REAL TIME AUTOMATION
2825 N. Mayfair Rd. Suite 111
Wauwatosa, WI 53222
(414) 453-5100
www.rtaautomation.com

EtherNet/IP - DeviceNet Master Gateway Catalog #455ED

MODBUS TCP - DeviceNet Master Gateway Catalog #455EMT

DEFAULT IP ADDRESS 192.168.0.100

Revision History

Date	Name	Revision	Description
08/24/2006	DJK	1.0	Initial Revision
06/28/2007	JDW	1.1	Expanded Modbus TCP support

Table of Contents

INTRODUCTION

Overview

The 455ED/455EMT Ethernet to DeviceNet Master Gateway converts a network of DeviceNet Slave devices to a single node of I/O on an EtherNet/IP network or a series of registers on Modbus TCP.

For EtherNet/IP Systems I/O data read from DeviceNet slave nodes is presented to an EtherNet/IP Client device as Input data. Output data transmitted by an EtherNet/IP Client is used to update the I/O data of DeviceNet Slave devices. The entire network of DeviceNet Slave devices appears to the EtherNet/IP Client as a single node of EtherNet/IP I/O.

The 455ED/455EMT DeviceNet Master can scan up to 32 DeviceNet Slave devices using the DeviceNet CAN communications. I/O data collected from the DeviceNet slave devices is combined to form input data on the EtherNet network. Output data from the EtherNet network is written to the DeviceNet Slave devices.

DeviceNet slave devices can be added, removed, and configured using a standard web based interface. Such things as DeviceNet baud rate and Mac ID can be set from the web server. The EtherNet/IP side can be configured also. The IP address, gateway, etc can be configured from the web server. The status and information about DeviceNet devices can be viewed from the web server once a device is added properly.

The drawing in **Figure 1** represents the devices that might be found in an EtherNet/IP - DeviceNet system. In this sample system, there are three DeviceNet Slave nodes connected to the DeviceNet Master communications port. The EtherNet/IP Server side of the 455ED/455EMT gateway is connected to a EtherNet/IP Client through an Ethernet switch. I/O data from the three DeviceNet Slave nodes appears to the PLC as I/O data also.

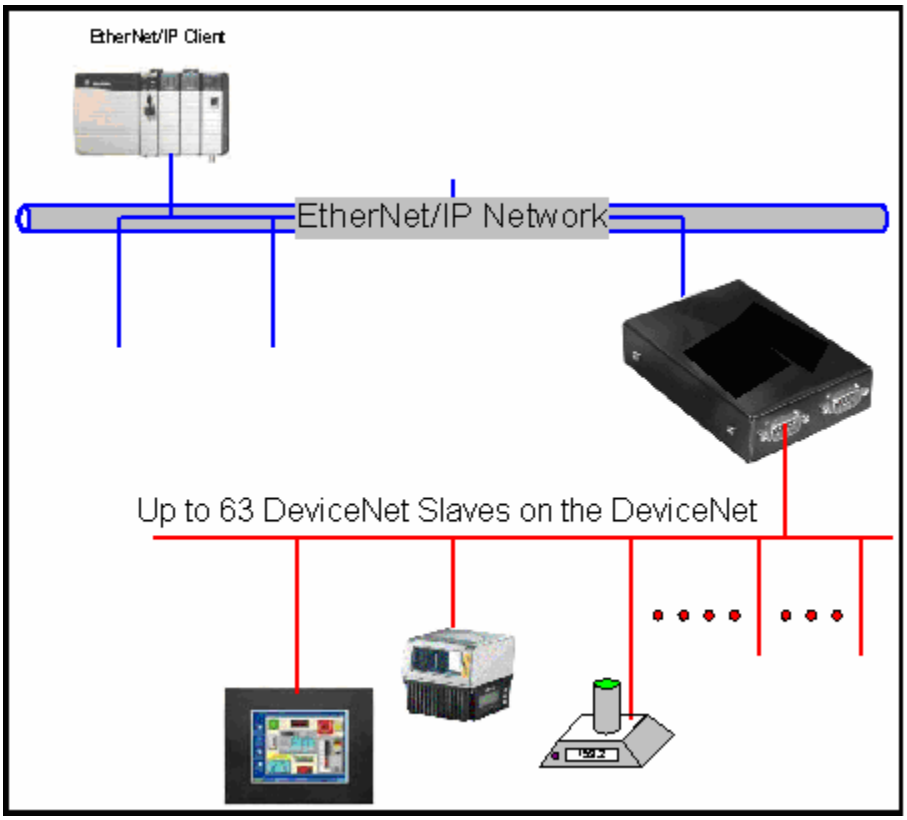


Figure 1 – Example EtherNet/IP DeviceNet System

Figure 2 illustrates how I/O data from the three DeviceNet devices in the previous example is mapped to EtherNet/IP. Four bytes from DeviceNet Slave 62, two bytes from DeviceNet Slave 3 and four bytes from DeviceNet Slave 19 form the input data on the EtherNet/IP network.

EtherNet/IP output data is mapped in exactly the same way. Some portion of the output data is mapped as I/O data and written to the first DeviceNet Slave. The next set of bytes in the EtherNet/IP Output data is mapped to the next DeviceNet Slave and so on until all the EtherNet/IP output data is mapped.

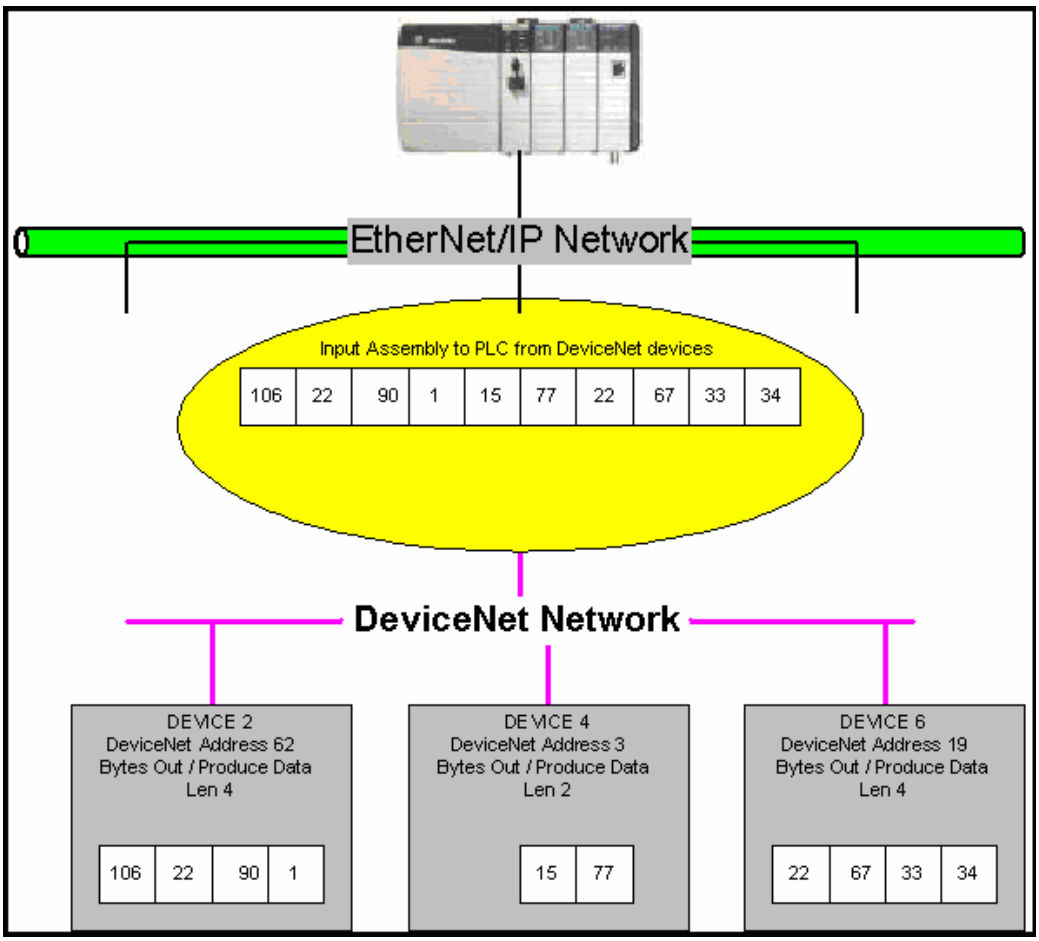


Figure 2 - EtherNet/IP DeviceNet Gateway Data Flow

Specifications

Ethernet Protocol:	EtherNet/IP Modbus TCP
DeviceNet Protocol:	DeviceNet Master
DeviceNet Slave Devices	1 to 32 devices
CAN Ports	1
DeviceNet Communications Interface:	DeviceNet CAN
Maximum EtherNet/IP Input / Output Size (bytes)	400
Connectors:	One DB9 Male, One TSTRIP5, One RJ-45
Supported Baud Rates:	125k, 250k, 500k
IP Addressing	Fixed via Web Server
Power Requirements:	24 VDC (from DeviceNet)

Contents of this Manual

This manual provides detailed instructions on the 455ED/455EMT Ethernet DeviceNet Master Gateway.

