CONTROL TECHNIQUES

Epsilon Ei DeviceNet Drive

Reference Manual

P/N 400501-08
Revision A2
Date: March 20,2002

EMERSON
Industrial Automation

Epsilon Ei DeviceNet Drive Reference Manual

P/N 400501-08
Revision A2
Date: March 20, 2002

Part Number: 400501-08

Revision A2

Date: March 2002

Printed in United States of America

This document has been prepared to conform to the current released version of the product. Because of our extensive development efforts and our desire to further improve and enhance the product, inconsistencies may exist between the product and documentation in some instances. Call your customer support representative if you encounter an inconsistency.

Customer Support

Control Techniques
12005 Technology Drive
Eden Prairie, Minnesota 55344-3620
U.S.A.

Telephone: (952) 995-8000 or (800) 397-3786

It is Control Techniques' goal to ensure your greatest possible satisfaction with the operation of our products. We are dedicated to providing fast, friendly, and accurate assistance. That is why we offer you so many ways to get the support you need. Whether it's by phone, fax or modem, you can access Control Techniques support information 24 hours a day, seven days a week. Our wide range of services include:

FAX	**(952) 995-8099**

You can FAX questions and comments to Control Techniques. Just send a FAX to the number listed above.

Website and Email	**www.emersonct.com**

Website: www.emersonct.com

Email: info@emersonct.com

If you have Internet capabilities, you also have access to technical support using our website. The website includes technical notes, frequently asked questions, release notes and other technical documentation. This direct technical support connection lets you request assistance and exchange software files electronically.

Technical Support	**(952) 995-8033 or (800) 397-3786**

Email: service@emersonct.com

Control Techniques' "Motion Made Easy" products are backed by a team of professionals who will service your installation. Our technical support center in Eden Prairie, Minnesota is ready to help you solve those occasional problems over the telephone. Our technical support center is available 24 hours a day for emergency service to help speed any problem solving. Also, all hardware replacement parts, if needed, are available through our customer service organization.

When you call, please be at your computer, with your documentation easily available, and be prepared to provide the following information:

- Product version number, found by choosing About from the **Help** menu
- The type of controller or product you are using

- Exact wording of any messages that appear on your screen
- What you were doing when the problem occurred
- How you tried to solve the problem

Need on-site help? Control Techniques provides service, in most cases, the next day. Just call Control Techniques' technical support center when on-site service or maintenance is required.

Training Services	**(952) 995-8000 or (800) 397-3786**

Email: training@emersonct.com

Control Techniques maintains a highly trained staff of instructors to familiarize customers with Control Techniques' "Motion Made Easy" products and their applications. A number of courses are offered, many of which can be taught in your plant upon request.

Application Engineering	**(952) 995-8000 or (800) 397-3786**

Email: applengr@emersonct.com

An experienced staff of factory application engineers provides complete customer support for tough or complex applications. Our engineers offer you a broad base of experience and knowledge of electronic motion control applications.

Customer Service (Sales)	**(952) 995-8000 or (800) 397-3786**

Email: customer.service@emersonct.com

Authorized Control Techniques distributors may place orders directly with our Customer Service department. Contact the Customer Service department at this number for the distributor nearest you.

Document Conventions

Manual conventions have been established to help you learn to use this manual quickly and easily. As much as possible, these conventions correspond to those found in other Microsoft® Windows® compatible software documentation.

Menu names and options are printed in bold type: the **File** menu.

Dialog box names begin with uppercase letters: the Axis Limits dialog box.

Dialog box field names are in quotes: "Field Name."

Button names are in italic: *OK* button.

Source code is printed in Courier font: `Case ERMS`.

In addition, you will find the following typographic conventions throughout this manual.

This	Represents
bold	Characters that you must type exactly as they appear. For example, if you are directed to type **a:setup**, you should type all the bold characters exactly as they are printed.
italic	Placeholders for information you must provide. For example, if you are directed to type *filename*, you should type the actual name for a file instead of the word shown in italic type.
ALL CAPITALS	Directory names, file names, key names, and acronyms.
SMALL CAPS	Non-printable ASCII control characters.
KEY1+KEY2 example: (Alt+F)	A plus sign (+) between key names means to press and hold down the first key while you press the second key.
KEY1,KEY2 example: (Alt,F)	A comma (,) between key names means to press and release the keys one after the other.

Note

For the purpose of this manual and product, “Note” indicates essential information about the product or the respective part of the manual.

Throughout this manual, the word “drive” refers to an Epsilon drive.

⚠ WARNING

“Warning” indicates a potentially hazardous situation that, if not avoided, could result in death or serious injury.

⚠ CAUTION

“Caution” indicates a potentially hazardous situation that, if not avoided, may result in minor or moderate injury.

CAUTION

“Caution” used without the safety alert symbol indicates a potentially hazardous situation that, if not avoided, may result in property damage.

Safety Instructions

General Warning

Failure to follow safe installation guidelines can cause death or serious injury. The voltages used in the product can cause severe electric shock and/or burns and could be lethal. Extreme care is necessary at all times when working with or adjacent to the product. The installation must comply with all relevant safety legislation in the country of use.

Qualified Person

For the purpose of this manual and product, a “qualified person” is one who is familiar with the installation, construction and operation of the equipment and the hazards involved. In addition, this individual has the following qualifications:

- Is trained and authorized to energize, de-energize, clear and ground and tag circuits and equipment in accordance with established safety practices.
- Is trained in the proper care and use of protective equipment in accordance with established safety practices.
- Is trained in rendering first aid.

Reference Materials

The following related reference and installation manuals may be useful with your particular system.

- *Epsilon Ei Drive Installation Manual* (P/N 400501-06)
- *Epsilon Ei and FM-2 Module Indexing Reference Manual* (P/N 400507-01)
- *FM-3 & FM-4 DeviceNet Module Reference Manual* (P/N 400508-03)
- *Epsilon and E Series Drive Parameters Reference Manual* (P/N 400504-01)
- *ODVA Specification Volumes I and II, Release 2.0*

Safety Considerations

Safety Precautions

This product is intended for professional incorporation into a complete system. If you install the product incorrectly, it may present a safety hazard. The product and system may use high voltages and currents, carry a high level of stored electrical energy, or are used to control mechanical equipment that can cause injury.

You should give close attention to the electrical installation and system design to avoid hazards either in normal operation or in the event of equipment malfunction. System design, installation, commissioning and maintenance must be carried out by personnel who have the necessary training and experience. Read and follow this safety information and instruction manual carefully.

Enclosure

This product is intended to be mounted in an enclosure that prevents access except by trained and authorized personnel and prevents the ingress of contamination. This product is designed for use in an environment classified as pollution degree 2 in accordance with IEC664-1. This means that only dry, non-conducting contamination is acceptable.

Setup, Commissioning and Maintenance

It is essential that you give careful consideration to changes to drive settings. Depending on the application, a change could have an impact on safety. You must take appropriate precautions against inadvertent changes or tampering. Restoring default parameters in certain applications may cause unpredictable or hazardous operation.

Safety of Machinery

Within the European Union all machinery in which this product is used must comply with Directive 89/392/EEC, Safety of Machinery.

The product has been designed and tested to a high standard. However the level of integrity offered by the product's control function – for example stop/start, forward/reverse and maximum speed – is not sufficient for use in safety-critical applications without additional independent channels of protection. All applications where malfunction could cause injury or loss of life must be subject to a risk assessment, and further protection must be provided where needed.

⚠WARNING

General warning
Failure to follow safe installation guidelines can cause death or serious injury. The voltages used in this unit can cause severe electric shock and/or burns, and could be lethal. Extreme care is necessary at all times when working with or adjacent to this equipment. The installation must comply with all relevant safety legislation in the country of use.

AC supply isolation device
The AC supply must be removed from the drive using an approved isolation device or disconnect before any servicing work is performed, other than adjustments to the settings or parameters specified in the manual. The drive contains capacitors which remain charged to a potentially lethal voltage after the supply has been removed. Allow at least 6 minutes for the Epsilon 205, 3 minutes for Epsilon 202/203 and 30 seconds for E Series drives after removing the supply before carrying out any work which may involve contact with electrical connections to the drive.

Products connected by plug and socket
A special hazard may exist where the drive is incorporated into a product which is connected to the AC supply by a plug and socket. When unplugged, the pins of the plug may be connected to the drive input, which is only separated from the charge stored in the bus capacitor by semiconductor devices. To avoid any possibility of electric shock from the pins, if they are accessible, a means must be provided for automatically disconnecting the plug from the drive (e.g., a latching contactor).

Grounding (Earthing, equipotential bonding)
The drive must be grounded by a conductor sufficient to carry all possible fault current in the event of a fault. The ground connections shown in the manual must be followed.

Fuses
Fuses or over-current protection must be provided at the input in accordance with the instructions in the manual.

Isolation of control circuits
The installer must ensure that the external control circuits are isolated from human contact by at least one layer of insulation rated for use at the applied AC supply voltage.

Table of Contents

Introduction

Purpose

This manual describes the Epsilon Ei DeviceNet (Ei-DN) Servo Drive and gives examples for connecting DeviceNet® with Epsilon Indexing drives. The reader should have knowledge of basic DeviceNet concepts. In addition, the user should be familiar with the functionality of the Epsilon Ei drive. The Ei-DN drive is capable of all functions of the Epsilon Ei, as described in the *Epsilon Ei Reference Manual* (P/N 400507-01). The Quick Start chapters of this manual offer specific examples of various messaging types as well as a programming reference that lists functional requirements for the Epsilon Ei-DN drive. For a list of definitions, consult the"Glossary" on page 75.

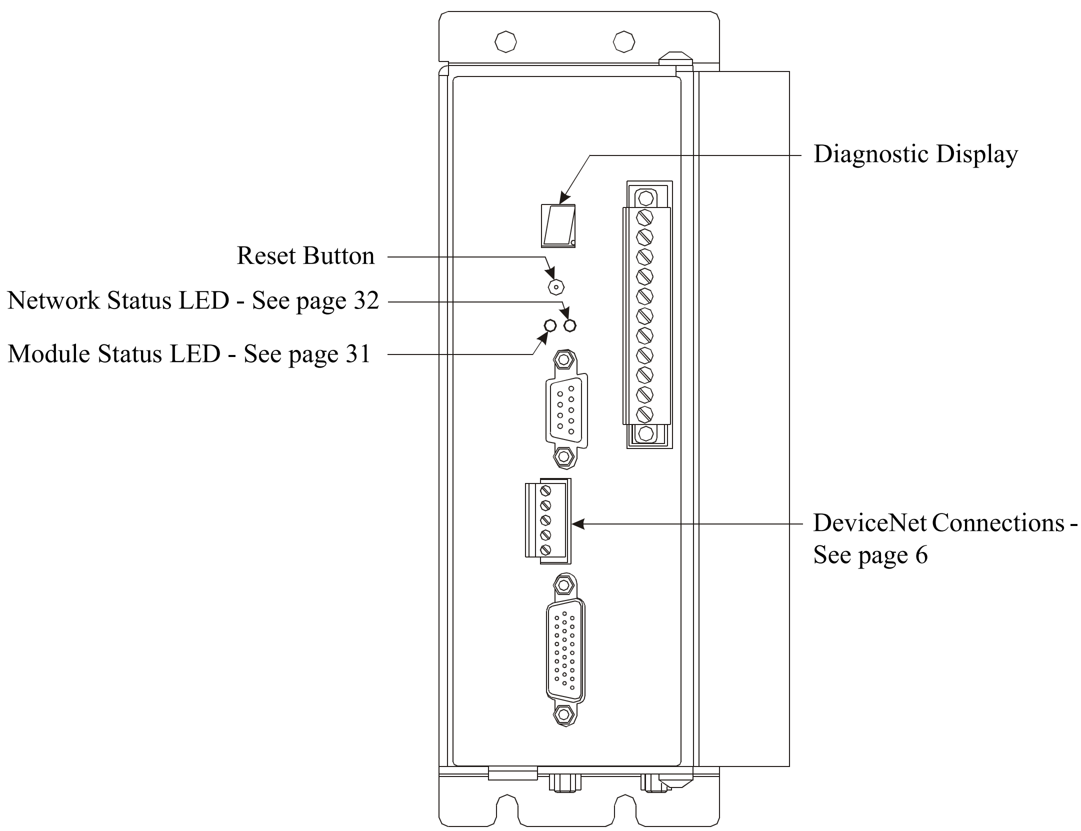


Figure 1: Epsilon Ei DeviceNet Drive - Ei-DN-205 Shown

The Epsilon Ei-DN configuration is easily set within PowerTools FM software. In PowerTools FM software, the user has the option to select between 8 predefined assembly blocks to be transferred using polled data (8 bytes input, 8 bytes output polled data length per drive). The Ei-DN additionally allows the master to set up an explicit messaging connection to access all user accessible parameters within the Epsilon drive through PLC logic.

The Epsilon Ei-DN drive has passed internal tests for conformance as a group 2 DeviceNet slave.

DeviceNet Overview

DeviceNet is a low-level network that provides connections between simple industrial devices (sensors, actuators) and higher level devices (industrial PLC's). DeviceNet standards and specifications are managed by the Open DeviceNet Vendors Assoc. (ODVA) which is an independent supplier organization that manages the DeviceNet specification and supports the worldwide growth of DeviceNet and Ethernet IP.

DeviceNet Communications Link

DeviceNet has two primary purposes:

1. Transport of control-oriented information associated with low-level devices.
2. Transport of other information that is indirectly related to the system being controlled, such as configuration parameters.

The list below presents a summary of the Physical/Media specific characteristics of DeviceNet:

1. Trunk-line/drop-line configuration (See Figure 2, "Master/Slave Relationship" on page 4.)
2. Support for up to 64 nodes
3. Node removal without severing the network
4. Simultaneous support for both network-powered (sensors) and self-powered (actuators) devices
5. Use of sealed or open-style connectors
6. Protection from wiring errors
7. Selectable data rates of 125k baud, 250k baud, and 500k baud
8. High current capability (up to 16 amps per supply)

9. Operation with off-the-shelf power supplies
10. Power taps that allow the connection of several power supplies from multiple vendors that comply with DeviceNet standards
11. Built-in overload protection
12. Power available along the bus: both signal and power lines contained in the trunk line

DeviceNet Messaging and Communications

Master/Slave Relationship

ODVA defines a DeviceNet Master as:

The device that gathers and distributes I/O data for the process controller. A Master contains a scan list it uses to poll slave devices.

ODVA defines a DeviceNet Slave as:

The slave returns I/O data to its Master when it is polled. With respect to the network, the Slave is a Group 2 Server or a Group 2 Only Server.

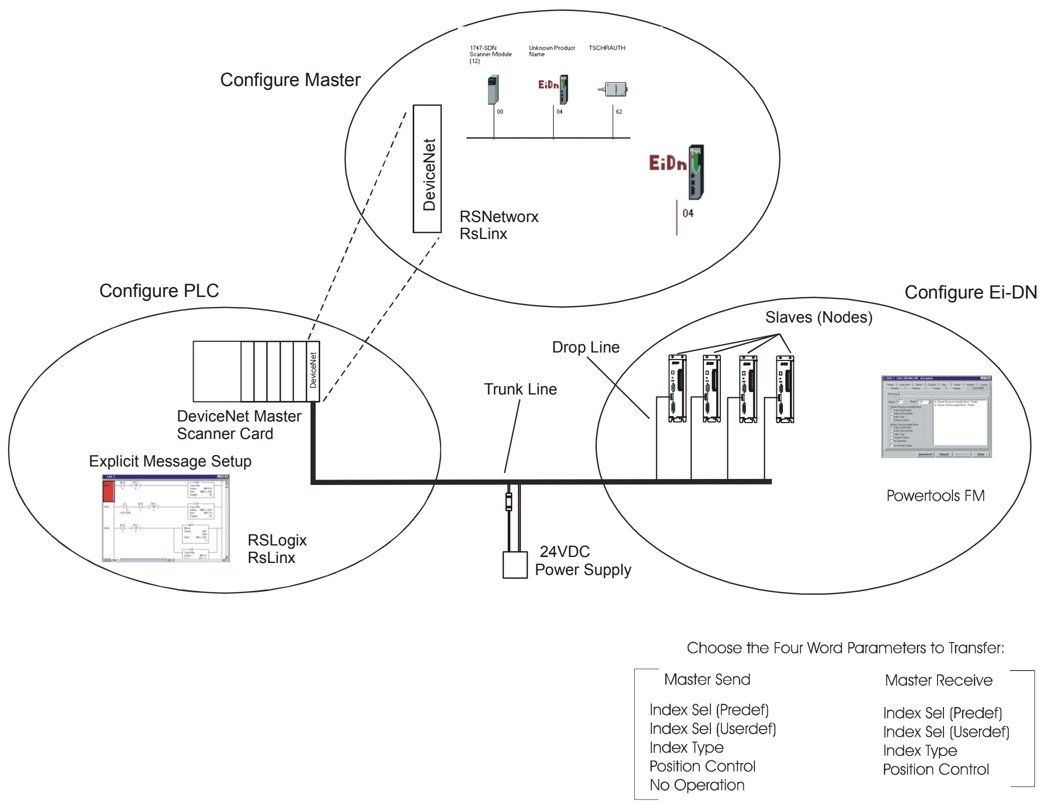


Figure 2: Master/Slave Relationship

Installation

Mechanical Installation

Follow the instructions for mechanical installation of an Epsilon Ei Drive as outlined in the *Epsilon Ei Installation Manual* (P/N 400501-06).

DeviceNet Hardware Components

The following components are necessary to design a DeviceNet cable system:

- Cables
- Nodes/Devices
- Connectors
- Power Supply
- Terminating Resistors

Cables

These cables can be obtained from the supplier of the DeviceNet Network Master. (For more details see the ODVA Specification Volumes I and II, Release 2.0.)

Thick Cable

The thick cable consists of two shielded pairs twisted on a common axis with a drain line in the center covered with an overall braid. The shield is commonly used as a trunk line when length is important.

