GE Fanuc Automation
Programmable Control Products

PACSystems® RX3i

Ethernet Network Interface Unit

User's Manual, GFK-2439

January 2006

Warnings, Cautions, and Notes as Used in this Publication

Warning

Warning notices are used in this publication to emphasize that hazardous voltages, currents, temperatures, or other conditions that could cause personal injury exist in this equipment or may be associated with its use.

In situations where inattention could cause either personal injury or damage to equipment, a Warning notice is used.

Caution

Caution notices are used where equipment might be damaged if care is not taken.

Note

Notes merely call attention to information that is especially significant to understanding and operating the equipment.

This document is based on information available at the time of its publication. While efforts have been made to be accurate, the information contained herein does not purport to cover all details or variations in hardware or software, nor to provide for every possible contingency in connection with installation, operation, or maintenance. Features may be described herein which are not present in all hardware and software systems. GE Fanuc Automation assumes no obligation of notice to holders of this document with respect to changes subsequently made.

GE Fanuc Automation makes no representation or warranty, expressed, implied, or statutory with respect to, and assumes no responsibility for the accuracy, completeness, sufficiency, or usefulness of the information contained herein. No warranties of merchantability or fitness for purpose shall apply.

The following are trademarks of GE Fanuc Automation, Inc.

Alarm Master	Genius	ProLoop	Series Six
CIMPLICITY	Helpmate	PROMACRO	Series Three
CIMPLICITY 90–ADS	Logicmaster	PowerMotion	VersaMax
CIMSTAR	Modelmaster	PowerTRAC	VersaPoint
Field Control	Motion Mate	Series 90	VersaPro
GEnet	PACSystems	Series Five	VuMaster
	Proficy	Series One	Workmaster

Contents

Chapter 1 Introduction

This manual describes installation and operation of the PACSystems RX3i Ethernet Network Interface Unit (ENIU). The Ethernet NIU makes it possible to use PACSystems RX3i and Series 90-30 I/O remotely on an Ethernet network. Once set up by configuration, I/O operation is completely automatic. Control of the I/O can be provided by any GE Fanuc master device capable of exchanging Ethernet Global Data. The Ethernet NIU automatically provides the controller with status information in each EGD exchange sent to the controller. Each EGD exchange received by the ENIU can provide appropriate commands to the Ethernet NIU.

The Ethernet NIU works in systems with a single controller or with redundant controllers. When used with redundant controllers, the ENIU automatically switches to the standby controller if the active controller becomes unavailable.

The ENIU can also receive commands to execute COMMREQs to intelligent modules in the ENIU. Only PACSystems RX7i and RX3i controllers can send Remote COMMREQ Calls (RCC) to the Ethernet NIU to take advantage of this COMMREQ capability.

The ENIU has available one Local User Block that can be customized to provide a local control application to be executed on the ENIU. The Local User Block is limited in size to 20K bytes.

Chapter 2: Introduction, summarizes steps for setting up the Ethernet NIU.

Chapter 3: Installation, summarizes basic installation steps.

Chapter 4: I/O Control, Status, and Data Formats, describes the content of the data exchanged by the Ethernet NIU and the controller for I/O operation.

Chapter 5: I/O Configuration, explains how to set up data exchange between the Ethernet NIU and one or two controllers. This chapter also describes how to set up optional default states or values for output data.

Chapter 6: I/O Diagnostics, describes how to view and clear fault information for the Ethernet NIU.

Chapter 7: Local Program Logic in the Ethernet NIU, describes the Local Logic feature of the Ethernet NIU.

Chapter 8: Remote COMMREQ Calls, describes the Remote COMMREQ Call (RCC) feature that allows PACSystems RX7i and RX3i controllers to pass COMMREQs to modules in an I/O Station via the Ethernet NIU.

Chapter 9: COMMREQs for remote COMMREQ Calls, describes the standard COMMREQs that can be sent to the Ethernet NIU in a Remote COMMREQ Call.

Chapter 10: Modbus Master, explains how to use an Ethernet NIU as a Modbus Master using a pre-coded "C" block.

Appendix A: I/O Quick Start Guide, uses an example system to give an overview of the steps needed to set up an Ethernet NIU application.

Appendix B: Configuration Worksheets, consists of two sample configuration worksheets that can be used to record configuration parameters.

Additional Documentation

The Ethernet NIU and associated equipment function as part of a larger control system. Additional documentation may be needed to complete the system installation and configuration:

***TCP/IP Ethernet Communications for PACSystems,* GFK-2224.** This manual provides general information about Ethernet communications for PACSystems RX3i and PACSystems RX7i equipment.

TCP/IP Ethernet Station Manager for PACSystems, GFK-2225. This manual describes how to access and use the built-in Station Manager features.

TCP/IP Ethernet Communications for Series 90-30 CPU374 PLUS, GFK-2382.

TCP/IP Ethernet Station Manager for Series 90-30 CPU374 PLUS, GFK-2383.

***TCP/IP Ethernet Communications for Series 90 PLCs*, GFK-1541.**

TCP/IP Communications for Series 90 PLCs, Station Manager Manual, GFK-1186

PACSystems RX3i System Manual, GFK-2314. This manual details installation procedures, and includes descriptions and specifications of PACSystems RX3i I/O and option modules.

***Series 90-30 PLC Installation and Hardware Manual*, GFK-0356.** This manual describes Series 90-30 hardware components and provides basic hardware installation procedures.