This manual does not describe the Rockwell Automation ControlLogix or CompactLogix controllers, DeviceNet slave devices, or how to troubleshoot an Ethernet or DeviceNet network.

WIRING THE DEVICENET NETWORK

Overview

DeviceNet Slave devices are connected to the 455ED/455EMT DeviceNet Master Gateway using CAN. Up to 32 DeviceNet Slave devices can be connected to the DeviceNet Master. This section describes the electrical connection of the slave devices on the DeviceNet network.

DeviceNet Wiring

The DeviceNet Slave network is connected to Serial Port 0 on the 455ED/455EMT module. The Port 0 pinout is shown in **Figure 3**. The DeviceNet CAN cable can only be used on **Port 0.**

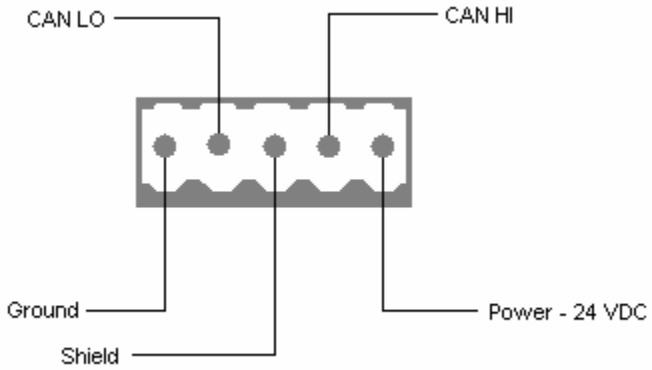


Figure 3 – CAN Port 0 Wiring for DeviceNet

CONFIGURING USING THE WEB SERVER

Overview

The 455ED/455EMT DeviceNet Master is configured using either the internal web server or the EtherNet/IP network. Configuration can be completed using either method.

The DeviceNet Master requires a list of DeviceNet Slave nodes to scan and their I/O data mapping.

This section describes how to enable the DeviceNet Master using the internal web server.

Accessing the Internal Web Server

The internal web server is accessed by entering the TCP/IP Address of the 455ED/455EMT Gateway into the URL Address box of your browser. Figure 4 illustrates how to enter the default address using Microsoft Internet Explorer. If you have modified the address, enter the current TCP/IP address instead of the default address.

Figure 4 - Accessing the Internal Web Server

Once your browser locates the 455ED/455EMT Gateway on your local network there are five steps to configure the DeviceNet network master.

Step 1 – Select DeviceNet Master Configuration

There are two Edit buttons and a Configuration button on the main screen (**Figure 5**) of the 455ED/455EMT Gateway. One Edit button configures the DeviceNet Serial parameters and the application description. The other Edit button configures the network settings of the 455ED/455EMT Module and is discussed in a later section. The configuration button modifies the DeviceNet Master operational settings.

Click on the “Config Page” button to start creating or changing your DeviceNet network configuration.

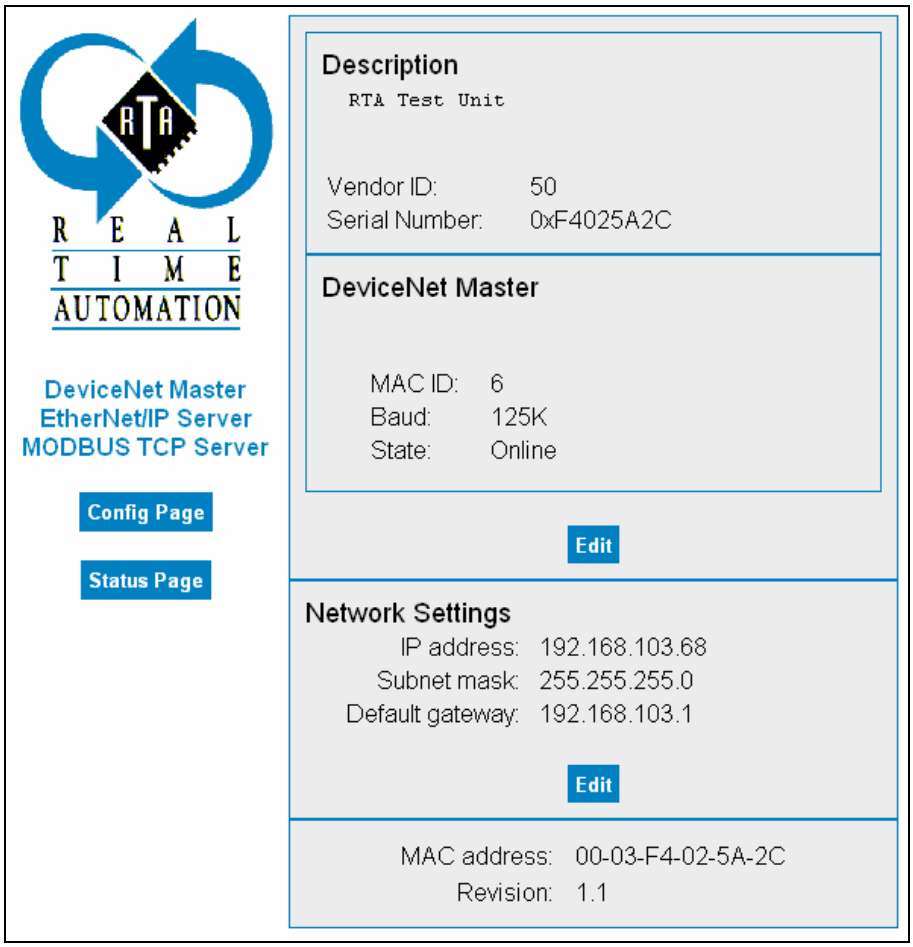


Figure 5 - 455ED/455EMT Web Server Main Page

Step 2 – Select a Device to Configure

From the DeviceNet Slave Configuration screen shown in **Figure 6** select a device to configure.

There are 32 possible devices to configure. The device number **is not** the DeviceNet Slave network address. It is simply a placeholder for the 32 possible devices. Any DeviceNet device can be assigned to any of the 32 possible devices. For example, DeviceNet Slave Address 19 can be assigned to Device 1, Address 5 can be assigned to Device 2 and so on in any order that you choose.

The device number is directly related to the mapping of the DeviceNet I/O data into EtherNet/IP I/O. Data from Device 1 is mapped first in the I/O structure. The data from Device 1 is followed by Device 2, Device 3 and so on.

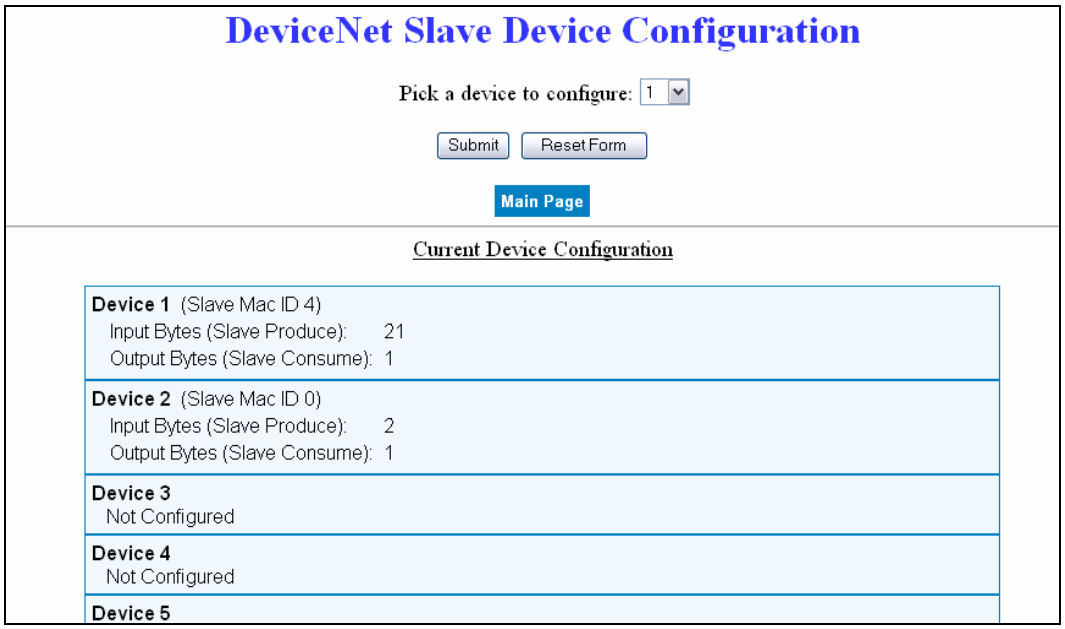


Figure 6 - 455ED/455EMT DeviceNet Master Configuration Page

Step 3 – Configure a DeviceNet Device

Once a Device is selected, there are five configuration parameters to set on the Device Configuration screen shown in Figure 7:

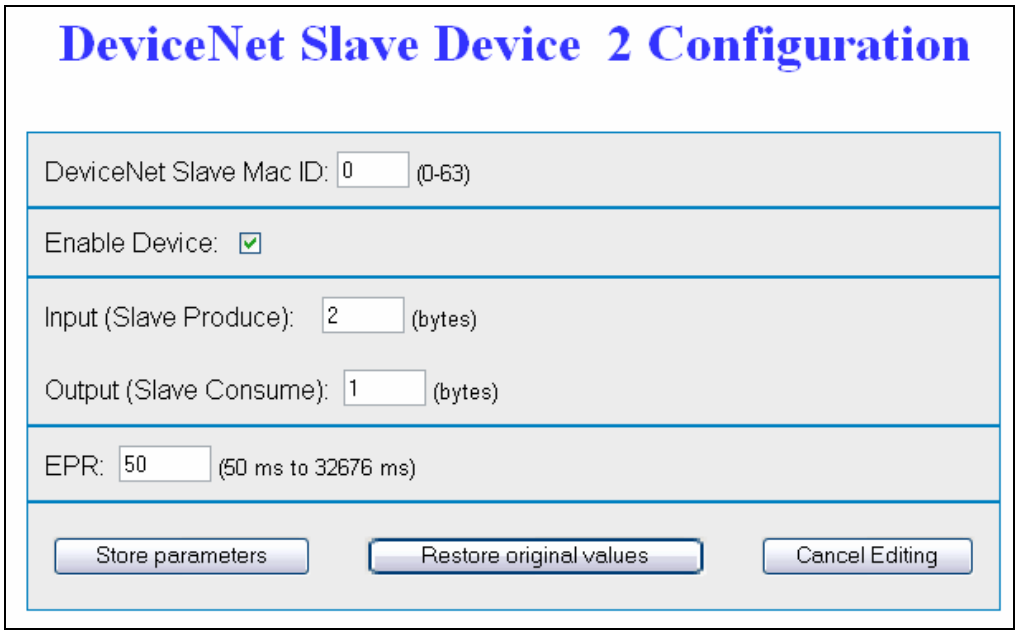


Figure 7 – Device Configuration Page

DeviceNet Slave Mac ID
This is the address of the DeviceNet slave on the DeviceNet network. This address cannot be the same as the DeviceNet master of the 455ED/455EMT.

Enable Device
This parameter is used to tell the DeviceNet master that the device on the DeviceNet network is enabled and the DeviceNet master of the 455ED/455EMT should communicate with the slave device. If a device is already configured and running and you decide, later on, to remove the node, just simply uncheck this value and the DeviceNet master of the 455ED/455EMT will stop communicating with the deice.

Input (Slave Produce)
This is the number of bytes the DeviceNet slave device produces as its input data to the DeviceNet network.

Output (Slave Consume)
This is the number of bytes the DeviceNet slave device consumes as its output data from the DeviceNet network.

EPR
The delay, in milliseconds, the DeviceNet master of the 455ED/455EMT waits before reading and writing I/O data to the DeviceNet slave. If there are a lot of devices on the DeviceNet network then you may want to set the number higher, if there are few devices on the DeviceNet network than you can set the number low.

Step 4 – Configure All Your Other DeviceNet Devices

Using the DeviceNet Master main configuration page configure the remainder of your devices. When complete, hit the “Store Parameters” button to return to the main page of the web server. Power must be cycled for any configuration changes to take effect.

Configuring CAN Communications

From the main web page shown in Figure 5 you can configure the CAN communications settings for the DeviceNet network.

Click on the Edit button under the DeviceNet Master Information block to edit the CAN communications settings or change the application description.

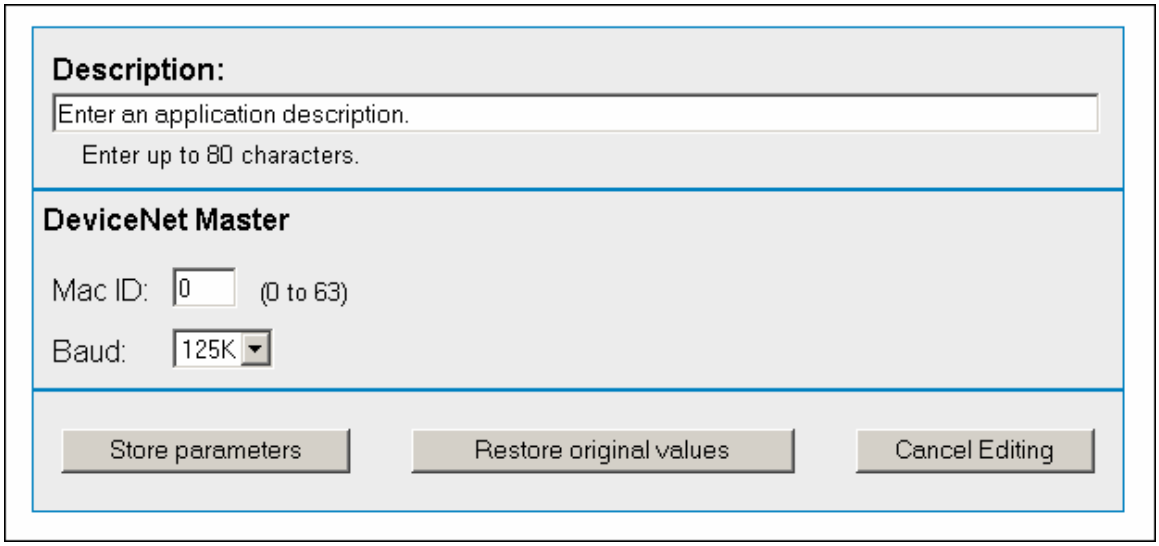


Figure 8 – DeviceNet serial configuration page

The serial communications screen shown in Figure 8 is used to set the following three serial communications parameters:

Description
This string is an optional text string that describes your application. The Application Description String is never sent over the network.

Mac ID
The Mac address of the DeviceNet master on the DeviceNet network, default value is 0.

Baud
The baud rate in which all devices on the DeviceNet network communicate with each other, the default value is 125 KB/s.

READING/WRITING DEVICENET DEVICES FROM ETHERNET/IP

Overview

The EtherNet/IP DeviceNet Gateway is a complete EtherNet/IP Server with all the required EtherNet/IP objects. It also includes the four vendor specific objects shown in

OBJECT CLASS	*Description*
64 Hex (100 Decimal)	Gateway Write Data Object
65 Hex (101 Decimal)	Gateway Read Data Object
66 Hex (102 Decimal)	Gateway Serial Configuration Object
67 Hex (103 Decimal)	Gateway Device Configuration Object

Figure 10 - EtherNet/IP Vendor Specific Objects

This section describes the EtherNet/IP Object Model and how to read/write the object data using EtherNet/IP Explicit and I/O messaging.

Object Model

The Object Model for the EtherNet/IP DeviceNet Gateway is shown in Figure 11:

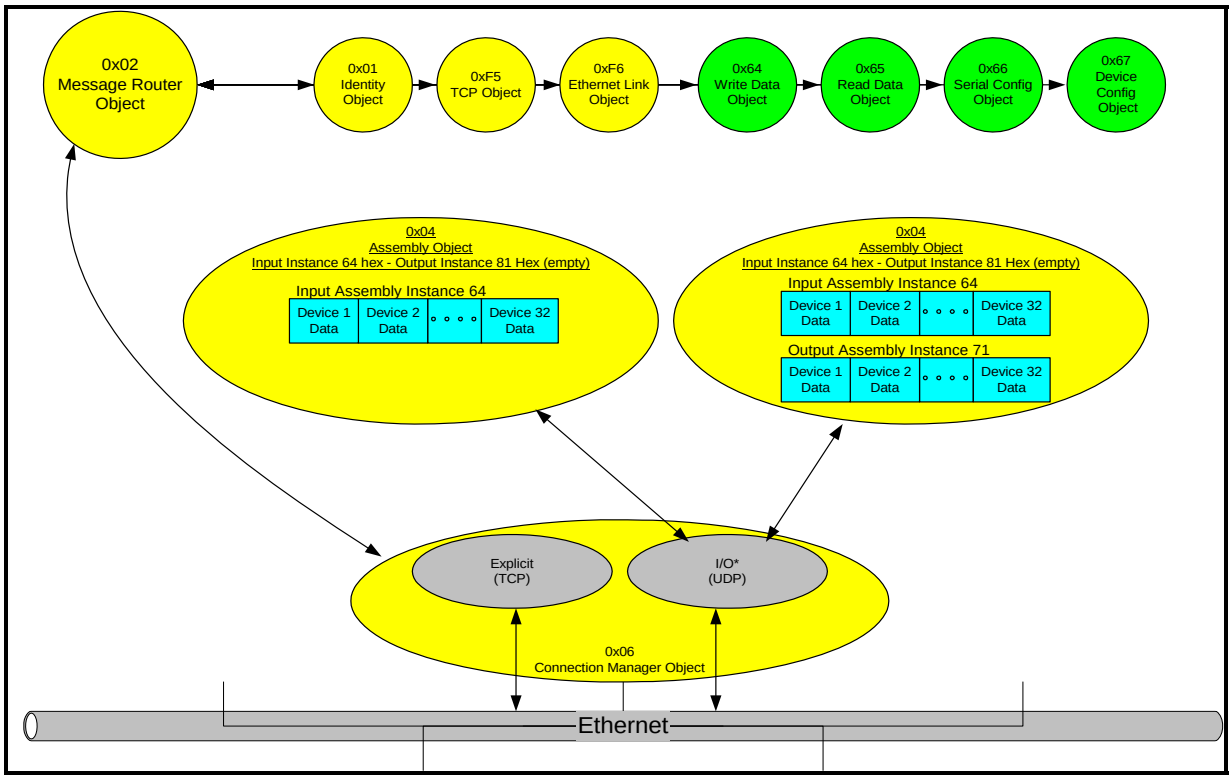


Figure 11 – EhterNet/IP DeviceNet Object Model

Explicit Messaging

EtherNet/IP Explicit Messaging Services are supported for all objects in the object model. The specific instances and attributes for each Object Class are presented in Appendix A.

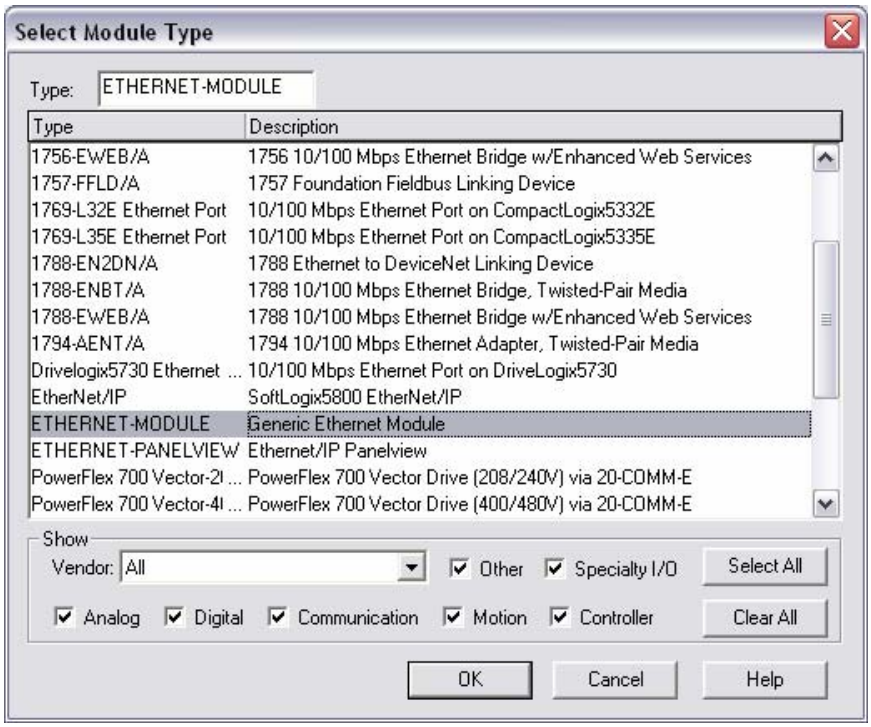
Assembly Object

There are two optional assembly instances. Option one supports input from the DeviceNet network to the EtherNet/IP network. Option two supports both input and output between the DeviceNet network and the EtherNet/IP network. The input and output assemblies are described in detail in the Appendix.

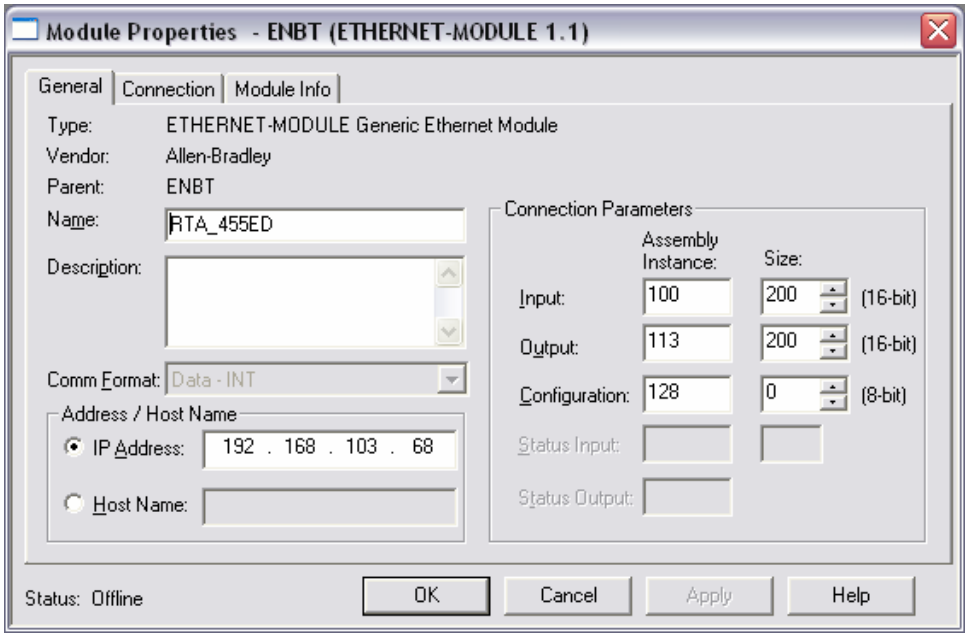
The gateway defaults to option 2 – input and output between the DeviceNet network and the EtherNet/IP network.

I/O Messaging (via ControlLogix)

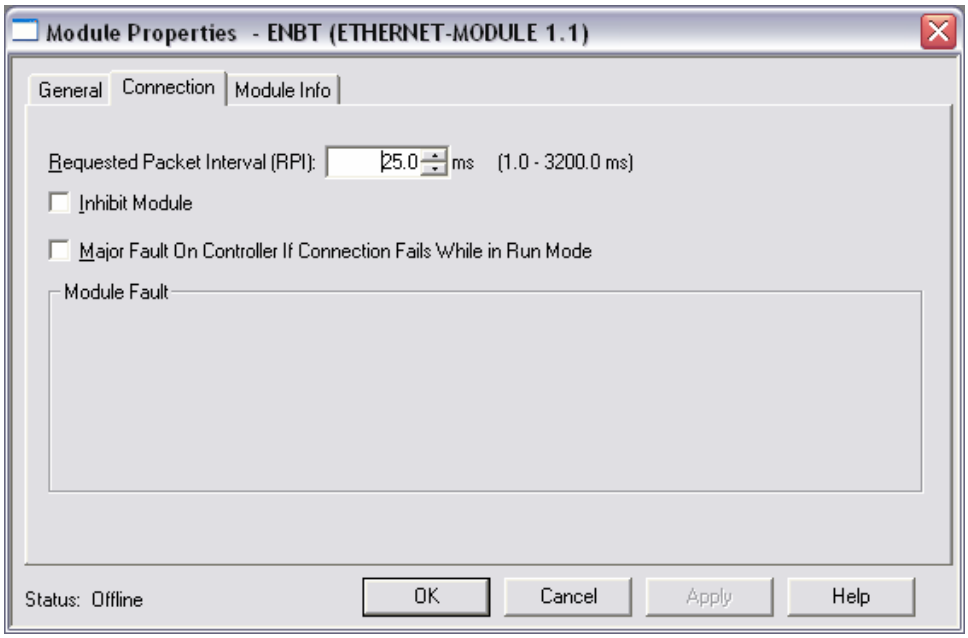
Using RsLogix5000, add an Ethernet module. Select “Generic Ethernet Module”



Configure the module to match the below window. Ensure you select “Data – INT’ as the Comm Format.



Select an appropriate RPI. This number should be the slowest acceptable update rate for any input or output. An RPI value faster than 25ms may cause intermittent timeouts. This value should be similar to the EPR on the DeviceNet network.



I/O Messaging (General)

I/O messaging is supported using the Option 2 Assembly Object, Input Instance 64 hex and Output Instance 71 hex.

One DeviceNet byte = 1 EtherNet/IP word

The input and output assemblies' instances are built by concatenating the data defined for each DeviceNet device (1 to 32) to form the input and output assemblies. For the input assemblies, DeviceNet device 1 data is stored in the input assembly followed by the data for device 2, device 3 and so on until all defined devices are added or 198 words are stored in the assembly. This assembly is the Input data transferred to the EtherNet/IP Client. The Input DeviceNet data appears as Input data in the Programmable Controller. The last 2 words of the assembly are a bit representation of the read/write error counters from the slave devices. If the bit is set, there are read and/or write errors.