Thin Cable

The thin cable is smaller and more flexible than the thick cable. It is commonly used for drop lines but can be used for shorter distances as a trunk line.

Nodes/Devices

A DeviceNet slave is any device that is addressable through DeviceNet and contains DeviceNet communications circuitry. DeviceNet Slaves must comply with the following:

A slave must be connected to the network by a tap and drop-line.

Slaves must be DeviceNet-compatible devices.

63 Slaves can be supported on one network.

Each Slave must be assigned a media access control identification number (MacID).

MacID’s on a single network must be unique.

Connectors

DeviceNet connectors can be either open-style (wires exposed) or sealed. These connectors can be obtained from the supplier of the DeviceNet Network Master. (For more details see the ODVA Specification Volumes I and II, Release 2.0.)

Power Supply Requirements

A separate 24 VDC power supply is needed for the DeviceNet network. The Ei-DN slave interface is powered using this "network power." Devices may take all of their required power off of the network per ODVA specifications.

The Ei-DN draws 25 mAmps from the DeviceNet power supply.

The power supply must have its own current limit protection.

Fuse protection must be provided for each segment of the cable system.

The power supply must be correctly sized to provide each device with its required power.

Power supplies should be distributed throughout the DeviceNet network to maintain a maximum of 4 Amps per trunk branch.

For more details, refer to the ODVA Specification Volumes I and II, Release 2.0.

Terminating Resistors

Terminating resistors are used to reduce the reflection of signals over the network. Each terminator must be 121 ohms and installed on both ends of the network between CAN_L (pin 2) and CAN_H (pin 3).

Electrical Installation

Physical Connections to the DeviceNet Network

A standard five wire configuration is used to connect the Ei-DN to the DeviceNet Network. A 24 Volt power supply should be connected between V+ and V-.

When multiple Ei-DN devices are present on the network, connection with all devices can be accomplished by either daisy chaining or by using a multiple tap box.

Pin Number	Name	Insulation Color
1	V-	Black
2	CAN_L	Blue
3	Drain	None
4	CAN_H	White
5	V+	Red

The connector provided for the DeviceNet connection is keyed: VT (pin 5) and should be wired to the keyed slot.

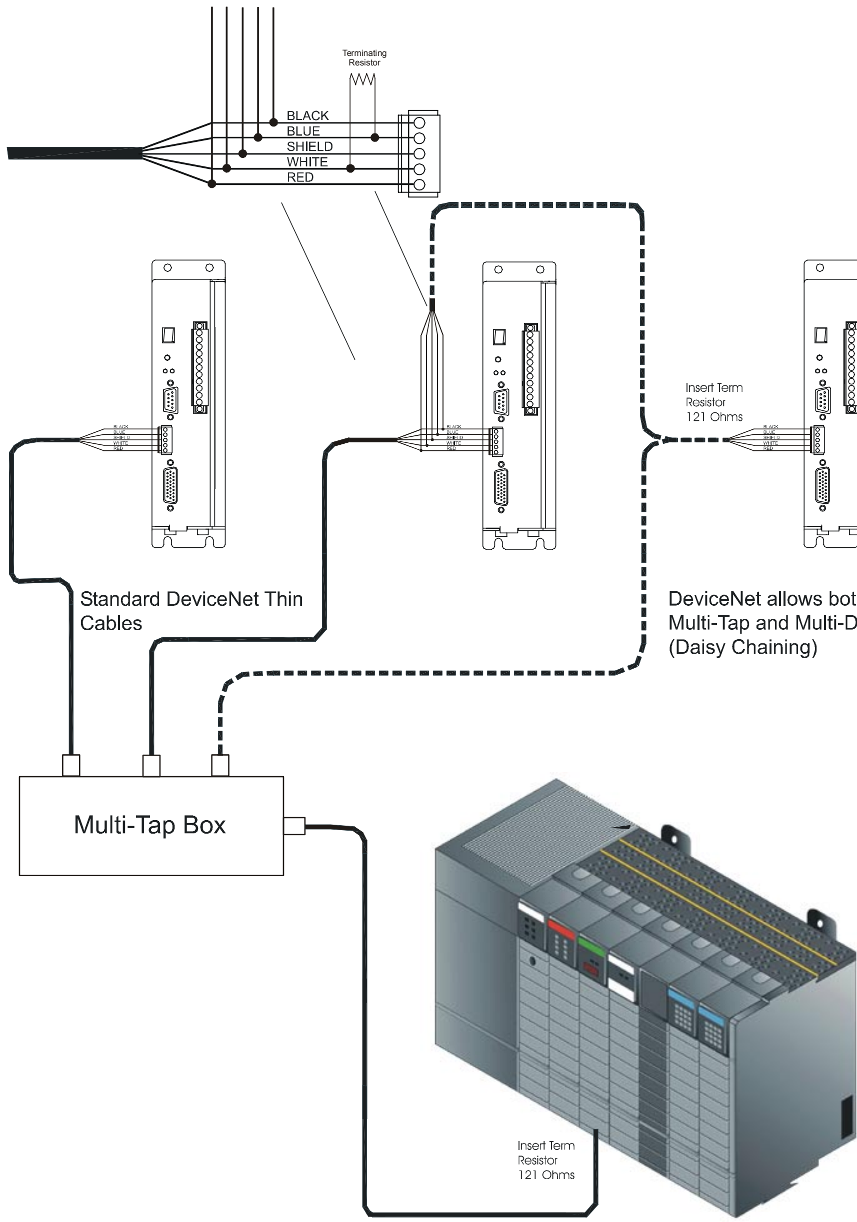


Figure 3: DeviceNet Wiring

Configuring the DeviceNet Network

Connection Types

The Epsilon Ei-DN DeviceNet interface was designed to communicate with a DeviceNet network in two ways. Polled I/O connections and Explicit Messaging connections can be used to access all read or read/write data from the Epsilon Ei-DN.

Polled I/O Connections (Implicit Messaging)

The Poll Command is an I/O Message that is transmitted by the Master. A Poll Command is directed towards a single, specific Slave (point-to-point). A Master must transmit a separate Poll Command Message for each one of its Slaves that is to be polled. The Poll Response is an I/O Message that a Slave transmits back to the Master.

A Polled Message contains I/O data that can be read by the PLC on every scan. Polled messages are used for high priority data and are typically used to transmit parameters, such as index initiates or position feedback.

Explicit Messaging

Explicit Messages are initiated from a user-created program inside a PLC or PC-based software program. In a PLC, these messages are sent and received using the PLC-specific explicit message setup. Using Explicit Messaging, all user parameters can be accessed in the Ei-DN drive. These messages allow parameters to be sent in messages that may take multiple scans of the PLC. Explicit Messages are lower priority than Polled I/O messaging connections and work well for transferring recipes or any 32 bit parameters to the drive.

Software Interface

This section discusses how to configure a DeviceNet Network with PowerTools FM software and discusses the parameters that appear on the tabs related to DeviceNet configuration. Other tabs are described in the *Epsilon Ei Indexing Drive and FM-2 Indexing Module Reference Manual* (P/N 400507-01).

DeviceNet Tab

The DeviceNet tab allows the user to configure DeviceNet parameters and watch the DeviceNet Parameters while online with the drive through PowerTools FM. The DeviceNet tab has two sub-tabs: DN Setup and Online.

The DN Setup tab allows the user to set up MacID, Baud Rate, choice of predefined assembly blocks, and host-mode enable as well as display the fixed assembly block word mapping.

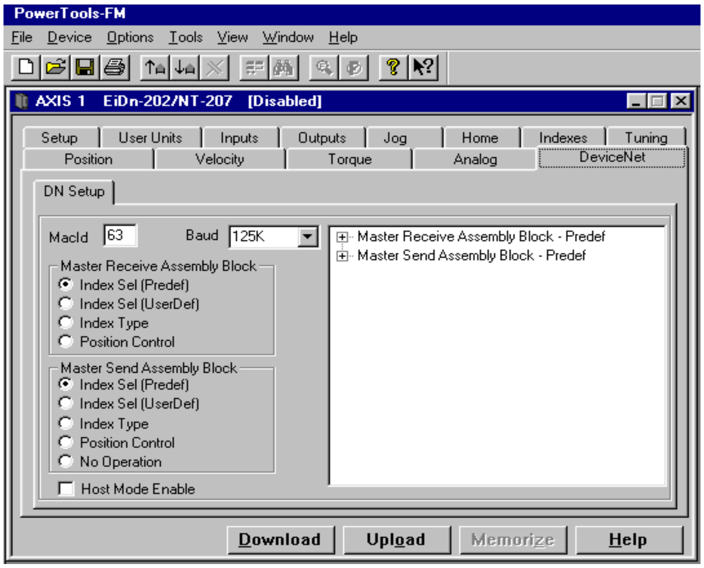


Figure 4: The DeviceNet Setup Tab

MacID

The MacID is the number assigned to a particular node. Every node on a DeviceNet network must have a unique MacID. The range is 0-63. The MacID is also read/write accessible with the one-touch configuration described in "Setting the Baud Rate and MacID Externally (One Touch)" on page 32.

Baud Rate

Three standard baud rates can be configured for the DeviceNet network: 125K, 250K, and 500K. The baud rate is also read/write accessible with the one touch configuration described in "Setting the Baud Rate and MacID Externally (One Touch)" on page 32.

Master Receive and Master Send Assembly Block Selections

In PowerTools FM the user is given the option to select from 9 predefined assembly blocks. Master Receive Assembly Block data translates to 8 bytes of Polled I/O data that will be transferred from the Ei-DN to the Master via DeviceNet, and Master Receive Assembly Block data translates to 8 bytes of polled I/O data that will be transferred from the Master to the Ei-DN.

Index Select Predefined (Index Sel (Predef))

Index Sel (Predef) is a fixed assembly block used for reading and writing basic functions to and from the Ei-DN. This fixed assembly block is the default selection in PowerTools FM and has many parameters premapped for quick setup and replacement. Index Sel (Predef) was created to keep everything simple and standard for the user. Bits that are generally used have been included in the assembly block for "no-work" access to DeviceNet I/O. The Index Sel (Predef) is set as default so the user can set Baud Rate, MacID, drive type and then start communication. The following tables display the data mapping for the Index Sel (PreDef) Assembly Blocks. See "Appendix" on page 71 for expanded versions of these tables.

Master ReceiveAssembly Block - Index Sel (Predef)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	End of Index	End of Chaining Count	End of Index Count	End of Index Motion	Travel Limit -	Travel Limit +	Reg Limit Distance Hit	Brake Release	Enable State	In - Motion	In + Motion	At Velocity	End of Home	Torque Limit	Fault	Drive OK
1	Reserved	Reserved	Reserved	Input Word Select Data Pointer (See page 14)					Registration Sensor 1 Status	Registration Sensor 2 Status	Home Sensor Status	Absolute Position Valid	Home Limit Dist Hit	Motion State Bit 2	Motion State Bit 1	Motion State Bit 0
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Index Sel (Predef)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Index Select Bit 3	Index Select Bit 2	Index Select Bit 1	Index Select Bit 0	Jog -	Jog +	Home Initiate	Start Index	DN Bit 7 MS	DN Bit 6 MS	DN Bit 5 MS	DN Bit 4 MS	DN Bit 3 MS	DN Bit 2 MS	DN Bit 1 MS	DN Bit 0 MS
1	Reserved	Reserved	Reset	Output Word Select Data Pointer (See page 15)					Enable	Stop	Define Home	Input Word Select Data Pointer (See page 14)				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Index Select User Defined (Index Sel (UserDef))

Index Sel (UserDef) is the fixed assembly block used when intiating multiples indexes as with a standard Ei drive. Assignments are made on the same screen as with the Index Sel (Predef). Index Sel (UserDef) is used for more advanced Ei applications. By allowing the user to set up his own DN bits, more advanced operations can be controlled right over the DeviceNet network. In addition to Home and Jog, any of the 16 indexes may be Initiated or changed. The following tables display the data mapping for the Index Sel (UserDef) Assembly Blocks. See "Appendix" on page 71 for expanded versions of these tables.

Master Receive Assembly Block - Index Sel (Userdef)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DN Bit 15 MR	DN Bit 14 MR	DN Bit 13 MR	DN Bit 12 MR	DN Bit 11 MR	DN Bit 10 MR	DN Bit 9 MR	DN Bit 8 MR	DN Bit 7MR	DN Bit 6MR	DN Bit 5 MR	DN Bit 4 MR	DN Bit 3 MR	DN Bit 2 MR	DN Bit 1 MR	DN Bit 0 MR
1	Reserved	Reserved	Reserved	Input Word Select Data Pointer (See page 14)					Index Select Bit 3	Index Select Bit 2	Index Select Bit 1	Index Select Bit 0	Enable State	Motion State Bit 2	Motion State Bit 1	Motion State Bit 0
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Index Sel (Userdef)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Index Select Bit 3	Index Select Bit 2	Index Select Bit 1	Index Select Bit 0	DN Bit 11 MS	DN Bit 10 MS	DN Bit 9 MS	DN Bit 8 MS	DN Bit 7 MS	DN Bit 6 MS	DN Bit 5 MS	DN Bit 4 MS	DN Bit 3 MS	DN Bit 2 MS	DN Bit 1 MS	DN Bit 0 MS
1	Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15)					Reserved	Reserved	Enable	Input Word Select Data Pointer (See page 14)				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Index Type

Index Type is the fixed assembly block used for reading and writing parameters when one index is running different index types (incremental, absolute, etc.). The Index Type Assembly Block is used to set up and initiate one index as a position controller but also provides the user with the additional functionality of the Epsilon Ei drive (i.e. home, jog, or most other I/O parameters). The Index Type Assembly Block gives the user the flexibility of being able to control the drive position by position but also gives Home and Jog functionality right within the DeviceNet I/O. See “Appendix” on page 71for expanded versions of these tables

Master Receive Assembly Block - Index Type

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DN Bit 7 MR	DN Bit 6 MR	DN Bit 5 MR	DN Bit 4 MR	DN Bit 3 MR	DN Bit 2 MR	DN Bit 1 MR	DN Bit 0 MR	Enable State	Absolute Position Valid	Home Limit Distance Hit	Reg Limit Distance Hit	End of Home	Torque Limit	Fault	Drive OK
1	Reserved	Reserved	Reserved	Input Word Select Data Pointer (See page 14)					Reserved	Reserved	Reserved	Travel Limit -	Travel Limit +	Motion State Bit 2	Motion State Bit 1	Motion State Bit 0
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Index Type

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DN Bit 7	DN Bit 6	DN Bit 5	DN Bit 4	DN Bit 3	DN Bit 2	DN Bit 1	DN Bit 0	Enable	Reset	Stop	Home Initiate	Index Type Bit 2	Index Type Bit 1	Index Type Bit 0	Start Index
1	Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15)					Jog Fast	Jog -	Jog +	Input Word Select Data Pointer (See page 14)				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Note

“Index Type” bit 1, 2, and 3 in Master Send Block determine the functionality if the specified index. As noted in the Drives Parameters Reference Manual (400504-01) the value of these bits reflect the following data: 000 Incremental Index, 001 Absolute Index, 010 Registration Index, 011 Rotary Plus, and 100 Rotary Minus.

Position Control

Position Control is the fixed assembly block designed using the “Position Controller” profile for DeviceNet as a guide. Parameters are transferred over the polled I/O connection. Using the position controller profile positions, velocities, accels, and decels can be written for a single index. The index can then be initiated. The Position Control object is used heavily when a central PLC is desired to control all of the Slaves. This type of configuration allows for quick parameter changes. The following tables display the data mapping for the Position Control Assembly Blocks. See “Appendix” on page 71 for expanded versions of these tables

Master Receive Assembly Block - Position Control

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Enable State	Valid Data = 1 Ignore Data = 0	Absolute Position Valid	Stop Input	Fault	End of Index Motion	Reserved	Trajectory Started
1	Reserved	Reserved	Reserved	Response Assembly Code (See page 16)					Command Error	Reserved	Trajectory Start Echo	Reserved	Reserved	CCW Hardware Limit (Travel Limit -)	CW Hardware Limit (Travel Limit +)	Drive OK
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Position Control

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Enable	Valid Data = 1 Ignore Data = 0	Reserved	Stop	Reserved	Absolute=0 Incremental=1	Reserved	Start Trajectory
1	Reserved	Reserved	Reserved	Response Assembly Code (See page 16)					Reserved	Reserved	Reserved	Command Assembly Code (See page 16)				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

No Operation

The No Operation Assembly Block is used when the user only needs to read back data. This assembly block allows the user to put any data into the corresponding PLC address without affect from the drive. This type of assembly block is used heavily in applications when the PLC is explicit messaging.

Master Send Assembly Block - No Operation

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0																
1																
2																
3																

Input Word Select Data Pointer

The Input Word Select Data Pointer may be used when a polled connection is established between the Ei-DN and the DeviceNet Master. Using the Input Data Pointer Bits, 32-bit parameters may be read from the Ei-DN by the PLC. Entering the code into the input select data pointer will place the corresponding parameter value within the two high words of the 4 word polled data packet. The Input Word Data Select Data Pointer is set by the master in word 1 of the Master Send Data.

Code	Description
0x00	No Operation
0x01	Position Feedback
0x02	Position Command
0x03	Velocity Feedback
0x04	Command Velocity
0x05	Reserved
0x06	Torque Command
0x07	Index Position
0x08	Index Velocity
0x09	Index Accel
0x0A	Index Decel
0x0B - 0x1D	Reserved
0x1E	Fault Status Bitmap
0x1F	Reserved

Output Word Select Data Pointer

The Output Word Select Data Pointer may be used when a polled connection is established between the Ei-DN and the DeviceNet master. Using the Output Word Select Data Pointer bits, 32-bit parameters may be read from the Ei-DN by the PLC. Entering the code into the Output Select Data Pointer will place the corresponding parameter value within the two high words of the 4 word polled data.

