAME "/LINE=DATA_HIGHWAY_PLUS" ATTRIBUTES "OC=DH_LINE
/PORT=LTA202:/ENQ_RETRIES=10/NAK_RETRIES=3/KE_TIMEOUT=128/SPEED=9600
/PARITY=none/MODEM=0/TRACE=0/KE_STATION=13/TIMEOUT=3"
REGISTER DIRECTORY NAME "/LINE=DH485" ATTRIBUTES "OC=DH_LINE
/PORT=LTA203:/ENQ_RETRIES=10/NAK_RETRIES=3/SPEED=9600
/PARITY=none/MODEM=0/TIMEOUT=3"

REGISTER DIRECTORY NAME "/PATH=PLC_2_15_PATH" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY)/STATION=20/TRACE=0"
REGISTER DIRECTORY NAME "/PATH=PLC_2_30_PATH" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY)/STATION=21/TRACE=0"
REGISTER DIRECTORY NAME "/PATH=PLC_3_PATH" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY)/STATION=22/TRACE=0"
REGISTER DIRECTORY NAME "/PATH=PLC_5_40_PATH" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY_PLUS)/STATION=20/TRACE=0"
REGISTER DIRECTORY NAME "/PATH=PLC_5_250_PATH" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY_PLUS)/STATION=21/TRACE=0"
REGISTER DIRECTORY NAME "/PATH=SLC_5_04" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DH485)/STATION=1"
```

You can modify the command for your site configuration. Table 2–3 lists ODS attributes, the class they belong to and valid values for the attributes. Attribute values are not checked until a connection is initiated with the VMD.

Table 2–3 REGISTER Attributes

Class	Attribute	Default	Value(s)
DH_LINE	PORT		TT01:, LTA201:, etc.[1]
	SPEED		110 - 19200
	PARITY		NONE, EVEN
	MODEM	NO	TRUE, ON, YES, 1 or FALSE, OFF, NO, 0
	TIMEOUT	3	2-6[2]
	KE_STATION[3]	0	0-377[4]
	KE_TIMEOUT	128	0-255[6]
	ENQ_RETRIES[7]	10	0-20
	NAK_RETRIES[8]	3	0-20
	TRACE[9]	NO	TRUE, ON, YES, 1 or FALSE, OFF, NO, 0

[1]LTAXXX: is used for LAT connections. TTXX: is used for direct connection.

[2]Seconds.

[3]If a KE Station address is specified, then the DAS will configure the KE/KF/KF-2 for the ENQ retries, NAK retries and KE timeout values specified. If a KE Station Address of 0 is specified, then the DAS does not configure the KE/KF/KF-2 and the KE timeout value is ignored. Do NOT configure the KE station address if connecting to a DH-485 network.

[4]Octal for PLC-2, PLC-3 or PLC-5 controllers.

[6]The KE timeout value is specified in cycles. Consult the KE/KF/KF-2 documentation to determine how long the time is in seconds.

[7]The ENQ retries is the number of times a request for a response is sent to the PLC before an error is returned to the user.

[8]The NAK retries is the number of times a retransmission is attempted when a recoverable error is detected in the message received by the PLC.

[9]Trace is used to allow troubleshooting of the PLC communications. See Section 2.4 for details.

(continued on next page)

Table 2–3 (Cont.) REGISTER Attributes

Class	Attribute	Default	Value(s)
DH_TELNET	IP_ADDRESS		xx.xx.xx.xx[11]
	PORT_NUMBER[12]		
	TIMEOUT	3	2-6[2]
	KE_STATION[3]	0	0-377[4]
	KE_TIMEOUT	128	0-255[6]
	ENQ_RETRIES[7]	10	0-20
	NAK_RETRIES[8]	3	0-20
	TRACE[9]	NO	TRUE, ON, YES, 1 or FALSE, OFF, NO, 0
DH_PATH	LINE_NAME		/LINE=xxx[10]
	STATION		0-377[4], 1-31[5]
	TRACE[9]	NO	TRUE, ON, YES, 1 or FALSE, OFF, NO, 0

[2]Seconds.

[3]If a KE Station address is specified, then the DAS will configure the KE/KF/KF-2 for the ENQ retries, NAK retries and KE timeout values specified. If a KE Station Address of 0 is specified, then the DAS does not configure the KE/KF/KF-2 and the KE timeout value is ignored. Do NOT configure the KE station address if connecting to a DH-485 network.

[4]Octal for PLC-2, PLC-3 or PLC-5 controllers.

[5]Decimal for SLC controllers. If a SLC controller is connected directly to a data highway plus network, the station is still specified in decimal, not octal.

[6]The KE timeout value is specified in cycles. Consult the KE/KF/KF-2 documentation to determine how long the time is in seconds.

[7]The ENQ retries is the number of times a request for a response is sent to the PLC before an error is returned to the user.

[8]The NAK retries is the number of times a retransmission is attempted when a recoverable error is detected in the message received by the PLC.

[9]Trace is used to allow troubleshooting of the PLC communications. See Section 2.4 for details.

[10]Specify a name registered earlier with ODS that specifies the line that the PLC with this station address will be using.

[11]The TCP/IP address must be in xx.xx.xx.xx notation where xx is a number from 0-255.

[12]Put a valid port number for a socket connection to the TELNET server.

2.3.1.2 Editing BASEstar Open Protocol Profile Definitions

The BASEstar Open protocol profile is used to associate a device with the correct DAS. Example 2–2 shows the configuration file commands that create protocol profile definitions.

Example 2–2 BASEstar Open Protocol Profile Configuration

```
CREATE PROTOCOL_PROFILE ab_plc2 \
      -APPLPROFID 51 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
CREATE PROTOCOL_PROFILE ab_plc3 \
      -APPLPROFID 52 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
```

(continued on next page)

Example 2–2 (Cont.) BASEstar Open Protocol Profile Configuration

```
CREATE PROTOCOL_PROFILE ab_plc5 \
      -APPLPROFID 53 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
CREATE PROTOCOL_PROFILE ab_slc \
      -APPLPROFID 54 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
```

The DAS for Allen-Bradley Data Highway uses a different Application Profile ID for each PLC family that it supports. When defining a VMD, be sure to specify the proper protocol profile for the family of PLC being defined.

The MAXPDUSIZE should be greater than or equal to the largest total packet sent by the PLC. If a packet is segmented into several smaller packets use the total packet size for this number.

SMA_MODE indicates that the DAS supports reading multiple variables with one read command. The DAS does not support writing multiple variables. Variables in a polling set are grouped and read in one function call by BASEstar Open if the SMA_MODE ENABLED flag is set.

2.3.1.3 Editing VMD Definitions

The VMD definition represents an Allen-Bradley PLC and references previously defined ODS and protocol profile definitions. Example 2–3 shows the configuration file commands that create VMD definitions.

Example 2–3 VMD Configuration

```
CREATE VMD plc_2_15 -MODEL "PLC_2_15" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_2_15_PATH":ab_plc2 \
                    -DESCRIPTION "PLC 2/15" \
                    -LOG
CREATE VMD plc_2_30 -MODEL "PLC_2_30" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_2_30_PATH":ab_plc2 \
                    -DESCRIPTION "PLC 2/30" \
                    -LOG
CREATE VMD plc_3 -MODEL "PLC_3" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_3_PATH":ab_plc3 \
                    -DESCRIPTION "PLC 3" \
                    -LOG
CREATE VMD plc_5_40 -MODEL "PLC_5_40" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_5_40_PATH":ab_plc5 \
                    -DESCRIPTION "PLC 5/40" \
                    -LOG
CREATE VMD plc_5_250 -MODEL "PLC_5_250" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_5_250_PATH":ab_plc5 \
                    -DESCRIPTION "PLC 5/250" \
                    -LOG
```

(continued on next page)

Example 2–3 (Cont.) VMD Configuration

```
CREATE VMD slc_5_04 -MODEL "SLC_5_04" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=SLC_5_04_PATH":ab_slc \
                    -DESCRIPTION "SLC 5/04" \
                    -LOG
```

An access point is created using the following syntax: `APPLICATION_SIMPLE_NAME:PROTOCOL_PROFILE_NAME`. The `APPLICATION_SIMPLE_NAME` is the definition registered in ODS. The `PROTOCOL_PROFILE_NAME` is defined using the BASEstar Open CLI as described in Section 2.3.1.2.

An access point tells the BASEstar Open software how to access the VMD. For more information on specifying client and DVM access points, refer to *BASEstar Open for OpenVMS Command Language Interface*.

Modify the command for your site configuration. Table 2–4 lists the modifiable device attributes.

Table 2–4 VMD CREATE

VMD	Parameter	Value(s)
PLC 2	MODEL	PLC_2, PLC_2_5, PLC_2_05, PLC_2_15, PLC_2_16, PLC_2_17, PLC_2_20, PLC_2_30
	VENDOR[1]	Allen-Bradley
	REVISION[1]	
	CLIENT_ACCESS_POINT[2]	
	DVM_ACCESS_POINT[2]	
	DESCRIPTION[1]	
PLC 3	MODEL	PLC_3
	VENDOR[1]	Allen-Bradley
	REVISION[1]	
	CLIENT_ACCESS_POINT[2]	
	DVM_ACCESS_POINT[2]	
	DESCRIPTION[1]	PLC-3
PLC 5	MODEL	PLC_5_10, PLC_5_11, PLC_5_12, PLC_5_15, PLC_5_16, PLC_5_20, PLC_5_25, PLC_5_26, PLC_5_30, PLC_5_36, PLC_5_40, PLC_5_40L, PLC_5_46, PLC_5_60, PLC_5_60L, PLC_5_66, PLC_5_80, PLC_5_86, PLC_5_250
	VENDOR[1]	Allen-Bradley
	REVISION[1]	
	CLIENT_ACCESS_POINT[2]	
	DVM_ACCESS_POINT[2]	
	DESCRIPTION[1]	

[1]The value of this field is informational only and is not checked by the DAS.

[2]See the *BASEstar Open for OpenVMS Command Language Interface* for information on specifying client and DVM access points.

(continued on next page)

Table 2–4 (Cont.) VMD CREATE

VMD	Parameter	Value(s)
SLC	MODEL	SLC_500, SLC_5_01, SLC_5_02, SLC_5_03, SLC_5_04, SLC_5_05
	VENDOR[1]	Allen-Bradley
	REVISION[1]	
	CLIENT_ACCESS_POINT[2]	
	DVM_ACCESS_POINT[2]	
	DESCRIPTION[1]	

[1]The value of this field is informational only and is not checked by the DAS.

[2]See the *BASEstar Open for OpenVMS Command Language Interface* for information on specifying client and DVM access points.

2.3.1.4 Configuration Files

Example 2–4 and Example 2–5 are samples of the ODS and BSTR configuration files for the DAS for Allen-Bradley Data Highway.

Example 2–4 ODS Configuration File

```
$!
$!
$!  Copyright 1994 Compaq Computer Corporation.
$!
$!  COMPAQ Registered in U.S. Patent and Trademark Office.
$!
$!  Confidential computer software.  Valid license from Compaq or authorized
$!  sulicensor required for possession, use or copying.  Consistent with
$!  FAR 12.211 and 12.212, Commercial Computer Software, Computer Software
$!  Documentation, and Technical Data for Commercial Items are licensed to
$!  the U.S. Government under vendor's standard commercial license.
$!
$!*********************************************************************!
$!                                                                     !
$!    This is a template for the Allen-Bradley-specific              !
$!    ODS configuration command file.  Rename and edit this command   !
$!    procedure template to reflect your site-specific               !
$!    configuration.  You must manually execute this command         !
$!    procedure to configure ODS to provide line configuration       !
$!    information for your devices.                                  !
$!                                                                     !
$!    Add comments to the site-specific changes here.                !
$!                                                                     !
$!*********************************************************************!
$!
$ODSCL := $ODS:[EXEC]ODSCL
$!
$!  Register the ODS definitions.
$!
$ODSCL
REGISTER DIRECTORY NAME "/LINE=DATA_HIGHWAY" ATTRIBUTES "OC=DH_LINE
/PORT=LTA201:/ENQ_RETRIES=10/NAK_RETRIES=3/KE_TIMEOUT=128/SPEED=9600
/PARITY=none/MODEM=0/KE_STATION=12/TIMEOUT=3"
```

(continued on next page)

Example 2–4 (Cont.) ODS Configuration File

```
REGISTER DIRECTORY NAME "/LINE=DATA_HIGHWAY_PLUS" ATTRIBUTES "OC=DH_LINE
/PORT=LTA202:/ENQ_RETRIES=3/NAK_RETRIES=3/KE_TIMEOUT=128/SPEED=9600
/PARITY=none/MODEM=0/KE_STATION=10/TIMEOUT=3"
REGISTER DIRECTORY NAME "/LINE=DH485" ATTRIBUTES "OC=DH_LINE
/PORT=LTA203:/ENQ_RETRIES=10/NAK_RETRIES=3/SPEED=9600
/PARITY=none/MODEM=0/TIMEOUT=3"

REGISTER DIRECTORY NAME "/PATH=PLC_2_15" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY)/STATION=20"
REGISTER DIRECTORY NAME "/PATH=PLC_2_30" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY)/STATION=21"
REGISTER DIRECTORY NAME "/PATH=PLC_3" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY)/STATION=22"
REGISTER DIRECTORY NAME "/PATH=PLC_5_40" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY_PLUS)/STATION=20"
REGISTER DIRECTORY NAME "/PATH=PLC_5_250" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DATA_HIGHWAY_PLUS)/STATION=21"
REGISTER DIRECTORY NAME "/PATH=SLC_5_04" ATTRIBUTES "OC=DH_PATH
/LINE_NAME=(/LINE=DH485)/STATION=1"
EXIT
```

To execute the ODS command file, enter the following command at the user prompt:

```
@ODS:[SCRIPTS]ODS_AB_CONFIG.TEMPLATE
```

Example 2–5 BSTR Configuration File

```
!
!  Copyright 1994 Compaq Computer Corporation
!
!  COMPAQ Registered in U.S. Patent and Trademark Office.
!
!  Confidential computer software.  Valid license from Compaq or authorized
!  sulicensor required for possession, use or copying.  Consistent with
!  FAR 12.211 and 12.212, Commercial Computer Software, Computer Software
!  Documentation, and Technical Data for Commercial Items are licensed to
!  the U.S. Government under vendor's standard commercial license.
!
!  Create the protocol profile definitions for the PLC-2,
!  PLC-3, PLC-5 and SLC family of PLCs.
!  Protocol profile IDs uniquely identify protocol support to OMNI.
!
CREATE PROTOCOL_PROFILE ab_plc2 \
      -APPLPROFID 51 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
CREATE PROTOCOL_PROFILE ab_plc3 \
      -APPLPROFID 52 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
CREATE PROTOCOL_PROFILE ab_plc5 \
      -APPLPROFID 53 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
```

(continued on next page)

Example 2–5 (Cont.) BSTR Configuration File

```
CREATE PROTOCOL_PROFILE ab_slc \
      -APPLPROFID 54 \
      -MAXPDUSIZE 1024 \
      -SMA_MODE ENABLED \
      -LOG
!
!  Create VMD definitions.
!  Use the ODS entries and protocol profile names created earlier.
!
CREATE VMD plc_2_15 -MODEL "PLC_2_15" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_2_15_PATH":ab_plc2 \
                    -DESCRIPTION "PLC 2/15" \
                    -LOG
CREATE VMD plc_2_30 -MODEL "PLC_2_30" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_2_30_PATH":ab_plc2 \
                    -DESCRIPTION "PLC 2/30" \
                    -LOG
CREATE VMD plc_3 -MODEL "PLC_3" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_3_PATH":ab_plc3 \
                    -DESCRIPTION "PLC 3" \
                    -LOG
CREATE VMD plc_5_40 -MODEL "PLC_5_40" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_5_40_PATH":ab_plc5 \
                    -DESCRIPTION "PLC 5/40" \
                    -LOG
CREATE VMD plc_5_250 -MODEL "PLC_5_250" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_5_250_PATH":ab_plc5 \
                    -DESCRIPTION "PLC 5/250" \
                    -LOG
CREATE VMD slc_5_04 -MODEL "SLC_5_04" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=SLC_5_04_PATH":ab_slc \
                    -DESCRIPTION "SLC 5/04" \
                    -LOG
!
!  Create actors/activities for the VMDs
!
CREATE NODE ab_node -PHYSICAL "your_node" -LOG
CREATE DOMAIN /plc_2_15
CREATE DOMAIN /plc_2_30
CREATE DOMAIN /plc_3
CREATE DOMAIN /plc_5_40
CREATE DOMAIN /plc_5_250
CREATE DOMAIN /slc_5_04
CREATE ACTOR /AB -LOG
SET CONTEXT /AB
CREATE PROGRAM ab_server -PROGRAM_KIND DATADEV -LOG
CREATE ACTIVITY plc_2_15 -PROGRAM ab_server \
                    -NODES (ab_node) \
                    -DOMAINS (/plc_2_15) \
                    -VMDS (plc_2_15) \
                    -LOG
```

(continued on next page)

Example 2–5 (Cont.) BSTR Configuration File

```
CREATE ACTIVITY plc_2_30 -PROGRAM ab_server \
                    -NODES (ab_node) \
                    -DOMAINS (/plc_2_30) \
                    -VMDS (plc_2_30) \
                    -LOG
CREATE ACTIVITY plc_3 -PROGRAM ab_server \
                    -NODES (ab_node) \
                    -DOMAINS (/plc_3) \
                    -VMDS (plc_3) \
                    -LOG
CREATE ACTIVITY plc_5_40 -PROGRAM ab_server \
                    -NODES (ab_node) \
                    -DOMAINS (/plc_5_40) \
                    -VMDS (plc_5_40) \
                    -LOG
CREATE ACTIVITY plc_5_250 -PROGRAM ab_server \
                    -NODES (ab_node) \
                    -DOMAINS (/plc_5_250) \
                    -VMDS (plc_5_250) \
                    -LOG
CREATE ACTIVITY slc_5_04 -PROGRAM ab_server \
                    -NODES (ab_node) \
                    -DOMAINS (/slc_5_04) \
                    -VMDS (slc_5_04) \
                    -LOG
```

To execute the BASEstar Open command file, enter the following command at the user prompt:

```
$CLI_BSTR -F BSTR_AB_CONFIG.TEMPLATE
```

For more information about maintaining protocol profile and device definitions, refer to the *BASEstar Open for OpenVMS Command Language Interface*. For more information about maintaining ODS definitions, refer to the *Compaq OMNI for OpenVMS Guide to Using Omni Directory Services*.

2.3.2 Configuring Ports

You can connect a PLC to a Local Area Transport (LAT) port. To use a LAT port with the DAS, you must define the LAT port on the OpenVMS system using the LAT control program.

For example, to define **port_2** on the LAT node **srvr4** as **LTA992**, enter the following:

```
$ MCR LATCP
LCP> CREATE PORT LTA992
LCP> SET PORT LTA992 /NODE=srvr4/PORT=port_2/NOQUEUE
```

Use the LAT port name defined above when defining the port to ODS.

You must also set the following SYSGEN parameters to access Allen-Bradley PLCs to reduce the potential for data overruns. Use the SYSGEN utility to display the current values and AUTOGEN to reset the values accordingly. For information on using SYSGEN and AUTOGEN, refer to the *OpenVMS System Management Utilities Reference Manual* for details.

Parameter	Value
TTY_ALTALARM	0
TTY_ALTYPAHD	1024

To enable use of the alternate typeahead buffer, type the following command from DCL:

```
$ SET TERM LTA992:/ALTYPEAHD/PERMANENT
```

For the DAS for Allen-Bradley Data Highway to send and receive data correctly, you must set and define LAT port characteristics to match the line parameters defined in ODS for the corresponding port. The following example shows the characteristics set at the LAT prompt:

```
Local> DEF PORT n SPEED 9600
Local> DEF PORT n AUTOBAUD DISABLED
Local> DEF PORT n AUTOCONNECT DISABLED
Local> DEF PORT n FLOW DISABLED
Local> DEF PORT n PREFERRED NONE
Local> DEF PORT n ACCESS REMOTE
Local> DEF PORT n CHAR 8
Local> DEF PORT n PARITY EVEN
```

2.3.3 Configuring TELNET Access

You can connect a PLC to a terminal server using the TELNET protocol. Using the TELNET protocol allows the use of non-proprietary protocols to access terminal servers to perform device communications.

The terminal server being used must support the BINARY transfer option (RFC 856). The DAS negotiates binary transfer as part of establishing a connection with the port. If the terminal server will not allow binary transfer, the negotiation fails and a connection is not established.

NOTE: Using TELNET communications is slower than either LAT or a directly connected device. Ensure that your throughput requirements will be satisfied before deciding to implement TELNET communications.

For the DAS for Allen-Bradley Data Highway to send and receive data correctly using TELNET, you must set up a port on the terminal server to receive a connect from the DAS. The following example shows setting up a port using a DECserver 700.