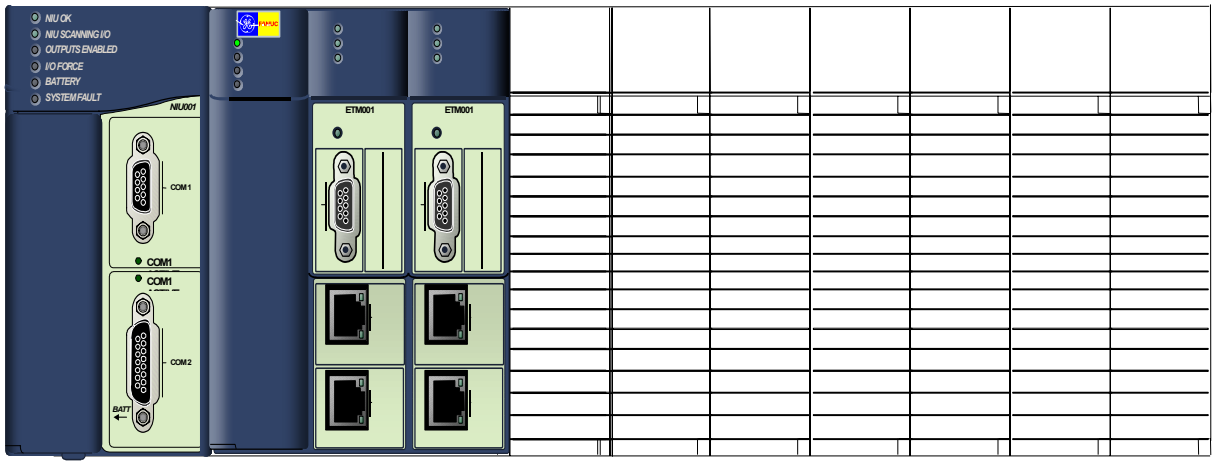
***Series 90-30 Module Specifications*, GFK-0898.** This manual is a collection of detailed module datasheets.

These user manuals, module datasheets, and other important product documents are available online at www.gefanuc.com. They are also included in the *Infolink for PLC* documentation library on CDs, catalog number IC690CDR002.

A PACSystems RX3i Ethernet NIU I/O Station

A PACSystems RX3i Ethernet NIU I/O Station consists of:

- an RX3i Universal Backplane (IC695CHS0xx)
- an RX3i power supply (IC695PSxxxx)
- the RX3i Ethernet NIU (IC695NIU001)
- one or more RX3i Ethernet Transmitter modules (IC695ETM001), which interface the I/O Station and NIU to the Ethernet network and to the controller.
- proprietary application software
- PACSystems RX3i and/or Series 90-30 modules, as appropriate for the application.



The system may also include optional Series 90-30 expansion backplanes.

In an RX3i I/O Station, the Ethernet NIU functions like a PLC CPU, controlling the activities of the modules in the station.

The PACSystems RX3i Ethernet NIU

The Ethernet NIU (IC695NIU001) makes it possible to use PACSystems RX3i and Series 90-30 I/O remotely on an Ethernet network. Once set up by configuration, data exchange is completely automatic. System control can be provided by any GE Fanuc master device capable of exchanging Ethernet Global Data. However, the Remote COMMREQ Call features of the Ethernet NIU are only accessible using PACSystem RX7i and RX3i controllers.

The Ethernet NIU automatically provides the controller with status information in each exchange. The application program logic in the controller can monitor this status data, and issue appropriate commands to the Ethernet NIU.

The PACSystems Ethernet NIU is compatible with the same types of modules, backplanes, and other equipment as a PACSystems RX3i CPU. For a list of compatible products, see the *PACSystems RX3i Hardware and Installation Manual,* GFK-2314.

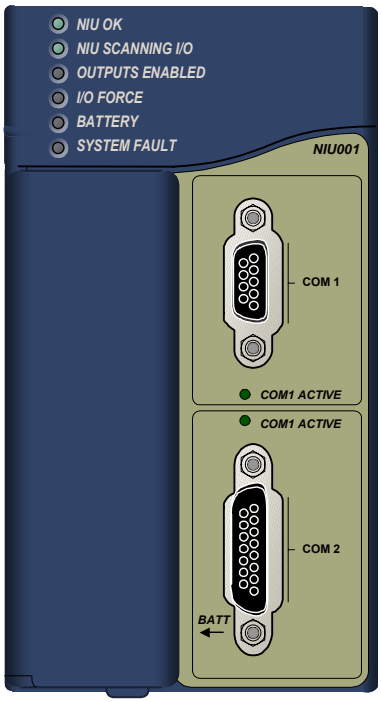
This module requires Machine Edition release 5.5 SIM 1 or later.

The Ethernet NIU can access one block of program logic called the ”Local Logic Block”, which can be up to 20K bytes in size.

Machine Edition automatically includes the proprietary logic blocks needed for the Ethernet NIU application.