Code	Description
0x00	No Operation
0x01	Index Position Registers
0x02	Index Velocity Register
0x03	Index Acceleration Register
0x04	Index Deceleration Register
0x05	Index Dwell Time
0x06	Reserved
0x07	Torque Limit
0x08	Rotary Rollover Value
0x09	Following Error Limit
0x0A	Jog Velocity
0x0B	Jog Fast Velocity
0x0C	Jog Acceleration
0x0D	Jog Deceleration
0x0E	Home Velocity
0x0F	Home Acceleration
0x10	Home Deceleration
0x11	Home Specified Offset
0x12	End of Home Position
0x13	Home Limit Distance
0x14	Chaining Count
0x15	Index Count
0x16	Next Index
0x17	RAM to NVM
0x18 - 0x1F	Reserved

Command Assembly Code

The Command Assembly Code is used only when the position controller object is in use and the Ei-DN is online and connected. The Command Assembly Code gives the drive a 32-bit parameter based on the code it is set for. The parameter is placed in words 2 and 3 of the polled I/O.

Code	Description
0x00	No Operation
0x01	Index 0 Position
0x02	Index 0 Velocity
0x03	Index 0 Accel
0x04	Index 0 Decel
0x05 to 0x10	Reserved
0x11	Reserved
0x12	Reserved
0x13 to 0x1D	Reserved
0x1E	Reset “Set Bit 0 to 1”
0x1F	Reserved

Response Assembly Codes

The Response Assembly Code is used only when the position controller object is in use and the Ei-DN is online and connected. The Response Assembly Code gives the drive a 32-bit parameter based on the code it is set for. The parameter is placed in words 2 and 3 of the polled I/O.

Code	Description
0x00	No Operation
0x01	Actual Position
0x02	Commanded Position
0x03	Actual Velocity
0x04 to 0x1D	Reserved
0x1E	Fault Status Bitmap
0x1F	Reserved

Data Processing/Order of Operations for Fixed Assembly Blocks

Since the transfer of parameters over the DeviceNet network has the potential to transfer on the same scan of the PLC, the user needs to take special consideration of the order of bits sent. When the Master sends two or more bits in the same DeviceNet message, the bits get implemented in the Ei-DN in the following order:

- Get Master Send Data Pointer
- Process Master Send Data Block (update word information)
- Process Master Send Bits **
- Send Input data to Master Receive FAB

**The following list represents the priority/order of bit data processed (first to last) sent from the Master.

- Stop Bit (Held until clear)
- Home Bits (Initiate a home)
- Index Bits (Initiate an Index)
- Jog Bits (Held until clear)

** If individual bits are changed simultaneously within each of the above group, the order of processing shall follow:

1. Home Bits
    - Home Initiate
    - Define Home
2. Index Bits
    - Index Select
    - Index Initiate
3. Jog Bits
    - Jog +
    - Jog-
    - Jog Fast

Note

An Implicit Poll initiates motion, and an immediate response is sent. This response does not wait for motion to complete, i.e., an Index Initiate Command sent before the drive completes an index will not be acknowledged by the drive.

Host Mode Enable

Host Mode Enable allows the user to decide where the control will occur from the DeviceNet network. Host Mode Enable allows the user to ignore hardwire inputs. Conversely, non-host mode logically OR's the Inputs with the DeviceNet function.

Enabling the "Host Mode" gives the host (DeviceNet Master) exclusive control of the following parameters:

Index Select and Index Initiate
Jog +, Jog -, Jog Fast,
Home Initiate
Define Home
Brake Control and Release

All other functions will be logically OR'ed with hardwire and Modbus inputs when in Host mode.

Host mode is useful when access to the hardwire I/O needs to be switched from "DeviceNet only" to "DeviceNet I/O or Hardware I/O."

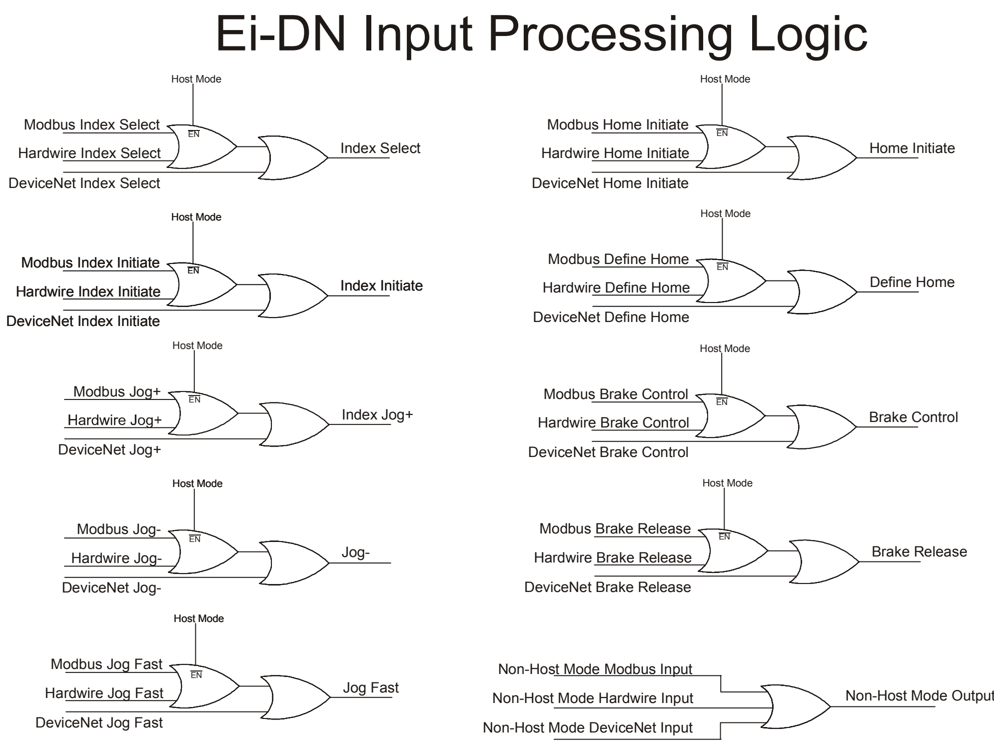


Figure 5: Ei-DN Input Processing Logic Diagram

DeviceNet Online Tab

The DeviceNet Online Tab is available to the user when PowerTools FM is online with the Ei-DN. This tab displays DeviceNet specific diagnostics and allows the user to view the status of the DeviceNet network.

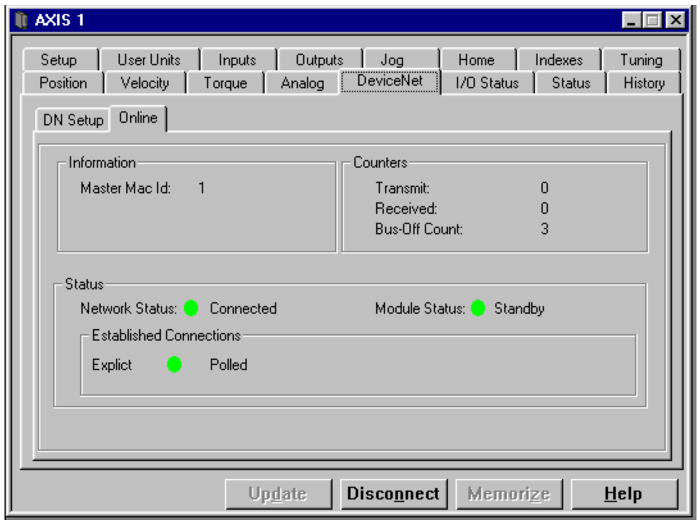


Figure 6: DeviceNet Online Tab

Information

Master MacID

The Master MacID is the MacID of the DeviceNet master to the Ei-DN.

Counters

Transmit/Receive Counter

The Transmit and Receive Counters note the successful data packet exchanges between the Ei-DN and its DeviceNet master.

Bus-Off Counter

The Bus-Off counter counts the number of bus failures in order to safeguard against a network that goes down. A Buss Off may occur due to errors in wiring, baud rate settings, or MacID errors on the network. Ten bus-offs in a row initiate the bus-off interrupt (major fault). This information is used for troubleshooting the Ei-DN.

Status

Network Status LED

Network Status vitual LED indicates whether the Ei-DN is communicating properly with the DeviceNet network. The table below will describe the current condition of this parameter.

Virtual LED State	Drive State	Indicates
Off	Not Powered/Not online	Device is not online: The device has not completed the Duplicate MacID test yet. The Device may not be powered, look at Module Status LED.
Flashing Green	Online, Not Connected	Device is online but has no connections in the established state. The device has passed the Duplicate MacID test and is online but has no established connections to a Master.
Green	Link OK Online, Connected	The device is online and has connections in the established state. This means the device is connected to a Master.
Flashing Red	Connection Time-Out	One or more I/O Connections are in the Timed-Out state.
Red	Critical Link Failure	Failed communication device. The device has detected an error that has rendered it incapable of communicating on the network (Duplicate MacID or Bus Off).

Module Status LED

Module Status virtual LED indicates whether the device is in standby or operational mode. Coinciding text will describe the current condition of this parameter.

Virtual LED State	Drive State	Indicates
Off	No Power	There is no power applied to the device.
Green	Device Operational	The device is operating in a normal condition.
Flashing Green	Device in Standby (The Device Needs Commissioning)	The device needs commissioning due to configuration missing, incomplete, or incorrect. The device may be in the standby state.
Flashing Red	Minor Fault	Recoverable Fault
Red	Unrecoverable Fault	The device has an unrecoverable fault, may need replacing
Flashing Red-Green	Device Self-Testing	The device is in self-test.

Established Connections

Polled/Explicit Messaging Virtual LED's

The virtual LED display for Polled and Explicit Messaging represents the status of these connections on the DeviceNet Network.

Virtual LED State	Indicates
Green	Connected state
Gray	Unconnected state

Input and Output Tab

DeviceNet Bits

DeviceNet bits are user-defined bits that are assigned within the inputs and outputs tabs. These bits are used with specific assembly blocks to transfer data via the polled connection. DeviceNet bit numbering is the same for all assembly blocks selected; therefore, DN bit 0 MR (Master Receive) under "Index Sel (User Def)" is the same as DN bit 0 MR (Master Receive) for "Index Type."

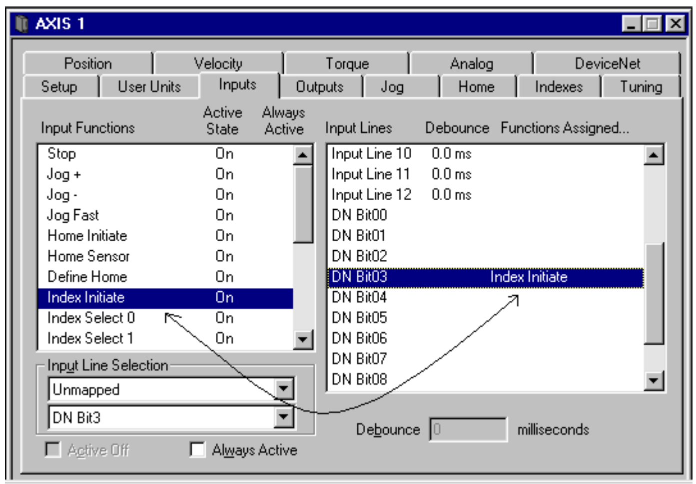


Figure 7: Inputs Tab

Multiple assignments may be made to the same DN bit, and the same I/O function may be mapped to a DN bit as well as to an I/O assignment. This allows the user to indicate drive status simultaneously to multiple devices via hardwire and DeviceNet. When multiple assignments are made, the DN bit is the logical OR condition at he assigned input function.

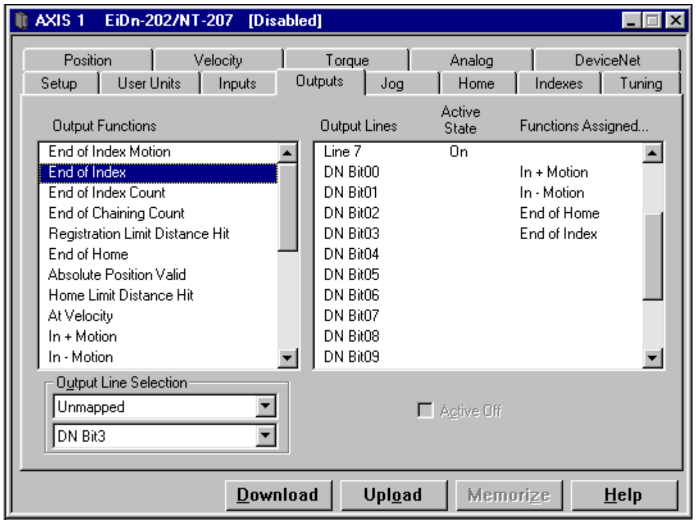


Figure 8: Outputs Tab

When an unsupported DN bit assignment is made, a warning message will pop up telling the user that the selected assembly block does not support that bit. When acknowledged, the assignment will be mapped to that bit anyway and will not be transferred via DeviceNet.

Faults and Diagnostic Display

PowerTools FM has implemented an extended set of fault and diagnostic parameters to aid in setup and monitoring of the DeviceNet Network. Figure 9 shows the faults and diagnostic parameters available to the user through PowerTools FM or using Explicit Messages on the DeviceNet network.

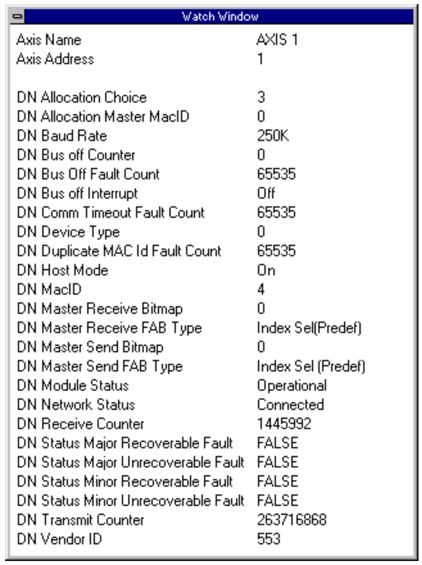


Figure 9: Watch Window

DN Allocation Choice (Connections Established)

Ei-DN **Modbus Address: 34003**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			BM16	WORD		No		RO

This parameter monitors the status of the established connections and displays the corresponding bitmap. Bit 0 Explicit Message, Bit1 Polled.

DN Allocation Master MacID

Ei-DN **Modbus Address: 34004**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-63		63	US16	UINT		No		RO

This parameter displays the MacID of the master used to control the Ei-DN. The DN Allocation Master MacID parameter defaults to 63 when no master is controlling it.

DN Baud Rate

Ei-DN **Modbus Address: 44002**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-2	Baud		ENM	WORD		Yes		RW

Three standard baud rates can be configured for the DeviceNet network: 125K, 250K, and 500K. The baud rate is also read/write accessible with the one touch configuration located directly on the drive.

Value	Baud
0	125K
1	250K
2	500K

DN Bus Off Counter

Ei-DN **Modbus Address: 34206**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-255	Counts		US16	UINT		No		RO

The Bus Off counter counts the number of bus failures in order to safeguard against a network that goes down. This information is used for troubleshooting the Ei-DN.

DN Bus Off Fault Count

Ei-DN **Modbus Address: 40719**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
$0\text{-}2^{31}\text{-}1$	Counts		US16	UINT		Yes		RO

This parameter indicates the total number of Bus Off Faults that have occurred on the DeviceNet network.

DN Bus Off Interrupt

Ei-DN **Modbus Address: 34205**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0, 1			ENM	Word		No		RO

Ten Bus-Off Faults in a row initiate the Bus-Off Interrupt (Major fault).

DN Comm Time-out Fault Count

Ei-DN **Modbus Address: 40717**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
$0\text{-}2^{15}\text{-}1$	Counts		US16	UINT		Yes		RO

This parameter indicates the total # of Communication Time-out Faults that have occurred on the DeviceNet network.

DN Device Type

Ei-DN **Modbus Address: 34002**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			US16	UINT		No		RO

DN Device Type indicates the ODVA (Open DeviceNet Vendors Association) definition for devices. The Ei-DN is set at a DN Device Type of 0.

DN Duplicate MacID Fault Count

Ei-DN **Modbus Address: 40718**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
$0\text{-}2^{15}\text{-}1$	Counts		US16	UINT		Yes		RO

The DN Duplicate MacID Fault Count indicates the total number of duplicate MacID faults that have occurred on the DeviceNet network.

DN FAB Master Receive Block

Ei-DN **Modbus Address: 34011-34014**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			US16	UINT		No		RO

The DN FAB Master Receive Block displays the polled data being transferred from the drive to the master via the Master Receive fixed assembly block (FAB).

Word	Modbus Address
0	34011
1	34012
2	34013
3	34014

DN FAB Master Send Block

Ei-DN **Modbus Address: 34007-34010**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			US16	UINT		No		RO

The DN FAB Master Send Block displays the polled data being transferred from the master to the drive via the Master Send fixed assembly block (FAB).

Word	Modbus Address
0	34007
1	34008
2	34009
3	34010

DN Host Mode

Ei-DN **Modbus Address: 44003**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0,1			ENM	Word		Yes		RW

Enabling the "Host Mode" gives the host (DeviceNet Master) exclusive control of the following Input functions:

- Index Select and Index Initiate
- Jog +, Jog -, Jog Fast
- Home Initiate
- Define Home
- Brake Control and Release

All other functions will be logically OR'ed with inputs and Modbus 3 when in Host mode.

Host mode is useful when access to the hardwire I/O needs to be switched from "DeviceNet only" to "DeviceNet I/O or Hardware I/O."

DN MacID

Ei-DN **Modbus Address: 44001**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-63			US16	UINT		Yes		RW

The DN MacID is the node address of the Ei-DN on the DeviceNet network. The range of this number is 0-63.

DN Master Receive Bitmap

Ei-DN **Modbus Address: 34006**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			US16	UINT		No		RO

This parameter returns the values for the selected master receive bits.

DN Master Receive FAB Type

Ei-DN **Modbus Address: 44005**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-3			ENM	Word		Yes		RW

The DN Master Receive FAB Type displays the chosen fixed assembly block configuration as shown.