```
Local> DEF TELNET LISTENER y PORT n
Local> DEF TELNET LISTENER y CONNECTIONS ENABLED
```

The following example shows setting up the server for TELNET access. A gateway needs to be configured only if the terminal server is on a different subnet than the host. In general, the terminal server should be on the same subnet as the host in order to reduce communications delays.

```
Local> DEF INTERNET ADDRESS xx.xx.xx.xx
Local> DEF INTERNET MASK xx.xx.xx.xx
Local> DEF INTERNET GATEWAY xx.xx.xx.xx
```

2.3.4 Setting Up Plant-Floor Equipment

To set up your plant-floor equipment, refer to the Allen-Bradley documentation for your specific PLC.

Table 2–5 provides switch settings to use in setting up your KE/KF/KF-2/KF-3 communications card.

Table 2–5 KE/KF/KF-2/KF-3 Switch Settings

Parameter	Supported	Values
Half Duplex	No	
Full Duplex	Yes	
Embedded Response	No	
BCC Error Check	Yes	
CRC Error Check	No	
RS-232C Handshaking	Yes[1]	
Station Number	Yes	1-377 (octal)[2], 1-31 (decimal)[3]
Computer Comm Rate	Yes	110 - 19200[4]

[1]If using the handshaking signals (RTS, CTS, DSR, DTR, DCD) be sure that the computer is also using those signals and that the cable has been wired properly.

[2]PLC-2, PLC-3, & PLC-5 controllers

[3]SLC controllers

[4]The 1771-KE,KF Series A Revision A-G support a maximum of 9600 baud. The 1771-KE,KF Series A Revision H and later, the 1770-KF2 Series B Revision F and later and the 1785-KE support a maximum of 19200 baud.

2.4 Tracing device communications

The DAS for Allen-Bradley Data Highway has built into it the capability to trace PLC communications and to send this output to either a file or to a user terminal. The purpose of this tracing is to allow the troubleshooting of PLC communications.

Tracing can be enabled for a device and/or a line. If tracing is enabled for a PLC, then only communications from/to that PLC are traced. If tracing is enabled for a line, then all communications from/to that line are traced. If both PLC and line tracing is enabled, then the trace will contain the output for both the PLC and the line, so the trace output will be duplicated.

To enable tracing do the following:

- Set the TRACE attribute in the ODS definition for the path or line being traced.
- Set the DAS_TRACE_OUTPUT logical name to the name of the file that you want to contain the trace output. The logical name must be defined in a scope that will be seen by the server the DAS is a part of.

```
DEFINE DAS_TRACE_OUTPUT MY_DISK:[MY_DIRECTORY]TRACE_OUTPUT.LOG
```

- Reinitiate communications to the PLC/line. The ODS definition is read when a connection to the device is initiated (device tracing) or when a connection to the line is initiated (line tracing), so communications must be reinitiated in order for tracing to occur.

The trace output is flushed to the file after every 10 lines that are traced, so trace output may lag actual device I/O.

All trace output is in hexadecimal and shows both input to and output from the PLC.

To turn tracing off, reverse the steps used to turn tracing on.

2.5 Failures During Product Use

If an error occurs while this product is in use and you believe the error is caused by a problem with the product, take one of the following actions:

- If you have a Software Product Services Support Agreement, contact your Customer Support Center (CSC) by telephone or by using the electronic means provided with your support agreement (such as DSNlink). The CSC provides telephone support for high-level advisory and remedial assistance. When you initially contact the CSC, indicate the following:
  - The name and version number of the operating system you are using
  - The version number of the product you are using
  - The version number of BASEstar Open you are using
  - The hardware system you are using (such as a model number)
  - The Allen-Bradley PLCs you are communicating with
  - A brief description of the problem (one sentence if possible)
  - How critical the problem is
- If you have a Self-Maintenance Software Agreement, you can submit a Software Performance Report (SPR).
- If you do not have any type of software services support agreement and you purchased this product within the past year, you can submit an SPR if you think the problem is caused by a software error.

When you submit an SPR, take the following steps:

1. Describe as accurately as possible the circumstances and state of the system when the problem occurred. Include the description and version number of the product being used. Demonstrate the problem with specific examples.
2. Reduce the problem to as small a size as possible.
3. Remember to include listings of any command files, INCLUDE files, or relevant data files, and so forth.
4. Report only one problem per SPR. This will facilitate a faster response.
5. Mail the SPR package to Compaq.

3
Using the DAS

This chapter provides information about the supported functions for Allen-Bradley PLCs, and how to access these functions.

3.1 Accessing Allen-Bradley Functions

DAS for Allen-Bradley Data Highway functions are accessed through the following interfaces:

- BASEstar Open Command Language Interface (CLI)
- BASEstar Open Application Programming Interface (API)
- OMNI Application Programming Interface (API)

Command Language Interface (CLI) To access the BASEstar Open CLI, enter the following command at the DCL prompt ($):

```
$ CLI_BSTR
```

For information about BASEstar Open commands, refer to the *BASEstar Open for OpenVMS Command Language Interface*.

Application Programming Interface (API) For information on accessing BASEstar Open Callable Services, refer to the *BASEstar Open for OpenVMS Application Programming Interface*.

For information on accessing OMNI Callable Services, refer to the *Compaq OMNI for OpenVMS Application Programmer's Interface*.

3.2 Supported Functions

A variety of functions are supported by the DAS for Allen-Bradley Data Highway. These functions may be supported as either client or server or both. If a function is supported as a server, then the application will receive an *indication* when the VMD is requesting the service.

The DAS for Allen-Bradley Data Highway supports the following client functions for all supported devices:

- Connect
- Read Variable
- Write Variable
- Abort
- Conclude

The DAS for Allen-Bradley Data Highway supports the following server functions (indications) for all supported devices:

- Write indication
- Abort

Note

Information Report is still supported for backward compatability but support will be dropped in a future release of the DAS. Support for SLC controllers does not include information report as an option. Configurations should be modified to use write indication support as soon as possible.

3.2.1 Connect, Conclude and Abort Functions

Table 3–1 lists the connect, conclude and abort functions available through the OMNI (API) interface.

Table 3–1 Connect, Conclude and Abort Interface Access

Interface	Function Name
API	omni_connect
	omni_connect_a
	omni_conclude
	omni_conclude_a
	omni_abort
	omni_abort_a

3.2.2 Read and Write Variable Functions

Use the read variable and write variable functions to read data from and write data to a specific address in PLC memory.

Table 3–2 lists the variable types supported by this DAS.

Table 3–2 Read and Write Variable Types

Variable Type	Address Type	Supported
UNNAMED_VARIABLE	NUMERIC	No
	SYMBOLIC	No
	UNCONSTRAINED	Yes
NAMED_VARIABLE		Yes[1]

[1]Named variables are not supported by the DAS, but they are used by BASEstar Open when accessing variables using the Device Manager Server. The BASEstar Open Device Manager Server converts named variables with an address type of UNCONSTRAINED to unnamed variables.

An UNNAMED_VARIABLE type describes a specific physical address on a VMD. This address has a predefined type based on the address. Variables of this type are an inherent part of the VMD architecture and are uniquely identified by their physical address.

An unnamed variable can be one of the following types:

- NUMERIC

  A numeric address must be an MMS unsigned integer.

- SYMBOLIC

  A symbolic address must be an MMS visible string.

- UNCONSTRAINED

  An unconstrained address can be any MMS octet (byte) string.

A NAMED_VARIABLE type describes a variable on a PLC. Variables of this type are uniquely identified by their name.

Table 3–3 lists the read and write variable functions available through the BASEstar Open (CLI) and OMNI (API) interfaces.

Table 3–3 Read and Write Variable Interface Access

Interface	Function Name
CLI	GET VALUE UNNAMED_VARIABLE
	PUT VALUE UNNAMED_VARIABLE
API	omni_get_value
	omni_get_value_a
	omni_put_value
	omni_put_value_a

Example 3–1 shows examples of creating unnamed variables using the CREATE UNNAMED_ VARIABLE command.

Example 3–1 UNNAMED_VARIABLE CREATE Command

```
BSTR> CREATE UNNAMED_VARIABLE plc_2_15.p1 \
      -DATATYPE INTEGER_16 \
      -ADDRESS "200" \
      -ADDRTYPE UNCONSTRAINED
BSTR> CREATE UNNAMED_VARIABLE plc_3.p1 \
      -DATATYPE INTEGER_16 \
      -ADDRESS "$N5:20" \
      -ADDRTYPE UNCONSTRAINED
BSTR> CREATE UNNAMED_VARIABLE plc_5_40.p1 \
      -DATATYPE BIT \
      -ADDRESS "$N7:30/1" \
      -ADDRTYPE UNCONSTRAINED
BSTR> CREATE ARRAY_DATATYPE WORD_ARRAY_10 \
      -DATATYPE INTEGER_16 \
      -SIZE 10
BSTR> CREATE UNNAMED_VARIABLE plc_5_40.p2 \
      -DATATYPE WORD_ARRAY_10 \
      -ADDRESS "$N7:30" \
      -ADDRTYPE UNCONSTRAINED
BSTR> CREATE STRUCTURE_DATATYPE PLC_5_COUNTER \
      -MEMBERS (CONTROL:INTEGER_16 PRE:INTEGER_16 ACC:INTEGER_16)
BSTR> CREATE UNNAMED_VARIABLE plc_5_40.p3 \
      -DATATYPE PLC_5_COUNTER \
      -ADDRESS "$C5:4" \
      -ADDRTYPE UNCONSTRAINED
BSTR> CREATE UNNAMED_VARIABLE slc_5_04.p1 \
      -DATATYPE INTEGER_16 \
      -ADDRESS "$N7:0" \
      -ADDRTYPE UNCONSTRAINED
```

Example 3–2 shows the information displayed when you enter the GET VALUE UNNAMED_VARIABLE command.

Example 3–2 UNNAMED_VARIABLE GET VALUE Output

```
BSTR> GET VALUE UNNAMED_VARIABLE plc_5_40.p1

20
BSTR> GET VALUE UNNAMED_VARIABLE plc_5_40.p2

array[10] of INTEGER_16
  [ 0]: 0
  [ 1]: 0
  [ 2]: 0
  [ 3]: 0
  [ 4]: 0
  [ 5]: 0
  [ 6]: 0
  [ 7]: 0
  [ 8]: 0
  [ 9]: 0
```

(continued on next page)

Example 3–2 (Cont.) UNNAMED_VARIABLE GET VALUE Output

```
BSTR> GET VALUE UNNAMED_VARIABLE plc_5_40.p3

structure PLC_5_COUNTER
{
  CONTROL: 0
  PRE: 0
  ACC: 0
}
```

Example 3–3 shows the information displayed when you enter the PUT VALUE UNNAMED_VARIABLE command.

Example 3–3 UNNAMED_VARIABLE PUT VALUE Output

```
BSTR> PUT VALUE UNNAMED_VARIABLE plc_5_40.p1

(Bit) : 1
BSTR> PUT VALUE UNNAMED_VARIABLE plc_5_40.p2

array[10] of INTEGER_16
  [ 0]: (Integer 16) : 1
  [ 1]: (Integer 16) : 2
  [ 2]: (Integer 16) : 3
  [ 3]: (Integer 16) : 4
  [ 4]: (Integer 16) : 5
  [ 5]: (Integer 16) : 6
  [ 6]: (Integer 16) : 7
  [ 7]: (Integer 16) : 8
  [ 8]: (Integer 16) : 9
  [ 9]: (Integer 16) : 10
```

3.2.3 Read and Write Data Point Functions

Use the read data point and write data point functions to read data from and write data to a device in a device independent way.

Table 3–4 lists the read and write data point functions available through the BASEstar Open (CLI) interface and BASEstar Open (API) interface.

Table 3–4 Read and Write Data Point Interface Access

Interface	Function Name
CLI	GET VALUE DATA_POINT
	PUT VALUE DATA_POINT
API	bstr_data_point_get_value
	bstr_data_point_put_value

Example 3–4 shows examples of creating a device data point using the CREATE DEVICE_DATA_POINT command.

Example 3–4 DEVICE_DATA_POINT CREATE Command

```
BSTR> CREATE DEVICE_DATA_POINT /plc_2_15/p1 \
                    -DATATYPE UNSIGNED_16 \
                    -UPDATE_POLICY SOLICITED \
                    -ACCESS_POLICY ALWAYS \
                    -DEVICE_ACCESS RDWR \
                    -VMD_NAME plc_2_15 \
                    -VAR_NAME p1 \
                    -VAR_CLASS UNNAMED
BSTR> CREATE DEVICE_DATA_POINT /plc_3/p1 \
                    -DATATYPE INTEGER_16 \
                    -UPDATE_POLICY SOLICITED \
                    -ACCESS_POLICY ALWAYS \
                    -DEVICE_ACCESS RDWR \
                    -VMD_NAME plc_3 \
                    -VAR_NAME p1 \
                    -VAR_CLASS UNNAMED
BSTR> CREATE DEVICE_DATA_POINT /plc_5_40/p1 \
                    -DATATYPE BIT \
                    -UPDATE_POLICY SOLICITED \
                    -ACCESS_POLICY ALWAYS \
                    -DEVICE_ACCESS RDWR \
                    -VMD_NAME plc_5_40 \
                    -VAR_NAME p1 \
                    -VAR_CLASS UNNAMED
BSTR> CREATE DEVICE_DATA_POINT /plc_5_40/p2 \
                    -DATATYPE WORD_ARRAY_10 \
                    -UPDATE_POLICY SOLICITED \
                    -ACCESS_POLICY ALWAYS \
                    -DEVICE_ACCESS RDWR \
                    -VMD_NAME plc_5_40 \
                    -VAR_NAME p2 \
                    -VAR_CLASS UNNAMED
BSTR> CREATE DEVICE_DATA_POINT /plc_5_40/p3 \
                    -DATATYPE PLC_5_COUNTER \
                    -UPDATE_POLICY SOLICITED \
                    -ACCESS_POLICY ALWAYS \
                    -DEVICE_ACCESS RDWR \
                    -VMD_NAME plc_5_40 \
                    -VAR_NAME p3 \
                    -VAR_CLASS UNNAMED
BSTR> CREATE DEVICE_DATA_POINT /slc_5_04/p1 \
                    -DATATYPE INTEGER_16 \
                    -UPDATE_POLICY SOLICITED \
                    -ACCESS_POLICY ALWAYS \
                    -DEVICE_ACCESS RDWR \
                    -VMD_NAME slc_5_04 \
                    -VAR_NAME p1 \
                    -VAR_CLASS UNNAMED
```

Example 3–5 shows the information displayed when you enter the GET VALUE DATA_POINT command.

Example 3–5 DATA_POINT GET VALUE Output

```
BSTR> GET VALUE DATA_POINT /plc_5_40/p1

20
BSTR> GET VALUE DATA_POINT /plc_5_40/p2

array[10] of INTEGER_16
  [ 0]: 0
  [ 1]: 0
  [ 2]: 0
  [ 3]: 0
  [ 4]: 0
  [ 5]: 0
  [ 6]: 0
  [ 7]: 0
  [ 8]: 0
  [ 9]: 0
BSTR> GET VALUE DATA_POINT /plc_5_40/p3

structure PLC_5_COUNTER
{
  CONTROL: 0
  PRE: 0
  ACC: 0
}
```

Example 3–6 shows the information displayed when you enter the PUT VALUE DATA_POINT command.

Example 3–6 DATA_POINT PUT VALUE Output

```
BSTR> PUT VALUE DATA_POINT /plc_5_40/p1

(Bit) : 1
BSTR> PUT VALUE DATA_POINT /plc_5_40/p2

array[10] of INTEGER_16
  [ 0]: (Integer 16) : 1
  [ 1]: (Integer 16) : 2
  [ 2]: (Integer 16) : 3
  [ 3]: (Integer 16) : 4
  [ 4]: (Integer 16) : 5
  [ 5]: (Integer 16) : 6
  [ 6]: (Integer 16) : 7
  [ 7]: (Integer 16) : 8
  [ 8]: (Integer 16) : 9
  [ 9]: (Integer 16) : 10
```

3.2.4 Receiving Indications

An indication is data that is sent to the DAS from the PLC without the DAS specifically requesting the data. The DAS supports write and abort indications.

Note

Information reports are also supported for PLC-2, PLC-3, and PLC-5 for backward compatability purposes only. Configurations should be migrated to use the write indication support as soon as possible.

Table 3–5 lists the indication functions available through the OMNI (API) interface.

Table 3–5 Indication Interface Access

Interface	Function Name
API	omni_get_indications
	omni_get_indications_a

3.2.4.1 Write Indications

The write indication is used when the PLC is sending data to the host. In order to use the write indication function, a variable must be created on the host's local VMD that matches the address and size of the data being sent by the PLC.

Example 3–7 shows an example of how to configure VMDs, actors and activities to receive write indications from a PLC-5 that has been configured to send PLC-2 type messages to the host. A PLC-2 VMD is configured in BASEstar Open to receive the write indications. Note that the PLC-2 has been configured using the PLC-5 protocol profile since it will be receiving write indications from a PLC-5.

Note

The path on the PLC-2 is not used since the VMD is only used to receive the write indications and is not used in establishing communications to the PLC.

Example 3–7 Write Indication Configuration

```
CREATE PROTOCOL_PROFILE ab_plc5 \
                    -APPLPROFID 53 \
                    -MAXPDUSIZE 1024 \
                    -SMA_MODE ENABLED \
                    -LOG
CREATE VMD plc_2_30 -MODEL "PLC_2_30" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_2_30_PATH":ab_plc5 \
                    -DESCRIPTION "PLC 2/30" \
                    -LOG
```

(continued on next page)

Example 3–7 (Cont.) Write Indication Configuration

```
CREATE VMD plc_5_40 -MODEL "PLC_5_40" \
                    -VENDOR "AB" \
                    -DVM_ACCESS_POINT "/PATH=PLC_5_40_PATH":ab_plc5 \
                    -DESCRIPTION "PLC 5/40" \
                    -LOG
!
!  Create actors/activities for the VMDs
!
CREATE NODE ab_node -PHYSICAL "your_node" -LOG
CREATE DOMAIN /ab_data
CREATE ACTOR /AB_IND -LOG
SET CONTEXT /AB_IND
CREATE PROGRAM ab_data_srv -PROGRAM_KIND DATADEV \
                    -PARAM AB_PARAM:AB.PAR \
                    -LOG
CREATE_PROGRAM ab_event_srv -PROGRAM_KIND EVENT \
                    -LOG
CREATE ACTIVITY ab_data_act -PROGRAM ab_data_srv \
                    -NODES (ab_node) \
                    -DOMAINS (/ab_data) \
                    -VMDS (plc_5_40) \
                    -LOG
CREATE ACTIVITY ab_event_act -PROGRAM ab_event_srv \
                    -NODES (ab_node) \
                    -DOMAINS (/ab_data) \
                    -LOG
```

The CREATE PROGRAM command shown in Example 3–7 has an additional attribute, the -PARAM attribute, that specifies a parameter file to be read by the ab_data_srv program. The parameter file contains the name of the local VMD to receive the write indications as shown in Example 3–8. See *BASEstar Open for OpenVMS Management Guide* for more information on possible contents of the parameter file.

Example 3–8 AB.PAR Contents