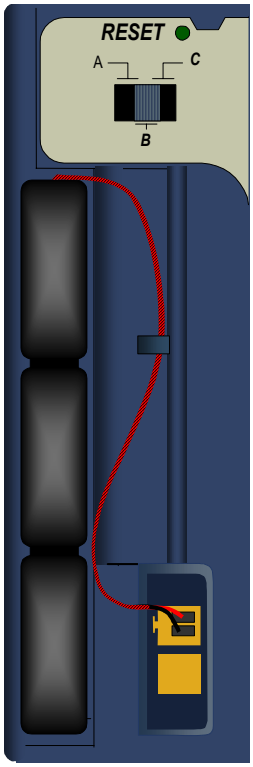
Ethernet NIU Features

- 20Kbytes of optional local logic. Supports all languages except C programming.
- 10 Mbytes of battery-backed CMOS RAM memory for local data storage.
- 10 Mbytes of built-in flash memory for local user data storage. Use of this flash memory is optional.
- Battery-backed calendar clock.
- In-system upgradeable firmware.
- RS-485 serial port and an RS-232 serial port.
- Data exchange using Ethernet Global Data (EGD)
- TCP/IP communication services using SRTP
- Supports operation with redundant controllers



Battery

A three cell lithium battery pack (IC698ACC701) is installed as shown at right. The battery maintains data memory when power is removed and operates the calendar clock. Program and initial values are always loaded from flash when the Ethernet NIU powers up. When replacing the battery, be sure to install a new battery before disconnecting the old one.



Disposal of lithium batteries must be done in accordance with federal, state, and local regulations. Be sure to consult with the appropriate regulatory agencies before disposing of batteries.

To avoid loss of RAM memory contents, routine maintenance procedures should include scheduled replacement of the Ethernet NIU's lithium battery pack. For information on estimating battery life, refer to the *PACSystems CPU Reference Manual,* GFK-2222.

Specifications for IC695NIU001

Current required from 5V bus	+3.3 VDC: 1.25 Amps nominal +5 VDC: 1.0 Amps nominal
Operating Temperature	0°C to 60°C (32°F to 140°F)
Floating point	Yes
Embedded communications	RS-232, RS-485
Serial Protocols supported	Modbus RTU Slave, SNP, Serial I/O, Modbus RTU Master via Serial I/O and "C" block.
Backplane	Dual backplane bus support: RX3i PCI and 90-30-style serial
PCI compatibility	System designed to be electrically compliant with PCI 2.2 standard

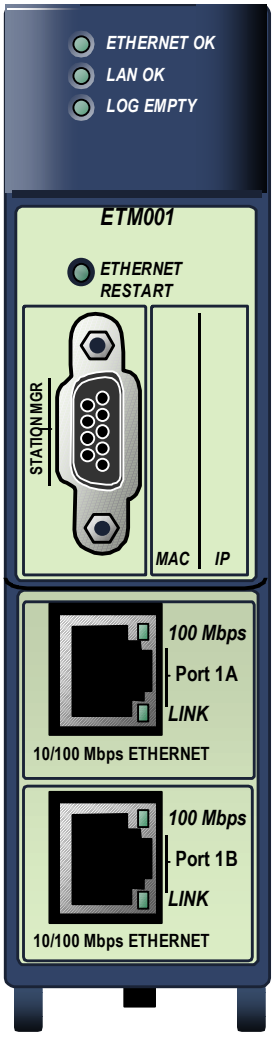
For environmental specifications and compliance to standards (for example, FCC or European Union Directives), refer to the *PACSystems RX3i Hardware and Installation Manual,* GFK-2314.

The Ethernet Transmitter Module

The Ethernet Transmitter Module, IC695ETM001, connects the Ethernet NIU's I/O Station to an Ethernet network. The Ethernet Transmitter Module enables the Ethernet NIU to communicate with other PACSystems equipment and with Series 90 and VersaMax controllers. The Ethernet Transmitter Module provides TCP/IP communications with other PLCs, host computers running the Host Communications Toolkit or programmer software, and computers running the TCP/IP version of the programming software. These communications use the GE Fanuc SRTP, Modbus TCP, and Ethernet Global Data (EGD) protocols over a four-layer TCP/IP (Internet) stack.

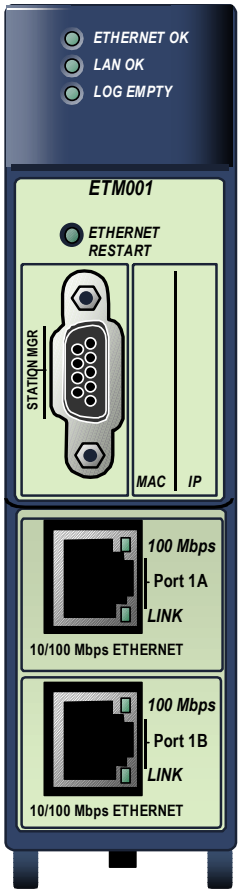
Features of the RX3i Ethernet Transmitter Module include:

- Implements EGD Class 1 and Class 2 capabilities.
- Firmware upgrades using the WinLoader software utility.
- Periodic data exchange using Ethernet Global Data (EGD).
- EGD Commands to read and write PLC and EGD exchange memory over the network.
- TCP/IP communication services using SRTP.
- Support for SRTP Channels, Modbus/TCP Server, and Modbus/TCP Client
- Built-in Station Manager for on-line supervisory access to the Ethernet Interface. Dedicated Station Manager port.
- Two auto-sensing 10Base T / 100Base TX RJ-45 shielded twisted-pair Ethernet ports for direct connection to either a 10BaseT or 100BaseTX IEEE 802.3 network without an external transceiver. There is only one interface to the network (only one Ethernet MAC address and only one IP address).
- Internal network switch with Auto negotiate, Sense, Speed, and crossover detection.
- Recessed Ethernet Restart pushbutton to manually restart the Ethernet firmware without power cycling the system.
- LEDs: Ethernet OK, LAN OK, Log Empty, individual port activity and speed LEDs.