Counter Registers (198 & 199)

First word of status (198) 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Device 16 ----------^
Device 15 ----------^
Device 14 ----------^
Device 1 ----------^
2nd word of status (199) 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Device 32 ----------^
Device 17 ----------^

Output Assembly data works in an identical fashion. The Output data received from the Programmable Controller or other EtherNet/IP client is written to each DeviceNet device one by one. First DeviceNet device 1 data is taken from the output data and written to DeviceNet device 1. Then the number of write data defined for device 2 is taken and written to the DeviceNet device defined as device 2. This continues until all the data in the output data is written to the DeviceNet network.

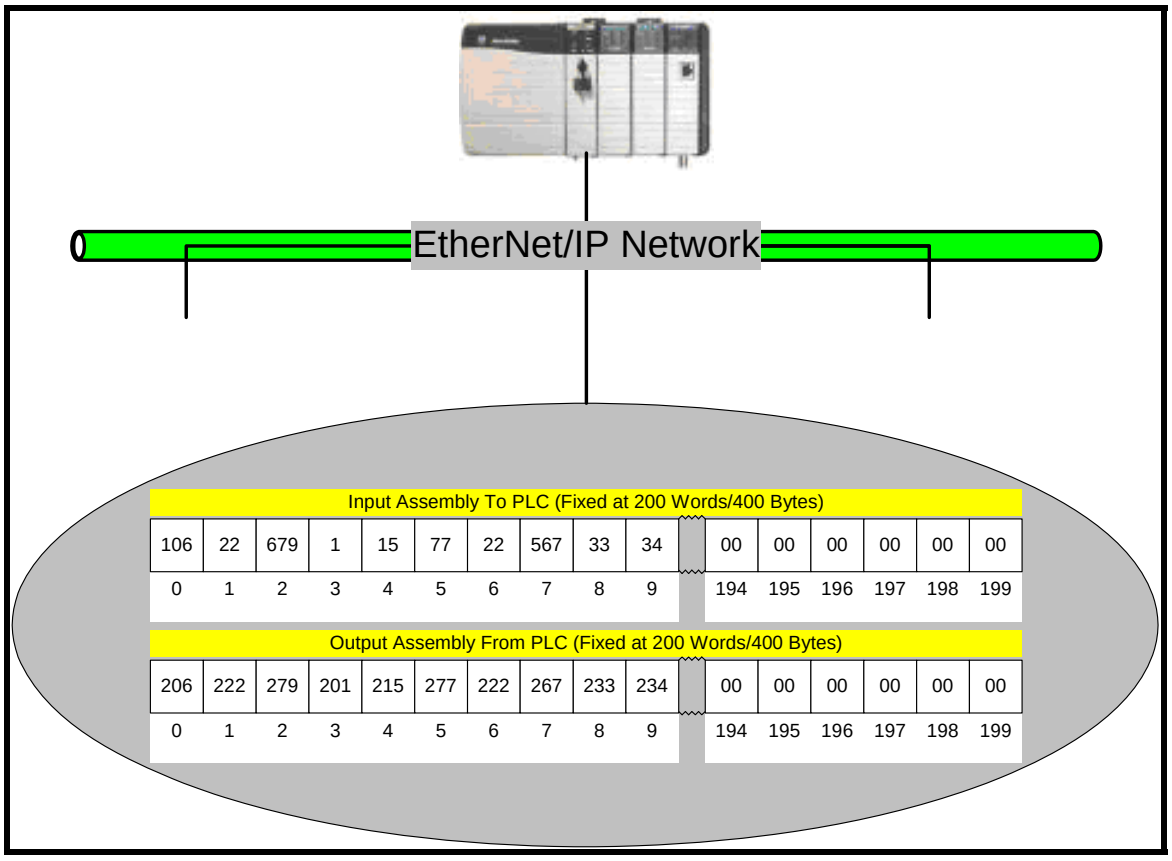


Figure 12 - EtherNet/IP Example Input/Output Assembly

Configuring Over EtherNet/IP

The EtherNet/IP DeviceNet Gateway can be configured over EtherNet/IP. See the section titled "CONFIGURING OVER ETHERNET/IP" for detailed instructions.

CONFIGURING OVER ETHERNET/IP

The Ethernet/IP DeviceNet Gateway can be configured from the internal web server or over the network.

To configure the device over the EtherNet/IP network you must access the objects described in the section using EtherNet/IP Explicit Messaging. See your EtherNet/IP Network tool documentation for an explanation of Explicit Messaging.

Configuration settings made using one method can be read and verified from the other method. For example, configuration settings made over EtherNet/IP can be displayed and modified using the internal web server. Configuration settings made using the internal web server can be read and revised using an EtherNet/IP network tool.

DeviceNet Network Configuration

The configuration for each of the 32 DeviceNet devices can be read or written over EtherNet/IP by accessing Object Class 67 hex. There are 32 instances of Object 67 hex, one for each DeviceNet device. For each device, Explicit messages can be sent to Read or Write the DeviceNet configuration including the DeviceNet Slave Address and I/O data.

See the complete description of Object 67 Hex in the appendix.

CONFIGURING MODULE NETWORK SETTINGS

Overview

There are three basic settings that must be configured to properly operate the 455ED/455EMT Gateway. These settings are configured using the internal web server and accessed by clicking on the "Edit" button under the Network Settings section of the server main page.

The Ethernet Network Settings are configured from the Network Edit Screen shown in Figure 13.

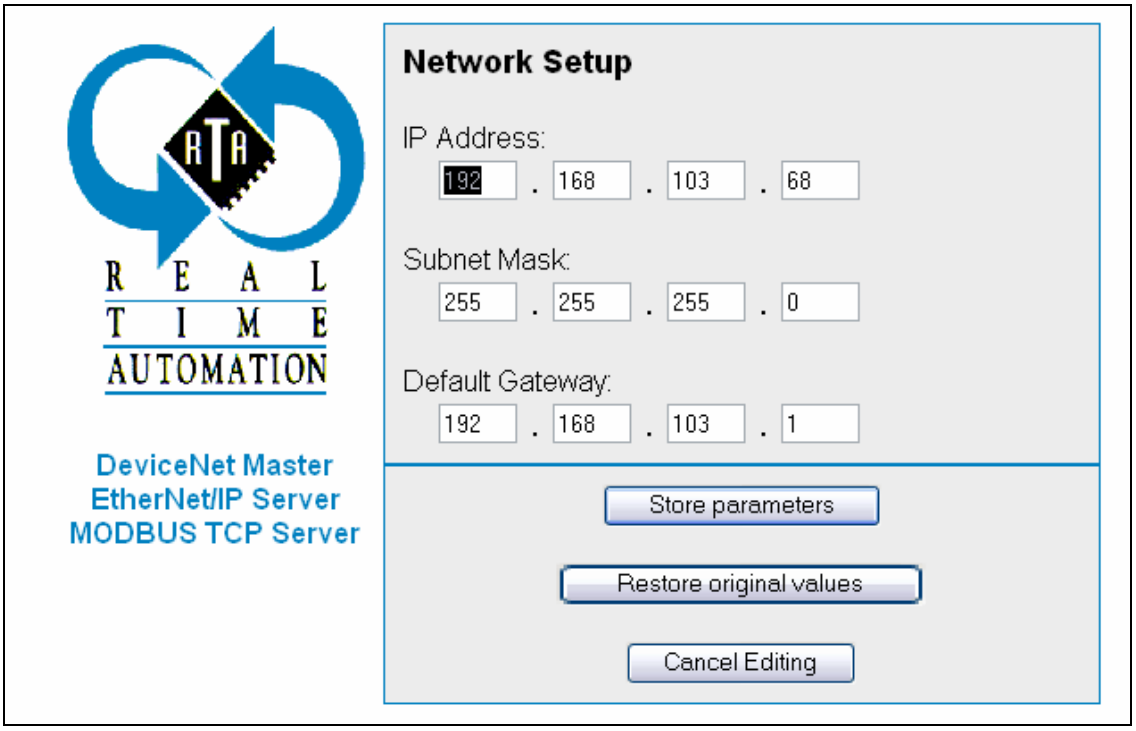


Figure 13 – 455ED/455EMT Network Settings

The three configuration parameters are:

IP Address
The IP Address is the IP address of the 455ED/455EMT module. This address normally exists on the same subnet as the EtherNet/IP Client.

Subnet Mask
The Subnet Mask is the address mask for the current subnet.

Default Gateway
The Default Gateway is the address of the device that serves as the gateway for non-local IP addresses. The default gateway IP address is usually the address of a router.