Value	Master Receive FAB Type
0	Index Select (Predef)
1	Index Select (Userdef)
2	Index Type
3	Position Control

DN Master Send Bitmap

Ei-DN **Modbus Address: 34005**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			US16	UINT		No		RO

This parameter returns the values for the selected master send bits.

DN Master Send FAB Type

Ei-DN **Modbus Address: 44004**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-4			ENM	Word		Yes		RW

The DN Master Send FAB Type displays the chosen fixed assembly block configuration as shown.

Value	Master Receive FAB Type
0	Index Select (Predef)
1	Index Select (Userdef)
2	Index Type
3	Position Control
4	No Operation

DN Module Status

Ei-DN **Modbus Address: 34208**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-31			ENM	Word		No		RO

The DN Module status indicates the ODVA specific status of the DN device as follows:

Value	Module Status
1	No Power
2	Operational
3	Standby
4	Minor Fault
5	Major Fault

DN Network Status

Ei-DN **Modbus Address: 34207**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
0-31			ENM	Word		No		RO

The DN Network status indicates the ODVA specific status of the DeviceNet network as follows:

Value	Network Status
1	No Power
2	Not Connected
3	Connected
4	Time-out
5	Link Failure

DN Receive Counter

Ei-DN **Modbus Address: 34203-34204**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
$0\text{-}2^{31}\text{-}1$	Counts		US32	UDINT		No		RO

The DN Receive Counter keeps a running total of all DeviceNet packets successfully received from the Ei-DN on the DeviceNet network.

DN Status Major Recoverable Fault

Ei-DN **Modbus Address: 14003**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
True/False			BIT	BOOL		No		RO

A Major Recoverable Fault disables the bridge and then re-enables the bridge when the fault has cleared. At the present, no Major Recoverable Faults are defined in the Ei-DN.

DN Status Major Unrecoverable Fault

Ei-DN **Modbus Address: 14004**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
True/False			BIT	BOOL		No		RO

A Major Unrecoverable Fault is implemented on the Ei-DN when one of the following faults occur: Power Stage Module, Invalid Configuration, Power Up Self Test, NVM Invalid, Motor Overtemp, Drive Overtemp, Duplicate MacID, and Bus-Off. Major Unrecoverable Faults disable the bridge and require a cycle of power to reset.

DN Status Minor Recoverable Fault

Ei-DN **Modbus Address: 14001**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
True/False			BIT	BOOL		No		RO

A Minor Recoverable Fault occurs when the drive experiences a connection time-out. This fault will not disable the bridge and will clear after the fault clears.

DN Status Minor Unrecoverable Fault

Ei-DN **Modbus Address: 14002**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
True/False			BIT	BOOL		No		RO

A Minor Unrecoverable Fault will initiate when any of the following faults occur: Encoder State, Encoder Hardware, Low DC Bus, High DC Bus, Overspeed, Following Error, Shunt Power RMS. Minor Unrecoverable Faults can only be reset with a cycle of power to the drive.

DN Transmit Counter

Ei-DN **Modbus Address: 34201-34202**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
$0\text{-}2^{31}\text{-}1$	Counts		US32	UDINT		No		RO

The DN Transmit Counter keeps a running total of all DeviceNet packets successfully transmitted to the Master on the DeviceNet network.

DN Vendor ID

Ei-DN **Modbus Address: 34001**

Range	Units	Default	Type	DN Type	Group	NVM	Res.	Access
			US16	UINT		No		RO

DN Vendor ID indicates the ODVA (Open DeviceNet Vendors Association) vendor specific number. The Motion Made Easy Vendor ID is 553.

Accessing the EDS File

The EDS file is used for configuration of the DeviceNet master. This file tells the Master software how much I/O data is supported for a particular DeviceNet device as well as Control Techniques' vendor code, module revision, etc. The EDS file for the Ei-DN also contains the Class, Instance, and Attribute ID's for all Ei drive parameters. These class, instance, and attribute ID's are used for explicit messaging and can be found in the back of this manual in the Drive Parameters chapter.

The EDS file is installed with PowerTools FM and can be found in the PTOOLSFM folder. An icon file is also located in this section for display on the master software configuration screen. These two files may also be downloaded from our website at www.emersonct.com.

Hardware Interface

Module Status LED

The Module Status LED located on the front of the drive provides device status according to the following table:

LED State	Drive State	Indicates
Off	No Power	There is no power applied to the device.
Green	Device Operational	The device is operating in a normal condition.
Flashing Green	Device in Standby (The Device Needs Commissioning)	The device need commissioning due to configuration missing, incomplete, or incorrect. The Device may be in the standby state.
Flashing Red	Minor Fault	Recoverable Fault
Red	Unrecoverable Fault	The device has an unrecoverable fault; may need replacing
Flashing Red-Green	Device Self Testing	The device is in self-test.

Network Status LED

The Network Status LED located on the front of the drive provides network status according to the following table:

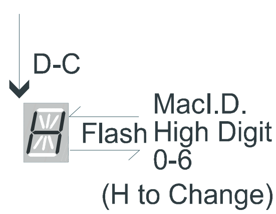
LED State	Drive State	Indicates
Off	Not Powered/Not online	Device is not online: The device has not completed the Dup MacID test yet. The device may not be powered, look at Module Status LED.
Flashing Green	Online, Not Connected	Device is online but has no connections in the established state. The device has passed the Dup MacID test, is online, but has no established connections to a Master.
Green	Link OK Online, Connected	The device is online and has connections in the established state. This means the device is connected to a Master.
Flashing Red	Connection Time-Out	One or more I/O Connections are in the Timed-Out state.
Red	Critical Link Failure	Failed communication device. The device has detected an error that has rendered it incapable of communicating on the network (Duplicate MacID or Bus Off).

Setting the Baud Rate and MacID Externally (One Touch)

The Baud Rate and MacID of the Ei-DN may easily be changed using PowerTools FM programming software or externally on the drive using a one-touch setup with the “Reset” button. The following is a full description of how to navigate through the DeviceNet edit mode externally on the Ei-DN.

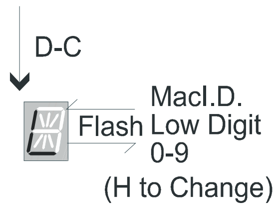
Viewing and Changing the Baud Rate and MacID

1. Double-click the Reset Button to enter the DeviceNet edit mode. The diagnostic display located on the front of drive will flash between “H” (high digit) and the high (tens) digit of the MacID. If the reset button is held in while the “H” is blinking on the diagnostic display (H to Change), the high digit of the MacID value will scroll. Let the reset button go when the display shows the correct number for the high digit of the MacID. (The display will now flash between the letter “H” and the new tens digit for MacID.)



2. Double-click the Reset Button to view the low digit of the MacID. The diagnostic display located on the front of the drive will flash between an “L” and the low (ones) digit of the Mac ID. If the reset button is held in while the “L” is flashing on the diagnostic display (H to Change), the low digit of the MacID value will scroll. Let the reset button go when

the display shows the correct number for the low digit of the MacID. (The display will now flash between the letter “L” and the new ones digit for MacID.)

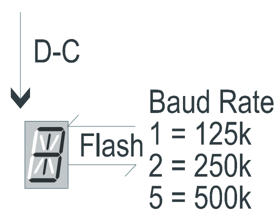


3. Double-click the Reset Button to view the Baud Rate selected. The display will flash between the letter B and the number of the baud rate that is currently configured.

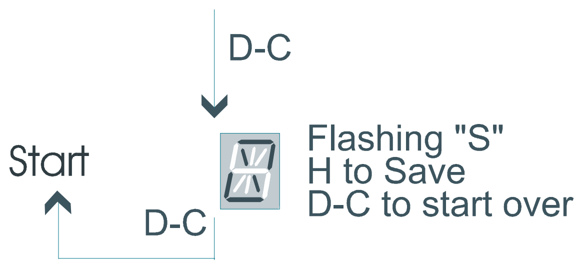
Display Value	Baud
1	125K
2	250K
3	500K

To change the baud rate hold the reset button down and scroll through these three options. Let the “Reset” button go when the displayed value reflects the baud rate for the DeviceNet network.

(Default Baud Rate is configured to “1” or “125K”)



4. Double-click the “Reset” button to reveal a flashing “S” on the diagnostic display. If the parameters entered reflect the proper DeviceNet network settings, the configuration can be saved by holding the “Reset” button while the “S” is flashing. After the parameters are saved, the display will flash “O” and then “K” and return the display back to either a disabled or enabled state. If the user does not want to save the changes, a 30-second time-out will return the drive back to the enabled or disabled display on the drive.



Note

If the user at any time wishes to discontinue the edit mode without saving changes , this can be accomplished by either allowing the drive a 30 second time-out period without touching the “Reset” button, or by cycling power to the drive.

Note

If the Baud Rate or MacID of the device is changed externally, power to the drive and the DeviceNet network must be cycled. Changes to the baud rate and Mac ID will not be reflected until a full power reset has been completed.

Network Configuration Quick Start

This chapter presents sample startups with a Epsilon Ei-DN drive using a 1747-SDN scanner card as the DeviceNet master in a SLC 500 PLC™. The examples presented are intended as a tutorial. For greater information about the steps in this chapter, please refer to the "Configuring the DeviceNet Network" chapter.

Equipment

Epsilon Ei-DN Digital Drive

NT-207 Motor

PowerTools FM Software

Allen Bradley SLC 500 PLC™, 1747-L542 (4 Slot Chassis), P1 Power Supply, 1747 SDN Scanner Card (inserted in slot #3)

1770 KFD RS 232 to DeviceNet to RS232 Interface module with 96881501 RS232 cable (included)

DeviceNet thin cable and terminating resistors

RS Logix 500 Industrial Programming Software (PLC)

Epsilon Ei-DN Parameter Tables from the "Drive Parameters" chapter of this manual

Epsilon Ei DeviceNet Drive Configuration Quick Start Process

1. In PowerTools FM, configure the software for the particular drive and motor (NT-207) that will be used.
2. Open the DeviceNet tab and configure the DeviceNet Setup.
    - Set the MacID and Baud Rate.
    - Select the appropriate Predefined Assembly Block for the transmission of polled I/O. (The tree to the right will display all parameters within the selected assembly block.)
3. Make assignments under the Inputs and Outputs tab for any DN bits that may be configured in each assembly block.
4. Download the configuration to the Ei-DN.

5. Install the .eds file in the Master Configuration software (RsNetworx, Wonderware, etc.).

6. Download a configuration for 8 polled input bytes and 8 polled output bytes to be transferred to/from the Ei-DN.

7. Use ladder logic commands to initiate explicit messaging commands over DeviceNet.

Polled I/O Setup

The following examples assume that the user is familiar with and can set up the Ei using the *Epsilon Ei Reference Manual* (P/N 400507-01). This application example will focus on the DeviceNet setup for the Ei-DN. The setups will demonstrate the Index Sel Predefined, Index Sel User Defined, and Index Type Position Control assembly blocks to transfer DeviceNet bits and initiate indexes.

Example #1: Index Sel(Predef)

This sample procedure for the Index Sel(Predef) assembly block will run preconfigured Index 1 and 2 and Jog while reading back Torque Command and Command Velocity respectively.

1. Start a new Ei-DNconfiguration in PowerTools FM and configure the drive for the particular motor that will be used. (For more information consult the *Epsilon Ei Drives Reference Manual* (P/N 400507-01).

2. Configure Index 0 as a Incremental Index, Index Count of 1, Distance of 20 revs at a Velocity of 1000 RPM, both Acceleration and Deceleration set to 1000 ms/Krevs/m and a Dwell Time of 0.

3. Enter the DeviceNet setup and set the MacID to 1 and the Baud Rate to 125K. Select the Index Sel (Predef) for both Master Send Assembly Block and Master Receive Assembly Block.

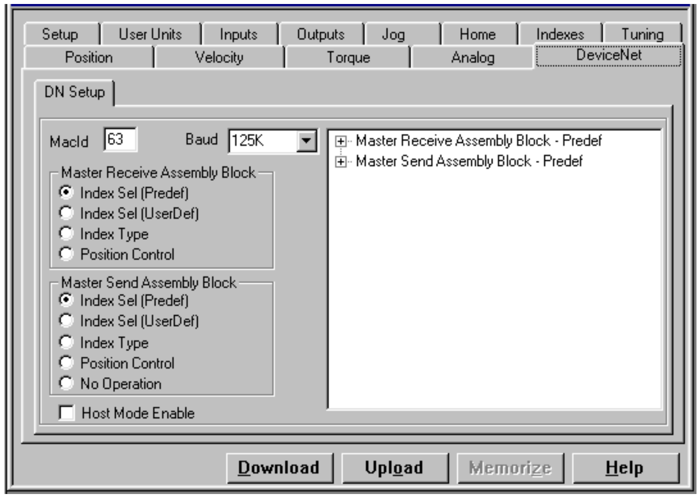


Figure 10: DN Setup / Index Sel (Predef)

4. Download this file to the Ei-DN.

5. Proceed to "Master Configuration" on page 48 before going any farther with this example. After configuring the master return to this point and continue.

CAUTION

The following will initiate motion.

While Online with both the Ei-DN via PowerTools FM and the PLC via RSLogix 500:

6. Set Output Bit O:1.2/7 high to software enable the drive.

7. Set Output Bit O:1.1/0 high to initiate Index 0.

8. Set Output Bit O:1.1/1 high to initiate Home.

9. Set Output Bit O:1.1/2 high to Jog +.

10. Set Output Bit O:1.1/3 high to Jog -.

11. Set Output Bit O:1.1/4 high to enable Jog Fast.

12. Set Output Bits O:1.1/12 - O:1.1/15 to select the index to initiate or to select the index to read from or write to using the Input/Output Word Select Data Pointer.

13. Using Output Word Select Data Pointer 0x0A to set Jog Velocity and Input Select Data Pointer 0x06 to read back the torque command, use the following table to construct the proper words for the read and write:

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x0A	Reserved	Jog -	Jog +	Input Word Select Data Pointer (See page 14) = 0x06
X	X	X	01010	X	X	X	00110

Final Word = XXX01010XXX00110

With zeros in for X's = 2566 (0xA06)

Write to Words O:1.3 - O:1.4 to write to the selected jog velocity. Data can be obtained from the online DN view described in the following step.

14. While online with PowerTools FM click the DeviceNet tab and open up the Master Receive and Master Send Assembly Blocks. Verify that the correct data is moving back and forth through the assembly blocks and that the data pointers are reading and writing the data as expected.

Example #2 Index Sel (Userdef)

This sample procedure for the Index Sel (Userdef) assembly block will use the DN bits set up from Example #1 to initiate indexes and a home. This procedure will also initiate Jog + and Jog - and select between Jog Fast. Corresponding to these initiates, DeviceNet will read back End of Index and End of Home bits, indicate in + Motion and in - Motion, Faults, and Drive OK.

Using the Input and Output Word Select Data Pointer, this procedure will send index velocities to the drive and receive position feedback from the drive.

1. Start a new Ei-DNconfiguration in PowerTools FM and configure the drive for the particular motor that will be used. For more information consult the *Epsilon Ei Drives Reference Manual* (P/N 400507-01.)

2. Enter the DeviceNet setup and set the MacID to 1 and the Baud Rate to 125K. Select the Index Sel (Userdef) in both Master Send Assembly Block and Master Receive Assembly Block setups.

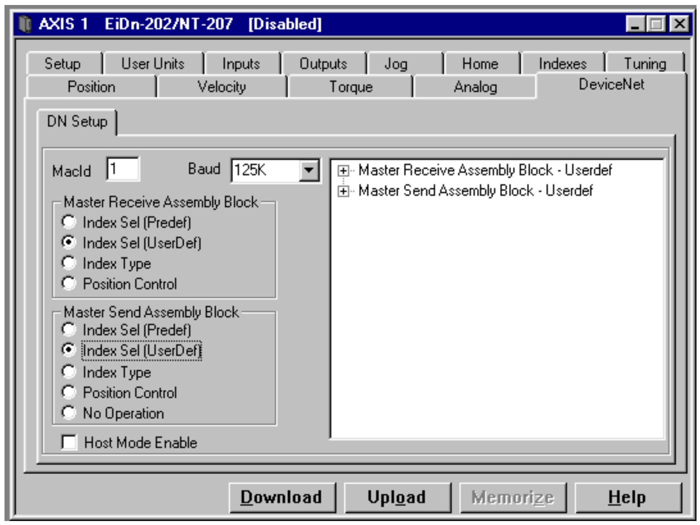


Figure 11: DN Setup / Index Sel (Userdef)

3. Click the Inputs tab to configure the DN bits that will be written from the PLC (Master Send). The following assignments will be used for this application example:

   DN Bit00 (MS) = Index Initiate

   DN Bit01 (MS) = Home Initiate

   DN Bit02 (MS) = Jog +

   DN Bit03 (MS) = Jog -

   DN Bit04 (MS) = Jog Fast

Note

Index Sel (Userdef) allows 16 DN bits to be configured. Although these bits are optional to assign, bandwidth used on the DeviceNet network remains the same.