Ethernet Transmitter Module Controls and Indicators

LEDs



- The **Ethernet OK** LED indicates whether the module is able to perform normal operation. This LED is On for normal operation and flashing for all other operations. If a hardware or runtime failure occurs, the EOK LED blinks a two-digit error.
- The **LAN OK** LED indicates access to the Ethernet network. The LAN LED blinks when data is being sent or received over the network directed to or from the Ethernet interface. It remains On when the Ethernet interface is not actively accessing the network but the Ethernet physical interface is available and one or both of the Ethernet ports is operational. It is Off otherwise unless software load is occurring.
- The **Log Empty** LED is On during normal operation. It is Off if an event has been logged.
- Two Ethernet network activity LEDS (**LINK**) indicate the network link status and activity.
- Two Ethernet network speed LEDS (**100Mbps**) indicates the network data speed (10 (off) or 100 Mb/sec (on)).

Ethernet Restart Pushbutton

This pushbutton is used to manually restart the Ethernet firmware without power cycling the entire system. It is recessed to prevent accidental operation.

Connectors

The module has two 10BaseT/100BaseTX Ethernet Network Port Connectors. There is only one interface to the network (only one Ethernet MAC address and only one IP address).

It also has a Station Manager (RS-232) Serial Port.

Ethernet Transmitter Module Specifications

Ethernet processor speed	200 MHz
Connectors	- Station Manager (RS-232) Port: 9-pin female D-connector - Two 10BaseT / 100BaseTX Ports: 8-pin female shielded RJ-45
LAN	IEEE 802.2 Logical Link Control Class I IEEE 802.3 CSMA/CD Medium Access Control 10/100 Mbps
Number of IP addresses	One
Number of Ethernet Port Connectors	Two, both are 10BaseT / 100BaseTX with auto-sensing RJ-45 connection.
Embedded Ethernet Switch	Yes – Allows daisy chaining of Ethernet nodes.
Serial Port	Station Manager Port: RS-232 DCE, 1200 - 115200 bps.

Refer to the *PACSystems RX3i System Manual* , GFK-2314, for product standards and general specifications.

Ethernet Interface Ports

The Ethernet Interface module has two auto-sensing 10Base T / 100Base TX RJ-45 shielded twisted pair Ethernet ports for connection to either a 10BaseT or 100BaseTX IEEE 802.3 network. The port automatically senses the speed (10Mbps or 100Mbps), duplex mode (half duplex or full duplex) and cable (straight-through or crossover) attached to it with no intervention required.

Ethernet Media

The Ethernet Interface can operate directly on 10BaseT/100BaseTX media via its network ports.

10BaseT: 10BaseT uses a twisted pair cable of up to 100 meters in length between each node and a switch, hub, or repeater. Typical switches, hubs, or repeaters support 6 to 12 nodes connected in a star wiring topology.

100BaseTX: 100BaseTX uses a cable of up to 100 meters in length between each node and a switch, hub, or repeater. The cable should be data grade Category 5 unshielded twisted pair (UTP) or shielded twisted pair (STP) cable. Two pairs of wire are used, one for transmission, and the other for collision detection and receive. Typical switches, hubs, or repeaters support 6 to 12 nodes connected in a star wiring topology.

Station Manager

The built-in Station Manager function of the Ethernet Transmitter Module provides on-line supervisory access to the Ethernet interface, through the Station Manager port or over the Ethernet cable. Station Manager services include:

- An interactive set of commands for interrogating and controlling the station.
- Unrestricted access to observe internal statistics, an exception log, and configuration parameters.
- Password security for commands that change station parameters or operation.

Refer to the *PACSystems TCP/IP Ethernet Communications Station Manager Manual*, GFK-2225 for complete information on the Ethernet Transmitter Module's Station Manager features.

Firmware Upgrades

The Ethernet Transmitter Module receives its firmware upgrades indirectly from the Ethernet NIU serial port using the WinLoader software utility. WInLoader is supplied with any updates to the Ethernet interface software.

Ethernet NIU COMMREQ Feature

The Ethernet NIU supports selected COMMREQs that are sent to it by a “C” block application in a PACSystems Rx7i or RX3i controller. This feature is not available with other types of controllers.

Ladder code is written in the RX7i or RX3i to interface to the “C” block which results in COMMREQ commands being sent via a EGD Exchange to the ENIU. The Ethernet NIU executes the COMMREQ and sends the results back to the RX7i or RX3i via another EGD exchange.

The following COMMREQs are supported:

- Modbus Master – function codes 1, 2, 3, 4, 5, 6, 7, 15, 16, 17
- Genius – enable/disable outputs, switch BSM, clear fault, clear all faults, assign monitor, read diagnostic
- Profibus Master – COMMREQs 1, 2, 4, 5, 6
- Motion (DSM314/DSM324) – load parameters
- High Speed Counter – Data command
- DeviceNet Master – COMMREQs 1, 4, 5, 6, 7, 9
- Analog Module – HART Protocol COMMREQs

Ethernet Global Data Features

The Ethernet NIU communicates with its controller via the Ethernet Transmitter Module, using Ethernet Global Data exchanges. One exchange is used to send outputs to the Ethernet NIU and another exchange is used to send inputs back to the controller. The Ethernet NIU supports receiving outputs from redundant controllers. By sending the Ethernet Global Data exchange to a group address, both controllers can receive the inputs. Up to 1300 bytes of outputs can be sent to a set of ENIUs from a controller. Each ENIU can send up to 1300 bytes of inputs to the controller.