The default settings for the module network entries are:

CONFIGURATION PARAMETER	DEFAULT VALUE
IP Address:	192.168.0.100
Subnet Mask:	255.255.255.0
Default Gateway Address:	192.168.0.1

Table 1 – Default IP settings

DIAGNOSTICS & TROUBLESHOOTING

A Status page is accessible from the main page of the internal web server. The Status page indicates the status of communications on the DeviceNet network. The status page is accessed by clicking on the button labeled "Status Page" on the main page of the internal web server. A sample status page is shown in Figure 15.

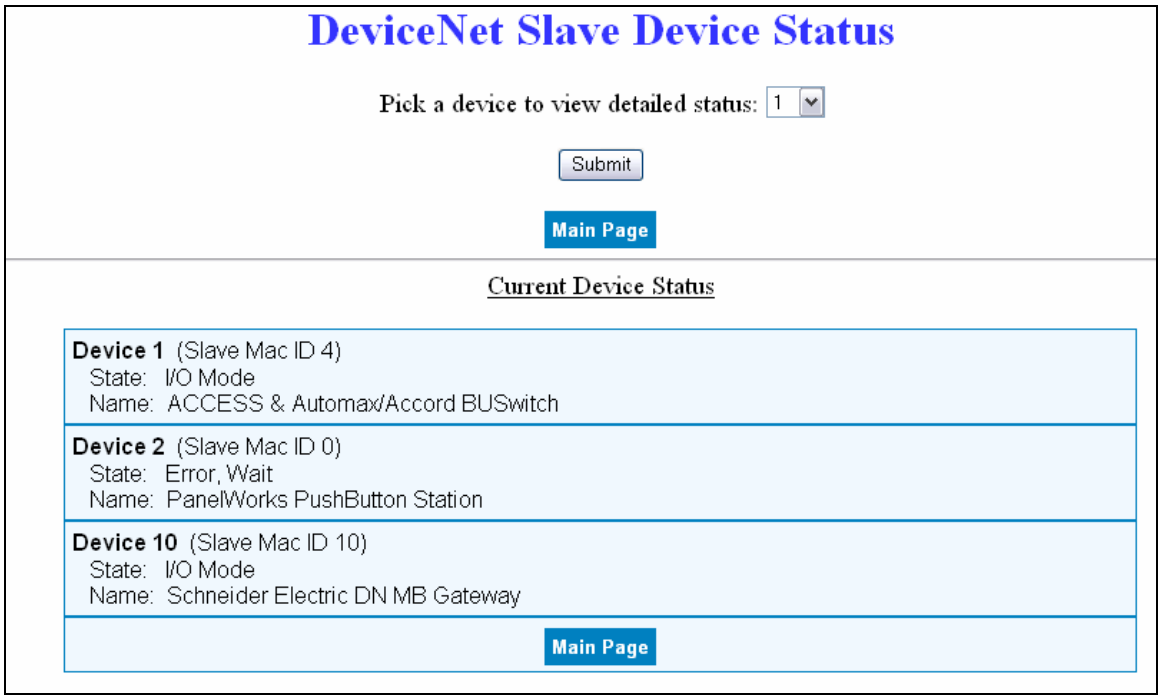


Figure 15 – DeviceNet Network Status Page

Status information is presented for each of the 32 possible devices on the DeviceNet network. The DeviceNet Slave Address, State, and Product Name are listed. Only configured devices will be shown on this page.

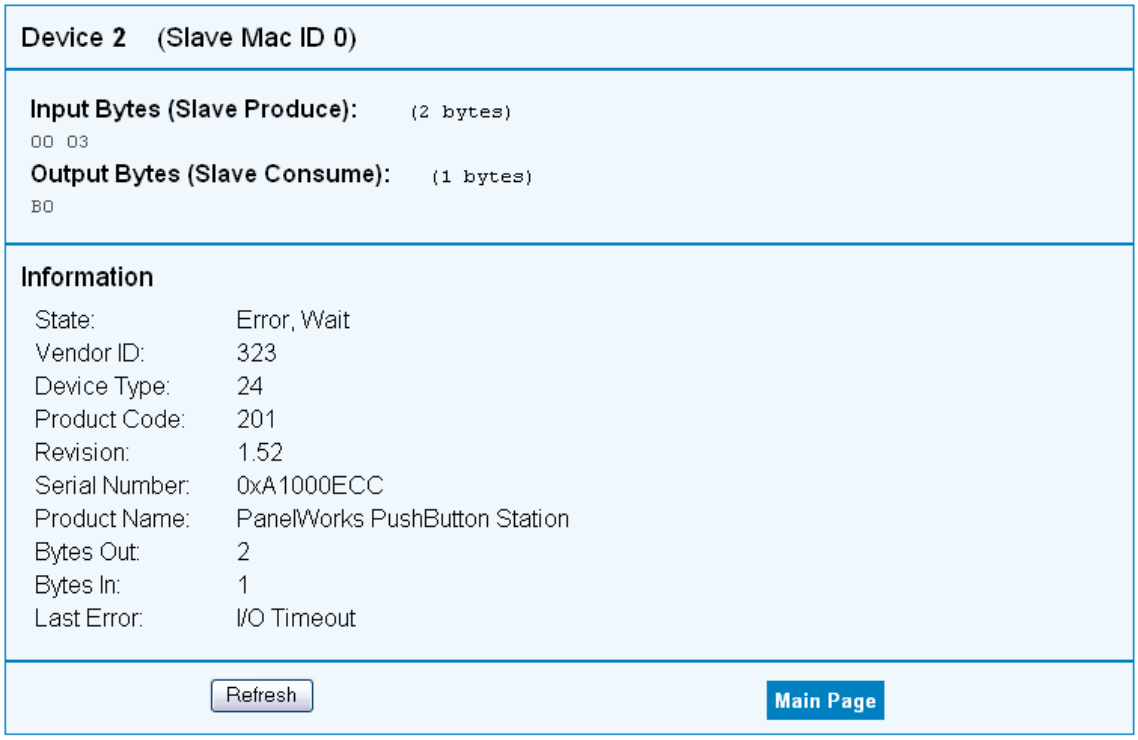


Figure 16 – Device Status Page

Here information about a device on the DeviceNet network can be viewed. The input and output data can be viewed under Bytes In and Bytes Out respectively. The Identity Object is displayed for each device as well as the current state of the device and the last error the device encountered. By pressing the Refresh button the user can see data being updated for the I/O data.

LED Status

There are two LEDs on the side of the 455 ED Gateway. One LED is for the DeviceNet Master status and the other is for the EtherNet/IP server status, both are shown in Figure 17:

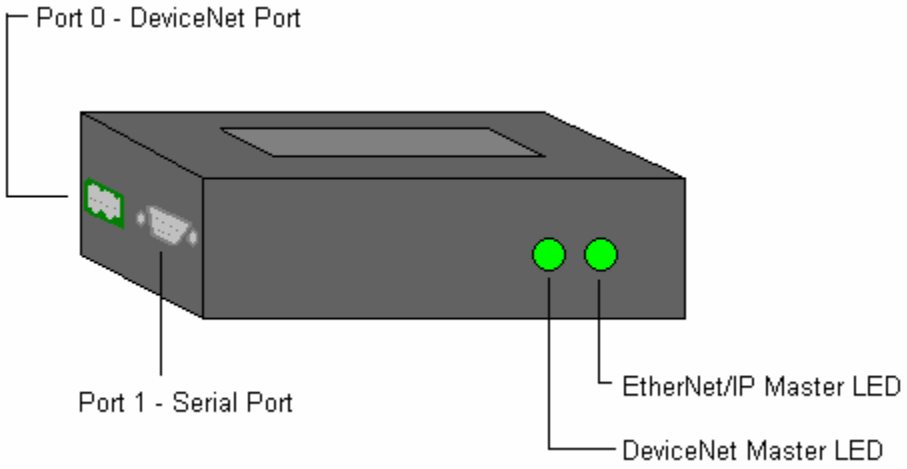


Figure 17 – LED Status

The LEDs behave in the following manner:

DeviceNet Master LED Status	
LED	Definition
Solid Green	The DeviceNet master is running and operational. All configured DevicNet slaves are communicating properly.
Flashing Green	The DeviceNet master is running and operational. There are no configured DeviceNet slaves.
Flashing Red	The DeviceNet master is running and operational. One or more DeviceNet slaves are not communicating properly.
Solid Red	There is a severe DeviceNet master error. The system may need to be restarted. The "DeviceNet Master State" on the main page reflects the cause of the fault.

Table 2 – DeviceNet Master LED Status

Ethernet Server LED Status	
LED	Definition
Solid Green	The EtherNet/IP server is running and operational. There is one or more connections on EtherNet/IP or Modbus TCP.
Flashing Green	The EtherNet/IP server is running and operational. There are no connections.