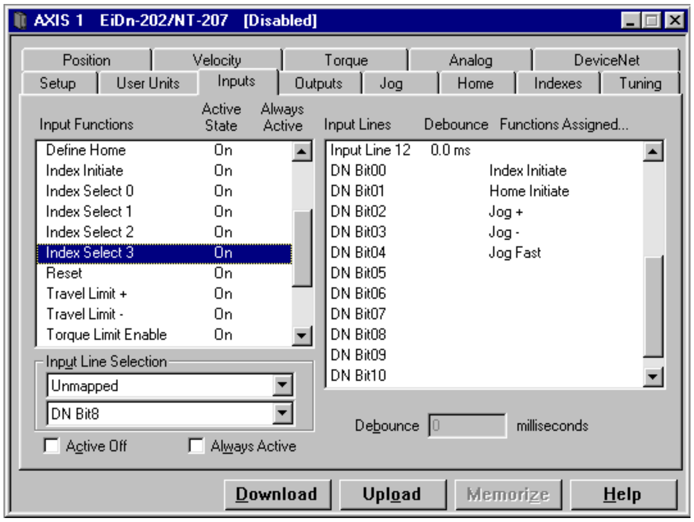


Figure 12: Inputs Tab

4. Click the Outputs tab to configure the DN bits that will be written to from the PLC (Master Receive).

DN Bit00 (MR) = End of Index

DN Bit01 (MR) = End of Home

DN Bit02 (MR) = In + Motion

DN Bit03 (MR) = In - Motion

DN Bit04 (MR) = Fault

DN Bit05 (MR) = Drive OK

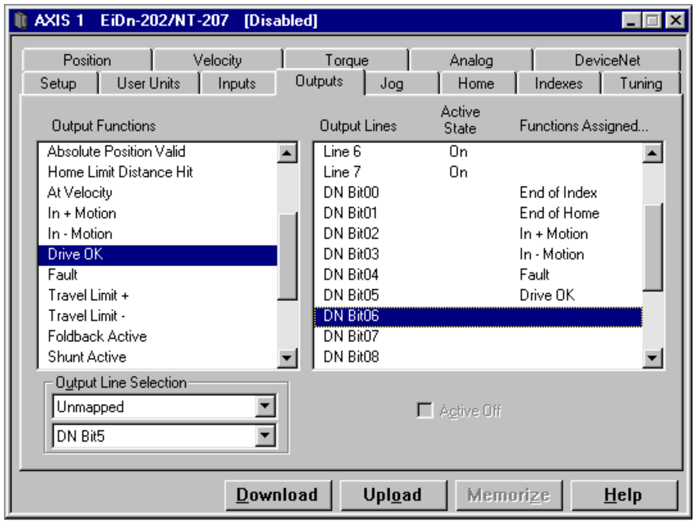


Figure 13: Outputs Tab

Note

Index Sel (userdef) allows 16 DN bits to be configured. Although these bits are optional to assign, bandwidth used on the DeviceNet network remains the same.

5. Once all parameters are configured, download this file to the Ei-DN.

6. Proceed to “Master Configuration” on page 48 before going any farther with this example. After configuring the master return to this point and continue.

CAUTION

The following will initiate motion.

While online with the Ei-DN via PowerTools FM and the PLC via RSLogix 500:

7. Set Output Bit O:1.2/5 high to software enable the drive.

8. Set Output Bit O:1.1/0 high to initiate Index 0.

9. Set Output Bit O:1.1/1 high to initiate Home.

10. Set Output Bit O:1.1/2 high to Jog +.

11. Set Output Bit O:1.1/3 high to Jog -.
12. Set Output Bit O:1.1/4 high to enable Jog Fast.
13. Set Output Bits O:1.1/12 - O:1.1/15 to select the index to initiate or to select the index to read from or write to using the Input/Output Word Select Data Pointer.
14. Using Output Word Select Data Pointer 0x02 to set the Index Velocity and Input Select Data Pointer 0x01 to read back the position feedback, use the following table to construct the proper words for the read and write:

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x02	Reserved	Reserved	Enable = 1	Input Word Select Data Pointer (See page 14) = 0x01
X	X	X	00010	X	X	1	00001

Final Word = XXX00010XX100001

With zeros in for reserved bits Final Word = 545 (0x221)

15. Write to Words O:1.3 - O:1.4 to write to the selected index velocity. Data can be obtained from the online DN view described in the following step.
16. While online with PowerTools FM click on the DeviceNet tab and open up the Master Receive and Master Send Assembly Blocks. Verify that the correct data is moving back and forth through the assembly block and that the data pointers are reading and writing the data as expected.

Example #3: Index Type

This sample procedure for the Index Type Assembly Block will initiate both incremental and absolute indexes.

The Input and Output Word Select Data Pointer will read velocity feedback from the drive and write Jog Acceleration to the drive.

1. Start a new Ei-DN configuration in PowerTools FM and configure the drive for the particular motor that will be used. For more information consult the *Epsilon Ei Drives Reference Manual* (P/N 400507-01.)

2. Enter the DN setup and set the MacID to 1 and the Baud Rate to 125K. Select Index Type for both Master Send Assembly Block and Master Receive Assembly Block Setups.

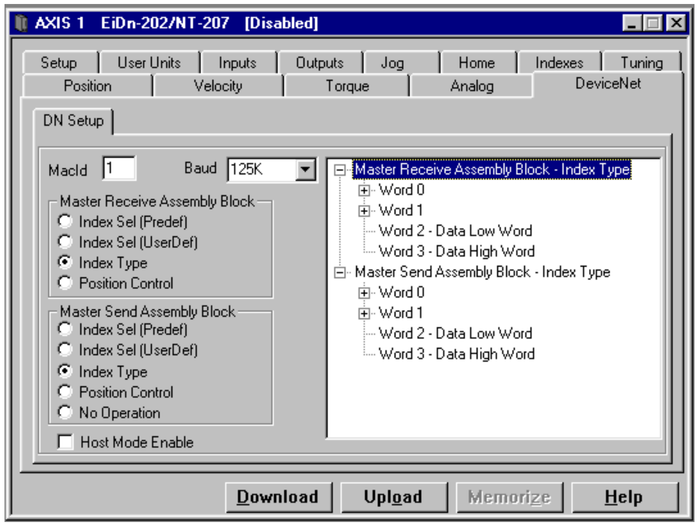


Figure 14: DN Setup / Index Type

3. Click the Inputs tab to configure the DN bits that will be written from the PLC (Master Send). The following assignments will be used for this application example:

   DN Bit00 (MS) = Reset

   DN Bit01 (MS) = Home Initiate

   DN Bit02 (MS) = Jog +

Note

Index type allows 8 DN bits to be configured. Although these bits are optional to assign, bandwidth used on the DeviceNet network remains the same.

---

4. Click the Outputs tab to configure the DN bits that will be written from the PLC (Master Receive).

   DN Bit00 (MR) = End of Index

   DN Bit01 (MR) = End of Home

   DN Bit02 (MR) = In + Motion

   DN Bit03 (MR) = In - Motion

   DN Bit04 (MR) = Fault

   DN Bit05 (MR) = Drive OK

5. Download this file to the Ei-DN.

6. Proceed to “Master Configuration” on page 48 before going any farther with this example. After configuring the master return to this point and continue.

While online with the Ei-DN via PowerTools FM and with the PLC via RSLogix 500:

7. Set Output Bit O:1.1/7 high to software enable the drive.

8. Set Output Bit O:1.1/1 high to indicate an incremental index.

9. Using Output Word Select Data Pointer 0x01 to set the Index Distance/Position Register and Input Select Data Pointer 0x07 to read back this same parameter in order to verify the transfer.

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x01	Reserved	Jog -	Jog +	Input Word Select Data Pointer (See page 14) = 0x07
X	X	X	00001	X	X	X	00111

Final Word = XXX00001XXX00111

With zeros in for reserved bits Final Word = 263 (0x107)

10. Using Output Word Select Data Pointer 0x02 to set the Index Velocity Register and Input Select Data Pointer 0x08 to read back this same parameter in order to verify the transfer.

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x02	Reserved	Jog -	Jog +	Input Word Select Data Pointer (See page 14) = 0x08
X	X	X	00010	X	X	X	01000

Final Word = XXX00010XXX01000

With zeros in for reserved bits Final Word = 520 (0x208)

11. Using Output Word Select Data Pointer 0x03 to set the Index Acceleration Register and Input Select Data Pointer 0x09 to read back this same parameter in order to verify the transfer.

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x03	Reserved	Jog -	Jog +	Input Word Select Data Pointer (See page 14) = 0x09
X	X	X	00011	X	X	X	01001

Final Word = XXX00011XXX01001

With zeros in for reserved bits Final Word = 777 (0x309)

12. Using Output Word Select Data Pointer 0x04 to set the Index Deceleration Register and Input Select Data Pointer 0x0A to read back this same parameter in order to verify the transfer.

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x04	Reserved	Jog -	Jog +	Input Word Select Data Pointer (See page 14) = 0x0A
X	X	X	00100	X	X	X	01010

Final Word = XXX00100XXX01010

With zeros in for reserved bits Final Word = 1034 (0x40A)

13. Set Output Bit O:1.1/0 high to initiate this incremental index.
14. Set Output Bit O:1/1/2 high and O:1.1/1 low to indicate an absolute index.
15. Using Output Word Select Data Pointer 0x01 to set the Index Distance/Position Register and Input Select Data Pointer 0x07 to read back this same parameter in order to verify the transfer.

Reserved	Reserved	Reserved	Output Word Select Data Pointer (See page 15) = 0x01	Reserved	Jog -	Jog +	Input Word Select Data Pointer (See page 14) = 0x07
X	X	X	00001	X	X	X	00111

Final Word = XXX00001XXX00111

With zeros in for reserved bits Final Word = 263 (0x107)

16. Set Output Bit O:1.1/0 high to initiate this absolute index.

Note

After the motion parameters have been configured (accel, decel, dist), these parameters remain static until they are changed using either explicit messaging or the data select pointers.

Example #4: Position Control

This sample procedure for the Position Controller Assembly Block will completely set up an index and then initiate it using the addresses assigned in previous examples.

1. Start a new Ei-DN configuration in PowerTools FM and configure the drive for the particular motor that will be used. For more information consult the *Epsilon Ei Drives Reference Manual* (P/N 400507-01.)

2. Enter the DN setup and set the MacID to 1 and the Baud Rate to 125K. Select PositionControl for both Master Send Assembly Block and Master Receive Assembly Block setups.

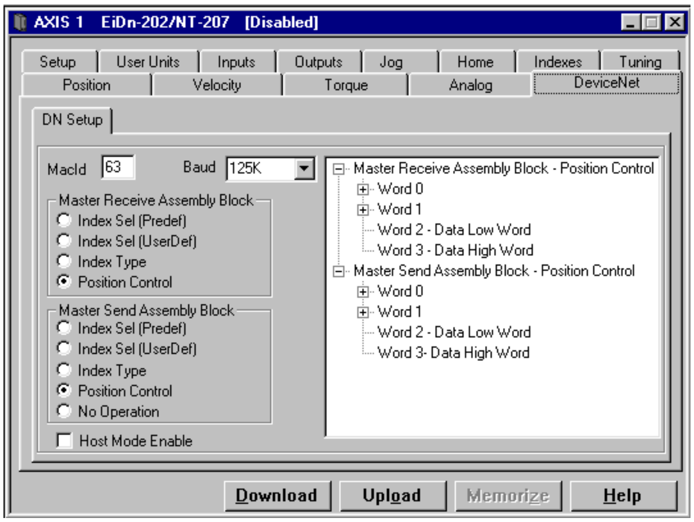


Figure 15: Dn Setup / Position Control

3. Download this file to the Ei-DN.

4. Proceed to “Master Configuration” on page 48 before going any farther with this example. After configuring the master return to this point and continue.

5. Set the Command Assembly code to send a value of 50 revs to the Index Position. O:1.2 = 0x01 (parameter) and O:1.3 = 0x32 (value). After these values are in place, toggle the Valid Data Bit (O:1.1/6).

6. Set the Command Assembly code to send a value of 1000 Revs/minute to the Index Velocity. O:1.2 = 0x02 (parameter) and O:1.3 = 0x3E8 (value). After these values are in place, toggle the Valid Data Bit (O:1.1/6).

7. Set the Command Assembly code to send a value of 500 revs/min^2 to the Index Accel. O:1.2 = 0x03 (parameter) and 0:1.3 = 0x1F4 (value). After these values are in place, toggle the Valid Data Bit (O:1.1/6).

8. Set the Command Assembly code to send a value of 500 revs/min^2 to the Index Decel. O:1.2 = 0x04 (parameter) and O:1.3 = 0x1F4 (value). After these values are in place, toggle the Valid Data Bit (O:1.1/6).

9. Set the Receive Assembly code to read the Actual Position O:1.2 = 0x02 (parameter). This response will be read back from I:1.3 and I:1.4 (low/high word response).

10. Set O:1.1/7 = 1 to Enable the Drive. Set O:1.1./2 = 0 indicating an Absolute Move. Set O:1.1/0 = 1 indicating start of the Move. After these values are in place, toggle the Valid Data Bit (O:1.1/6).

Using the parameter set previously, a second motion can be initiated with one Valid Data Bit toggle. This move will also read back the actual Velocity from I:1.3 and I:1.4.

11. Set O:1.2 = 0x03, this will set the Response Assembly code to read back Velocity feedback and the Command Assembly code to write to the Index Position. Set O:1.3 = 0x0 (value) and then toggle the Valid Data Bit (O:1.1/6).

Master Configuration

RS Networx™ will be used to install the Ei-DN onto the DeviceNet network as a slave. In order to have RS Networx recognize the Ei-DN as a node on the network, an .eds file must be installed using the .eds wizard found under the Tools menu. The .eds file is located in the Emerson directory under Emerson\PtoolsFM\Ei.eds. An icon file can also be found in this directory under Emerson\PtoolsFM\Ei.ico. This icon file will display the Ei-DN graphically in RSnetworx.

Once the .eds file is installed, add a 1747.SDN DeviceNet master to the network set for MacID 0 and an Ei-DN set for MacID 1.

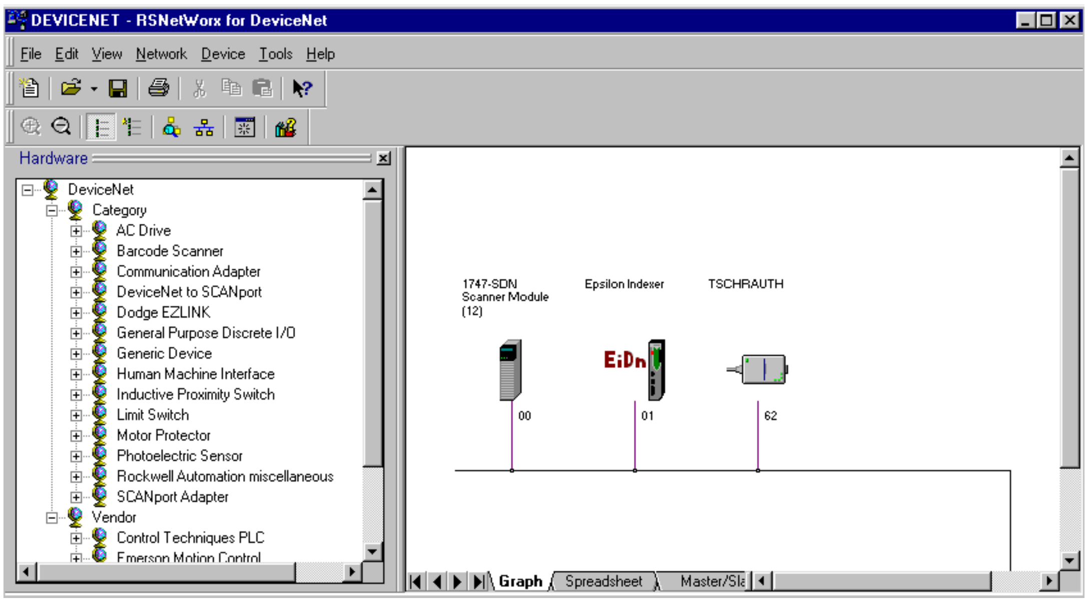


Figure 16: RS Networx Configuration

Double-click the 1747.SDN to bring up the Master Configuration.

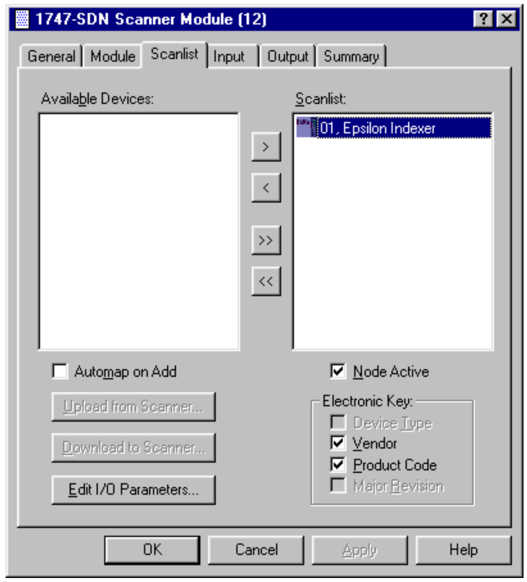


Figure 17: Scanlist Tab

Click the Input tab and map the 4 words of data from the Ei-DN to I:1.1 - I:1.4. These addresses are the addresses in the PLC where the Ei-DN data will be mapped.

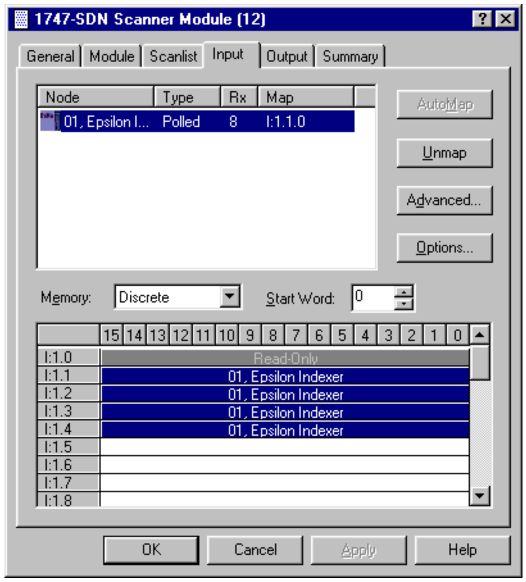


Figure 18: Input Tab

Click the Output tab and map the 4 words of data from the Ei-DN to O:1.1 - O:1.4 as shown. These addresses are the addresses in the PLC where the Ei-DN will pull its data from.