```
/AB_IND/ab_data_srv.env: BSTR_LOCAL_VMD_NAME=plc_2_30
```

Only one parameter file can be specified per program so BASEstar Open limits the number of local VMDs that can be specified to one per program. This limitation means that **all** write indications for a particular program must be directed to one local VMD, so all PLCs connected to that line must use unique addresses to ensure that the data from the PLC is uniquely identified. Also, mixing of types writes (i.e. PLC-2 and PLC-5) is not supported.

Example 3–9 shows an example that creates variables, device data points, events, triggers and enboxes to receive write indications from the PLC-5 configured in Example 3–7.

Example 3–9 Write Indication Example

```
BSTR> EXECUTE ACTOR /AB_IND -WAIT
BSTR> CREATE UNNAMED_VARIABLE plc_2_30.p1 \
      -DATATYPE UNSIGNED_16 \
      -ADDRESS "200" \
      -ADDRTYPE UNCONSTRAINED
BSTR> CREATE DEVICE_DATA_POINT /ab_data/p1 \
                    -DATATYPE UNSIGNED_16 \
                    -UPDATE_POLICY UNSOLICITED \
                    -ACCESS_POLICY ON_REQUEST \
                    -DEVICE_ACCESS READ \
                    -VMD_NAME plc_2_30 \
                    -VAR_NAME p1 \
                    -VAR_CLASS UNNAMED
BSTR> CREATE ENBOX /ab_data/enbox1 \
                    -QUEUE_KIND SEQUENTIAL \
                    -ENBOX_KIND SYSTEM_BUFFERING \
                    -MAX_NOTIFICATIONS 10
BSTR> CREATE EVENT /ab_data/e1 \
                    -PARAMETER_TYPES (INTEGER_16)
BSTR> CREATE TRIGGER /ab_data/t1 \
                    -STATE ENABLED \
                    -TRIGGER_EVALUATION ON_WRITE \
                    -DATA_POINT /ab_data/p1 \
                    -EVENT /ab_data/e1 \
                    -EVE_PARAMETERS (/ab_data/p1:VALUE)
BSTR> ADD SUBSCRIPTION EVENT /ab_data/e1 /ab_data/enbox1 \
                    -NOTIFICATION PARAMETERS_LIST \
                    -STATE ENABLED
BSTR> CONNECT ENBOX /ab_data/enbox1
BSTR> RECEIVE NOTIFICATION ENBOX /ab_data/enbox1
Event               : /ab_data/e1
Origin              : /ab_data/t1
Priority            : 0
Confirmation        : NOT_CONFIRMED
Notification_id     : 1
User_parameter      : 0
Notification_format: PARAMETERS_LIST
Time                : Fri Mar 17 08:26:50 2000