Table 3 – EtherNet/IP Server LED Status

APPENDIX A – EtherNet/IP Object Model

Device Object Model

The Device Object Model is the logical organization of attributes (parameters), classes (objects) and services supported by a device. Objects are composed of attributes and services. There are three types of objects in any CIP device: Required Objects, Application Objects and Vendor Specific Objects.

Required Objects are object classes that must be supported by all devices on EtherNet/IP.

Applications Objects are classes that must be supported by all devices using the same profile. An example of a profile is a Discrete I/O device or an AC Drive. This ensures that devices from different vendors but with the same profile have a common interface to EtherNet/IP Client devices. For example, every AC Drive device must have a motor object among its required application objects. The attribute numbers for the maximum motor frequency and other motor data are predefined by the AC Drive profile to simply access to any device supporting the AC Drive profile.

Vendor Specific Objects are classes that add attributes and services that don't fit into the Required or Application Objects.

The default object representation for End User systems using the RTA Gateway for EtherNet/IP access supports 6 Required Objects and 4 Vendor Specific Objects:

Required Objects (defined by EtherNet/IP Specification)

- Identity Object (0x01)
- Message Router Object (0x02)
- Assembly Object (0x04)
- Connection Manager Object (0x06)
- TCP Object (0xF5)
- Ethernet Link Object (0xF6)

Vendor Specific Objects (defined by RTA to support End User system operation)

- Gateway Write Data Object (0x64)
- Gateway Read Data Object (0x65)
- Gateway Serial Configuration Object (0x66)
- Gateway Device Configuration Object (0x67)

Identity Object (01_{HEX} - 1 Instance)

The Identify Object provides read only data that describes the Gateway. Data includes the EtherNet/IP Vendor number, the major and minor revision and the serial number of the gateway. The End User system has no direct control of any attributes in this object.

Class Attributes

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get

Instance Attributes

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Vendor Number	UINT	50_{DEC}	Get
2	Device Type	UINT	$0C_{HEX}$	Get
3	Product Code Number	UINT	451_{DEC}	Get
4	Product Major Revision Product Minor Revision	USINT USINT	01 25	Get
5	Status Word (see below for definition)	WORD	See Below	Get
6	Serial Number	UDINT	Unique 32 Bit Value	Get
7	Product Name **Structure of:** Product Name Size Product Name String	 USINT USINT[32]	 32 String	Get

Status Word

Bit	Bit = 0	Bit = 1
0	No I/O Connection	I/O Connection Allocated
1 – 15	Unused	Unused

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get_Attribute_Single
05_{HEX}	No	Yes	Reset

Assembly Object (04_{HEX} – 4 Instances)

An EtherNet/IP Assembly Object assembles data from other objects into input and output packages that are exchanged with the Programmable Controllers. Input objects refer to the collection of data items that are transferred from the Server (Gateway) to the Client (usually a Programmable Controller). Output refers to the collection of data items that are transferred from the Client (Programmable Controller) to the Server (Gateway).

The Gateway provides two options for transferring data from the End User system to the EtherNet/IP Client (typically a Programmable Controller). When a Programmable Controller is used, the option is selected by a configuration choice in the Programmable Controller configuration tool. The PLC programmer enters the selected instance numbers defined below to choose a specific input and output assembly option.

Other less sophisticated EtherNet/IP Clients must choose an instance number from the list of input and output instances and include those numbers in the EtherNet/IP Forward Open command.

Assembly Options for the Application

Option 1: INPUT ONLY ASSEMBLY

Input Instance: 64 Hex
Output Instance: 81 Hex

This option provides the DeviceNet data as input I/O to the EtherNet/IP Client. The data for all the DeviceNet nodes is packed into one package of 200 words of data. The size of the input instance is fixed at 200. If there are less than 200 words in all of the DeviceNet buffers assigned to the gateway, the remainder of the assembly data is set to zero. If the DeviceNet data totals more than 200 registers the extra data is discarded.

In this option the output Assembly data length is zero, providing only a “heartbeat” type communication with the client.

Option 2: INPUT OUPUT ASSEMBLY

Input Instance: 64 Hex
Output Instance: 71 Hex

Option 2 contains both input and output instances. The input assistance is identical to Option 1.

The output instance contains the Programmable Controller output data to write to the DeviceNet devices as register data. Up to 200 words of output data can be sent from the EtherNet/IP Client to the Gateway for distribution to the DeviceNet devices as register data. The size of the output instance is fixed at 200 words. If the total number of DeviceNet buffer data is less than 200 words, the remainder of the output assembly data is ignored.

Class Attributes

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get
2	Max Instance	UINT	0x81	Get

Instance 0x64 Attributes (Input Instance)

Attribute ID	Name	Data Type	Default Data Value	Access Rule
3	Serial Read Data **Structure of** Node Read Data 1 … Node Read Data n Error Status (2 words)	WORD[200]	All 0's	Get

The input data is packed into the 200 words of data based on the DeviceNet slave device read configuration information. The data is packed in device order (device 1 data bytes, device 2 data bytes, etc.). Devices with read buffer length equal to zero are skipped. There may be more register data configured than will fit in the 198 words– this data will be truncated. The last 2 words are for the error status, each bit represents 1 device node.

Instance 0x71 Attributes (Output Instance)

Attribute ID	Name	Data Type	Default Data Value	Access Rule
3	Serial Write Data **Structure of** Node Write Data 1 … Node Write Data n Error Status (2 words)	WORD[200]	All 0's	Get / Set

The output data is packed into the 200 words of data based on the DeviceNet slave device write configuration information. The data is packed in device order (device 1 write data bytes, device 2 write data bytes, etc.). Devices with write buffer length equal to zero are skipped. There may be more data configured than will fit in the 198 words. The last 2 words are for the error status, each bit represents 1 device node. Writing the appropriate bit will clear that devices error counters, clearing that bit will allow the counters to operate.

Instance 0x80 Attributes (Configuration Instance)

Most I/O clients include a Configuration path when opening an I/O connection to a server. There is no Configuration data needed.

Instance 0x81 Attributes (Heartbeat Instance – Input Only)

There is no data in Instance 81 hex.

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get_Attribute_Single
10_{HEX}	No	Yes	Set_Attribute_Single

TCP Object ($F5_{HEX}$ - 1 Instance)

The TCP Object provides read only data that describes the TCP connection between the Gateway and the EtherNet/IP Client. The End User system has no direct control of any attributes in this object.

Class Attributes

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get

Instance Attributes

Attribute ID	Name	Data Type	Default Data Value	Access Rule
1	Status[1]	DWORD	1	Get
2	Configuration Capability[2]	DWORD	0	Get
3	Configuration Control[3]	DWORD	0	Get
4	Physical Link Object[4] **Structure of:** Path Size Path	 UINT Padded EPATH	 2 0x20F6 0x2401	Get
5	Interface Configuration[5] **Structure of:** IP Address Network Mask Gateway Address Name Server Name Server 2 Domain Name Size Domain Name	 UDINT UDINT UDINT UDINT UDINT UINT STRING	 0 0 0 0 0 0 0	Get
6	Host Name[6] **Structure of:** Host Name Size Host Name	 UINT STRING	 0 0	Get

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get_Attribute_Single

[1] See section 5-3.2.2.1 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[2] See section 5-3.2.2.2 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[3] See section 5-3.2.2.3 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[4] See section 5-3.2.2.4 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[5] See section 5-3.2.2.5 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[6] See section 5-3.2.2.6 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.

Ethernet Link Object ($F6_{HEX}$ - 1 Instance)

The Ethernet Link Object provides read only data that describes the status of the physical Ethernet link. The End User system has no direct control of any attributes in this object.

Class Attributes

Attribute ID	Name	Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get

Instance Attributes

Attribute ID	Name	Data Type	Default Data Value	Access Rule
1	Interface Speed[7]	UDINT	100	Get
2	Interface Flags[8]	DWORD	3	Get
3	Physical Address[9]	USINT Array[6]	0	Get

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get_Attribute_Single

[7] See section 5-4.2.2.1 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[8] See section 5-4.2.2.2 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.
[9] See section 5-4.2.2.3 of "Volume 2: EtherNet/IP Adaptation of CIP" from ODVA for more details on this attribute.

Gateway Write Data Object (64_{HEX} – 32 Instances)

The Gateway Write Data Object maps the data transferred from the EtherNet/IP to the End User system over the DeviceNet Master interface.

Class Attributes (Instance 0)

The Write Object Class Attributes provide read only data that describes the number of words per attribute, and number of attributes per instance. The End User system has no direct control of any attributes in this object.

Attribute ID	Name	Data Type	Default Data Value	Access Rule
1	Revision	UINT	1	Get
2	Maximum Write Data Buffer Size (in words)	UINT	100	Get
3	Number of Words per Attribute	UINT	100	Get
4	Number of Data Attributes per Instance *(doesn't include the Write Data Size attribute)*	UINT	1	Get

Instance Attributes (Instances 1-32)

The Write Object Instance Attributes provides read/write data that contains the data transferred from an EtherNet/IP Client.