A typical system might consist of a controller with five Ethernet NIU I/O Stations. The controller sends 1300 bytes of outputs and each Ethernet NIU sends 100 bytes of inputs to the controller. This typical system would have its I/O updates occur in less than 25 milliseconds. If the controller scan time is greater than 25 milliseconds, the update occurs at the controller’s scan rate. This performance timing is a guideline, not a guarantee, and assumes that there is no other traffic on the Ethernet link to the I/O.

Ethernet Global Data (EGD)

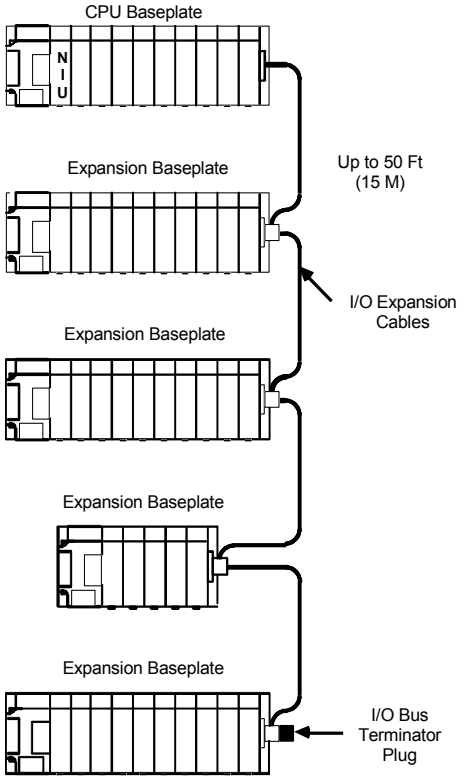
EGD exchanges are configured using the programmer and stored into the PLC. Both Produced and Consumed exchanges can be configured. PACSystems Ethernet Interfaces support both selective consumption of EGD exchanges and EGD exchange production and consumption to the broadcast IP address of the local subnet.

The Ethernet Interface can be configured to use SNTP to synchronize the timestamps of produced EGD exchanges.

The Ethernet Interface implements the capabilities of a Class 1 and Class 2 EGD device. COMMREQ-driven EGD Commands can be used in the application program to read and write data into the CPU or other EGD Class 2 devices.

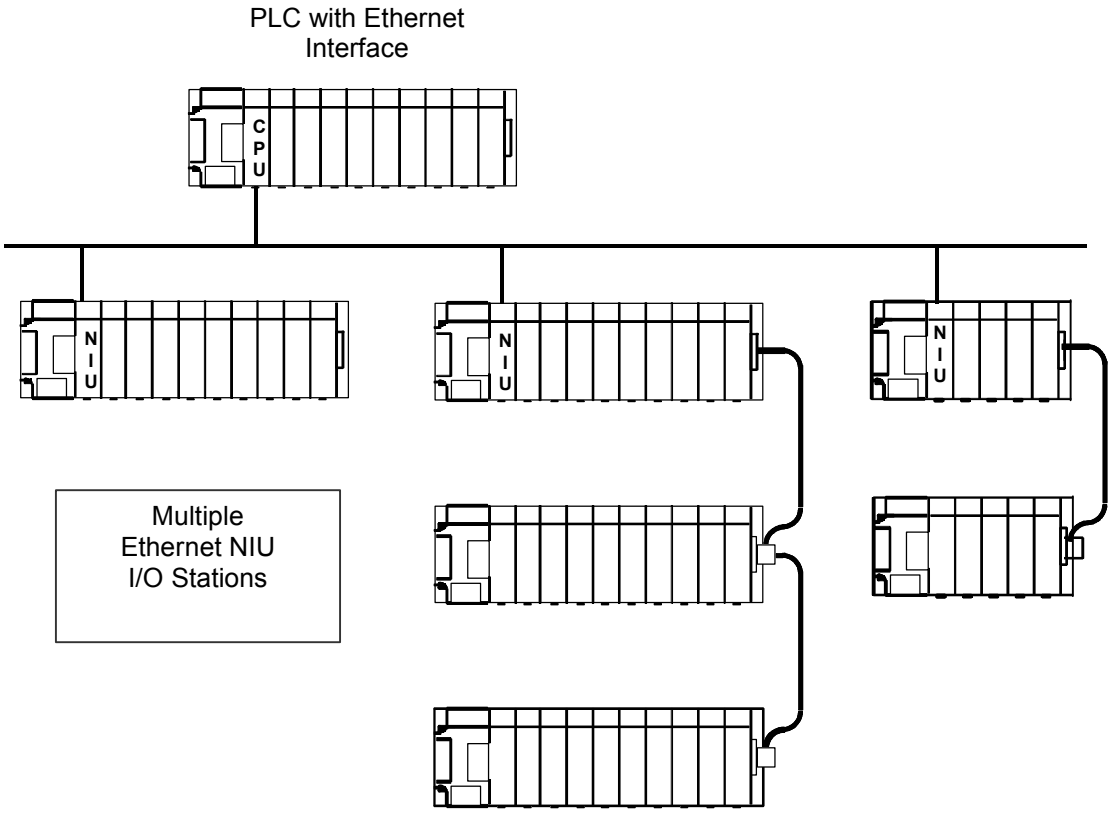
Modules and Baseplates in the I/O Station

The I/O Station can consist of just the main baseplate with NIU and modules, or a main baseplate and additional Expansion baseplates and Remote baseplates with modules as appropriate for the application.