Parameter 1 value:
1
BSTR> DISCONNECT ENBOX /ab_data/enbox1
BSTR> REMOVE SUBSCRIPTION EVENT /ab_data/e1 /ab_data/enbox1
```

The use of the value INIT_UNSOLICITED for the -UPDATE_POLICY qualifier is not supported for write indications since the address of the variable in the remote VMD is not available. If an application requires the value of an unsolicited variable when starting, before the value has been updated by the device, create a device data point and unnamed variable on the remote VMD and use that to read the initial value.

If a PLC has been programmed to send data to the host and a variable is found that matches the address of the data being sent, a write indication is generated specifying the variable that was found. If more than one variable is configured that matches the address of the data being sent, then an indication is generated for only the first variable found.

Table 3–6 lists the functions to use when programming the PLC to send data to the host for each PLC family. Sending PLC-2 protected or unprotected writes is supported for all PLC families.

For SLC controllers use the 485CIF target device to send PLC-2 messages. Also for SLC controllers the destination address is specified as a decimal byte offset. This value must be converted to a word octal address to obtain the unnamed variable address to use for the message coming from the device.

Table 3–6 Supported PLC Write Functions

PLC Family			
Remote VMD	**Local VMD**	**Function Name**	**Address Type**
PLC 2	PLC 2	Protected or Unprotected Write	PLC-2 address
PLC 3	PLC 2	Protected or Unprotected Write	PLC-2 address
PLC 3	PLC 3	Word Range Write	PLC-3 Logical binary address
PLC 5	PLC 2	Protected or Unprotected Write	PLC-2 address
PLC 5	PLC-5	Typed Write	PLC-5 Logical binary or ASCII address
SLC	PLC 2	Protected or Unprotected Write	PLC-2 address
SLC	SLC	Protected Typed Logical Write	SLC address

3.2.4.2 Abort Indications

An abort indication is generated by the DAS when it loses a connection to the PLC. The DAS detects that a connection is broken under the following conditions:

- When a request to the KE/KF card times out. In this case all connections using this communication card are aborted.
- When an error is returned from a QIO call. In this case all connections using this port are aborted.
- When a LOCAL status code of X'02' is returned from the KE/KF card. In this case the KE/KF cards is unable to communicate to the remote PLC. In this case only the connection to the PLC for which the X'02' status code is returned is aborted.

3.3 Addressing Syntax and Supported Data Types

Data types are defined at the device and application levels. The device data type is referred to as the *MMS data type*. The MMS data type is mapped to an application data type.

Data types can be either *simple* or *complex*. Simple data types are native to the OMNI environment. Complex data types are divided into either arrays or structures. Complex data types must be created by a user before they can be used in defining data points or variables.

The following sections describe the supported data types and addressing syntax for each of the supported PLC families.

3.3.1 PLC-2 Addressing Syntax and Supported Data Types

Figure 3–1 shows the structure of a legal address for the PLC-2 family.

Figure 3–1 General PLC-2 Address Format

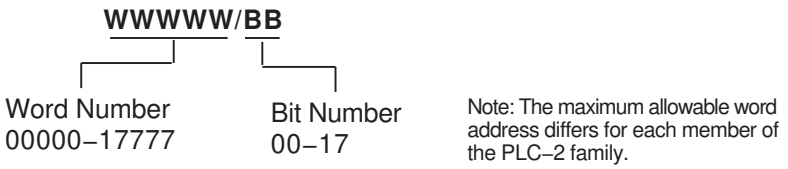


Table 3–7 shows examples of valid addresses for the PLC-2 family.

Table 3–7 Example PLC-2 Data Types

Example Address	MMS Type[1]	Description
377	UNSIGNED_16	Word 377 octal including status bits[2]
20	BCD3[3]	Word 20 octal[2]
1777/17	BIT_STRING	Bit 17 octal of word 1777 octal[2]
33	Array of 10 BCD3	Ten words beginning at word 33 octal

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]The number must be a valid BCD number from 0-999. Status bits are cleared.

3.3.2 PLC-3 Addressing Syntax and Supported Data Types

The PLC-3 family supports the following addressing syntaxes:

- Data table addressing
- Extended addressing

Section 3.3.2.1 and Section 3.3.2.2 describe these addressing syntaxes.

3.3.2.1 PLC-3 Data Table Addressing

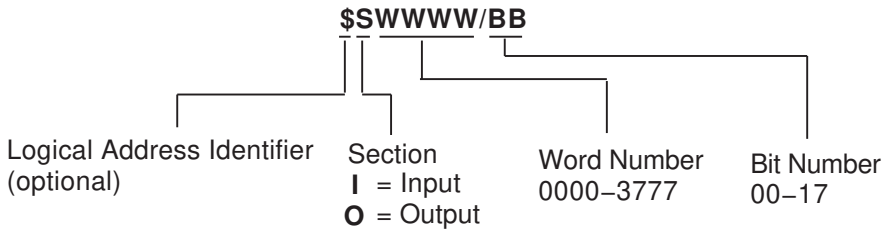
The structure of a legal data table address for the PLC-3 differs depending on the table that is being addressed. The PLC-3 supports the following data table sections:

I - Input
O - Output
C - Counter
T - Timer
A - ASCII
B - Binary
D - Decimal
F - Floating-Point
H - High-Order Integer
N - Integer
S - Status
P - Pointer

Figure 3–2 shows the structure of a legal I/O data table address for the PLC-3.

Table 3–8 shows examples of legal I/O data table addresses for the PLC-3.

Figure 3–2 PLC-3 I/O Data Table Address Format



Note: Addresses 0000–3777 are reserved for the I/O image. The remaining words of file 0 and all of files 1–999 can be used for data storage.

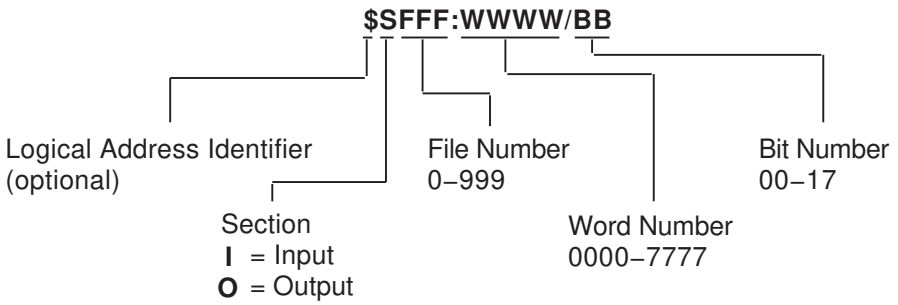


Table 3–8 Example PLC-3 I/O Data Table Addresses

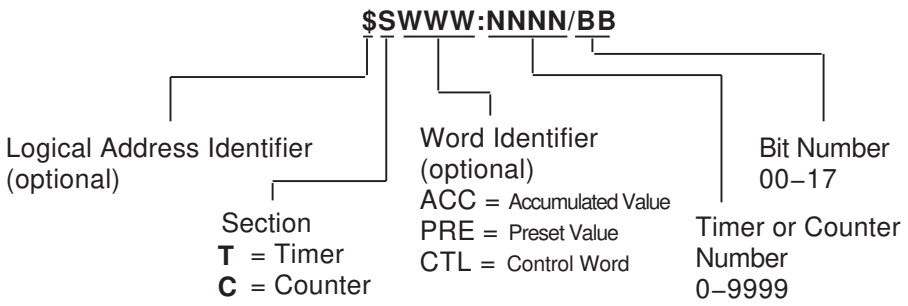
Example Address	MMS Type[1]	Description
I20	UNSIGNED_16	Input image rack 2 group 0 octal
I377	INTEGER_16	Input image rack 37 group 7 octal
I1:1777/17	BIT_STRING	Bit 17 of word 1777 octal in file 1 decimal
$O33:77/1	BIT_STRING	Bit 1 of word 77 octal in file 33 decimal
O99:167	INTEGER_16	Word 167 octal in file 99 decimal
I0	Array of 10 UNSIGNED_16	Ten words beginning at rack 0 group 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–3 shows the structure of a legal Timer or Counter data table address for the PLC-3. Figure 3–4 shows a PLC-3 Timer structure. Figure 3–5 shows a PLC-3 Counter structure.

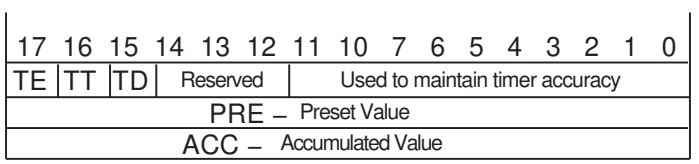
Table 3–9 shows examples of legal Timer or Counter data table addresses for the PLC-3.

Figure 3–3 PLC-3 Timer and Counter Data Table Address Format



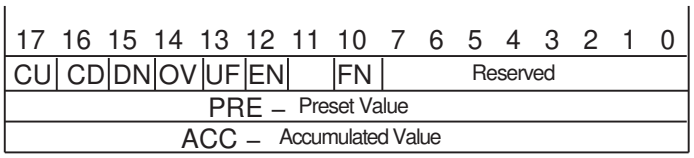
Note: If a word identifier is not specified, then the whole structure is addressed.

Figure 3–4 PLC-3 Timer Structure



TE – Timer Enabled
TT – Timer Timing
TD – Timer Done

Figure 3–5 PLC-3 Counter Structure



CU – Count Up Enable
CD – Count Down Enable
DN – Counter Done
OV – Counter Overflow
UF – Counter Underflow
EN – File Enable
FN – File Found

Table 3–9 Example PLC-3 Timer and Counter Data Table Addresses

Example Address	MMS Type[1]	Description
$TACC20	UNSIGNED_16	Timer 20 accumulated value
CPRE:91	INTEGER_16	Counter 91 preset value
TCTL:20/17	BIT_STRING	Timer 20 Timer Enable bit[2]
T20.TD	BIT_STRING	Timer 20 Timer Done bit[2]
C5	Structure[3] UNSIGNED_16 UNSIGNED_16 UNSIGNED_16	All of counter 5. Control Preset value Accumulated value

[1]Arrays are **not** supported.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]Write operations are not allowed to structures. Fields in the structure can be any combination of three INTEGER_16 or UNSIGNED_16. The control, preset and accumulated values are read in that order.

Figure 3–6 shows the structure of a legal ASCII, Binary, Decimal, Floating-Point, High-Order Integer or Status data table address for the PLC-3.

Table 3–10 shows examples of legal ASCII, Binary, Decimal, Floating-Point, High-Order Integer or Status data table addresses for the PLC-3.

Figure 3–6 PLC-3 ASCII, Binary, Decimal, Floating-Point, High-Order Integer, Integer and Status Data Table Address Format

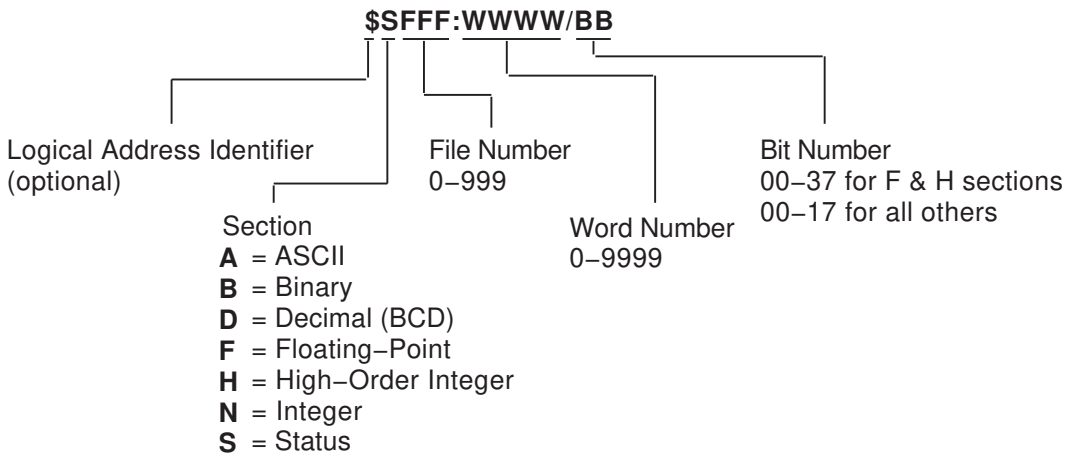


Table 3–10 Example PLC-3 ASCII, Binary, Decimal, Floating-Point, High-Order Integer, Integer and Status Data Table Address Format

Example Address	MMS Type[1]	Description
$S0:0	UNSIGNED_16	Word 0 of status file 0
N20:0	INTEGER_16	Word 0 of integer file 20
D20:780	BCD4[2]	Word 780 of decimal (BCD) file 20
F39:19	FLOAT[3]	Word 19 of floating-point file 39
H8:20	UNSIGNED_32[4]	Word 20 of high-order integer file 8
H99:0	INTEGER_32[4]	Word 0 of high-order integer file 99
A10:20	VISIBLE_STRING[5]	Word 20 of ASCII file 10
A7:0	OCTET_STRING[5]	Word 0 of ASCII file 7
B5:20/10	BIT_STRING[6]	Bit 10 octal of word 20 of binary file 5

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

[2]Decimal (BCD) section only. The data must be a valid BCD number from 0-9999.

[3]Floating-point section only.

[4]High-order integer section only.

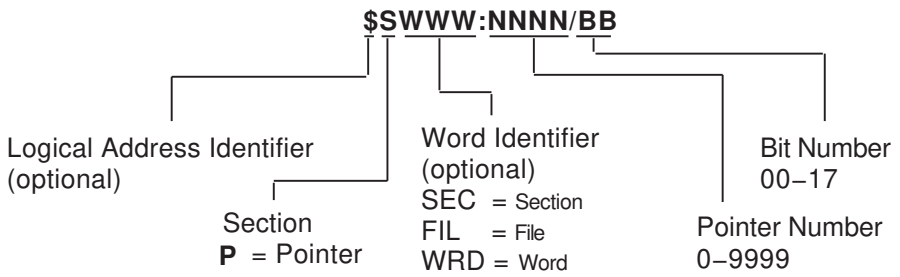
[5]ASCII section only.

[6]Bit writes that span more than one word are **not** supported.

Figure 3–7 shows the structure of a legal Pointer data table address for the PLC-3. Figure 3–8 shows a PLC-3 Pointer structure.

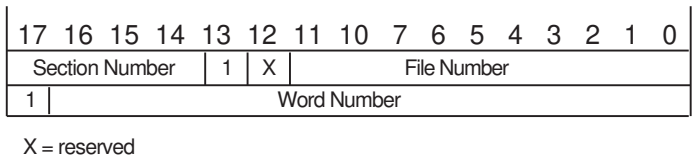
Table 3–11 shows examples of legal Pointer data table addresses for the PLC-3.

Figure 3–7 PLC-3 Pointer Data Table Address Format



Note: If a word identifier is not specified, then the whole structure is addressed.

Figure 3–8 PLC-3 Pointer Structure



X = reserved

Table 3–11 Example PLC-3 Pointer Data Table Addresses

Example Address	MMS Type[1]	Description
$PFIL:91	INTEGER_16T	Pointer 91 file number
PSEC:20	UNSIGNED_16	Pointer 20 section number
P4	Structure[2] UNSIGNED_16 UNSIGNED_16 UNSIGNED_16	All of pointer 4 Section File Word

[1]Writes are **not** allowed to the pointer file. Arrays are **not** supported.

[2]Fields in the structure can be any combination of three INTEGER_16 or UNSIGNED_16. The section, file and word values are read in that order.

3.3.2.2 PLC-3 Extended Addressing

Extended addressing is an alternative form of addressing memory in the PLC-3. Only data table area addresses are supported (area 3).

Figure 3–9 shows the structure of legal extended addresses.

Table 3–12 shows example addresses and supported data types for the PLC-3.

Figure 3–9 PLC-3 Extended Address Format

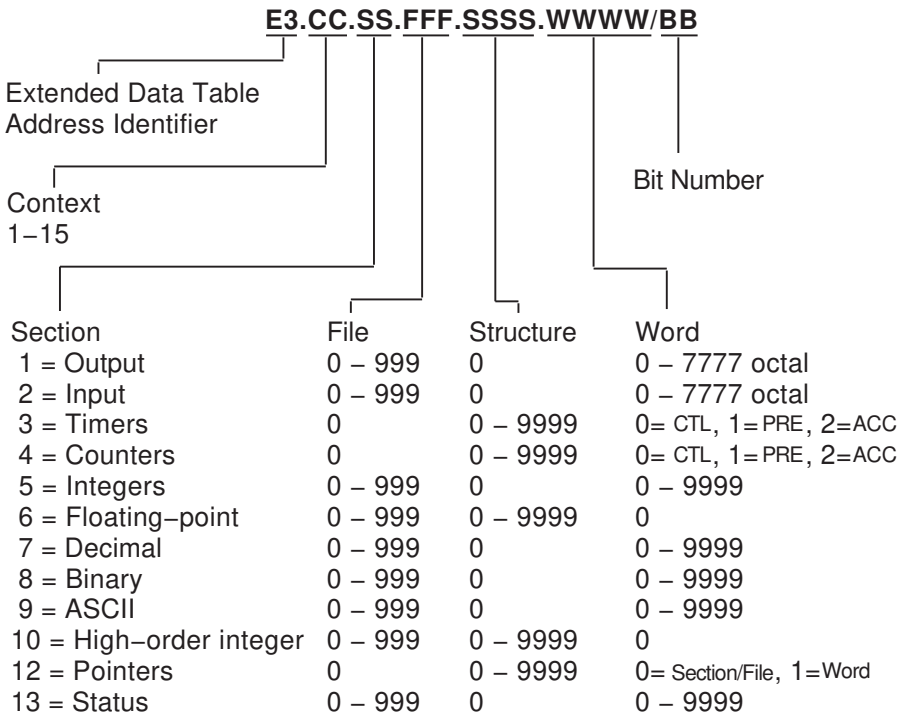


Table 3–12 Example PLC-3 Extended Addresses

Example Address	MMS Type[1]	Description
E3.1.1.0.0.77	INTEGER_16	Word 77 octal of the output image (1) file
E3.1.9.0.0.30	VISIBLE_STRING	Word 30 of ASCII (9) file 0
E3.1.10.8.99.0	INTEGER_32	Longword 99 of high-order integer (10) file 8
E3.1.4.0.99.2	UNSIGNED_16	Accumulated value of counter (4) 99
E3.1.8.8.0.183/16	BIT_STRING	Bit 16 octal of word 183 of binary (8) file 8

[1]Support for data types and arrays follow the same rules as when using data table addressing. Structures are **not** supported.

3.3.3 PLC-5 Addressing Syntax and Supported Data Types

This section describes legal data table addresses and data types for the 1785 PLC-5 family. For information on addresses and supported data types for the PLC-5/250 PLC, refer to Section 3.3.4.

The structure of a legal data table address for the PLC-5 differs depending on the table that is being addressed. The PLC-5 supports the following data table sections:

I - Input
O - Output
S - Status
B - Binary
T - Timer
C - Counter
R - Control
N - Integer
F - Floating-Point
A - ASCII
D - Decimal
ST - String
SC - SFC Status
BT - Block-Transfer Control
TD - Token Data
PD - PID Control
MG - Message Control

Figure 3–10 shows the structure of a legal I/O data table address for the PLC-5.

Table 3–13 shows examples of legal I/O data table addresses for the PLC-5.

Figure 3–10 PLC-5 I/O Data Table Address Format

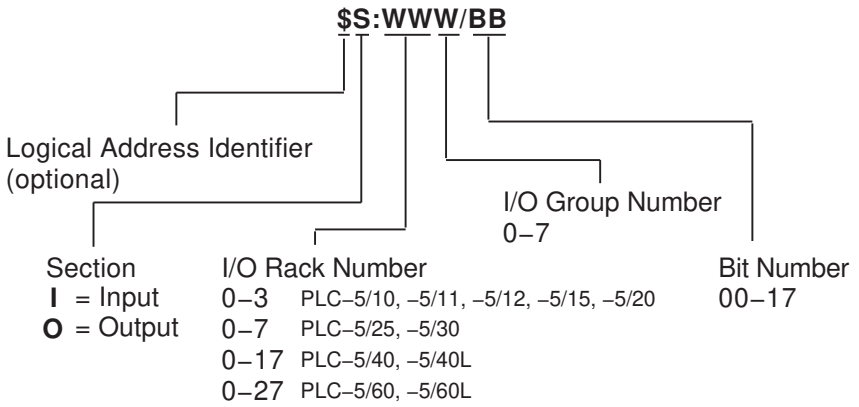


Table 3–13 Example PLC-5 I/O Data Table Addresses

Example Address	MMS Type[1]	Description
I:20	UNSIGNED_16	Input image rack 2 group 0 octal
I:277	INTEGER_16	Input image rack 27 group 7 octal
I:177/17	BIT_STRING	Bit 17 of rack 17 group 7 octal
$O:77/1	BIT_STRING	Bit 1 of rack 7 group 7 octal
I:0	Array of 10 UNSIGNED_16	Ten words beginning at rack 0 group 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–11 shows the structure of a legal Status data table address for the PLC-5.

Table 3–14 shows examples of legal Status data table addresses for the PLC-5.

Figure 3–11 PLC-5 Status Data Table Address Format

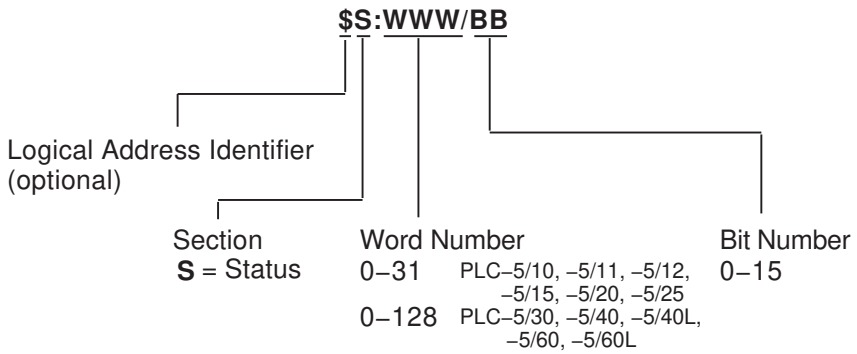


Table 3–14 Example PLC-5 Status Data Table Addresses

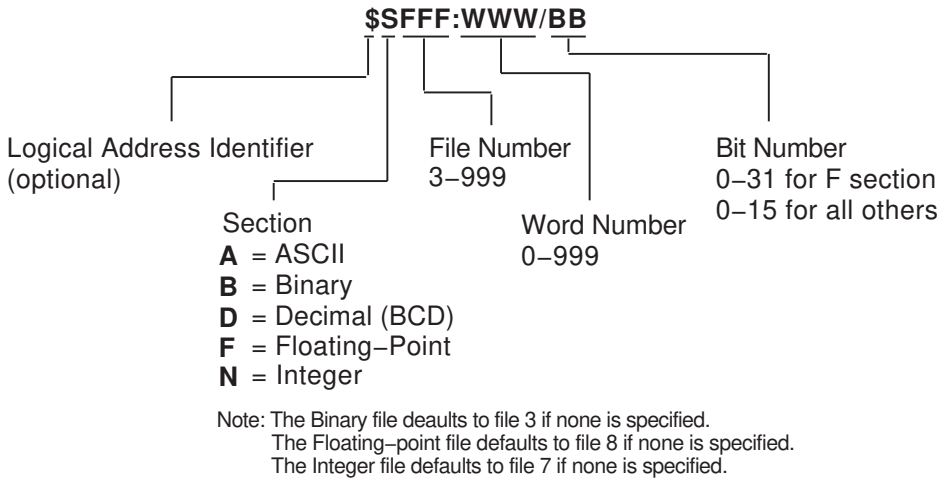
Example Address	MMS Type[1]	Description
S:20	UNSIGNED_16	Word 20 of the status file
S:12	INTEGER_16	Word 12 of the status file
$S:125/1	BIT_STRING	Bit 1 of word 125 of the status file
S:0	Array of 10 UNSIGNED_16	Ten words beginning at word 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–12 shows the structure of a legal ASCII, Binary, Decimal, Floating-Point or Integer data table address for the PLC-5.

Table 3–15 shows examples of legal ASCII, Binary, Decimal, Floating-Point and Integer data table addresses for the PLC-5.

Figure 3–12 PLC-5 ASCII, Binary, Decimal, Floating-Point and Integer Data Table Address Format



Note: The Binary file deaults to file 3 if none is specified.
The Floating–point file defaults to file 8 if none is specified.
The Integer file defaults to file 7 if none is specified.

Table 3–15 Example PLC-5 ASCII, Binary, Decimal, Floating-Point and Integer Data Table Address Format

Example Address	MMS Type[1]	Description
$N20:0	INTEGER_16	Word 0 of integer file 20
D20:780	BCD4[2]	Word 780 of decimal (BCD) file 20
F39:19	FLOAT[3]	Word 19 of floating-point file 39
A10:20	VISIBLE_STRING[5]	Word 20 of ASCII file 10
A11:0	OCTET_STRING[4]	Word 0 of ASCII file 11
B91:20/10	BIT_STRING[5]	Bit 10 of word 20 of binary file 91
B:33	UNSIGNED_16	Word 33 of the default binary file (3)
N:91	UNSIGNED_16	Word 91 of the default integer file (7)
F:3	FLOAT	Word 3 of the default floating-point file (8)
B/155	BIT_STRING	Bit 11 of word 9 in the default binary file (3)

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

[2]Decimal (BCD) section only. The data must be a valid BCD number from 0-9999.

[3]Floating-point section only.

[4]ASCII section only.

[5]Bit writes that span more than one element are **not** supported.

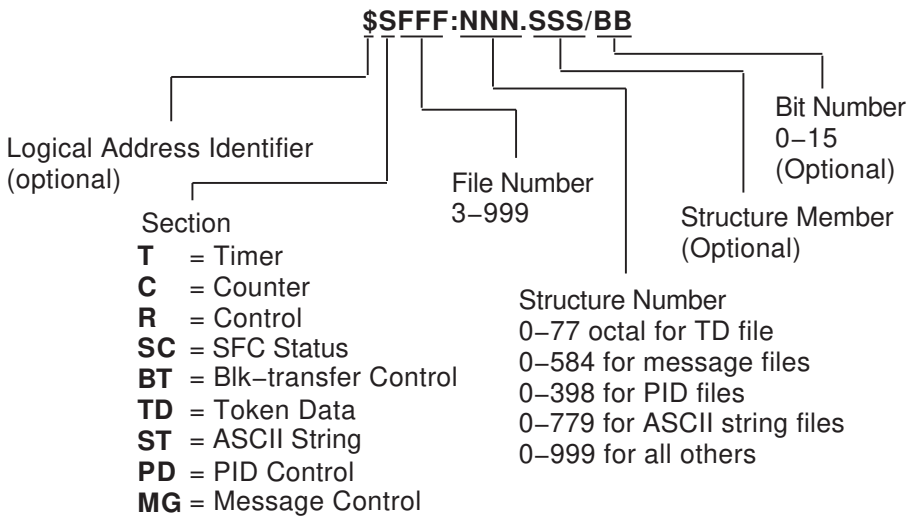
Figure 3–13 shows the structure of a legal Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control or Message Control data table address for the PLC-5.

Figure 3–14 shows a PLC-5 Timer structure.
Figure 3–15 shows a PLC-5 Counter structure.

Figure 3–16 shows a PLC-5 Control structure.
Figure 3–17 shows a PLC-5 SFC Status structure.
Figure 3–18 shows a PLC-5 Block-Transfer Control structure.
Figure 3–19 shows a PLC-5 Token Data structure.
Figure 3–20 shows a PLC-5 ASCII String structure.
Figure 3–21 shows a PLC-5 PID Control structure.
Figure 3–22 shows a PLC-5 Message Control structure.

Table 3–16 shows examples of legal Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control data table addresses for the PLC-5.

Figure 3–13 PLC-5 Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control Data Table Address Format



Note: If a structure member is not specified, then the whole structure is addressed. Structure members are described in following figures.

Note: SC, BT, TD, ST, PD and MG file types are not available on PLC–5/10, –5/11, –5/12, –5/20, and –5/25 processors.

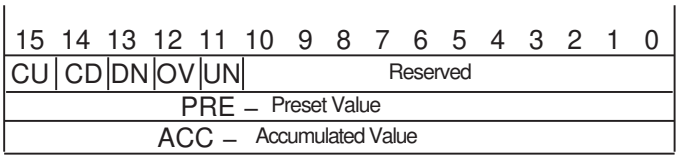
Note: The Timer file defaults to file 4 if not specified.
The Counter file defaults to file 5 if not specified.
The Control file defaults to file 6 if not specified.

Figure 3–14 PLC-5 Timer Structure

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	TT	TD	Reserved					Used to maintain timer accuracy							
PRE – Preset Value															
ACC – Accumulated Value															

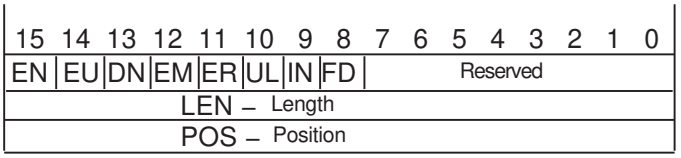
EN – Timer Enabled
TT – Timer Timing
TD – Timer Done

Figure 3–15 PLC-5 Counter Structure



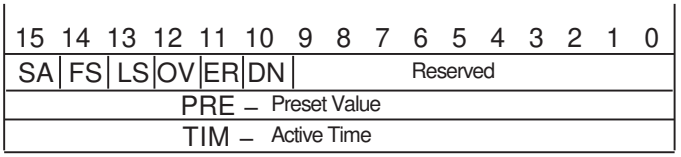
CU – Count Up Enable
CD – Count Down Enable
DN – Counter Done
OV – Counter Overflow
UN – Counter Underflow

Figure 3–16 PLC-5 Control Structure



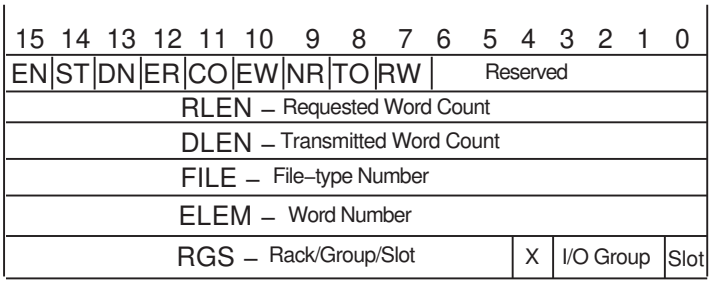
EN – Enable
EU – Enable Unloading
DN – Done
EM – Empty
ER – Error
UL – Unload
IN – Inhibit Comparison
FD – Found

Figure 3–17 PLC-5 SFC Status Structure



SA – Scan Active
FS – First Scan
LS – Last Scan
OV – Timer Overflow
ER – Step Errored
DN – Done

Figure 3–18 PLC-5 Block-Transfer Control Structure



EN – Enable
ST – Start
DN – Done
ER – Error
CO – Continue
EW – Enabled Waiting
NR – No Response
TO – Time Out
RW – Read/Write

Figure 3–19 PLC-5 Token Data Structure

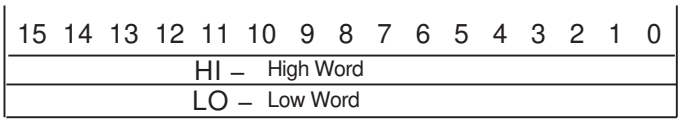


Figure 3–20 PLC-5 ASCII String Structure

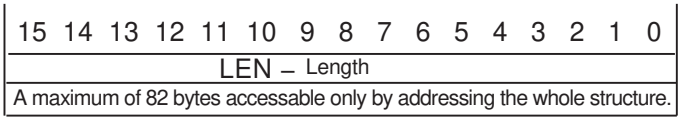
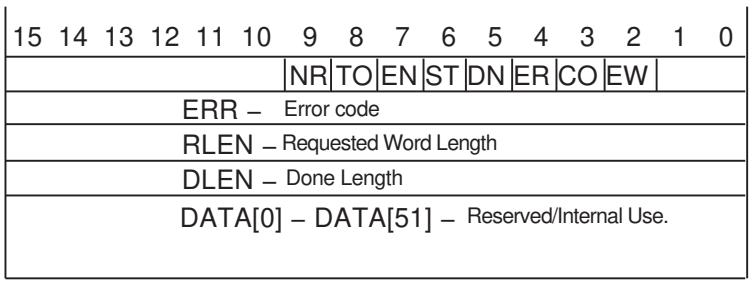


Figure 3–21 PLC-5 PID Control Structure

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
EN CT CL PVT DO SWM CA MO PE NI SPOR DLL OLH EWD DVNA DVPA PVLA PVHA
SP – Set Point
KP – Proportional Gain
KI – Initial Gain
KD – Derivative Gain
BIAS – Output Bias %
MAXS – Maximum Scaled Value
MINS – Minimum Scaled Value
DB – Dead Band
SO – Set Output %
MAXO – Maximum Output LImit
MINO – Minimum Output Limit
UPD – Update Time
PV – Process Variable
ERR – Error: Scaled
OUT – Output
PVH – PV Alarm High
PVL – PV Alarm Low
DVP – Deviation Alarm +
DVN – Deviation Alarm –
PVDB – PV Alarm Dead Band
DVDB – Deviation Alarm Dead Band
MAXI – Maximum Input
MINI – Minimum Input
TIE – Tieback %
ADDR[0] – ADDR[3] – Master to Slave Address
DATA[0] – DATA[13] – Reserved/Internal Use

Figure 3–22 PLC-5 Message Control Structure



NR – No Response
TO – Time Out
EN – Enable
ST – Start Transmission
DN – Done
ER – Error
CO – Continuous
EW – Enabled Waiting

Table 3–16 Example PLC-5 Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control Data Table Address Formats

Example Address	MMS Type[1]	Description
$T20:0.ACC	UNSIGNED_16	Timer 0 in file 20 accumulated value
C91:3.PRE	INTEGER_16	Counter 91 in file 3 preset value
T:20.CTL/15	BIT_STRING	Timer 20 in the default timer file (4) Enable bit[2]
R6:0.UL	BIT_STRING	Control 0 in file 6 Unload bit[2]
T4:20.TD	BIT_STRING	Timer 20 in file 4 Timer Done bit[2]
R:1.LEN	INTEGER_16	Control 1 in the default control file (6) Length value