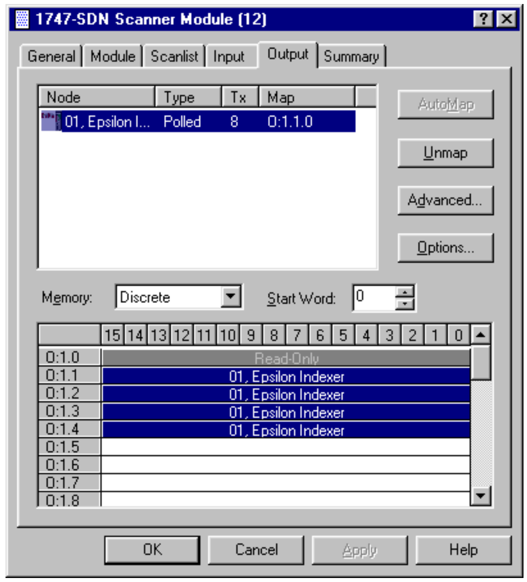


Figure 19: Output Tab

After all of the data is mapped click OK, go online, and then download this RsNetworx file to the DeviceNet scanner. (Make sure processor is in program mode.)

Note

If, after a download, errors occur on the DeviceNet Master (scanner card), clear all mappings in the scanlist and download this “empty” file to the Master. The network should display two green LED’s (if not check wiring... etc.). After this has occurred repeat the above procedure to configure the Ei-DN onto the DeviceNet network.

Explicit Messaging Quick Start

To send explicit messages, the Epsilon Ei-DN drive should be configured and successfully connected to the DeviceNet system as a unique node. This chapter will present examples that should be used as a tutorial.

Equipment

Ei-DN DeviceNet Digital Drive

NT 207 Motor

Allen Bradley SLC 500 PLC™, 1747-L542 (4 Slot Chassis), P1 Power Supply, 1747 SDN Scanner Card (slot three)

1770 KFD RS 232 to DeviceNet Interface module with 96881501 RS232 cable

DeviceNet thin cable

RS Logix 500 Industrial Programming Software (PLC)

Ei DeviceNet Drive Parameter Tables from the "Drive Parameters" chapter of this manual

How to Send Explicit Messages

An Explicit Message is compiled using an integer file (for example N10). Format the message using the following table.

	Integer File N10	High Byte	Low Byte
Message Header	N10:0	TXID	COMMAND
	N10:1	PORT	SIZE OF FIELD
	N10:2	SERVICE	MacID
Message Data	N10:3	CLASS	
	N10:4	INSTANCE	
	N10:5	ATTRIBUTE	
	N10:6	VALUE LOW WORD	
	N10:7	VALUE HIGH WORD	

Once formatted, use the copy command to transfer it to the M0 file. The copy initiates the explicit message transfer from the scanner to the Ei DeviceNet drive. The Ei drive response will be placed into the M1 file by the scanner. Using the scanner message received status flag (in the example of this chapter the location would be I:3/15), copy the M1 file to an integer file (N11). Once the message has been copied, the results may be examined from the N11 file.

TXID is set at 0x01 for all of the following examples.

PORT will be set to 0x00 indicating a 1 port network.

COMMAND	
0x01	EXECUTE
0x04	CLEAR RESPONSE BUFFER

Size of field is the size of the message data section and does not contain the message header. For GET Service command, size should be 6 bytes; for SET Service command, size should be 10 [0x0A] bytes.

SIZE OF DATA FIELD	
Get Fields	0x06 bytes
Set Fields	0x0A bytes

The service parameter determines whether the data will be sent to or received from the slave.

SERVICE	
0x0E	GET DATA
0x10	SET DATA

The MacID indicates the node number of the device to be read/written to.

MacID
0x00-0x3F

The class instance and attribute IDs for each parameter in the Ei DeviceNet drive can be found in the back of this manual.

CLASS
INSTANCE
ATTRIBUTE

Note

In order to transmit a new explicit message, the response buffer must be cleared by setting TXID/Command portion of the message header to 104 hex in the integer file and initiating a file copy. This copy must be completed every time an explicit message is sent or received. The Ladder Logic given initiates this command automatically.

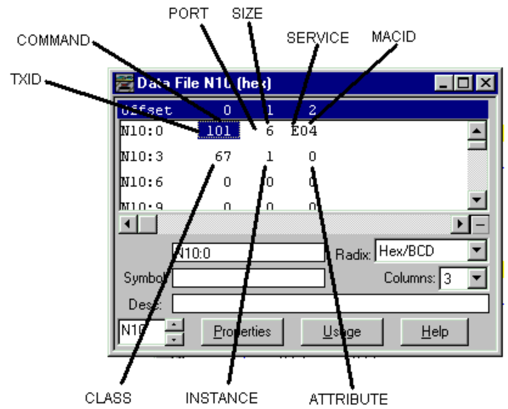


Figure 20: Description of N10 Integer File

Explicit Messaging Examples

The following two examples of DeviceNet Explicit Messaging will be assuming a PLC configuration as follows:

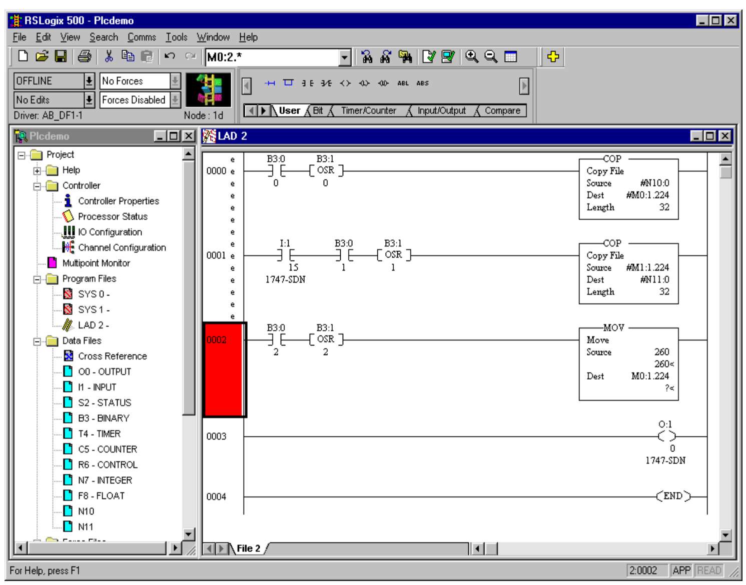


Figure 21: PLC Configuration for Explicit Messaging

Explanation of PLC Program

The copy command in Rung 0 will activate when B3:0/0 is turned on. This command will send out a request to the DeviceNet master by copying the N10 integer file into the M0: file.

After this request gets sent out, the PLC indicates when it has all data returned by setting bit I:3/15 high. Once this occurs bit B3:0/1 can be set high to view the results via a copy command from the scanner card M1: to integer file N11:.

Once the data is received, the response buffer must be cleared before the next explicit message can be sent. This is accomplished by setting bit B3:0/2 high. When this is implemented, the I:3/15 bit will be cleared by the scanner card and the explicit messaging system is free to use again.

Note

After each send a clear response buffer message must be sent in order to continue communication.

Examples

Example 1 Read Index 0 Velocity

This example will show how to use explicit messaging to capture the current index velocity in an Ei-DN drive.

Create a PowerTools Pro configuration file ensuring that both the MacID (04) and the network BaudRate (125 KB) are configured. For this example, set the Index 0 Velocity is to 1000 units. Download this to the drive.

Index 0 Velocity is mapped to class 0x68, instance 0x01, attribute 0x09, as shown in the table below.

The following parameters will be used to set up this explicit message.

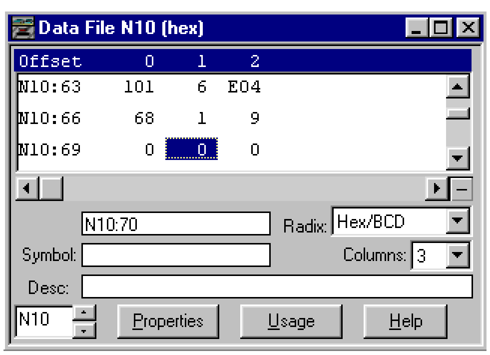


Figure 22: Integer Data File N10

Integer File N10	High Byte	Low Byte
N10:0	TXID= 0x01	COMMAND= 0x01
N10:1	PORT= 0x00	SIZE OF DATA FIELD= 0x06
N10:2	SERVICE= 0x0E (GET)	MacID= 0x04
N10:3	CLASS= 0x68	
N10:4	INSTANCE= 0x01	
N10:5	ATTRIBUTE= 0x09	

After the N10 file is set, B3:0/0 can be set high to send the explicit message to the scanner card. After bit I:3/15 is set by the PLC, the message can be viewed by setting B3:0/2 high which copies the M1: file to the N11: file.

The response from the drive should resemble the following.

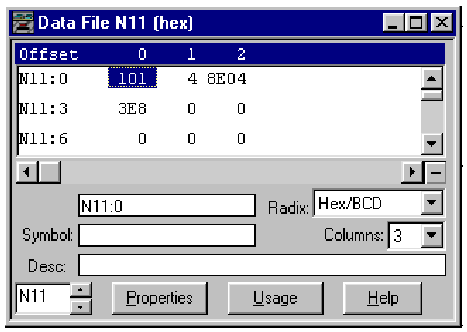


Figure 23: PLC Configuration Example 1

The following table reflects the results of the Index 0 Velocity example. Note that the Low Data Word is 1000 (0x3E8), the velocity requested.

Integer File N11	High Byte	Low Byte
N11:0	TXID=0x01	COMMAND=0x01
N11:1	PORT=0x00	SIZE OF DATA FIELD=0x04
N11:2	SERVICE=0x8E (Successful Response)	MacID=0x04
N11:3	LOW DATA WORD = 0x3E8	
N11:4	HIGH DATA WORD = 0x00	

Example 2 Write Index 1 Velocity

This example will show how to use explicit messaging to write the current index velocity to an Ei-DN drive.

Assume the drive configuration for example 1.

Index 1 Velocity is mapped to class 0x68, instance 0x02, attribute 0x09, as shown in the table below.

The following parameters will be used to set up this explicit message.

Interger File N10	High Byte	Low Byte
N10:0	TXID=0x01	COMMAND=0x01
N10:1	PORT=0x00	SIZE OF DATA FIELD=0x0A
N10:2	SERVICE=0x10 (SET)	MacID=0x04
N10:3	CLASS=0x68	
N10:4	INSTANCE=0x02	
N10:5	ATTRIBUTE=0x09	

After the N10 file is set, B3:0/0 can be set high to send the explicit message to the scanner card. After bit I:3/15 is set by the PLC, the message can be viewed by setting B3:0/2 high which copies the M1: file to the N11: file.

The response from the drive should resemble the following.

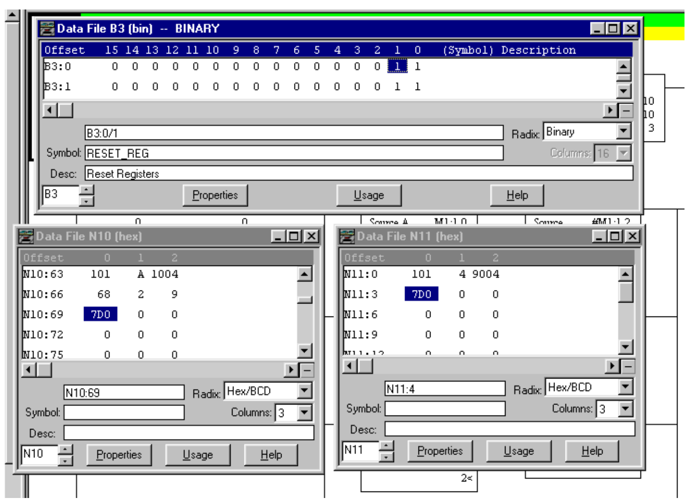


Figure 24: PLC Configuration Example 2

The following table reflects the results of the Index 1 Velocity example. These results are verifiable via the keypad interface.

Integer File N11	High Byte	Low Byte
N11:0	TXID=0x01	COMMAND=0x01
N11:1	PORT=0x00	SIZE OF DATA FIELD=0x04
N11:2	SERVICE=0x90 (Successful Transmit)	MacID=0x04
N11:3	LOW DATA WORD = 0x7D0	
N11:4	HIGH DATA WORD = 0x00	

Drive Parameters

This section lists all parameters available for Epsilon Ei DeviceNet drive. The tables provide the following information about each parameter:

Name

The parameter's name

DeviceNet Data Type

INT, UINT = 16 bit value
DINT, UDINT = 32 bit value

BOOL = Bit
Word = Bitmap 16
DWord = Bitmap 32

String: XXX = ASCII characters of XXX length, padded with white space

Class, Instance, Attribute

The path necessary to access the parameter.

By Name

Note 1

The instance number will be one more than the number of the index, jog, input or output. For example, Index0 will have an instance of 1 (0x01) and Index 14 will have an instance of 15 (0x0F). If the parameter is unnumbered, the instance will be one.

Name	DN Data Type	Class	Instance	Attribute
AnalogOutput.AnalogOutputInstance#.Channel	INT	100 (0x64)	See Note 1	1 (0x01)
AnalogOutput.AnalogOutputInstance#.Offset	DINT	100 (0x64)	See Note 1	2 (0x02)
AnalogOutput.AnalogOutputInstance#.Scale	DINT	100 (0x64)	See Note 1	3 (0x03)
AnalogOutput.AnalogOutputInstance#.Select	INT	100 (0x64)	See Note 1	4 (0x04)
DeviceNet.DeviceNetIdentityObject.DeviceStatus	UINT	1 (0x01)	1 (0x01)	5 (0x05)
DeviceNet.DeviceNetIdentityObject.DeviceType	UINT	1 (0x01)	1 (0x01)	2 (0x02)
DeviceNet.DeviceNetIdentityObject.HeartbeatInterval	DINT	1 (0x01)	1 (0x01)	10 (0x0A)
DeviceNet.DeviceNetIdentityObject.VendorID	UINT	1 (0x01)	1 (0x01)	1 (0x01)
Execute.ExecuteClearFault	BOOL	101 (0x65)	1 (0x01)	1 (0x01)
Execute.ExecuteReadBaseDriveNVMtoRAM	BOOL	101 (0x65)	1 (0x01)	2 (0x02)

Name	DN Data Type	Class	Instance	Attribute
Execute.ExecuteStopAllMotion	BOOL	101 (0x65)	1 (0x01)	3 (0x03)
Execute.ExecuteWriteBaseDriveRAMtoNVM	BOOL	101 (0x65)	1 (0x01)	4 (0x04)
Fault.FaultInstance#.Code	WORD	102 (0x66)	See Note 1	1 (0x01)
Fault.FaultInstance#.PowerUpCount	UINT	102 (0x66)	See Note 1	2 (0x02)
Fault.FaultInstance#.PowerUpTime	UDINT	102 (0x66)	See Note 1	3 (0x03)
Fault.FaultStatus	DWORD	102 (0x66)	1 (0x01)	15 (0x0F)
Home.BackOffSensorBeforeHoming	WORD	103 (0x67)	1 (0x01)	1 (0x01)
Home.EndofHomePosition	DINT	103 (0x67)	1 (0x01)	2 (0x02)
Home.HomeAcceleration	UDINT	103 (0x67)	1 (0x01)	3 (0x03)
Home.HomeDeceleration	UDINT	103 (0x67)	1 (0x01)	4 (0x04)
Home.HomeLimitDistance	UDINT	103 (0x67)	1 (0x01)	5 (0x05)
Home.HomeLimitDistanceEnable	WORD	103 (0x67)	1 (0x01)	6 (0x06)
Home.HomeOffset	DINT	103 (0x67)	1 (0x01)	7 (0x07)
Home.HomeOffsetEnable	WORD	103 (0x67)	1 (0x01)	8 (0x08)
Home.HomeReference	WORD	103 (0x67)	1 (0x01)	9 (0x09)
Home.HomeVelocity	DINT	103 (0x67)	1 (0x01)	10 (0x0A)
Index.ChainingCount	UINT	104 (0x68)	1 (0x01)	23 (0x17)
Index.CurrentChainingCount	UINT	104 (0x68)	1 (0x01)	24 (0x18)
Index.CurrentIndexCount	UINT	104 (0x68)	1 (0x01)	21 (0x15)
Index.CurrentIndexNumber	UINT	104 (0x68)	1 (0x01)	22 (0x16)
Index.IndexInstance#.Acceleration	UDINT	104 (0x68)	See Note 1	2 (0x02)
Index.IndexInstance#.ChainNext	UINT	104 (0x68)	See Note 1	1 (0x01)
Index.IndexInstance#.ControlRegister	UINT	104 (0x68)	See Note 1	3 (0x03)
Index.IndexInstance#.Count	UINT	104 (0x68)	See Note 1	4 (0x04)
Index.IndexInstance#.Deceleration	UDINT	104 (0x68)	See Note 1	5 (0x05)
Index.IndexInstance#.Distance	DINT	104 (0x68)	See Note 1	6 (0x06)
Index.IndexInstance#.Dwell	UINT	104 (0x68)	See Note 1	7 (0x07)
Index.IndexInstance#.Type	WORD	104 (0x68)	See Note 1	8 (0x08)
Index.IndexInstance#.Velocity	UDINT	104 (0x68)	See Note 1	9 (0x09)
Index.InfiniteChaining	WORD	104 (0x68)	1 (0x01)	25 (0x19)
Index.Instance#.RegistrationOffset	DINT	104 (0x68)	See Note 1	10 (0x0A)
InputFunction.InputFunctionAlwaysActiveBitMap	WORD	105 (0x69)	1 (0x01)	21 (0x15)
InputFunction.InputFunctionInstance#.ActiveOffArray	BOOL	105 (0x69)	See Note 1	1 (0x01)
InputFunction.InputFunctionInstance#.AlwaysActive Array	BOOL	105 (0x69)	See Note 1	2 (0x02)
InputFunction.InputFunctionInstance#.Mapping	WORD	105 (0x69)	See Note 1	3 (0x03)
InputFunction.InputFunctionInstance#.StatusArray	BOOL	105 (0x69)	See Note 1	4 (0x04)
InputFunction.InputFunctionPolarityBitMap	WORD	105 (0x69)	1 (0x01)	22 (0x16)
InputFunction.InputFunctionStatusBitMap	WORD	105 (0x69)	1 (0x01)	23 (0x17)