- An Ethernet NIU can support up to 2048 discrete inputs, 2048 discrete outputs, 1268 analog inputs and 512 analog outputs. Additional I/O in the system can be located in other I/O Stations on the same network.
- There can be up to 50 feet (15 meters) of cable interconnecting Expansion baseplates and the main baseplate. The maximum number of Expansion baseplates in the I/O Station is 7. The actual number that can be used in an application depends on the amount of I/O capacity available on the network and the memory capacity of the NIU. Expansion baseplates are available in two versions; 5-slot (IC693CHS398) and 10-slot (IC693CHS392). All Expansion baseplates must be connected to a common ground, as described in the hardware installation manual.
- If a baseplate must be located more than 50 feet from the NIU, a Remote baseplate must be used. There can be up to 700 feet of cable connecting all baseplates in a system that has Remote baseplates. Up to 7 Remote baseplates can be used in the system. Remote baseplates are available in two sizes; 5-slot (IC693CHS398) and 10-slot (IC693CHS392). The cable type recommended for use with Remote baseplates must be used throughout the system. I/O Stations on the Network.

An Ethernet network can serve more than one NIU I/O Station.

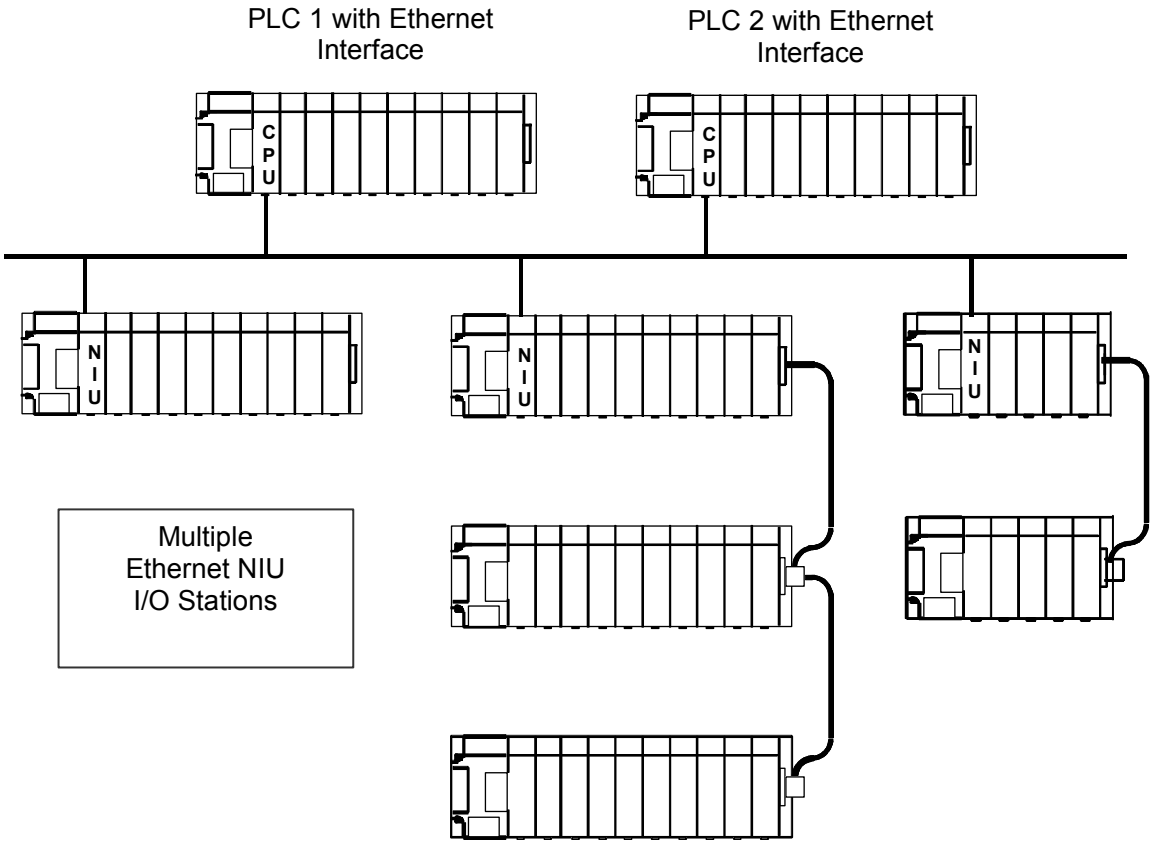


The Ethernet Interface in the master PLC sees all of the modules on the network without regard to their location in a specific I/O Station. That means each module must be assigned *unique* I/O references during configuration. The application program in the PLC sends output data on the Ethernet network, and each NIU consumes all of the output data. Each NIU then maps the output data to its own output memory. During the output portion of each NIU's I/O scan, it automatically sends the appropriate output data to the modules in its I/O Station.