SC10:0.TIM	INTEGER_16	SFC status 0 in file 10 Active Time value
BT11:3.RGS	UNSIGNED_16	Block Transfer Control 3 in file 11 Rack/Group/Slot
TD15:2.HI	INTEGER_16	Token Data 2 in file 15 High Word value
ST16:0.LEN	INTEGER_16	String 0 in file 16 Length value
ST16:2	VISIBLE_STRING[3]	String 2 in file 16
ST16:3	OCTET_STRING[3]	String 3 in file 16

[1]Arrays are **not** supported for structures other than ASCII strings.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]Only Varying length strings are supported.

(continued on next page)

Table 3–16 (Cont.) Example PLC-5 Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control Data Table Address Formats

Example Address	MMS Type[1]	Description
PD17:2.DVDB	FLOAT	PID Control 2 in file 17 Deviation Alarm Dead Band
PD17:2.EN	BIT_STRING	PID Control 2 in file 17 Enable
PD17:1.ADDR[0]	INTEGER_16	PID Control 1 in file 17 Master-to-slave address 0
MG18:3.RLEN	INTEGER_16	Message Control 3 in file 18 Requested Length
T:5	Structure[4] UNSIGNED_16 UNSIGNED_32 UNSIGNED_32	All of timer 5 in file 4 Control Preset value Accumulated value
R8:10	Structure[4] UNSIGNED_16 INTEGER_16 INTEGER_16	All of control 10 in file 8 Control Length Position
SC9:5	Structure[4] INTEGER_16 UNSIGNED_16 UNSIGNED_16	All of SFC status 5 in file 9 Control Preset value Active Time
BT11:2	Structure[4] UNSIGNED_16 INTEGER_16 INTEGER_16 INTEGER_16 INTEGER_16 UNSIGNED_16	All of BT 2 in file 11 Control Requested word count Transmitted word count File-type number Word number Rack/Group/Slot
TD15:2	Structure[4] UNSIGNED_16 UNSIGNED_16	All of Token Data 2 in file 15 High Word Low Word

[1]Arrays are **not** supported for structures other than ASCII strings.

[4]Write operations are **not** allowed to structures other than to ASCII strings.

(continued on next page)

Table 3–16 (Cont.) Example PLC-5 Timer, Counter, Control, SFC Status, Block-Transfer Control, Token Data, ASCII String, PID Control and Message Control Data Table Address Formats

Example Address	MMS Type[1]	Description
PD17:0	Structure[4]	All of PID Control 0 in file 17
	UNSIGNED_32	Control
	FLOAT	Set point
	FLOAT	Proportional gain
	FLOAT	Initial gain
	FLOAT	Derivative gain
	FLOAT	Output bias %
	FLOAT	Maximum scaled value
	FLOAT	Minimum scaled value
	FLOAT	Dead band
	FLOAT	Set output %
	FLOAT	Maximum output limit
	FLOAT	Minimum output limit
	FLOAT	Update time
	FLOAT	Process variable
	FLOAT	Error:scaled
	FLOAT	Output
	FLOAT	PV alarm high
	FLOAT	PV alarm low
	FLOAT	Deviation alarm +
	FLOAT	Deviation alarm -
	FLOAT	PV alarm dead band
	FLOAT	Deviation alarm dead band
	FLOAT	Maximum input
	FLOAT	Minimum intput
	FLOAT	Tieback %
	Array of 4	
	UNSIGNED_16	Master-to-slave address
	Array of 14	
	FLOAT	Reserved/internal use
MG18:2	Structure[4]	All of Message Control 2 in file 18
	UNSIGNED_16	Control
	INTEGER_16	Error code
	INTEGER_16	Requested length
	INTEGER_16	Done length
	Array of 52	
	INTEGER_16	Reserved/internal use

[1]Arrays are **not** supported for structures other than ASCII strings.

[4]Write operations are **not** allowed to structures other than to ASCII strings.

3.3.4 PLC-5/250 Addressing Syntax and Supported Data Types

This section describes legal addresses and data types for the PLC-5/250 PLC. For information on addresses and supported data types for the 1785 PLC-5 family, refer to Section 3.3.3.

The PLC-5/250 supports the following four types of modules:

Resource Manager
Logic Processor (1-4)
Remote Scanner (1-4)
CVIM (1-4)

Each module can be addressed and supports different types of memory sections. The structure of a legal address for the PLC-5/250 differs depending on the area of memory that is being addressed.

The PLC-5/250 supports the following memory sections for the Resource Manager and Logic Processor modules:

B - Binary
N - Integer
L - Long Integer
F - Floating-Point
T - Timer
C - Counter
R - Control
ST - String
PD - PID Control
MSG - Message Control
S - Status

The PLC-5/250 supports the following memory sections for the Remote Scanner module:

I - Input
O - Output
BTD - Block-Transfer Data
BR - Block-Transfer Read Control
BW - Block-Transfer Write Control
AS - Adapter Status
IS - Internal Storage

The PLC-5/250 supports the following memory section for the CVIM module:

SD - Shared Data

Figure 3–23 shows the structure of a legal Binary, Floating-Point, Long Integer or Integer table address for the PLC-5/250.

Table 3–17 shows examples of legal Binary, Floating-Point, Long Integer and Integer table addresses for the PLC-5/250.

Figure 3–23 PLC-5/250 Binary, Floating-Point, Long Integer and Integer Address Format

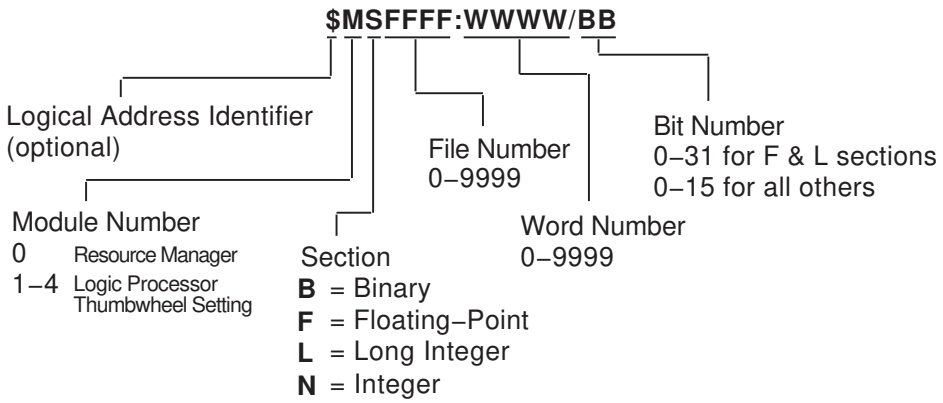


Table 3–17 Example PLC-5/250 Binary, Floating-Point, Long Integer and Integer Address Format

Example Address	MMS Type[1]	Description
$0N20:0	INTEGER_16	Word 0 of integer file 20
0L5:3	UNSIGNED_32[2]	Word 3 of long integer file 5
1L3:145	SIGNED_32[2]	Word 145 of long integer file 3
0F39:19	FLOAT[3]	Word 19 of floating-point file 39
0B91:20/10	BIT_STRING	Bit 10 of word 20 of binary file 91
1B3:33	UNSIGNED_16	Word 33 of binary file 3
0N7:91	UNSIGNED_16	Word 91 of integer file 7
0F8:3	FLOAT	Word 3 of floating-point file 8
0B2/124	BIT_STRING	Bit 12 of word 7 in the binary file 2
0B3:3/3	Array of 10 BIT[4]	Ten bits beginning at word 3 of binary file 3

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

[2]Long integer section only.

[3]Floating-point section only.

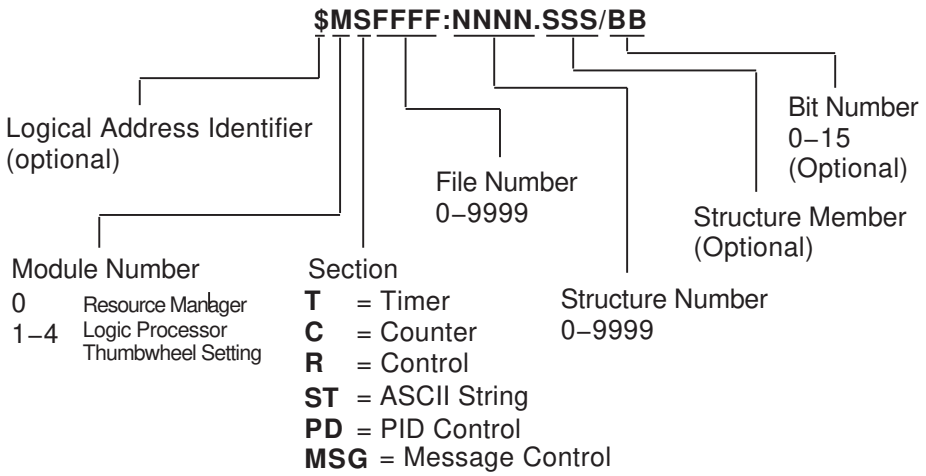
[4]Bit writes that span more than one element are **not** supported.

Figure 3–24 shows the structure of a legal Timer, Counter, Control, ASCII String, PID Control or Message Control data table address for the PLC-5/250.

Figure 3–25 shows a PLC-5/250 Timer structure.
Figure 3–26 shows a PLC-5/250 Counter structure.
Figure 3–27 shows a PLC-5/250 Control structure.
Figure 3–28 shows a PLC-5/250 ASCII String structure.
Figure 3–29 shows a PLC-5/250 PID Control structure.
Figure 3–30 shows a PLC-5/250 Message Control structure.
Figure 3–25 shows a PLC-5/250 Timer structure.
Figure 3–25 shows a PLC-5/250 Timer structure.

Table 3–18 shows examples of legal Timer, Counter, Control, ASCII String, PID Control and Message Control data table addresses for the PLC-5/250.

Figure 3–24 PLC-5/250 Timer, Counter, Control, ASCII String, PID Control and Message Control Address Format



Note: If a structure member is not specified, then the whole structure is addressed. Structure members are described in following figures.

Figure 3–25 PLC-5/250 Timer Structure

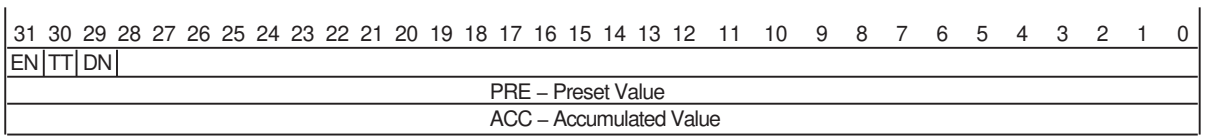
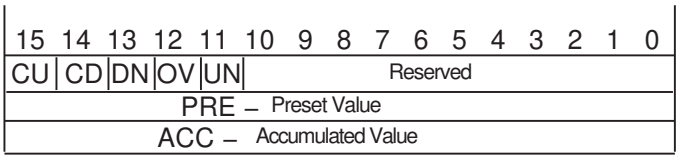
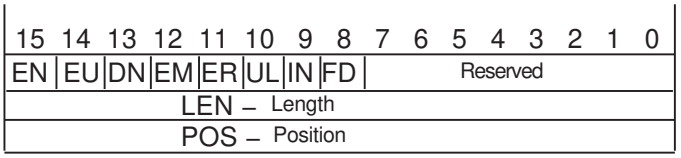


Figure 3–26 PLC-5/250 Counter Structure



CU – Count Up Enable
CD – Count Down Enable
DN – Counter Done
OV – Counter Overflow
UN – Counter Underflow

Figure 3–27 PLC-5/250 Control Structure



EN – Enable
EU – Enable Unloading
DN – Done
EM – Empty
ER – Error
UL – Unload
IN – Inhibit Comparison
FD – Found

Figure 3–28 PLC-5/250 ASCII String Structure

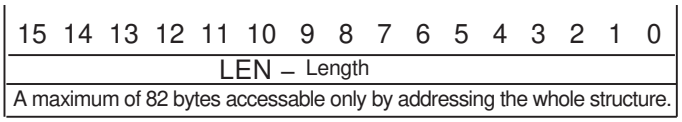
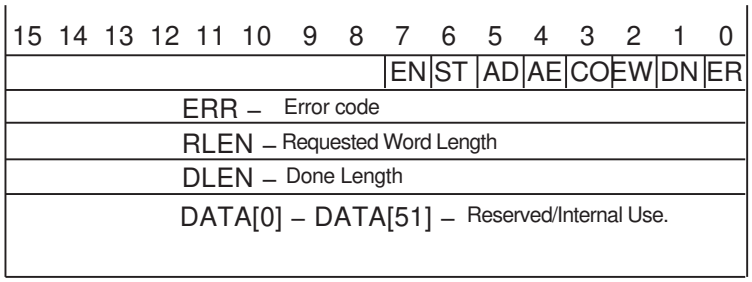


Figure 3–29 PLC-5/250 PID Control Structure

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
EN CT CL PVT DO SWM CA MO PE NI SPOR DLL OLH EWD DVNA DVPA PVLA PVHA
SP – Set Point
KP – Proportional Gain
KI – Initial Gain
KD – Derivative Gain
BIAS – Output Bias %
MAXS – Maximum Scaled Value
MINS – Minimum Scaled Value
DB – Dead Band
SO – Set Output %
MAXO – Maximum Output LImit
MINO – Minimum Output Limit
UPD – Update Time
PV – Process Variable
ERR – Error: Scaled
OUT – Output
PVH – PV Alarm High
PVL – PV Alarm Low
DVP – Deviation Alarm +
DVN – Deviation Alarm –
PVDB – PV Alarm Dead Band
DVDB – Deviation Alarm Dead Band
MAXI – Maximum Input
MINI – Minimum Input
TIE – Tieback %
ADDR[0] – ADDR[3] – Master to Slave Address
DATA[0] – DATA[13] – Reserved/Internal Use

Figure 3–30 PLC-5/250 Message Control Structure



EN – Enable
ST – Start Transmission
AD – Asynchronous Done
AE– Asynchronous Error
CO – Continuous
EW – Enabled Waiting
DN– Done
ER– Error

Table 3–18 Example PLC-5/250 Timer, Counter, Control, ASCII String, PID Control and Message Control Address Formats

Example Address	MMS Type[1]	Description
$0T20:0.ACC	UNSIGNED_32	Timer 0 in file 20 accumulated value
0C91:3.PRE	INTEGER_16	Counter 91 in file 3 preset value
0T3:20.CTL/31	BIT_STRING	Timer 20 in timer file 3 Enable bit[2]
1R6:0.UL	BIT_STRING	Control 0 in file 6 unload bit[2]
0T4:20.TD	BIT_STRING	Timer 20 in file 4 timer done bit[2]
1R6:1.LEN	INTEGER_16	Control 1 in control file 6 length value
0ST16:0.LEN	INTEGER_16	String 0 in file 16 length value
0ST16:2	VISIBLE_STRING[3]	String 2 in file 16
0ST16:3	OCTET_STRING[3]	String 3 in file 16
0PD17:2.DVDB	FLOAT	PID Control 2 in file 17 Deviation Alarm Dead Band
0PD17:2.EN	BIT_STRING	PID Control 2 in file 17 Enable
0PD17:1.ADDR[0]	INTEGER_16	PID Control 1 in file 17 Master-to-slave address 0
0MSG18:3.RLEN	INTEGER_16	Message Control 3 in file 18 Requested Length

[1]Arrays are **not** supported for structures other than ASCII strings.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]Only varying length strings are supported.

(continued on next page)

Table 3–18 (Cont.) Example PLC-5/250 Timer, Counter, Control, ASCII String, PID Control and Message Control Address Formats

Example Address	MMS Type[1]	Description
1T0:5	Structure[4]	All of timer 5 in file 0
	UNSIGNED_16	Control
	UNSIGNED_32	Preset value
	UNSIGNED_32	Accumulated value
0R8:10	Structure[4]	All of control 10 in file 8
	UNSIGNED_16	Control
	INTEGER_16	Length
	INTEGER_16	Position
0PD17:0	Structure[4]	All of PID Control 0 in file 17
	UNSIGNED_32	Control
	FLOAT	Set point
	FLOAT	Proportional gain
	FLOAT	Initial gain
	FLOAT	Derivative gain
	FLOAT	Output bias %
	FLOAT	Maximum scaled value
	FLOAT	Minimum scaled value
	FLOAT	Dead band
	FLOAT	Set output %
	FLOAT	Maximum output limit
	FLOAT	Minimum output limit
	FLOAT	Update time
	FLOAT	Process variable
	FLOAT	Error:scaled
	FLOAT	Output
	FLOAT	PV alarm high
	FLOAT	PV alarm low
	FLOAT	Deviation alarm +
	FLOAT	Deviation alarm -
	FLOAT	PV alarm dead band
	FLOAT	Deviation alarm dead band
	FLOAT	Maximum input
	FLOAT	Minimum intput
	FLOAT	Tieback %
	Array of 4	
	UNSIGNED_16	Master-to-slave address
	Array of 14	
	FLOAT	Reserved/internal use
0MSG18:2	Structure[4]	All of Message Control 2 in file 18
	UNSIGNED_16	Control
	INTEGER_16	Error code
	INTEGER_16	Requested length
	INTEGER_16	Done length
	Array of 52	
	INTEGER_16	Reserved/internal use

[1]Arrays are **not** supported for structures other than ASCII strings.

[4]Write operations are **not** allowed to structures other than to ASCII strings.

Figure 3–31 shows the structure of a legal Status data table address for the PLC-5/250.

Table 3–19 shows examples of legal Status data table addresses for the PLC-5 /250.

Figure 3–31 PLC-5/250 Status Address Format

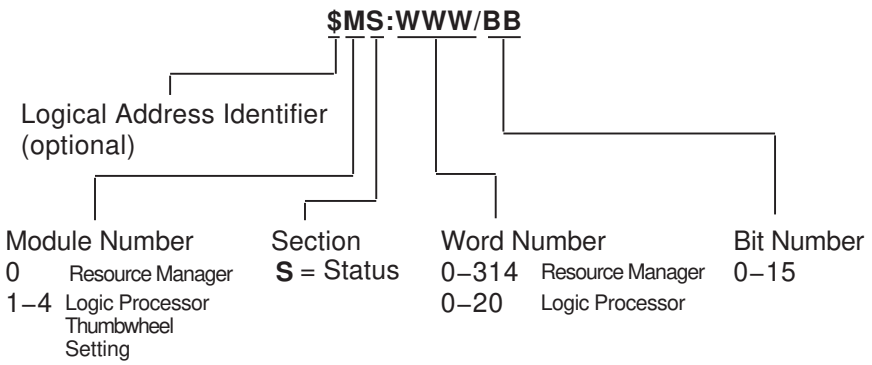


Table 3–19 Example PLC-5/250 Status Addresses

Example Address	MMS Type[1]	Description
0S:20	UNSIGNED_16	Word 20 of the status file in the resource manager
1S:12	INTEGER_16	Word 12 of the status file in logic processor 1
$0S:125/1	BIT_STRING	Bit 1 of word 125 of the status file in the resource manager
0S:0	Array of 10 UNSIGNED_16	Ten words beginning at word 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–32 shows the structure of a legal I/O data table address for the PLC-5 /250.

Table 3–20 shows examples of legal I/O data table addresses for the PLC-5/250.

Figure 3–32 PLC-5/250 I/O Address Format

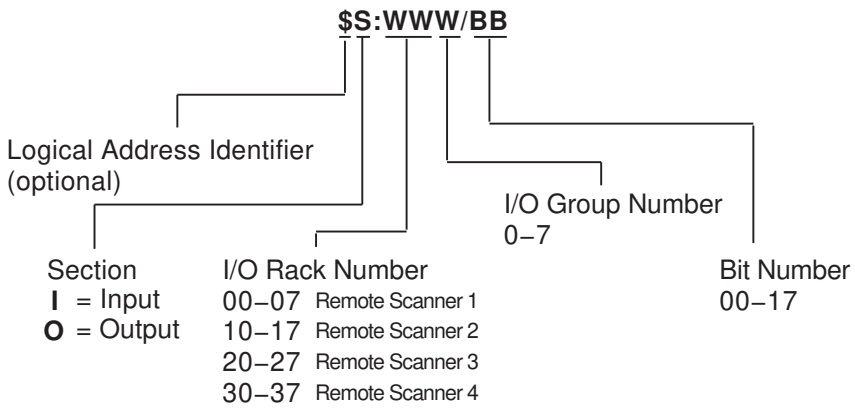


Table 3–20 Example PLC-5/250 I/O Addresses

Example Address	MMS Type[1]	Description
I:20	UNSIGNED_16	Input image rack 2 group 0 octal
I:277	INTEGER_16	Input image rack 27 group 7 octal
I:177/17	BIT_STRING	Bit 17 of rack 17 group 7 octal
$O:77/1	BIT_STRING	Bit 1 of rack 7 group 7 octal
I:0	Array of 10 UNSIGNED_16	Ten words beginning at rack 0 group 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–33 shows the structure of a legal Block-Transfer data table address for the PLC-5/250.

Table 3–20 shows examples of legal Block-Transfer data table addresses for the PLC-5/250.

Figure 3–33 PLC-5/250 Block-Transfer Data Address Format

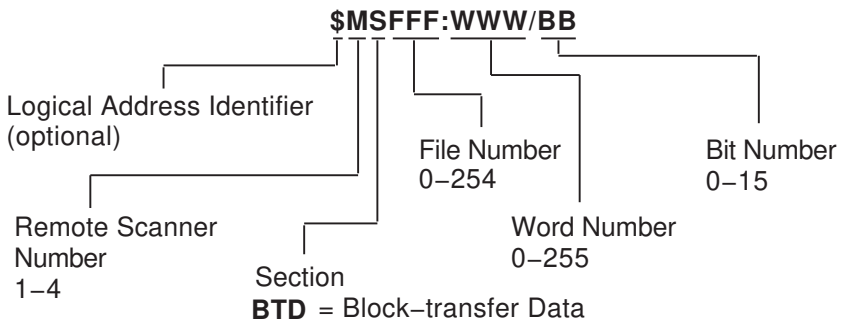


Table 3–21 Example PLC-5/250 Block-Transfer Data Addresses

Example Address	MMS Type[1]	Description
1BTD0:20	UNSIGNED_16	Word 20 of file 0 in remote scanner 1
1BTD2:12	INTEGER_16	Word 12 of file 2 in remote scanner 1
$2BTD30:125/1	BIT_STRING	Bit 1 of word 125 of file 30 in remote scanner 2
1BTD0:0	Array of 10 UNSIGNED_16	Ten words beginning at word 0 of file 0 in remote scanner 1

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–34 shows the structure of a legal Block-Transfer Read/Write Control data table address for the PLC-5/250. Figure 3–35 shows a PLC-5/250 Block-Transfer Read/Write Control structure.

Table 3–22 shows examples of legal Block-Transfer Read/Write Control data table addresses for the PLC-5/250.

Figure 3–34 PLC-5/250 Block-Transfer Read/Write Control Address Format

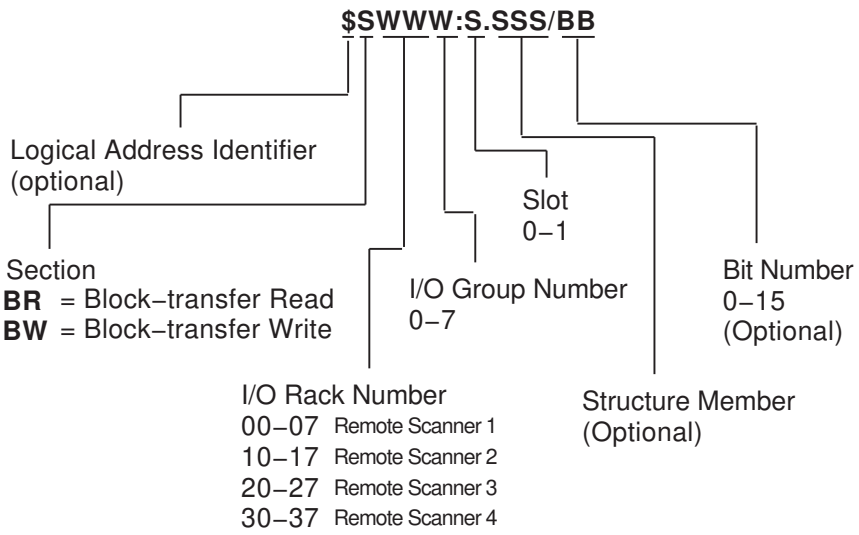


Figure 3–35 PLC-5/250 Block-Transfer Read/Write Control Structure

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EC – Error code								EN	ST	AD	AE	CO	EW	DN	ER
DLEN – Done Length															
PLEN – Programmed Length															
TOUT – Current/Programmed Time–out															
FILE – File Number/File Offset															
IDX – File Index															

EN – Enable
ST – Start Transmission
AD – Asynchronous Done
AE– Asynchronous Error
CO – Continuous
EW – Enabled Waiting
DN– Done
ER– Error

Table 3–22 Example PLC-5/250 Block-Transfer Read/Write Control Addresses

Example Address	MMS Type[1]	Description
BR01:1.DLEN	UNSIGNED_16	I/O rack 0, group 1, remote scanner 1, slot 1 done length.
BW16:0.PLEN	INTEGER_16	I/O rack 1, group 6, remote scanner 1, slot 1 programmed length.
BR103:1.EN	BIT_STRING[2]	IO rack 10, group 3, remote scanner 2, slot 1 enable bit.
$BW27:1	Structure[3]	All of Block-Transfer Write Control for I/O rack 2, group 7, remote scanner 1, slot 1.
	UNSIGNED_16	Error code/status bits.
	UNSIGNED_16	Done/programmed length.
	UNSIGNED_16	Time stamp.
	UNSIGNED_16	Current/programmed time-out.
	UNSIGNED_16	File number/file offset.
	UNSIGNED_16	Reserved.
	UNSIGNED_16	BT directory offset.
	UNSIGNED_16	BT directory length.

[1]Arrays are **not** supported for structures other than ASCII strings.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]Write operations are **not** allowed to Block-Transfer Read/Write Control structures.

Figure 3–36 shows the structure of a legal Adapter Status data table address for the PLC-5/250. Figure 3–37 shows a PLC-5/250 Adapter Status structure.

Table 3–23 shows examples of legal Adapter Status data table addresses for the PLC-5/250.

Figure 3–36 PLC-5/250 Adapter Status Address Format

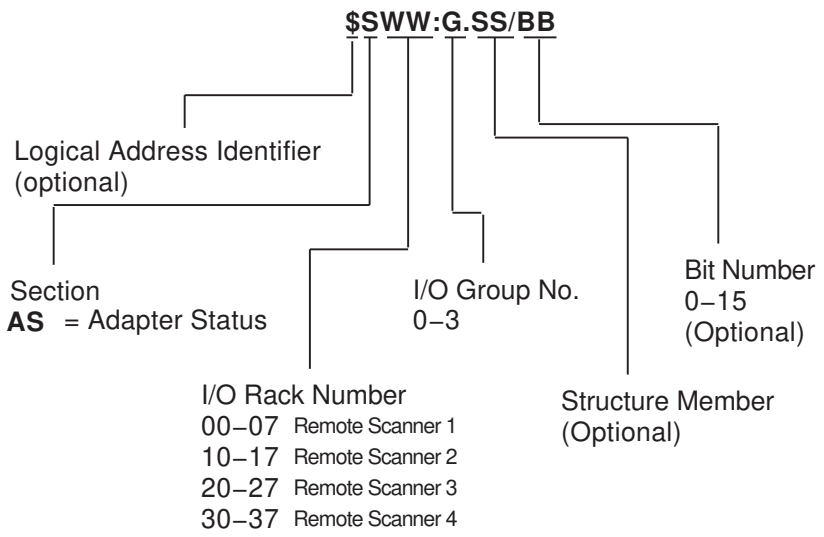
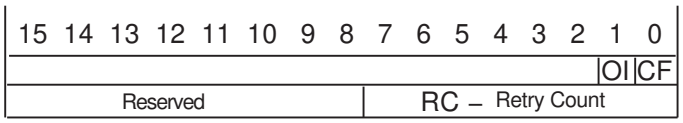


Figure 3–37 PLC-5/250 Adapter Status Structure



OI – Ouput Inhibit
CF – Fault

Table 3–23 Example PLC-5/250 Adapter Status Addresses

Example Address	MMS Type[1]	Description
AS01:1.RC	UNSIGNED_16	I/O rack 1, group 1, remote scanner 1 retry count
AS10:0.OI	BIT_STRING[2]	I/O rack 10, group 0, remote scanner 1 output inhibit
$AS27:1	Structure[3]	All of adapter status for I/O rack 27 group 1, remote scanner 3
	UNSIGNED_16	Control bits
	UNSIGNED_16	Retry count

[1]Arrays are **not** supported for structures other than ASCII strings.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]Write operations are **not** allowed to Adapter Status structures.

Figure 3–38 shows the structure of a legal Internal Storage data table address for the PLC-5/250.

Table 3–24 shows examples of legal Internal Storage data table addresses for the PLC-5/250.

Figure 3–38 PLC-5/250 Internal Storage Address Format

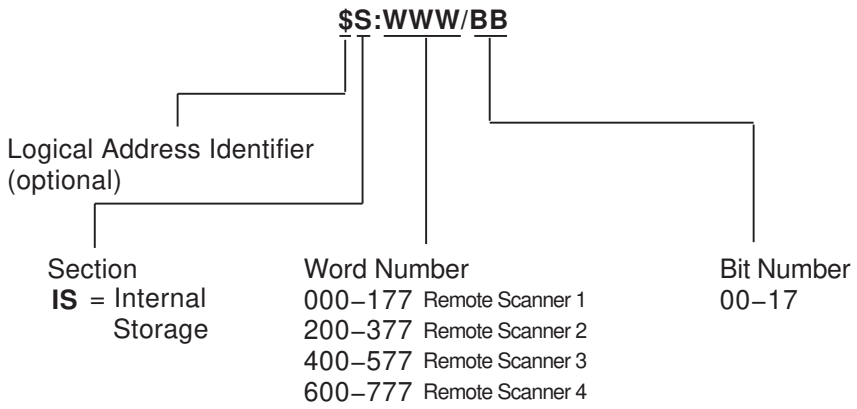


Table 3–24 Example PLC-5/250 Internal Storage Addresses

Example Address	MMS Type[1]	Description
IS:20	UNSIGNED_16	Word 20 in remote scanner 1
IS:12	INTEGER_16	Word 12 in remote scanner 1
$IS:203/10	BIT_STRING	Bit 10 octal of word 203 octal in remote scanner 2
IS:0	Array of 10 UNSIGNED_16	Ten words beginning at word 0 in remote scanner 1

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–39 shows the structure of a legal Shared Data data table address for the PLC-5/250.

Table 3–25 shows examples of legal Shared Data data table addresses for the PLC-5/250.

Figure 3–39 PLC-5/250 Shared Data Address Format

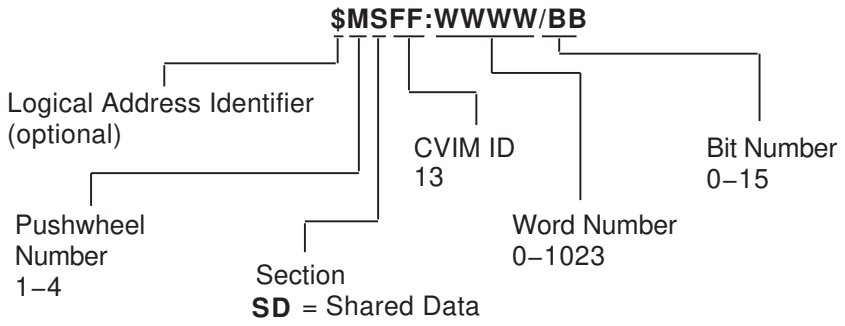


Table 3–25 Example PLC-5/250 Shared Data Addresses

Example Address	MMS Type[1]	Description
1SD13:20	UNSIGNED_16	Word 20 in CVIM module 1
2SD13:155	INTEGER_16	Word 155 in CVIM module 2
$1SD13:0/10	BIT_STRING	Bit 10 of word 0 in CVIM module 1
1SD13:100	Array of 10 UNSIGNED_16	Ten words beginning at word 100 in CVIM module 1

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

3.3.5 SLC Addressing Syntax and Supported Data Types

This section describes legal data table addresses and data types for the 1747 SLC 500 family.

The structure of a legal data table address for the SLC differs depending on the table that is being addressed. The SLC supports the following data table sections:

I - Input
O - Output
S - Status
B - Binary
T - Timer
C - Counter
R - Control
N - Integer
F - Floating-Point
A - ASCII
ST - String

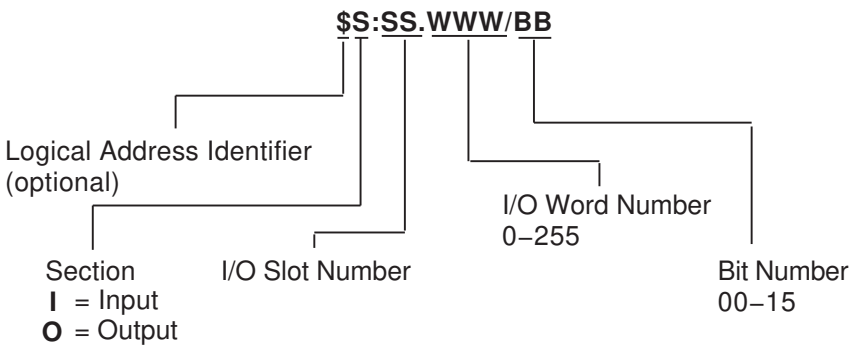
Note

ASCII, ASCII string and floating-point are available only on SLC 5/03 Series C with OS301 and SLC 5/04 Series A with OS400.

Figure 3–40 shows the structure of a legal I/O data table address for the SLC.