Name	DN Data Type	Class	Instance	Attribute
InputLine.DriveEnableInputDebouncedStatus	BOOL	106 (0x6A)	1 (0x01)	9 (0x09)
InputLine.DriveEnableInputRawStatus	BOOL	106 (0x6A)	1 (0x01)	10 (0x0A)
InputLine.DriveEnableInputStatus	BOOL	106 (0x6A)	1 (0x01)	11 (0x0B)
InputLine.EnableDebounceTime	UINT	106 (0x6A)	1 (0x01)	8 (0x08)
InputLine.InputLineInstance#.DebouncedStatusArray	BOOL	106 (0x6A)	See Note 1	1 (0x01)
InputLine.InputLineInstance#.DebounceTime	UINT	106 (0x6A)	See Note1	4 (0x04)
InputLine.InputLineInstance#.ForceOn/OffCommandArray	BOOL	106 (0x6A)	See Note 1	2 (0x02)
InputLine.InputLineInstance#.ForceOn/OffEnableArray	BOOL	106 (0x6A)	See Note 1	3 (0x03)
InputLine.InputLineInstance#.RawStatusArray	BOOL	106 (0x6A)	See Note 1	0 (0x00)
InputLine.InputLineInstance#.StatusArray	BOOL	106 (0x6A)	See Note 1	0 (0x00)
InputLine.InputLinesDebouncedBitMap	WORD	106 (0x6A)	1 (0x01)	21 (0x15)
InputLine.InputLinesOverrideActive	UINT	106 (0x6A)	1 (0x01)	22 (0x16)
InputLIne.InputLinesOverrideStatus	UINT	106 (0x6A)	1 (0x01)	23 (0x17)
InputLine.InputLinesRawBitMap	WORD	106 (0x6A)	1 (0x01)	24 (0x18)
InputLine.InputLinesStatusBitMap	WORD	106 (0x6A)	1 (0x01)	25 (0x19)
Jog.JogAcceleration	UDINT	107 (0x6B)	1 (0x01)	1 (0x01)
Jog.JogDeceleration	UDINT	107 (0x6B)	1 (0x01)	2 (0x02)
Jog.JogFastVelocity	UDINT	107 (0x6B)	1 (0x01)	3 (0x03)
Jog.JogVelocity	UDINT	107 (0x6B)	1 (0x01)	4 (0x04)
ModuleFirmwareRevision	String:4	1 (0x01)	1 (0x01)	4 (0x04)
ModuleSerialNumber	String:16	1 (0x01)	1 (0x01)	6 (0x06)
OutputFunction.OutputFunctionInstance#.Mapping32	UINT	108 (0x6C)	See Note 1	1 (0x01)
OutputFunction.OutputFunctionStatusBitMap	WORD	108 (0x6C)	1 (0x01)	21 (0x15)
OutputLine.OutputLinesStatusBitMap	WORD	109 (0x6D)	1 (0x01)	4 (0x04)
OutputLine.OutputLinesOverrideActive	UINT	109 (0x6D)	1 (0x01)	1 (0x01)
OutputLine.OutputLinesOverrideStatus	UINT	109 (0x6D)	1 (0x01)	2 (0x02)
OutputLine.OutputLinesPolarityBitMap	WORD	109 (0x6D)	1 (0x01)	3 (0x03)
Position.FollowingError	DINT	110 (0x6E)	1 (0x01)	1 (0x01)
Position.FollowingErrorEnable	WORD	110 (0x6E)	1 (0x01)	2 (0x02)
Position.FollowingErrorLimit	DINT	110 (0x6E)	1 (0x01)	3 (0x03)
Position.InPositionTime	UINT	110 (0x6E)	1 (0x01)	7 (0x07)
Position.InPositionWindow	UDINT	110 (0x6E)	1 (0x01)	6 (0x06)
Position.RolloverPosition	UDINT	110 (0x6E)	1 (0x01)	4 (0x04)
Position.RolloverPositionEnable	WORD	110 (0x6E)	1 (0x01)	5 (0x05)
ProductID.FirmwareRevisionBase	String	111 (0x6F)	1 (0x01)	1 (0x01)
ProductID.FirmwareRevisionOption	String	111 (0x6F)	1 (0x01)	2 (0x02)
ProductID.InterfaceRevisionBase	UINT	111 (0x6F)	1 (0x01)	3 (0x03)
ProductID.InterfaceRevisionOption	UINT	111 (0x6F)	1 (0x01)	4 (0x04)

Name	DN Data Type	Class	Instance	Attribute
ProductID.Option1IDFunctionModule	UINT	111 (0x6F)	1 (0x01)	5 (0x05)
ProductID.Option2IDAnybus1	UINT	111 (0x6F)	1 (0x01)	6 (0x06)
ProductID.Option3IDAnybus2	UINT	111 (0x6F)	1 (0x01)	7 (0x07)
ProductID.ProductGroup	UINT	111 (0x6F)	1 (0x01)	8 (0x08)
ProductID.ProductID	UINT	111 (0x6F)	1 (0x01)	9 (0x09)
ProductID.ProductSerialNumber	String	111 (0x6F)	1 (0x01)	10 (0x0A)
ProductID.ProductSubGroup	UINT	111 (0x6F)	1 (0x01)	11 (0x0B)
ProductSubGroup	UINT16	1 (0x01)	1 (0x01)	3 (0x03)
Setup.DriveAmbientTemperature	UINT	112 (0x70)	1 (0x01)	1 (0x01)
Setup.DriveAxisName	String	112 (0x70)	1 (0x01)	2 (0x02)
Setup.EncoderOutput	UINT	112 (0x70)	1 (0x01)	3 (0x03)
Setup.EncoderOutputEnable	WORD	112 (0x70)	1 (0x01)	4 (0x04)
Setup.ExecuteReadFMNVMtoRAM	BOOL	112 (0x70)	1 (0x01)	21 (0x15)
Setup.LowDCBusEnable	WORD	112 (0x70)	1 (0x01)	5 (0x05)
Setup.PositiveDirection	WORD	112 (0x70)	1 (0x01)	6 (0x06)
Status.AbsoluteRotorPosition	DINT	113 (0x71)	1 (0x01)	1 (0x01)
Status.BusVoltage	UINT	113 (0x71)	1 (0x01)	2 (0x02)
Status.CommutationAngleCorrection	INT	113 (0x71)	1 (0x01)	3 (0x03)
Status.CommutationTrackAngle	UINT	113 (0x71)	1 (0x01)	4 (0x04)
Status.CommutationVoltage	INT	113 (0x71)	1 (0x01)	5 (0x05)
Status.EncoderCounts	DINT	113 (0x71)	1 (0x01)	6 (0x06)
Status.HeatsinkRMS	UINT	113 (0x71)	1 (0x01)	7 (0x07)
Status.MotionState	WORD	113 (0x71)	1 (0x01)	8 (0x08)
Status.OptionSerialNumber	String	113 (0x71)	1 (0x01)	13 (0x0D)
Status.PowerUpCount	UINT	113 (0x71)	1 (0x01)	9 (0x09)
Status.PowerUpTime	UDINT	113 (0x71)	1 (0x01)	10 (0x0A)
Status.RotorVelocity	DINT	113 (0x71)	1 (0x01)	11 (0x0B)
Status.SegmentDisplayCharacter	UINT	113 (0x71)	1 (0x01)	12 (0x0C)
Status.TorqueFeedback	INT	113 (0x71)	1 (0x01)	14 (0x0D)
Status.TotalCommandChangeOut	DINT	113 (0x71)	1 (0x01)	15 (0x0E)
Status.TotalPowerUpTime	UDINT	113 (0x71)	1 (0x01)	16 (0x10)
Status.VelocityCommand	DINT	113 (0x71)	1 (0x01)	17 (0x11)
Torque.FoldbackRMSCurrentLevel	UINT	114 (0x72)	1 (0x01)	1 (0x01)
Torque.LimitedTorqueCommand	INT	114 (0x72)	1 (0x01)	2 (0x02)
Torque.TorqueCommand	INT	114 (0x72)	1 (0x01)	3 (0x03)
Torque.TorqueLevel1	UINT	114 (0x72)	1 (0x01)	4 (0x04)
Torque.TorqueLevel2	UINT	114 (0x72)	1 (0x01)	5 (0x05)
Torque.TorqueLimit	UINT	114 (0x72)	1 (0x01)	6 (0x06)

Name	DN Data Type	Class	Instance	Attribute
Tuning.EnableFeedforwards	WORD	115 (0x73)	1 (0x01)	1 (0x01)
Tuning.Friction	UINT	115 (0x73)	1 (0x01)	2 (0x02)
Tuning.LoadIntertia	UINT	115 (0x73)	1 (0x01)	3 (0x03)
Tuning.LowPassFilterEnable	WORD	115 (0x73)	1 (0x01)	4 (0x04)
Tuning.LowPassFilterFrequency	UINT	115 (0x73)	1 (0x01)	5 (0x05)
Tuning.PositionErrorIntegral	DINT	115 (0x73)	1 (0x01)	6 (0x06)
Tuning.PositionErrorIntegralEnable	WORD	115 (0x73)	1 (0x01)	7 (0x07)
Tuning.PositionErrorIntegralTimeConstant	UINT	115 (0x73)	1 (0x01)	8 (0x08)
Tuning.ResponseLevel	UINT	115 (0x73)	1 (0x01)	10 (0x0A)
UserUnits.AccelerationDecimalPoint	UINT	115 (0x73)	1 (0x01)	1 (0x01)
UserUnits.PositionDecimalPoint	UINT	116 (0x74)	1 (0x01)	2 (0x02)
UserUnits.TimeBasedIndication	UINT	116 (0x74)	1 (0x01)	3 (0x03)
UserUnits.UserDefinedBitmap	UINT	116 (0x74)	1 (0x01)	21 (0x15)
UserUnits.UserDefinedBits	BOOL	116 (0x74)	1 (0x01)	22 (0x16)
UserUnits.UserDefinedRegisters	UINT	116 (0x74)	1 (0x01)	23 (0x17)
UserUnits.UserUnitDefinition	UINT	116 (0x74)	1 (0x01)	4 (0x04)
UserUnits.UserUnitString	String	116 (0x74)	1 (0x01)	5 (0x05)
UserUnits.UsingUserUnits	BOOL	116 (0x74)	1 (0x01)	6 (0x06)
UserUnits.VelocityDecimalPoint	UINT	116 (0x74)	1 (0x01)	7 (0x07)
Velocity.InMotionVelocity	UINT	117 (0x75)	1 (0x01)	1 (0x01)
Velocity.OverspeedVelocity	UINT	117 (0x75)	1 (0x01)	2 (0x02)
Velocity.StopDeceleration	UDINT	117 (0x75)	1 (0x01)	3 (0x03)
Velocity.TravelLimitDeceleration	UDINT	117 (0x75)	1 (0x01)	4 (0x04)

By DeviceNet Class

NOTE 1

The instance number will be one more than the number of the index, jog, pls, program, input or output. For example, Index0 will have an instance of 1 (0x01) and Index 14 will have an instance of 15 (0x0F). If the parameter is unnumbered, the instance will be one (see the tables in this chapter).

Name	DN Data Type	Class	Instance	Attribute
DeviceNet.DeviceNetIdentityObject.VendorID	UINT	1 (0x01)	1 (0x01)	1 (0x01)
DeviceNet.DeviceNetIdentityObject.DeviceType	UINT	1 (0x01)	1 (0x01)	2 (0x02)
ProductSubGroup	UINT16	1 (0x01)	1 (0x01)	3 (0x03)
ModuleFirmwareRevision	String:4	1 (0x01)	1 (0x01)	4 (0x04)
DeviceNet.DeviceNetIdentityObject.DeviceStatus	UINT	1 (0x01)	1 (0x01)	5 (0x05)
ModuleSerialNumber	String:16	1 (0x01)	1 (0x01)	6 (0x06)
DeviceNet.DeviceNetIdentityObject.HeartbeatInterval	DINT	1 (0x01)	1 (0x01)	10 (0x0A)
AnalogOutput.AnalogOutputInstance#.Channel	INT	100 (0x64)	See Note 1	1 (0x01)
AnalogOutput.AnalogOutputInstance#.Offset	DINT	100 (0x64)	See Note 1	2 (0x02)
AnalogOutput.AnalogOutputInstance#.Scale	DINT	100 (0x64)	See Note 1	3 (0x03)
AnalogOutput.AnalogOutputInstance#.Select	INT	100 (0x64)	See Note 1	4 (0x04)
Execute.ExecuteClearFault	BOOL	101 (0x65)	1 (0x01)	1 (0x01)
Execute.ExecuteReadBaseDriveNVMtoRAM	BOOL	101 (0x65)	1 (0x01)	2 (0x02)
Execute.ExecuteStopAllMotion	BOOL	101 (0x65)	1 (0x01)	3 (0x03)
Execute.ExecuteWriteBaseDriveRAMtoNVM	BOOL	101 (0x65)	1 (0x01)	4 (0x04)
Fault.FaultInstance#.Code	WORD	102 (0x66)	See Note 1	1 (0x01)
Fault.FaultInstance#.PowerUpCount	UINT	102 (0x66)	See Note 1	2 (0x02)
Fault.FaultInstance#.PowerUpTime	UDINT	102 (0x66)	See Note 1	3 (0x03)
Fault.FaultStatus	DWORD	102 (0x66)	1 (0x01)	15 (0x0F)
Home.BackOffSensorBeforeHoming	WORD	103 (0x67)	1 (0x01)	1 (0x01)
Home.EndofHomePosition	DINT	103 (0x67)	1 (0x01)	2 (0x02)
Home.HomeAcceleration	UDINT	103 (0x67)	1 (0x01)	3 (0x03)
Home.HomeDeceleration	UDINT	103 (0x67)	1 (0x01)	4 (0x04)
Home.HomeLimitDistance	UDINT	103 (0x67)	1 (0x01)	5 (0x05)
Home.HomeLimitDistanceEnable	WORD	103 (0x67)	1 (0x01)	6 (0x06)
Home.HomeOffset	DINT	103 (0x67)	1 (0x01)	7 (0x07)
Home.HomeOffsetEnable	WORD	103 (0x67)	1 (0x01)	8 (0x08)
Home.HomeReference	WORD	103 (0x67)	1 (0x01)	9 (0x09)
Home.HomeVelocity	DINT	103 (0x67)	1 (0x01)	10 (0x0A)
Index.IndexInstance#.ChainNext	UINT	104 (0x68)	See Note 1	1 (0x01)
Index.IndexInstance#.Acceleration	UDINT	104 (0x68)	See Note 1	2 (0x02)
Index.IndexInstance#.ControlRegister	UINT	104 (0x68)	See Note 1	3 (0x03)
Index.IndexInstance#.Count	UINT	104 (0x68)	See Note 1	4 (0x04)
Index.IndexInstance#.Deceleration	UDINT	104 (0x68)	See Note 1	5 (0x05)
Index.IndexInstance#.Distance	DINT	104 (0x68)	See Note 1	6 (0x06)
Index.IndexInstance#.Dwell	UINT	104 (0x68)	See Note 1	7 (0x07)
Index.IndexInstance#.Type	WORD	104 (0x68)	See Note 1	8 (0x08)

Name	DN Data Type	Class	Instance	Attribute
Index.IndexInstance#.Velocity	UDINT	104 (0x68)	See Note 1	9 (0x09)
Index.Instance#.RegistrationOffset	DINT	104 (0x68)	See Note 1	10 (0x0A)
Index.CurrentIndexCount	UINT	104 (0x68)	1 (0x01)	21 (0x15)
Index.CurrentIndexNumber	UINT	104 (0x68)	1 (0x01)	22 (0x16)
Index.ChainingCount	UINT	104 (0x68)	1 (0x01)	23 (0x17)
Index.CurrentChainingCount	UINT	104 (0x68)	1 (0x01)	24 (0x18)
Index.InfiniteChaining	WORD	104 (0x68)	1 (0x01)	25 (0x19)
InputFunction.InputFunctionInstance#.ActiveOffArray	BOOL	105 (0x69)	See Note 1	1 (0x01)
InputFunction.InputFunctionInstance#.AlwaysActive Array	BOOL	105 (0x69)	See Note 1	2 (0x02)
InputFunction.InputFunctionInstance#.Mapping	WORD	105 (0x69)	See Note 1	3 (0x03)
InputFunction.InputFunctionInstance#.StatusArray	BOOL	105 (0x69)	See Note 1	4 (0x04)
InputFunction.InputFunctionAlwaysActiveBitMap	WORD	105 (0x69)	1 (0x01)	21 (0x15)
InputFunction.InputFunctionPolarityBitMap	WORD	105 (0x69)	1 (0x01)	22 (0x16)
InputFunction.InputFunctionStatusBitMap	WORD	105 (0x69)	1 (0x01)	23 (0x17)
InputLine.InputLineInstance#.RawStatusArray	BOOL	106 (0x6A)	See Note 1	0 (0x00)
InputLine.InputLineInstance#.StatusArray	BOOL	106 (0x6A)	See Note 1	0 (0x00)
InputLine.InputLineInstance#.DebouncedStatusArray	BOOL	106 (0x6A)	See Note 1	1 (0x01)
InputLine.InputLineInstance#.ForceOn/OffCommandArray	BOOL	106 (0x6A)	See Note 1	2 (0x02)
InputLine.InputLineInstance#.ForceOn/OffEnableArray	BOOL	106 (0x6A)	See Note 1	3 (0x03)
InputLine.InputLineInstance#.DebounceTime	UINT	106 (0x6A)	See Note1	4 (0x04)
InputLine.EnableDebounceTime	UINT	106 (0x6A)	1 (0x01)	8 (0x08)
InputLine.DriveEnableInputDebouncedStatus	BOOL	106 (0x6A)	1 (0x01)	9 (0x09)
InputLine.DriveEnableInputRawStatus	BOOL	106 (0x6A)	1 (0x01)	10 (0x0A)
InputLine.DriveEnableInputStatus	BOOL	106 (0x6A)	1 (0x01)	11 (0x0B)
InputLine.InputLinesDebouncedBitMap	WORD	106 (0x6A)	1 (0x01)	21 (0x15)
InputLine.InputLinesOverrideActive	UINT	106 (0x6A)	1 (0x01)	22 (0x16)
InputLIne.InputLinesOverrideStatus	UINT	106 (0x6A)	1 (0x01)	23 (0x17)
InputLine.InputLinesRawBitMap	WORD	106 (0x6A)	1 (0x01)	24 (0x18)
InputLine.InputLinesStatusBitMap	WORD	106 (0x6A)	1 (0x01)	25 (0x19)
Jog.JogAcceleration	UDINT	107 (0x6B)	1 (0x01)	1 (0x01)
Jog.JogDeceleration	UDINT	107 (0x6B)	1 (0x01)	2 (0x02)
Jog.JogFastVelocity	UDINT	107 (0x6B)	1 (0x01)	3 (0x03)
Jog.JogVelocity	UDINT	107 (0x6B)	1 (0x01)	4 (0x04)
OutputFunction.OutputFunctionInstance#.Mapping32	UINT	108 (0x6C)	See Note 1	1 (0x01)
OutputFunction.OutputFunctionStatusBitMap	WORD	108 (0x6C)	1 (0x01)	21 (0x15)
OutputLine.OutputLinesOverrideActive	UINT	109 (0x6D)	1 (0x01)	1 (0x01)
OutputLine.OutputLinesOverrideStatus	UINT	109 (0x6D)	1 (0x01)	2 (0x02)
OutputLine.OutputLinesPolarityBitMap	WORD	109 (0x6D)	1 (0x01)	3 (0x03)