Similarly, when the master PLC receives data from the NIUs, it maps the I/O data into PLC memory at the appropriate addresses. Therefore, it is important to be sure that all of the input references are unique to prevent input data being accidentally overwritten.

Controllers on the Network

Many applications will use one master to control one or more I/O Stations on the network. However, it is also possible to have two masters, with one serving as the primary controller and the other as a secondary controller to provide backup operation should communications with the primary controller be lost. When using more than one master, it is important to balance the needs of the application against the greater complexity of coordinating the controllers.



Any GE Fanuc Ethernet interface master capable of exchanging Ethernet Global Data messages, such as a PAC Systems, Series 90-30 or Series 90-70 CPU, or PC Control can function as a controller for the Ethernet NIU. However, some communications are only available with PACSystems RX7i or RX3i controllers, In a system that uses a primary and secondary controller, it is not necessary for the controllers to be the same type.

Communications Overview

The mechanism used for communications between the controller (or two controllers) and Ethernet NIU I/O Stations on the network is Ethernet Global Data exchanges.

Ethernet Global Data provides periodic data transfer over an Ethernet network. It supports fast, efficient communications because it is connectionless and is not acknowledged.

Caution

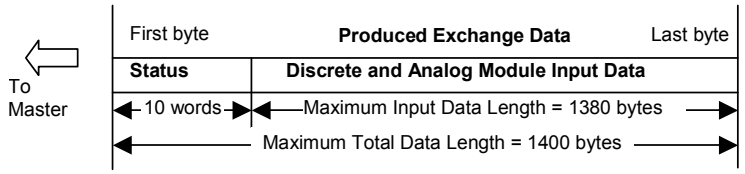
Ethernet Global Data (EGD) communication is connectionless and is not acknowledged. It is important to include error-checking and interlocking circuitry in the application to ensure the safety of personnel and equipment in the event that EGD data is lost. Failure to heed this warning could result in injury to personnel and damage to equipment.

In EGD communications, a device (called a producer) shares a portion of its memory contents periodically with one or more other devices (called consumers). This sharing of memory between devices is called an exchange.

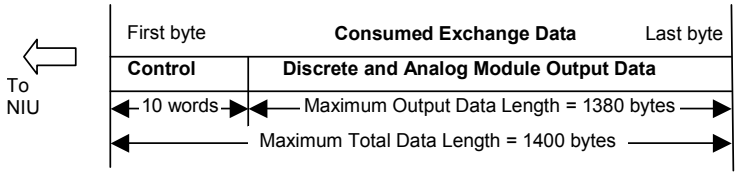
EGD Exchanges

An Ethernet NIU uses one EGD consumed data exchange and one produced data exchange. Each exchange begins with 10 words of NIU status data or CPU control data, followed by up to 1380 bytes of input or output data. The overall maximum length of a single exchange is 1400 bytes.

- The NIU's produced data exchanged consists of status data and the input data being sent to the controller.



- The NIUs consumed data exchange consists of control data and output data from the controller.



Chapter 4 describes the content of the status and control data, and explains how it can be used in the application. Chapter 5 describes how to configure EGD exchanges. If the system includes both a primary and secondary controller, EGD exchanges must be configured for both the primary and secondary controllers. In addition, if the system includes a secondary controller, the Ethernet NIU must be configured for two consumed exchanges. However, the ENIU uses data from only one controller at a time.

Chapter 2 Step-by-Step Ethernet NIU Setup

This chapter summarizes steps for setting up the Ethernet NIU. The instructions are written for one Ethernet NIU; for a system with multiple ENIUs, repeat the steps as appropriate. The setup steps focus first on the Ethernet NIU, then go on to the controller(s).

The list on the next page is an overview of the setup steps. In the overview:

- Boxes indicate steps that are linked to more detailed descriptions in this chapter.
- *Italics* indicate “Remote COMMREQ Call” (RCC) setup for the controller.
- Underlines indicate setup steps for an optional secondary controller.

At the end of this section, you’ll find instructions for converting a COMMREQ to a Remote COMMREQ Call command.

Overview of Steps

1. Make sure you are using Proficy™ Machine Edition release 5.5 SIM 1 or later. No earlier versions are compatible with the RX3i Ethernet NIU.
2. Determine IP Addresses for Primary and Secondary Controllers and Ethernet NIU(s).
3. Create Proficy Folder for Controllers, then add the ENIU target(s).
4. Set IP Address and Subnet Mask on Ethernet Transmitter module(s) ETM001 in Ethernet NIU, Set Gateway IP Address if required.
5. Set the Ethernet Global Data Local Producer ID in the Ethernet NIU
6. Add input and output modules to the Ethernet NIU configuration. (If you add or change modules later in the project, EGD Exchanges in the ENIU and controller will probably need to be updated).
7. Complete the Ethernet Global Data Exchanges in the ENIU
    - Repeat steps 4 through 8 for each ENIU
8. Store to ENIU
9. Set IP Address and Subnet Mask on ETM001 in controller(s), set Gateway IP Address if required. This could be on the CPU Embedded Ethernet Interface or on a separate Ethernet Transmitter module (IC69xETM001).
10. Add EGD to Controller(s)
11. Set the Ethernet Global Data Local Producer ID in Controller(s)
12. Create EGD Exchanges in Primary controller to match the EGD exchanges in the ENIU
13. Create EGD Exchanges in Secondary controller to match the EGD exchanges in the ENIU
14. Add RCC Parameterized “C” block to controller, If Remote COMMREQ Calls will not be used, skip to step 18.
15. Set up parameters on “C” block.
16. Add Call to RCC Parameterized “C” block and set up inputs and outputs to “C” block
17. Add logic to execute RCC commands with sequencing and checking results of RCC commands.
18. Store to controller
19. Verify EGD Exchanges are working
20. Verify RCC commands are working