Table 3–26 shows examples of legal I/O data table addresses for the SLC.

Figure 3–40 SLC I/O Data Table Address Format



Note

Reading and writing of I/O data table addresses is supported only for SLC-5/03 and SLC-5/04 processors.

Table 3–26 Example SLC I/O Data Table Addresses

Example Address	MMS Type[1]	Description
I:4	UNSIGNED_16	Input image I/O slot 4
I:5.1	INTEGER_16	Input image I/O slot 5 word 1
I:10.1/15	BIT_STRING	Bit 31 of I/O slot 10
$O:8/1	BIT_STRING	Bit 1 of I/O slot 8
I:0	Array of 10 UNSIGNED_16	Ten words beginning at I/O slot 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–41 shows the structure of a legal Status data table address for the SLC.

Table 3–27 shows examples of legal Status data table addresses for the SLC.

Figure 3–41 SLC Status Data Table Address Format

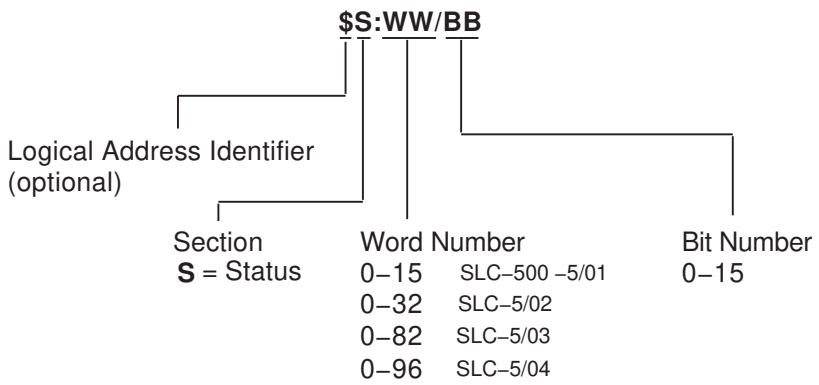


Table 3–27 Example SLC Status Data Table Addresses

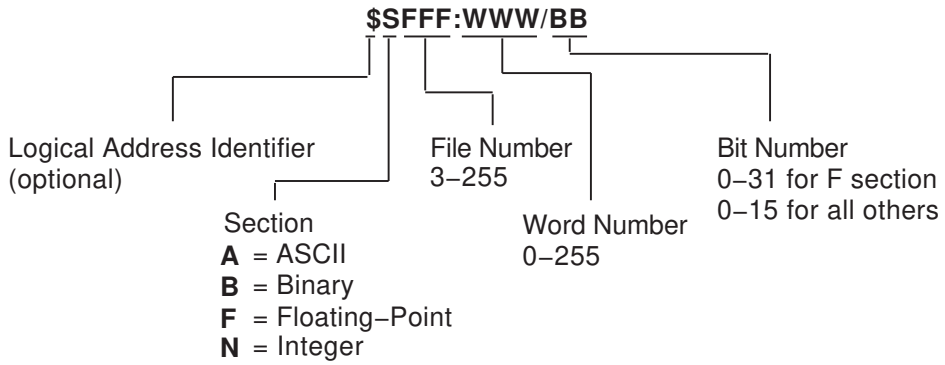
Example Address	MMS Type[1]	Description
S:20	UNSIGNED_16	Word 20 of the status file
S:12	INTEGER_16	Word 12 of the status file
$S:5/1	BIT_STRING	Bit 1 of word 5 of the status file
S:0	Array of 10 UNSIGNED_16	Ten words beginning at word 0

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

Figure 3–42 shows the structure of a legal ASCII, Binary, Floating-Point or Integer data table address for the SLC.

Table 3–28 shows examples of legal ASCII, Binary, Floating-Point and Integer data table addresses for the SLC.

Figure 3–42 SLC ASCII, Binary, Floating-Point and Integer Data Table Address Format



Note: The Binary file deaults to file 3 if none is specified.
The Floating–point file defaults to file 8 if none is specified.
The Integer file defaults to file 7 if none is specified.
The ASCII and Floating–point files are valid on SLC–5/03 OS301, SLC–5/04 OS400 andSLC–5/05 processors only.

Note

ASCII and floating-point are available only on SLC 5/03 Series C with OS301 and SLC 5/04 Series A with OS400.

Table 3–28 Example SLC ASCII, Binary, Floating-Point and Integer Data Table Address Format

Example Address	MMS Type[1]	Description
$N20:0	INTEGER_16	Word 0 of integer file 20
F39:19	FLOAT[2]	Word 19 of floating-point file 39
A10:20	VISIBLE_STRING[3]	Word 20 of ASCII file 10
A11:0	OCTET_STRING[3]	Word 0 of ASCII file 11
B91:20/10	BIT_STRING[4]	Bit 10 of word 20 of binary file 91
N7:20	UNSIGNED_32[5]	Words 20 and 21 of integern file 7
B:33	UNSIGNED_16	Word 33 of the default binary file (3)
N:91	UNSIGNED_16	Word 91 of the default integer file (7)
F:3	FLOAT	Word 3 of the default floating-point file (8)
B/155	BIT_STRING	Bit 11 of word 9 in the default binary file (3)

[1]Arrays are supported. Structures are **not** supported. Variable size arrays are **not** supported.

[2]Floating-point section only.

[3]ASCII section only.

[4]Bit writes that span more than one element are **not** supported.

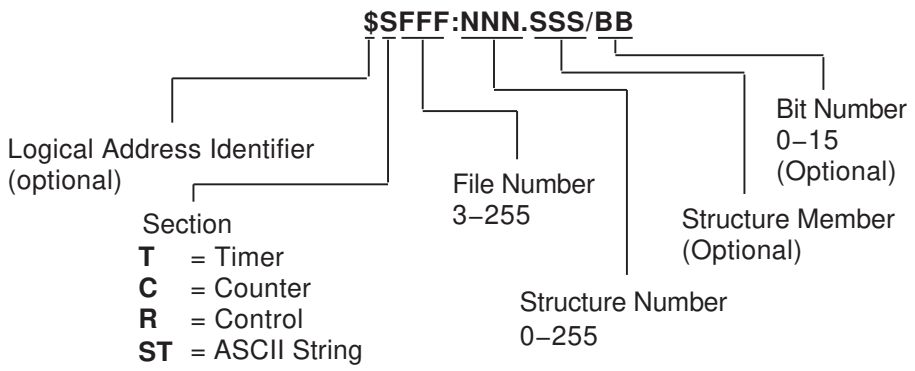
[5]Word 20 is the least significant word. Word 21 is the most significant word in the longword. Both signed and unsigned longwords are supported.

Figure 3–43 shows the structure of a legal Timer, Counter, Control or ASCII String data table address for the SLC.

Figure 3–44 shows a SLC Timer structure.
Figure 3–45 shows a SLC Counter structure.
Figure 3–46 shows a SLC Control structure.
Figure 3–47 shows a SLC ASCII String structure.

Table 3–29 shows examples of legal Timer, Counter, Control and ASCII String table addresses for the SLC.

Figure 3–43 SLC Timer, Counter, Control and ASCII String Data Table Address Format



Note: If a structure member is not specified, then the whole structure is addressed. Structure members are described in following figures.

Note: ST is aviilable only on SLC–5/03 OS301 and SLC–5/04 OS400 processores only.

Note: The Timer file defaults to file 4 if not specified.
The Counter file defaults to file 5 if not specified.
The Control file defaults to file 6 if not specified.

Note

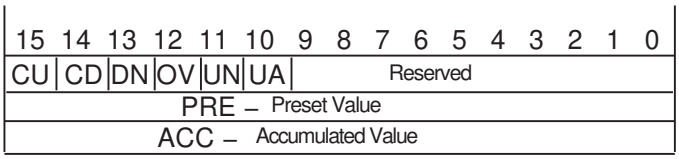
ASCII string is available only on SLC 5/03 Series C with OS301 and SLC 5/04 Series A with OS400.

Figure 3–44 SLC Timer Structure

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	TT	TD	Reserved					Used to maintain timer accuracy							
PRE – Preset Value															
ACC – Accumulated Value															

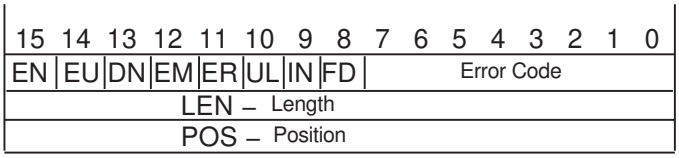
EN – Timer Enabled
TT – Timer Timing
TD – Timer Done

Figure 3–45 SLC Counter Structure



CU – Count Up Enable
CD – Count Down Enable
DN – Counter Done
OV – Counter Overflow
UN – Counter Underflow
UA – Update Accum. Value

Figure 3–46 SLC Control Structure



EN – Enable
EU – Enable Unloading
DN – Done
EM – Empty
ER – Error
UL – Unload
IN – Inhibit Comparison
FD – Found

Figure 3–47 SLC ASCII String Structure

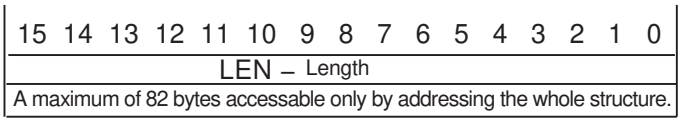


Table 3–29 Example SLC Timer, Counter, Control and ASCII String Data Table Address Formats

Example Address	MMS Type[1]	Description
$T20:0.ACC	UNSIGNED_16	Timer 0 in file 20 accumulated value
C91:3.PRE	INTEGER_16	Counter 91 in file 3 preset value

[1]Arrays are **not** supported for structures other than ASCII strings.

(continued on next page)

Table 3–29 (Cont.) Example SLC Timer, Counter, Control and ASCII String Data Table Address Formats

Example Address	MMS Type[1]	Description
T:20.CTL/15	BIT_STRING	Timer 20 in the default timer file (4) Enable bit[2]
R6:0.UL	BIT_STRING	Control 0 in file 6 Unload bit[2]
T4:20.TD	BIT_STRING	Timer 20 in file 4 Timer Done bit[2]
R:1.LEN	INTEGER_16	Control 1 in the default control file (6) Length value
ST16:0.LEN	INTEGER_16	String 0 in file 16 Length value
ST16:2	VISIBLE_STRING[3]	String 2 in file 16
ST16:3	OCTET_STRING[3]	String 3 in file 16
T:5	Structure[4] UNSIGNED_16 UNSIGNED_16 UNSIGNED_16	All of timer 5 in file 4 Control Preset value Accumulated value
R8:10	Structure[4] UNSIGNED_16 INTEGER_16 INTEGER_16	All of control 10 in file 8 Control Length Position

[1]Arrays are **not** supported for structures other than ASCII strings.

[2]Control bits should be treated as read only and modified only under PLC program control. Modifying these bits from a computer can give unpredictable results.

[3]Only Varying length strings are supported.

[4]Write operations are **not** allowed to structures other than to ASCII strings.

A

I/O Error Messages

This appendix contains returned values and associated error messages for the low level I/O routines.

allocerror, An error occurred when allocating an OpenVMS port,

Explanation: Error. An error occurred when allocating an OpenVMS port. The allocation is done when a connection is initiated with the VMD.

User Action: An error message detailing the error should have been logged. Examine the error to determine the proper corrective action.

allocstatus, Status for port XX returned from SYS$ALLOC, XX,

Explanation: Error. An error occurred when allocating a port.

User Action: Examine the error message to determine the cause of the error and take appropriate action based on the error message.

assignerror, An error occurred when assigning a channel to an OpenVMS port,

Explanation: Error. An error occurred when assigning a channel to an OpenVMS port. The assignment is done when a connection is initiated with the VMD.

User Action: An error message detailing the error should have been logged. Examine the error to determine the proper corrective action.

assignstatus, Status for port XX returned from SYS$ASSIGN, XX,

Explanation: Error. An error occurred when assigning a channel.

User Action: Examine the error message to determine the cause of the error and take appropriate action based on the error message.

connecterror, Error connecting to Internet address,

Explanation: Error. An error occurred when attempting to connect to an Internet address.

User Action: Examine the log file (BASEstar Open) or stderr to determine the error that occurred. Then take corrective action based on the error that was found.

dataoverrun, Data overrun occurred on the line,

Explanation: Error. A data overrun occurred on the line.

User Action: Use a slower baud rate or switch to a port that has faster hardware.

dont, TELNET server refused to allow TELNET option,

Explanation: Error. The TELNET server responded with "DON'T" to a request, "WILL", for the requested TELNET option.

User Action: Enable the apropriate option on the TELNET server or substitute a server that supports requested option.

dontdobin, TELNET server refused to allow binary data reception,

Explanation: Error. The TELNET server responded with "DON'T" to a request, "WILL", for the host to send binary data (TELNET option RFC 856).

User Action: Enable binary mode transmission on the TELNET server or substitute a server that supports binary mode transmission/reception.

econnrefused, The attempt to connect was rejected for port XX,

Explanation: Error. An attempt to connect resulted was rejected by the remote server.

User Action: If using TELNET, check the terminal server to ensure that the listener port has been properly configured and enabled.

ehostunreach, No route to host is present for port XX,

Explanation: Error. An attempt to connect to a TELNET port resulted in an unreachable status being returned.

User Action: Make sure that the IP address and port number are properly defined and that a gateway is defined on the host.

enetunreach, No route to the network or host is present for port XX,

Explanation: Error. An attempt to connect to a TELNET port resulted in an unreachable status being returned.

User Action: Make sure that the IP address and port number are properly defined and that a gateway is defined on the host.

etimedout, Connection timed out for port XX,

Explanation: Error. An attempt to connect to a port or socket timed out.

User Action: If using LAT, make sure that the port is defined in the LAT Control Program and that noone else is using the port. Then attempt to reconnect to the port. If using TELNET, make sure that the IP address and port number are defined properly on the terminal server and that the listener port is enabled. Also ensure that a gateway is defined if the host and the terminal server are not on the same subnet.

getdvistatus, Status for port XX returned from SYS$GETDVI, XX,

Explanation: Error. An error occurred when getting device information.

User Action: Examine the error message and the port definition to determine the cause of the error and take appropriate action based on the error message.

inaddrerror, Error when converting an Internet address to binary form,

Explanation: Error. An error occurred when convertng an Internet address

User Action: Examine the Internet address to determine the error and then correct the Internet address.

invalidport, An invalid port definition was specified,

Explanation: Error. A port definition that was defined in ODS is not valid. Either the port is not a valid TT type device or device characteristics could not be obtained for the port.

User Action: Examine the log file (BASEstar) or stderr to determine the error status and the function which detected the error. Take action based on the error message.

invalidspeed, An invalid baud rate was specified,

Explanation: Error. A baud rate that was defined in ODS is not valid.

User Action: Check the documentation to determine valid baud rates for the hardware platform being used. Then modify the baud rate in ODS to a valid value.

latcode, Status for port XX returned from SYS$QIOW (IO$_TTY_PORT | IO$M_LT_CONNECT) = X,

Explanation: Error. An error occurred when connecting to a LAT port.

User Action: Examine the error code to determine the cause of the error and take appropriate action based on the error code.

laterror, An error occurred when connecting to a LAT port,

Explanation: Error. An error occurred when connecting to a LAT port. The connection is done when a connection is initiated with the VMD.

User Action: An error message detailing the error should have been logged. Examine the error to determine the proper corrective action.

latmodeminv, Invalid attempt to enable modem signals on a LAT port,

Explanation: Error. LAT ports do not support setting of the "modem" ODS parameter to "TRUE".

User Action: Change the port to a non-LAT port or change the "modem" parameter to "FALSE", "0" or "OFF".

latstatus, Status for port XX returned from SYS$QIOW (IO$_TTY_PORT | IO$M_LT_CONNECT), XX,

Explanation: Error. An error occurred when connecting to a LAT port.

User Action: Examine the error message to determine the cause of the error and take appropriate action based on the error message.

negfailed, TELNET option negotiation failed,

Explanation: Error. An I/O error occurred when attempting to negotiate TELNET options.

User Action: Examine the log file (BASEstar Open) or stderr to determine the cause of the error and then take corrective action based on the error found.

noalttypeahead, "/ALTYPEAHD" must be specified for the given port,

Explanation: Error. The DAS requires that /ALTYPEAHD be specified for the port that is being used to communicate to the device. ALTYPEAHD helps to ensure that characters transmitted by the device are not lost when the host is busy.

User Action: Use DCL to "SET TERM/PERM/ALTYPEAHD port_name" on the port being used to communicate to the device(s).

nolicense, No software license,

Explanation: Error. The DAS requires a software license.

User Action: Install the license PAK for the DAS.

parity, Parity error,

Explanation: Error. A parity error occurred on the port.

User Action: Check parity of the device and make sure it matches of the host. (LAT and TELNET parity is set in the terminal server.) If parity is set correctly, then verify that all cabling is grounded and terminated properly.

portsyncerror, Unable to change file descriptor to blocking,

Explanation: Error. An error occurred when using ioctl( ) to place a file descriptor in blocking mode.

User Action: Examine the log file (BASEstar Open) or stderr to determine the error and then take corrective action based on the error found.

portsyncstatus, Unable to change file descriptor to blocking for port XX, errno = X,

Explanation: Error. An error occurred when using ioctl( ) to place a file descriptor in blocking mode.

User Action: Look up the error and then take corrective action based on the error found.

portsyntax, Improper syntax for intenet address,

Explanation: Error. There is an error in the syntax of the port address.

User Action: Check the syntax of the address and correct it to be a well formed internet address.

recverror, Read (recv) error occurred on the port,

Explanation: Error. An error was found when reading from the port. Examine the log file (BASEstar Open) or stderr to determine the errno returned by the read.

User Action: Examine the log file (BASEstar Open) or stderr to determine the errno returned by the recv call. Then take action based on the error found.

recvstatus, Read (recv) error on port XX, errno = X,

Explanation: Error. An error was found when reading from the port. The error number for the error is given in the message.

User Action: Look up the error and take action based on the error found.

socketerror, Error in obtaining a socket for port,

Explanation: Error. An error was found when issuing a 'socket' call to obtain a socket for a TCP/IP connection.

User Action: Examine the log file (BASEstar Open) or stderr to determine the errno returned by the socket call. Then take action based on the error found.

socketstatus, Error in obtaining a socket for port XX, errno = X,

Explanation: Error. An error was found when issuing a 'socket' call to obtain a socket for a TCP/IP connection.

User Action: Look up the error and take action based on the error found.

telnetiac, TELNET command XX received on port XX,

Explanation: Error. A TELNET command was received when in binary mode. In binary mode no TELNET commands are expected.

User Action: No user action should be necessary. The DAS will abort the connection and reset binary mode when it reconnects.

ttbreakerror, An error occurred when sending a break to a port,

Explanation: Error. An error was returned when sending a break to the port.

User Action: Examine the log file (BASEstar Open) or stderr to determine the error that occurred. Then take action based on the error found.

ttcharerror, An error occurred when defining TT characteristics,

Explanation: Error. An error occurred when setting up TT characteristics. The TT characteristics are set when a connection is initiated with the VMD.

User Action: An error message detailing the error should have been logged. Examine the error to determine the proper corrective action.

ucxseterror, Error returned from SYS$QIO - IO$_SETMODE,

Explanation: Error. An error occurred when attempting to set the socket address. This UCX call is equivalent to the C Socket socket ( ).

User Action: Examine the log file (BASEstar Open) or stderr to determine the error that occurred. Then take corrective action based on the error that was found.

ucxsetstatus, Status for port XX returned from SYS$QIO - IO$_SETMODE, XX,

Explanation: Error. An error occurred when attempting to set the socket address. This UCX call is equivalent to the C Socket socket ( ).

User Action: Examine the error status and look up the error to determine the cause and corrective action to take.

unexpectiac, An unexpected "IAC" was received while in TELNET binary mode,

Explanation: Error. An "IAC" (Is A command) character was received after a TELNET line was negotiated to transmit and receive in binary mode. In binary mode, the only expected TELNET control command is the IAC character itself.