Name	DN Data Type	Class	Instance	Attribute
OutputLine.OutputLinesStatusBitMap	WORD	109 (0x6D)	1 (0x01)	4 (0x04)
Position.FollowingError	DINT	110 (0x6E)	1 (0x01)	1 (0x01)
Position.FollowingErrorEnable	WORD	110 (0x6E)	1 (0x01)	2 (0x02)
Position.FollowingErrorLimit	DINT	110 (0x6E)	1 (0x01)	3 (0x03)
Position.RolloverPosition	UDINT	110 (0x6E)	1 (0x01)	4 (0x04)
Position.RolloverPositionEnable	WORD	110 (0x6E)	1 (0x01)	5 (0x05)
Position.InPositionWindow	UDINT	110 (0x6E)	1 (0x01)	6 (0x06)
Position.InPositionTime	UINT	110 (0x6E)	1 (0x01)	7 (0x07)
ProductID.FirmwareRevisionBase	String	111 (0x6F)	1 (0x01)	1 (0x01)
ProductID.FirmwareRevisionOption	String	111 (0x6F)	1 (0x01)	2 (0x02)
ProductID.InterfaceRevisionBase	UINT	111 (0x6F)	1 (0x01)	3 (0x03)
ProductID.InterfaceRevisionOption	UINT	111 (0x6F)	1 (0x01)	4 (0x04)
ProductID.Option1IDFunctionModule	UINT	111 (0x6F)	1 (0x01)	5 (0x05)
ProductID.Option2IDAnybus1	UINT	111 (0x6F)	1 (0x01)	6 (0x06)
ProductID.Option3IDAnybus2	UINT	111 (0x6F)	1 (0x01)	7 (0x07)
ProductID.ProductGroup	UINT	111 (0x6F)	1 (0x01)	8 (0x08)
ProductID.ProductID	UINT	111 (0x6F)	1 (0x01)	9 (0x09)
ProductID.ProductSerialNumber	String	111 (0x6F)	1 (0x01)	10 (0x0A)
ProductID.ProductSubGroup	UINT	111 (0x6F)	1 (0x01)	11 (0x0B)
Setup.DriveAmbientTemperature	UINT	112 (0x70)	1 (0x01)	1 (0x01)
Setup.DriveAxisName	String	112 (0x70)	1 (0x01)	2 (0x02)
Setup.EncoderOutput	UINT	112 (0x70)	1 (0x01)	3 (0x03)
Setup.EncoderOutputEnable	WORD	112 (0x70)	1 (0x01)	4 (0x04)
Setup.LowDCBusEnable	WORD	112 (0x70)	1 (0x01)	5 (0x05)
Setup.PositiveDirection	WORD	112 (0x70)	1 (0x01)	6 (0x06)
Setup.ExecuteReadFMNVMtoRAM	BOOL	112 (0x70)	1 (0x01)	21 (0x15)
Status.AbsoluteRotorPosition	DINT	113 (0x71)	1 (0x01)	1 (0x01)
Status.BusVoltage	UINT	113 (0x71)	1 (0x01)	2 (0x02)
Status.CommutationAngleCorrection	INT	113 (0x71)	1 (0x01)	3 (0x03)
Status.CommutationTrackAngle	UINT	113 (0x71)	1 (0x01)	4 (0x04)
Status.CommutationVoltage	INT	113 (0x71)	1 (0x01)	5 (0x05)
Status.EncoderCounts	DINT	113 (0x71)	1 (0x01)	6 (0x06)
Status.HeatsinkRMS	UINT	113 (0x71)	1 (0x01)	7 (0x07)
Status.MotionState	WORD	113 (0x71)	1 (0x01)	8 (0x08)
Status.PowerUpCount	UINT	113 (0x71)	1 (0x01)	9 (0x09)
Status.PowerUpTime	UDINT	113 (0x71)	1 (0x01)	10 (0x0A)
Status.RotorVelocity	DINT	113 (0x71)	1 (0x01)	11 (0x0B)
Status.SegmentDisplayCharacter	UINT	113 (0x71)	1 (0x01)	12 (0x0C)

Name	DN Data Type	Class	Instance	Attribute
Status.OptionSerialNumber	String	113 (0x71)	1 (0x01)	13 (0x0D)
Status.TorqueFeedback	INT	113 (0x71)	1 (0x01)	14 (0x0D)
Status.TotalCommandChangeOut	DINT	113 (0x71)	1 (0x01)	15 (0x0E)
Status.TotalPowerUpTime	UDINT	113 (0x71)	1 (0x01)	16 (0x10)
Status.VelocityCommand	DINT	113 (0x71)	1 (0x01)	17 (0x11)
Torque.FoldbackRMSCurrentLevel	UINT	114 (0x72)	1 (0x01)	1 (0x01)
Torque.LimitedTorqueCommand	INT	114 (0x72)	1 (0x01)	2 (0x02)
Torque.TorqueCommand	INT	114 (0x72)	1 (0x01)	3 (0x03)
Torque.TorqueLevel1	UINT	114 (0x72)	1 (0x01)	4 (0x04)
Torque.TorqueLevel2	UINT	114 (0x72)	1 (0x01)	5 (0x05)
Torque.TorqueLimit	UINT	114 (0x72)	1 (0x01)	6 (0x06)
Tuning.EnableFeedforwards	WORD	115 (0x73)	1 (0x01)	1 (0x01)
Tuning.Friction	UINT	115 (0x73)	1 (0x01)	2 (0x02)
Tuning.LoadIntertia	UINT	115 (0x73)	1 (0x01)	3 (0x03)
Tuning.LowPassFilterEnable	WORD	115 (0x73)	1 (0x01)	4 (0x04)
Tuning.LowPassFilterFrequency	UINT	115 (0x73)	1 (0x01)	5 (0x05)
Tuning.PositionErrorIntegral	DINT	115 (0x73)	1 (0x01)	6 (0x06)
Tuning.PositionErrorIntegralEnable	WORD	115 (0x73)	1 (0x01)	7 (0x07)
Tuning.PositionErrorIntegralTimeConstant	UINT	115 (0x73)	1 (0x01)	8 (0x08)
Tuning.ResponseLevel	UINT	115 (0x73)	1 (0x01)	10 (0x0A)
UserUnits.AccelerationDecimalPoint	UINT	116 (0x74)	1 (0x01)	1 (0x01)
UserUnits.PositionDecimalPoint	UINT	116 (0x74)	1 (0x01)	2 (0x02)
UserUnits.TimeBasedIndication	UINT	116 (0x74)	1 (0x01)	3 (0x03)
UserUnits.UserUnitDefinition	UINT	116 (0x74)	1 (0x01)	4 (0x04)
UserUnits.UserUnitString	String	116 (0x74)	1 (0x01)	5 (0x05)
UserUnits.UsingUserUnits	BOOL	116 (0x74)	1 (0x01)	6 (0x06)
UserUnits.VelocityDecimalPoint	UINT	116 (0x74)	1 (0x01)	7 (0x07)
UserUnits.UserDefinedBitmap	UINT	116 (0x74)	1 (0x01)	21 (0x15)
UserUnits.UserDefinedBits	BOOL	116 (0x74)	1 (0x01)	22 (0x16)
UserUnits.UserDefinedRegisters	UINT	116 (0x74)	1 (0x01)	23 (0x17)
Velocity.InMotionVelocity	UINT	117 (0x75)	1 (0x01)	1 (0x01)
Velocity.OverspeedVelocity	UINT	117 (0x75)	1 (0x01)	2 (0x02)
Velocity.StopDeceleration	UDINT	117 (0x75)	1 (0x01)	3 (0x03)
Velocity.TravelLimitDeceleration	UDINT	117 (0x75)	1 (0x01)	4 (0x04)

Appendix

Index Select Predefined

Master Receive Assembly Block - Index Sel (Predef)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	End of Index	End of Chaining Count	End of Index Count	End of Index Motion	Travel Limit -	Travel Limit +	Reg Limit Distance Hit	Brake Release	Enable State	In - Motion	In + Motion	At Velocity	End of Home	Torque Limit	Fault	Drive OK
1	Reserved	Reserved	Reserved	Input Word Select Data Pointer					Registration Sensor 1 Status	Registration Sensor 2 Status	Home Sensor Status	Absolute Position Valid	Home Limit Dist Hit	Motion State Bit 2	Motion State Bit 1	Motion State Bit 0
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Index Sel (Predef)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Index Select Bit 3	Index Select Bit 2	Index Select Bit 1	Index Select Bit 0	Jog -	Jog +	Home Initiate	Start Index	DN Bit 7 MS	DN Bit 6 MS	DN Bit 5 MS	DN Bit 4 MS	DN Bit 3 MS	DN Bit 2 MS	DN Bit 1 MS	DN Bit 0 MS
1	Reserved	Reserved	Reset	Output Word Select Data Pointer					Enable	Stop	Define Home	Input Word Select Data Pointer				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Index Select (User Defined)

Master Receive Assembly Block - Index Sel (User Def)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DN Bit 15 MR	DN Bit 14 MR	DN Bit 13 MR	DN Bit 12 MR	DN Bit 11 MR	DN Bit 10 MR	DN Bit 9 MR	DN Bit 8 MR	DN Bit 7 MR	DN Bit 6 MR	DN Bit 5 MR	DN Bit 4 MR	DN Bit 3 MR	DN Bit 2 MR	DN Bit 1 MR	DN Bit 0 MR
1	Reserved	Reserved	Reserved	Input Word Select Data Pointer					Index Select Bit 3	Index Select Bit 2	Index Select Bit 1	Index Select Bit 0	Enable State	Motion State Bit 2	Motion State Bit 1	Motion State Bit 0
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Index Sel (User Def)

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Index Select Bit 3	Index Select Bit 2	Index Select Bit 1	Index Select Bit 0	DN Bit 11 MS	DN Bit 10 MS	DN Bit 9 MS	DN Bit 8 MS	DN Bit 7 MS	DN Bit 6 MS	DN Bit 5 MS	DN Bit 4 MS	DN Bit 3 MS	DN Bit 2 MS	DN Bit 1 MS	DN Bit 0 MS
1	Reserved	Reserved	Reserved	Output Word Select Data Pointer					Reserved	Reserved	Enable	Input Word Select Data Pointer				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Index Type

Master Receive Assembly Block - Index Type

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DN Bit 7 MR	DN Bit 6 MR	DN Bit 5 MR	DN Bit 4 MR	DN Bit 3 MR	DN Bit 2 MR	DN Bit 1 MR	DN Bit 0 MR	Enable State	Absolute Position Valid	Home Limit Distance Hit	Reg Limit Distance Hit	End of Home	Torque Limit	Fault	Drive OK
1	Reserved	Reserved	Reserved	Input Word Select Data Pointer					Reserved	Reserved	Reserved	Travel Limit -	Travel Limit +	Motion State Bit 2	Motion State Bit 1	Motion State Bit 0
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block - Index Type

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	DN Bit 7 MS	DN Bit 6 MS	DN Bit 5 MS	DN Bit 4 MS	DN Bit 3 MS	DN Bit 2 MS	DN Bit 1 MS	DN Bit 0 MS	Enable	Reset	Stop	Home Initiate	Index Type Bit 2	Index Type Bit 1	Index Type Bit 0	Start Index
1	Reserved	Reserved	Reserved	Output Word Select Data Pointer					Jog Fast	Jog -	Jog +	Input Word Select Data Pointer				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Position Control

Master Receive Assembly Block - Position Control

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved								Enable State	Valid Data = 1 Ignore Data = 0	Absolute Position Valid	Stop Input	Fault	End of Index Motion	Reserved	Trajectory Started
1	Reserved	Reserved	Reserved	Response Assembly Code					Command Error	Reserved	Trajectory Start Echo	Reserved	Reserved	CCW Hardware Limit (Travel Limit -)	CW Hardware Limit (Travel Limit +)	Drive OK
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Master Send Assembly Block -Position Control

Word	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved								Enable	Valid Data = 1 Ignore Data = 0	Reserved	Stop	Reserved	Absolute=0 Incremental=1	Reserved	Start Trajectory
1	Reserved			Response Assembly Code					Reserved			Command Assembly Code				
2	Data Low Word															LS Bit
3	MS Bit	Data High Word														

Glossary

Application Objects

These implement the intended purpose of the product.

Attribute

A sub-classification for a parameter or bit. The attribute is grouped directly under the more broad category of class. Example: Class = Index, Attribute = Index Dwell. Each Attribute accessible to the user is assigned a number (See “Drive Parameters” on page 61.)

Baud Rate

The number of times the communication signal changes per second. In the case of a digital communication signal, it is equal to the number of bits per second.

Class

A top level DeviceNet classification for all parameters and bits. Each class is given a unique number ID found in the chart #.

Connection Class

This allocates and manages internal resources associated with both I/O and Explicit Messaging connections.

Connection Object

This manages the communication specific aspects associated with both I/O and Explicit Messaging connections.

Daisy Chain

A slang term for a wiring process that goes from one device to the next connecting wire A to A, B to B, etc.

DeviceNet

A communications link to connect industrial devices (limit switches, photoelectric switches, motor drives, and position controllers) as well as control devices (programmable controllers and computers) to a network. All references to DeviceNet in this document refer to ODVA DeviceNet Specifications Volumes 1 and 2, release 2.0.

DeviceNet Object

This provides the configuration and status of a physical DeviceNet network connection.

Drop Line

A DeviceNet cable that runs from the trunk of a system to a device.

EDS

Electronic Data Sheet file is a formatted ASCII file that contains configuration information of the device.

Epsilon Drive

The Epsilon drive is a digital positioning drive. In addition to the basic single axis motion control features, it provides advanced diagnostics and high speed communication capabilities.

Explicit Message

A message constructed within the master to poll a device for a single parameter. Explicit Messages occur in the background of implicit messaging.

Explicit Messaging Connections

Provide generic, multi-purpose communication paths between two devices. Explicit Messages provide the typical request/response oriented network communications.

Implicit Message

A message sent from the master to the slave as a way of passing data. Implicit messages transfer a predefined amount of data to and from the Master at a constant rate.

Index

An index is a set of parameters that defines position based motion including target position and velocity, and other parameters. The different types of indexes specify basic operation. Indexes are associated with digital input and outputs for the purposes of providing control and indication of the Index status.

Indexer

Another term for the Epsilon Ei drive, which is intended for indexing.

Instance

A sub-classification for a parameter or bit. The instance is grouped directly under the more broad category of attribute and allows for multiple occurrences of a parameter. Example: Class = Index, Attribute = Index Dwell, Instance1 = Index 0 Dwell, Instance2 = Index 1 Dwell. Each instance is assigned a unique number under its respective attribute.

I/O Connections

These provide dedicated, special-purpose communication paths between a producing application and one or more consuming applications. Application specific I/O data moves through these ports.

Link Consumer Object

This object is used by a Connection Object to receive data from DeviceNet.

Link Producer Object

This object is used by a Connection Object to transmit data onto DeviceNet.

MacID

DeviceNet specific term for a node address.

Message Router

This distributes Explicit Request Messages to the appropriate handler object.

Nodes

Each Device on a DeviceNet network is called a node

ODVA

The Open DeviceNet Vendor Association is an organization that manages the DeviceNet specification and supports the worldwide growth of DeviceNet.

One touch Configuration

Process for changing the baud rate and MacID on the Ei-DN using the reset button on the front of the Ei-DN

PLC

Programmable Logic Controller. Used to control Inputs and Outputs in a systematic fashion on a device.

Trunk Line

The main branch of a DeviceNet network. This branch generally attaches to the main power supply and is capable of carrying more current than the drop lines hanging off of it.

Terminating Resistors

Resistors used to decrease reflection and noise in a serial communications network. Generally terminating resistors are placed at the ends of the network on the lines that transmit messages.

UCMM

The Unconnected Message Manager processes DeviceNet Unconnected Explicit messages.

Index

A

B

C

D

E

F

W

Since 1979, the "Motion Made Easy" products, designed and manufactured in Minnesota U.S.A., are renowned in the motion control industry for their ease of use, reliability and high performance.

For more information about Control Techniques "Motion Made Easy" products and services, call (800) 397-3786 or contact our website at www.emersonct.com.

Control Techniques Drives, Inc
Division of EMERSON Co.
12005 Technology Drive
Eden Prairie, Minnesota 55344
U.S.A.

Customer Service

Phone: (952) 995-8000 or (800) 397-3786

Fax: (952) 995-8129
Technical Support

Phone: (952) 995-8033 or (800) 397-3786

Fax (952) 9995-8020