Each instance of the Write Data Object maps to a node number accessible by the End User system. The thirty two write instances map to device nodes 1 to 32 of the Gateway which are mapped to the End User system DeviceNet interface via the Configuration Object and/or the Gateway web page setup.

Attribute ID	Name	Data Type	Default Data Value	Access Rule
1	Write Data Size (in words)	UINT	0	Get / Set
2	Write Data [0-99]	WORD[100]	0	Get / Set

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get Attribute Single
10_{HEX}	No	Yes	Set Attribute Single

Gateway Read Data Object (65_{HEX} – 32 Instances)

The Gateway Read Data Object maps the data transferred from the End User system to an EtherNet/IP Client over the DeviceNet Master register interface.

Class Attributes (Instance 0)

The Read Object Class Attributes provide read only data that describes the number of words per attribute and the number of attributes per instance. The End User system has no direct control of any attributes in this object.

Attribute ID	Name	Data Type	Default Data Value	Access Rule
1	Revision	UINT	1	Get
2	Maximum Read Data Buffer Size (in words)	UINT	100	Get
3	Number of Words per Attribute	UINT	100	Get
4	Number of Data Attributes per Instance *(doesn't include the Read Data Size attribute)*	UINT	1	Get

Instance Attributes (Instances 1-32)

The Read Object Instance Attributes provides read/write data that describes contains the data to transfer to an EtherNet/IP Client.

Each instance of the Read Data Object maps to a node number accessible by the End User system. The thirty two read instances map to device nodes 1 to 32 of the Gateway which are mapped to the End User system DeviceNet interface via the Configuration Object and/or the Gateway web page setup

Attribute ID	Name	Data Type	Default Data Value	Access Rule
1	Read Data Size (in words) *(ONLY writes of "0" are allowed to this attribute)*	UINT	0	Get / Set
2	Read Data [0-99]	WORD[100]	0	Get

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get Attribute Single
10_{HEX}	No	Yes	Set Attribute Single

Gateway Serial Configuration Object (66_{HEX} – 1 Instance)

Class Attributes (Instance 0)

Attribute ID	Name	DeviceNet Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get

Instance Attributes (Instance 1)

Changes to these parameters will take place immediately.

Attribute ID	Name	DeviceNet Data Type	Data Value	Access Rule
1	Baud Rate: 0 = 125K 1 = 250K 2 = 500K	UINT	6	Get/Set

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get Attribute Single
10_{HEX}	No	Yes	Set Attribute Single

Gateway Device Configuration Object (67 $_{HEX}$ – 32 Instances)

Class Attributes (Instance 0)

Attribute ID	Name	DeviceNet Data Type	Data Value	Access Rule
1	Revision	UINT	1	Get

Instance Attributes (Instances 1-32)

Changes to these parameters will take place after the next reboot of the Gateway.

Attribute ID	Name	DeviceNet Data Type	Data Value	Access Rule
1	Node Slave Address (1-254)	UINT	0	Get/Set
2	Device Enabled (0=Disable)	UINT	0	Get/Set
3	Node Read Buffer Length (1-100)	UINT	0	Get/Set
4	Device Enabled (0=Disable)	UINT	0	Get/Set
5	Node Write Buffer Length (1-100)	UINT	0	Get/Set

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
$0E_{HEX}$	Yes	Yes	Get Attribute Single
10_{HEX}	No	Yes	Set Attribute Single

Explicit Message Passthrough (67 $_{HEX}$)

Sends an explicit message through the 455 ED Gateway onto a DeviceNet slave. An explicit message can be sent from a PC or PLC to the 455ED/455EMT. The 455ED/455EMT will send the message to the appropriate DeviceNet slave and return the response from the salve device back to the PC or PLC.

Class Attributes (Instance 0)

Attribute ID	Name	DeviceNet Data Type	Data Value	Access Rule
0 to 63	DeviceNet slave address	UINT	X	0x32

Common Services

Service Code	Implemented for		Service Name
	Class Level	Instance Level	
32_{HEX}	Yes	No	Explicit Message Pass through

When sending a message onto a DeviceNet slave, fill in the data portion of the explicit message to the 455 ED Gateway as follows:

SC	CID	IID	SIZE	ATTR	D_0	…	D_X

All of the above cells represent a singe byte and are defined as:

SC: Service Code
CID: Class ID
IID: Instance ID
SIZE: Number of data bytes plus the ATTR byte
ATTR: Attribute
$D_{0...X}$: Data bytes being sent

Example: Let's say a PC wants to read the Vendor ID of DeviceNet slave 4. The PC is connected to the 455ED/455EMT through EtherNet/IP. The explicit message would be assembled as such:

EtherNet/IP Portion:
SC = 0x32
CID = 0x67
IID = 0x0
ATTR = 0x4 DeviceNet Slave address

DeviceNet Portion, this is stored in the data block of the message:
SC = 0xE SIZE = 0x1
CID = 0x1 ATTR = 0x1
IID = 0x1 $D_{0...X}$ = NULL

APPENDIX B – 455EMT – MODBUS TCP

Introduction

The 455EMT is identical to the 455ED/455EMT from the point of view of DeviceNet and the Web server configuration. The difference is the method for accessing the data via Ethernet.

General Data Access

There is a one to one relationship between the Device ID in the DeviceNet scan list and the Unit ID on Modbus TCP. If DeviceNet Slave 10 is entered as Device 2 in the 455EMT scanlist, Unit ID 2, **NOT 10**, is used!

The starting address is always 400001 (0 on the Modbus TCP wire) for **both** the inputs and the outputs. This means the output data is write only and the input data is read only.

Each byte on DeviceNet is represented as a single register (16-bit), **not** two bytes per register. The read/write length must match the number of bytes of DeviceNet data exactly or an error code is returned.

Example 1 – Node 2 on DeviceNet; 2 Input Bytes and 1 Output Byte
Write 1 byte (0x54)
Read 2 bytes (returns 0x23 and 0x34)

READ HOLDING REGISTERS	
UNIT ID	2
LENGTH	2
REGISTER	**DATA VALUE**
400001	0x0023
400002	0x0034

WRITE HOLDING REGISTERS	
UNIT ID	2
LENGTH	1
REGISTER	**DATA VALUE**
400001	0x0054

Error Processing

The following are the supported Modbus TCP error codes:

0x01 – Illegal Function (Function Not Supported)
0x02 – Illegal Address (Combination of Starting Address and Length Invalid)
0x06 – Slave Busy (Explicit Message Pass Through in progress or format error)
0x0B – Gateway Slave Missing (Corresponding DeviceNet Slave is offline)

DeviceNet Explicit Message Pass Through

READ HOLDING REGISTERS	
UNIT ID	99
LENGTH	3-103
REGISTER	**DESCRIPTION**
401001	DN Slave Address
401002	Response Service Code
401003	Data Length
401004 – 401103	Data

WRITE HOLDING REGISTERS	
UNIT ID	99
LENGTH	6-106
REGISTER	**DESCRIPTION**
400001	Message Timeout
400002	DN Slave Address
400003	Service Code
400004	Class ID
400005	Instance ID
400006	Data Length
400007 - 400106	Data

Definitions

Message Timeout – 5 milliseconds per tick. Writing a value > 0 causes the explicit message to attempt queuing to DeviceNet Master and clears the response registers to 0. If no response is received within the allotted time, a timeout occurs and an error message "0x94 0x07 0xFF" is returned. **Once a response is ready, this register is cleared to 0.**

DN Slave Address – Actual DeviceNet Slave Address (0-63), not the 455EMT Device ID (1-32).

Service Code / Response Service Code – ODVA Service Code (i.e. 0x0e = "Get Attribute Single" and 0x10 = "Set Attribute Single"). As successful Response Service Code echoes the requested Service Code with bit 7 on (i.e. 0x8E = "Successful Get Attribute Single").

Class ID – ODVA Class ID

Instance ID – ODVA Instance ID

Data Length – Data length in bytes (0-100). **If an Attribute ID is needed, it must be stored as the first byte.**

Data – 0-100 bytes of data. 1 byte per register.

Example – Read Vendor ID from Slave 10, wait up to 100ms (return val is 0x0032)

REQUEST

WRITE HOLDING REGISTERS		
UNIT ID	99	
LENGTH	7	
REGISTER	**DESCRIPTION**	**DATA VALUE (decimal)**
400001	Message Timeout	20
400002	DN Slave Address	10
400003	Service Code	14
400004	Class ID	1
400005	Instance ID	1
400006	Data Length	1
400007	Data	1

RESPONSE

READ HOLDING REGISTERS		
UNIT ID	99	
LENGTH	5	
REGISTER	**DESCRIPTION**	**DATA VALUE (decimal)**
401001	DN Slave Address	10
401002	Response Service Code	142
401003	Data Length	2
401004	Data 1	0x0032
401005	Data 2	0x0000