User Action: No action should be necessary. The DAS will abort the connection if this should occur and then attempt to renegotiate binary transmission/reception when reconnecting.

vmsconnectstatus, Status for port XX returned from SYS$QIO - IO$_ACCESS, XX,

Explanation: Error. An error occurred when attempting to connect to an Internet address. This UCX call is equivalent to the C connect ( ).

User Action: Look up the error status to determine the error that occurred. Then take corrective action based on the error that was found.

vmsreaderror, QIO read error on port,

Explanation: Error. An error was found when reading from an OpenVMS port. Examine the log file (BASEstar Open) or stderr to determine the status returned by the read.

User Action: Look up the error and take action based on the error found.

vmsreadstatus, QIO read error on port XX, XX,

Explanation: Error. An error occurred when reading data from an OpenVMS port.

User Action: Examine the error message and take corrective action based on the error message.

vmsrecverror, Error returned from SYS$QIO - IO$_READVBLK,

Explanation: Error. An error was found when reading from the port. This UCX call is equivalent to the C Socket recv ( ).

User Action: Examine the log file (BASEstar Open) or stderr to determine the error that occurred. Then take action based on the error found.

vmsrecvstatus, Status for port XX:X returned from SYS$QIO - IO$_READVBLK, XX,

Explanation: Error. An error was found when reading from the port. This UCX call is equivalent to the C Socket recv ( ).

User Action: Look up the error status to determine the error that occurred. Then take corrective action based on the error that was found.

vmsttbreakstatus, Status for port XX returned from SYS$QIO - IO$_SETMODE, XX,

Explanation: Error. An error was returned when sending a break to the port.

User Action: Look up the error status to determine the error that occurred. Then take corrective action based on the error that was found.

vmsttcharstatus, Status for port XX returned from SYS$QIOW (IO$_SETMODE), XX,

Explanation: Error. An error occurred when setting port characteristics.

User Action: Look up the error number to determine the cause of the error and take appropriate action based on the error code.

wont, TELNET server refused to accept TELNET option,

Explanation: Error. The TELNET server responded with "WON'T" to a request, "DO", for the requested TELNET option.

User Action: Enable the apropriate option on the TELNET server or substitute a server that supports the requested option.

wontdobin, TELNET server refused to send binary data,

Explanation: Error. The TELNET server responded with "WON'T" to a request, "DO", for the server to send binary data (TELNET option RFC 856).

User Action: Enable binary mode transmission on the TELNET server or substitute a server that supports binary mode transmission/reception.

wrongdevice, An invalid device type was detected for a port,

Explanation: Error. A device type other than a terminal was detected for the indicated port.

User Action: Modify the ODS port definition to use a valid TT type device.

B

DAS Error Messages

This appendix contains returned values and associated error messages specific to the DAS for Allen-Bradley Data Highway.

aborting, The connection to the device is being aborted,

Explanation: Error. The connection to the device is being aborted by an error or by user request. No more requests will be processed by the DAS until the connection is restored.

User Action: If the abort was caused by a device error, take action based on the error that occurred. If the abort was a result of a user request. Reconnect to the device before attempting further operations.

addresstoolong, Variable address string too long,

Explanation: Error. A variable address that is being defined is too long. An address must be 30 characters of less in length.

User Action: Redefine the variable using a shorter address.

apptypeunsupp, The given application type is not supported for this DAS,

Explanation: Error. The application type that was specified in the variable definition is not supported by this DAS.

User Action: Change the application type to a valid type for this DAS.

badaddress, PLC XX sent data for address XX, but this address does not exist on the host,

Explanation: Error. A variable was received from a PLC and the address did not match any variables defined on the host.

User Action: Define a variable with the correct address on the host or change the address of the variable being sent from the PLC.

badbcdvalue, Bad BCD value,

Explanation: Error. A request was made to convert from or to a BCD number and the number was not a valid BCD value.

User Action: Modify the value to be converted into a valid BCD number.

badmodel, The given PLC model is not supported,

Explanation: Error. The model number that was specified in the device definition is not supported by this DAS.

User Action: Change the model number to one that is supported by this DAS.

datatoolarge, Application data too large for the data size of the variable,

Explanation: Error. The actual size of a string variable being sent to a device is larger than that defined by the variable definition. An actual size that is smaller than the variable definition is allowed but not one that is larger.

User Action: Define a variable with the correct size as the data being sent or change the size of the data being sent by the application.

datatoosmall, PLC data too small for the data size of the variable,

Explanation: Error. A variable was received from a PLC and the address matched a host variable, but the size of the data defined for the variable is larger than the size of the data actually received.

User Action: Define a variable with the correct size as the data being sent or change the size of the data being sent by the PLC.

datatrunc, PLC data from XX to XX too large for variable XX,

Explanation: Error. A variable was received from a PLC and the address matched a host variable, but the size of the data defined for the variable is smaller than the size of the data actually received. The data returned is truncated to fit into the variable.

User Action: Define a variable with the correct size as the data being sent or change the size of the data being sent by the PLC.

duptrans, Duplicate transaction received on line XX, transaction number = !UW,

Explanation: Warning. A duplicate transaction number was received on the data highway line. In most cases this is probably caused by too long a delay in responding to a message sent by the KE/KF card to the host. The duplicate message is discarded by the host.

User Action: Increase the KE/KF card timeout value by modifying the KE_TIMEOUT line attribute in ODS.

exterrsts, PLC XX sent extended status error = X'XX',

Explanation: Error. An Allen-Bradley extended status error occurred.

User Action: Consult the Allen-Bradley PLC communications manual to determine the cause of the error and take appropriate action based on the error code.

extrachars, Extraneous characters before message start on line XX,

Explanation: Warning. Extra characters were received before the DLE STX on the line. The most likely cause is the remains of a timed out previous transaction. It is also possible that there is noise on the line.

User Action: If this message persists, then check shielding, length of cabling or other factors that may result in a noisy communications signal. Correct any problems that are found.

extsts06, Ext Sts X06 - address doesn't point to something usable,

Explanation: Error. An attempt was made to read or write to a PLC address that isn't a valid PLC data table address for this PLC.

User Action: Define a variable with an address that matches actual PLC memory or reconfigure the PLC memory to include the requested address.

extsts07, Ext Sts X07 - file is wrong size,

Explanation: Error. An attempt was made to write to a PLC address that is beyond the end of a PLC data table address.

User Action: Define a variable with an address that matches actual PLC memory or reconfigure the PLC memory to include the requested address.

extsts0a, Ext Sts X0A - transaction size plus word address is too large,

Explanation: Error. An attempt was made to read from a PLC address that is beyond the end of a PLC data table address.

User Action: Define a variable with an address and size that matches actual PLC memory or reconfigure the PLC memory to include the requested address.

extsts11, Ext Sts X11 - illegal data type,

Explanation: Error. An attempt was made to read from a PLC address that does not exist in the PLC or that is not valid for the table being accessed.

User Action: Define a variable with an address that matches the actual PLC memory or reconfigure the PLC memory to include the requested address.

extsts12, Ext Sts X12 - invalid parameter or invalid data,

Explanation: Error. An invalid command or data was sent to the PLC.

User Action: Submit an SPR describing the circumstances that caused the error.

extstserr, PLC extended status error,

Explanation: Error. An Allen-Bradley extended status error occurred.

User Action: Use a data scope to capture the error code and take appropriate action based on the error code.

internalerror, Internal error,

Explanation: Error. An internal error occurred in the DAS.

User Action: Submit an SPR describing the circumstances that caused the error.

invaddresstype, Invalid address type,

Explanation: Error. A variable that is being defined is defined as an illegal address type. Only unconstrained address types are allowed.

User Action: Redefine the variable using the unconstrained address type.

invalidaddress, The specified PLC address is not supported by this DAS,

Explanation: Error. An address was sent by a PLC but the address is not supported by this DAS.

User Action: Reprogram the PLC and specify a valid address for this DAS. Consult the DAS documentation to determine a valid address.

invalidbitwrite, Bit writes cannot span more than one element,

Explanation: Error. An attempt was made to do a bit write that spanned more than one element. The DAS supports writes to only one element at a time.

User Action: Change the starting bit or change the number of bits that are being written so that the write spans only one element.

invalidcommand, Invalid command code received in a message from the PLC,

Explanation: Error. A message was received from the PLC, but the command code used by the PLC in the write command is not supported by the DAS.

User Action: Reprogram the PLC to send the correct command code.

invaliddrop, Invalid drop time specified in DH_TELNET ODS entry,

Explanation: Error. The drop time specified is not a valid number. The drop time must be a number greater than 0.

User Action: Use ODS to enter a valid drop time attribute.

invalidenqretries, Invalid ENQ retries number in the DH_LINE ODS entry,

Explanation: Error. The ENQ retries attribute must be between 0-20.

User Action: Use ODS to enter an ENQ retries number between 0-20.

invalidfunction, Invalid function code received in a message from the PLC,

Explanation: Error. A message was sent from the PLC, but the function code used by the PLC in the write command is not supported by the DAS.

User Action: Reprogram the PLC to send the correct function code.

invalidgroup, Variables in a group must all belong to the same file,

Explanation: Error. A group read was attempted without all the variables in the group belonging to the same file type and file number or a subelement of a structure file was included in the group.

User Action: Examine the members of the group to determine which variable does not belong, remove that variable from the list and retry the group read.

invalidgroupmem, Variable XX in a group read for device XX does not belong,

Explanation: Error. A group read was attempted without all the variables in the group belonging to the same file type and file number.

User Action: Remove the incorrect variable from the list and retry the group read.

invalidipaddress, Invalid IP_ADDRESS ODS attribute for the given line,

Explanation: Error. The IP_ADDRESS must be less than 50 characters in length.

User Action: Enter an IP address that is 50 characters or less in length.

invalidkestation, Invalid KE station number in the DH_LINE ODS entry,

Explanation: Error. AB station numbers must be an octal number between 0-377.

User Action: Use ODS to enter a station number between 0-377 octal.

invalidmodem, Invalid modem attribute specified in the DH_LINE ODS entry,

Explanation: Error. The modem attribute must be one of TRUE, ON, YES, 1 or FALSE, OFF, NO, 0.

User Action: Use ODS to enter a valid modem attribute.

invalidnakretries, Invalid NAK retries number in the DH_LINE ODS entry,

Explanation: Error. The NAK retries attribute must be between 0-20.

User Action: Use ODS to enter a NAK retries number between 0-20.

invalidparity, Invalid parity specified in the DH_LINE ODS entry,

Explanation: Error. The parity defined must be either EVEN or NONE.

User Action: Use ODS to enter a valid parity (EVEN, NONE).

invalidport, Invalid port name specified in the DH_LINE ODS entry,

Explanation: Error. Port names must be 10 characters or less.

User Action: Use ODS to enter a port name that is less than 10 characters.

invalidportnum, Invalid PORT_NUMBER ODS attribute for the given line,

Explanation: Error. The PORT_NUMBER must be less than 10 characters in length.

User Action: Enter a PORT_NUMBER address that is 10 characters or less in length.

invalidprobe, Invalid probe time specified in DH_TELNET ODS entry,

Explanation: Error. The probe time specified is not a valid number. The probe time must be a number greater than 0.

User Action: Use ODS to enter a valid probe time attribute.

invalidspeed, Invalid speed (baud rate) specified in the DH_LINE ODS entry,

Explanation: Error. Baud rates from 110 to 19200 are supported for the AB DAS.

User Action: Use ODS to enter a speed from 110 to 19200.

invalidstation, Invalid station number in the DH_PATH ODS entry,

Explanation: Error. AB station numbers must be an octal number between 0-377.

User Action: Use ODS to enter a station number between 0-377 octal.

invalidstructread, Variables in a group must read the entire structure,

Explanation: Error. A group read was attempted without all the variables in the group reading a complete structure element. For example, a timer element is being read, and an accumulator subelement was specified in the address.

User Action: Examine the variables in the group and specify an address for the entire structure for all members of the group.

invalidtimeout, Invalid timeout value in the DH_LINE ODS entry,

Explanation: Error. An invalid timeout value was entered for the timeout value in ODS. Timeouts must be between 2 and 6 seconds.

User Action: Use ODS to enter a timeout value between 2-6.

invalidtrace, Invalid trace attribute specified in the DH_LINE or DH_PATH ODS entry,

Explanation: Error. The trace attribute must be one of TRUE, ON, YES, 1 or FALSE, OFF, NO, 0.

User Action: Use ODS to enter a valid trace attribute.

iotimeout, AB Data Highway I/O timeout on line XX,

Explanation: Error. The PLC began to respond but the host timed out when attempting to read the complete response from the PLC. The response could be part of a read or a write operation.

User Action: Check all cabling to make sure that it is properly shielded, grounded and seated. Verify that the communication parameters are set up properly on both PLC and the host.

linenoise, Noise encountered on data highway communications line XX,

Explanation: Error. An unexpected character was received on the data highway communications line. The DAS ignores the character received.

User Action: Check all cabling to make sure that it is properly shielded, grounded and seated.

locerrsts, PLC XX sent local status error = X'XX',

Explanation: Error. An Allen-Bradley local status error occurred.

User Action: Consult the Allen-Bradley PLC communications manual to determine the cause of the error and take appropriate action based on the error code.

locsts02, Loc Sts X02 - link layer timed out or received a NAK,

Explanation: Error. A local error occurred when communicating with the KE/KF card. This error is usually caused by the not being at the address specified in the STATION ODS attribute.

User Action: Check the PLC station ID in ODS and on the PLC to make sure that they match.

locstserr, PLC local status error,

Explanation: Error. An Allen-Bradley local status error occurred.

User Action: Examine the log file (BASEstar Open) or stderr to determine the exact error status returned. Then consult the Allen-Bradley PLC communications manual to determine the cause of the error and take appropriate action based on the error found.

mmsaddrtypeunsupp, The given MMS type is unsupported for the given address,

Explanation: Error. The MMS type is not supported for the variable address given.

User Action: Modify the MMS type to one supported for the given address or modify the address to one supported for the MMS type.

mmstypeunsupp, The given MMS type is not supported for this DAS,

Explanation: Error. The MMS type that was specified in the variable definition is not supported by by this DAS.

User Action: Change the MMS type to a valid type for this DAS.

msgtooshort, A message received from the PLC was too short,

Explanation: Error. A message was received from the PLC, but the message couldn't be decoded because it didn't contain enough data. This error is most likely caused by improper programming of the PLC or by communication errors.

User Action: Check the PLC program for errors and check that all communications plugs are properly seated and all communications cabling is not shorted, not broken and is properly terminated and shielded.

nak, Bad checksum - NAK sent on line XX, transaction number = !UW,

Explanation: Warning. A bad checksum was detected, so a NAK was sent in response to the message. A bad checksum is most likely the result of noisy communications.

User Action: If NAK messages persist, then check shielding, length of cabling or other factors that may result in a noisy communications signal. Correct any problems that are found.

neterr, AB Data Highway Network error occurred,

Explanation: Error. A network error on the Allen-Bradley Data Highway occurred.

User Action: Examine the network status in the returned IOSB to determine the specific network error that occurred and take action based on that error.

noenqretries, No "ENQ_RETRIES" ODS entry for the given line,

Explanation: Error. The AB DAS requires an ENQ_RETRIES attribute defined in ODS. The ENQ_RETRIES attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the ENQ_RETRIES attribute for this line. Use a valid number when defining the ENQ_RETRIES attribute.

noipaddress, No "IP_ADDRESS" ODS entry for the given line,

Explanation: Error. The AB DAS requires an IP_ADDRESS attribute defined in ODS. The IP_ADDRESS attribute is defined for the DH_TELNET class within ODS.

User Action: Use the DH_TELNET object class within ODS to define the IP_ADDRESS attribute for this line. Use a valid IP address for this attribute.

nokestation, No "KE_STATION" ODS entry for the given line,

Explanation: Error. The AB DAS requires a KE_STATION attribute defined in ODS. The KE_STATION attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the KE_STATION attribute for this line. Use a valid number when defining the KE_STATION attribute.

noketimeout, No "KE_TIMEOUT" ODS entry for the given line,

Explanation: Error. The AB DAS requires a KE_TIMEOUT attribute defined in ODS. The KE_TIMEOUT attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the KE_TIMEOUT attribute for this line. Use a valid number when defining the KE_TIMEOUT attribute.

nolicense, No software license,

Explanation: Error. The AB DAS requires a software license.

User Action: Install the license PAK for the AB DAS.

noline, No "/LINE=" ODS entry for the given application simple name,

Explanation: Error. The AB DAS requires a /LINE= attribute defined for the ODS entry corresponding to the application simple name entry for the VMD. The /LINE= attribute is defined for DH_PATH object class within ODS.

User Action: Use the DH_PATH object class within ODS to define the application simple name for this VMD. Use a valid line name when defining the /LINE= attribute.

nomodem, No "MODEM" ODS entry for the given line,

Explanation: Error. The AB DAS requires a MODEM attribute defined in ODS. The MODEM attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the MODEM attribute for this line. Modem control is either enabled (TRUE, ON, YES, 1) or disabled (FALSE, OFF, NO, 0).

nonakretries, No "NAK_RETRIES" ODS entry for the given line,

Explanation: Error. The AB DAS requires a NAK_RETRIES attribute defined in ODS. The NAK_RETRIES attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the NAK_RETRIES attribute for this line. Use a valid number when defining the NAK_RETRIES attribute.

noparity, No "PARITY" ODS entry for the given line,

Explanation: Error. The AB DAS requires a PARITY attribute defined in ODS. The PARITY attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the PARITY attribute for this line. Use a valid parity (EVEN, NONE) when defining the PARITY attribute.

noport, No "PORT" ODS entry for the given line,

Explanation: Error. The AB DAS requires a PORT attribute defined in ODS. The PORT attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the PORT attribute for this line. Use a valid port name when defining the PORT attribute.

noportnumber, No "PORT_NUMBER" ODS entry for the given line,

Explanation: Error. The AB DAS requires a PORT_NUMBER attribute defined in ODS. The PORT_NUMBER attribute is defined for the DH_ TELNET class within ODS.

User Action: Use the DH_TELNET object class within ODS to define the PORT_NUMBER attribute for this line. Use a valid IP port number for this attribute.

noresponse, No response from the PLC within the timeout period,

Explanation: Error. A message was sent to the PLC and was acknowledged, but no confirmation message was received by the host in the required timeframe.

User Action: Ensure that the timeout value in the line parameters is large enough based on the data highway network load.

nospeed, No "SPEED" ODS entry for the given line,

Explanation: Error. The AB DAS requires a SPEED attribute defined in ODS. The SPEED attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the SPEED attribute for this line. Use a valid baud rate when defining the SPEED attribute.

nostation, No "STATION" ODS entry for the given application simple name,

Explanation: Error. The AB DAS requires a STATION attribute defined for the ODS entry corresponding to the application simple name entry for the VMD. The STATION attribute is defined for DH_PATH object class within ODS.

User Action: Use the DH_PATH object class within ODS to define the application simple name for this VMD. Use a valid data highway station number when defining the STATION attribute.

notimeout, No "TIMEOUT" ODS entry for the given line,

Explanation: Error. The AB DAS requires a TIMEOUT attribute defined in ODS. The TIMEOUT attribute is defined for the DH_LINE object class within ODS.

User Action: Use the DH_LINE object class within ODS to define the TIMEOUT attribute for this line. Use a valid number when defining the TIMEOUT attribute.

notrace, No "TRACE" ODS entry,

Explanation: Error. The AB DAS requires a TRACE attribute defined in ODS. The TRACE attribute is defined for the DH_PATH object class and for the DH_LINE object class within ODS.

User Action: Use the DH_LINE and DH_PATH object classes within ODS to define the TRACE attribute for this line. Trace is either enabled (TRUE, ON, YES, 1) or disabled (FALSE, OFF, NO, 0).

parsefailed, The parsing of an ASCII address failed,

Explanation: Error. When defining a variable, the parsing of the ASCII address failed. Either the syntax or the value of a portion of the address is incorrect.

User Action: Consult the user documentation to determine a valid address for the variable being defined and redefine the variable using the correct syntax.

rcvthreaddied, Receiver thread died for line XX, status = X,

Explanation: Error. Receiver thread exited with an error.

User Action: Take action based on the error code.

rcvthreadexcep, Receiver thread for line XX caught an exception, status = X,

Explanation: Error. Receiver thread caught an exception. The exception that was caught is in this message.

User Action: Examine the exception that was caught and take action based on the error code.

remerrsts, PLC XX sent remote status error = X'XX',

Explanation: Error. An Allen-Bradley remote status error occurred.

User Action: Consult the Allen-Bradley PLC communications manual to determine the cause of the error and take appropriate action based on the error code.

remsts60, Rem Sts X60 - function disallowed due to command protection selection,

Explanation: Error. The PLC is in the wrong protection mode for the command being attempted.

User Action: Change the protection mode to one appropriate for the command.

remsts70, Rem Sts X70 - processor is in program mode,

Explanation: Error. The PLC is in the wrong processor mode for the command being attempted.

User Action: Change the processor mode to one appropriate for the command.

remstserr, PLC remote status error,

Explanation: Error. An Allen-Bradley remote status error occurred.

User Action: Examine the log file (BASEstar Open) or stderr to determine the exact error status returned. Then consult the Allen-Bradley PLC communications manual to determine the cause of the error and take appropriate action based on the error found.

reqpending, Conclude rejected because of a pending read/write request,

Explanation: Error. The conclude request was rejected because either a read or a write request was currently pending.

User Action: Either abort the connection or wait for the current request(s) to complete before retrying the conclude.

retriesexceeded, The NAK retry count was exceeded,

Explanation: Error. When trying to communicate to the device, the NAK retry count was exceeded. The NAK retry being exceeded means that the message being sent to the device was garbled during transmission and did not match the checksum.

User Action: Check cabling to ensure there are no shorts or breaks. Check/replace KE/KF card.

retriesexcline, The NAK retry count was exceeded for line XX,

Explanation: Error. When trying to communicate to the device, the NAK retry count was exceeded. The NAK retry being exceeded means that the message being sent to the device was garbled during transmission and did not match the checksum.

User Action: Check cabling to ensure there are no shorts or breaks. Check/replace KE/KF card.

structwriteunsup, Structure writes are not supported,

Explanation: Error. Writes to structure variables are not supported. Reads of structure variables are supported, but not writes.

User Action: Write only to elements of the structure.

timeout, AB Data Highway timeout when reading from the PLC,

Explanation: Error. The PLC did not respond when attempting to read a response from the PLC. The response could be part of a read or a write operation.

User Action: Check all cabling to make sure that it is properly shielded, grounded and seated. Verify that the communication parameters are set up properly on both PLC and the host. Verify that the correct station address has been selected on the PLC and on the host.

timeoutreq, AB Data Highway request timeout on line XX,

Explanation: Error. The PLC did not respond when attempting to read a response from the PLC. The response could be part of a read or a write operation.

User Action: Check all cabling to make sure that it is properly shielded, grounded and seated. Verify that the communication parameters are set up properly on both PLC and the host. Verify that the correct station address has been selected on the PLC and on the host.

timeoutresp, AB Data Highway response timeout on line XX,

Explanation: Error. The PLC did not respond when attempting to read a response from the PLC. The response was part of a write indication confirmation.

User Action: Check all cabling to make sure that it is properly shielded, grounded and seated. Verify that the communication parameters are set up properly on both PLC and the host. Verify that the correct station address has been selected on the PLC and on the host.

tmrthreadcancel, Timer thread was cancelled,

Explanation: Error. Timer thread was cancelled.

User Action: Restart the OMNI application.

tmrthreadexcep, Timer thread caught an exception, status = X,

Explanation: Error. Timer thread caught an exception. The exception code that was caught is in the message.

User Action: Examine the exception that was caught and take action based on the error code.

transnotfound, Matching transaction number not found for data highway message on line XX, station XX, transaction number = !UW,

Explanation: Error. A response was received from the PLC but there wasn't found a matching request transaction number to the transaction number in the response.

User Action: Check all cabling to make sure that it is properly shielded, grounded and seated. If the transaction had previously timed out, take steps to reduce Data Highway network traffic or the load on the host or both.

transtimedout, No response from VMD XX, transaction number = !UW,

Explanation: Error. A message was sent to the PLC and was acknowledged, but no confirmation message was received by the host in the required timeframe.

User Action: Ensure that the timeout value in the line parameters is large enough based on the data highway network load.

unexpcommand, Unexpected command received from PLC XX. Expected command code was X'XX'; received command code was X'XX',

Explanation: Error. An unexpected command was received in a message from the PLC.

User Action: Reprogram the PLC to send the correct command code.

unexpfunction, Unexpected function received from PLC XX. Expected function code was X'XX'; received function code was X'XX',

Explanation: Error. An unexpected function code was received in a message from the PLC.

User Action: Reprogram the PLC to send the correct function code.

unsupported, The requested function is not supported by the DAS,

Explanation: Error. An attempt was made to execute an OMNI function that is not supported by this DAS.

User Action: Consult the user documentation to determine which functions are supported by the DAS and limit the OMNI functions used to those supported by the DAS.

variablenotfound, Variable not found with the specified address,

Explanation: Error. A variable was received from a PLC and the address did not match any variables defined on the host.

User Action: Define a variable with the correct address on the host or change the address of the variable being sent from the PLC.

wrongindication, An indication was received from an unknown PLC - line XX, station XX,

Explanation: Error. An indication was received from a PLC station address, but the station is not known by the DAS.

User Action: Create a VMD with the station address of the PLC sending data to the DAS, or reprogram the PLC that is sending the data so that it is no longer sends data to the DAS.

Index

A

Abort function, 3–2
Abort indication, 3–11
Accessing DAS functions, 3–1
Address syntax, 3–12
  PLC-2, 3–12
  PLC-3, 3–12
  PLC-5, 3–19
  PLC-5/250, 3–29
  SLC, 3–44

C

CLI
  accessing, 3–1
Conclude function, 3–2
Configuration
  editing configuration files, 2–8
  ODS registrations, 2–8
  protocol profile, 2–10
  VMD definitions, 2–11
Configuration files, 2–13
Configuring ports, 2–16
Configuring TELNET, 2–17
Connect function, 3–2

D

DAS
  See Device access software
Data table addressing
  PLC-3, 3–13
  PLC-5, 3–19
  SLC, 3–44
Data types, 3–12
Device access software
  accessing functions, 3–1
  address syntax, 3–12
  description of, 1–1
  functions, 1–2
  installation of, 2–1
  installation requirements, 2–1
  overview, 1–1
  supported client functions, 3–2
  supported data types, 3–12
  supported devices, 1–2
Device access software (cont'd)
  supported functions, 1–2, 3–1
  supported server functions, 3–2
  using, 3–1
Device communications
  cabling, 2–1
  overview, 1–1
  tracing, 2–18
Devices
  PLCs supported, 1–2

E

Extended addressing
  PLC-3, 3–17

F

Files, created during installation, 2–6
Functions, 3–1
  abort, 3–2
  conclude, 3–2
  connect, 3–2
  read data point, 3–5
  read variable, 3–2
  supported, 3–1
  write data point, 3–5
  write variable, 3–2

I

Indications
  abort, 3–11
  receiving, 3–8
  write, 3–8
Installation, 2–1
  files created, 2–6
  messages, 2–6
  procedure, 2–4
Installation requirements, 2–1
  disk space, 2–4
  hardware, 2–1
  software, 2–3

L

LAT, 2–16

M

Messages
  DAS error and return values, B–1
  I/O error and return values, A–1
  installation, 2–6
MMS data type, 3–12

O

ODS registrations, 2–8

P

Plant-floor equipment, setting up, 2–18
PLC-2
  address syntax, 3–12
  supported data types, 3–12
PLC-3
  address syntax, 3–12
  data table addressing, 3–13
  extended addressing, 3–17
  supported data types, 3–12
PLC-3 data table address
  ASCII, 3–16
  binary, 3–16
  counter, 3–15
  decimal, 3–16
  floating-point, 3–16
  high-order integer, 3–16
  I/O, 3–14
  integer, 3–16
  pointer, 3–17
  status, 3–16
  timer, 3–14
PLC-5
  address syntax, 3–19
  data table addressing, 3–19
  supported data types, 3–19
PLC-5/250
  address syntax, 3–29
  supported data types, 3–29
  supported modules, 3–29
PLC-5/250 address format
  adapter status, 3–41
  ASCII string, 3–34
  binary, 3–30
  block-transfer, 3–38
  block-transfer read/write control, 3–39
  control, 3–34
  counter, 3–34
  floating-point, 3–30
  I/O, 3–37
PLC-5/250 address format (cont'd)
  integer, 3–30
  internal storage, 3–42
  long integer, 3–30
  message control, 3–34
  PID control, 3–34
  shared data, 3–43
  status, 3–36
  timer, 3–34
PLC-5 data table address
  ASCII, 3–21
  ASCII string, 3–26
  binary, 3–21
  block-transfer control, 3–26
  control, 3–26
  counter, 3–26
  decimal, 3–21
  floating-point, 3–21
  I/O, 3–20
  integer, 3–21
  message control, 3–26
  PID control, 3–26
  SFC status, 3–26
  status, 3–20
  timer, 3–26
  token data, 3–26
Postinstallation tasks, 2–8
Problem reporting, 2–19
Product failure, 2–19
Protocol profile, 2–10

R

Read data point function, 3–5
Read variable function, 3–2

S

SLC
  address syntax, 3–44
  data table addressing, 3–44
  supported data types, 3–44
SLC data table address
  ASCII, 3–46
  ASCII string, 3–48
  binary, 3–46
  control, 3–48
  counter, 3–48
  floating-point, 3–46
  I/O, 3–44
  integer, 3–46
  status, 3–45
  timer, 3–48
Supported data types, 3–12
  PLC-2, 3–12
  PLC-3, 3–12
  PLC-5, 3–19
  PLC-5/250, 3–29

Supported data types (cont'd)
SLC, 3–44

T

TELNET, 2–17
binary transfer, 2–17
Trace
ODS configuration, 2–8
using, 2–18

V

VMD definitions, 2–11

W

Write data point function, 3–5
Write indication, 3–8
Write variable function, 3–2