More Details for Individual Steps

(Refer to list for steps 1 – 4)

Step 5: Set the ENIU Ethernet Global Data Local Producer ID

In the Machine Edition Navigator tree view, click on EGD for the Ethernet NIU target(s)

In the Property Inspector enter the Local Producer ID. It may be easiest to use the IP Address of the Ethernet NIU as the Local Producer ID. The Local producer ID can also be entered as a number. The Local Producer ID should only be entered once, either as the dotted decimal (IP Address type) or as a number. Leave the use configuration server property set to false. If the Property inspector is not open, right click on EGD in the navigator and then click on Properties to open the Property Inspector.

(Refer to list for step 6)

Step 7: Complete the Ethernet Global Data Exchanges in the ENIU

Outputs_Pri_to_ENIU – This exchange provides outputs from the Primary controller to be consumed by all Ethernet NIUs. Each ENIU retrieves its piece of the exchange.

In the Navigator tree view, in the ENIU section, double-click on Consumed Exchanges, then click on Outputs_Pri_to_ENIU.

In the Property Inspector, enter the Producer ID (Note this is the Producer ID of the Primary controller; usually the controller IP address is used). (The controller will be set up in step 11 below).

Leave the Group ID and Exchange ID parameters at their defaults.

If the Ethernet NIU has multiple Ethernet Transmitter ETM001 modules, select the correct ETM module in the adaptor name property that will receive (or consume) the exchange.

The Update Timeout property is set to 32 milliseconds. For systems with five or less Ethernet NIUs, this is fine. For systems with larger numbers of ENIUs, refer to “Timing for Ethernet Global Data Exchanges” in chapter 5 for more information.

The configuration screen for data ranges for the ENIU is correct. Unless some special case exists, it should not need to be adjusted.

Outputs_Sec_to_ENIU – This exchange produces outputs from the secondary controller to be consumed by all ENIUs. Each ENIU retrieves its piece of the exchange. If a Secondary Controller will not be used, this exchange can be deleted.

In the Navigator tree view , in the ENIU section, click on Outputs_Sec_to_ENIU

In the Property Inspector, enter the Producer ID (Note this is the Producer ID of the controller, usually the controller IP address is used).

Leave the Group ID and Exchange ID parameters at their defaults.

If the Ethernet NIU has multiple Ethernet Transmitter Modules, select the correct Ethernet Transmitter Module that will receive (or consume) the exchange in the adaptor name property.

The Update Timeout property is set to 32 milliseconds. For systems with up to five Ethernet NIUs, this will be fine. If the system has more than five Ethernet NIUs, refer to “Timing for Ethernet Global Data Exchanges” in chapter 5 for more information.

The configuration screen for data ranges for the ENIU is correct. Unless some special case exists, it should not need to be adjusted.

RCC _Pri_request_to_ENIU_xx *– This exchange produces an RCC command from the Primary controller to be consumed by Ethernet NIU "xx". For systems with multiple ENIUs, using RCC the "xx" should be changed to an identifier for the ENIU.*

In the Navigator tree view, in the ENIU section, click on RCC _Pri_request_to_ENIU_xx

In the Property Inspector, enter the Producer ID (Note this is the Producer ID of the Primary controller, usually the controller IP address is used).

Leave the Group ID parameter at its default.

If there are multiple Ethernet NIUs using the Remote COMMREQ Calls feature, the Exchange ID needs to be different for each ENIU. Change the Exchange ID if needed.

If the Ethernet NIU has multiple Ethernet Transmitter Modules, select the correct Ethernet Transmitter Module that will receive (or consume) the exchange in the adaptor name property.

The Update Timeout property is set to 150 milliseconds. For systems with up to five Ethernet NIUs, this is fine. If the system has more than five Ethernet NIUs, refer to "Timing for Ethernet Global Data Exchanges" in chapter 5 for more information.

The configuration for the data ranges for the ENIU is correct. Unless some special case exists, it should not need to be adjusted.

RCC _Sec_request_to_ENIU_xx *– This exchange produces a Remote COMMREQ Call (RCC) command from the Secondary controller to be consumed by Ethernet NIU "xx". For systems with multiple ENIUs using RCC the "xx" should be changed to an identifier for the ENIU.*

In the Navigator tree view click on RCC _Sec_request_to_ENIU_xx.

In the Property Inspector, enter the Producer ID (Note this is the Producer ID of the Secondary controller, usually the controller IP address is used).

Leave the Group ID parameter at its default.

If there are multiple Ethernet NIUs using RCC the Exchange ID needs to be different for each ENIU. Change the Exchange ID if needed.

If the Ethernet NIU has multiple Ethernet Transmitter Modules, select the correct Ethernet Transmitter Module that will receive (consume) the exchange in the adaptor name property.

The Update Timeout property is set to 150 milliseconds. For systems with up to five Ethernet NIUs, this is fine. If the system has more than five Ethernet NIUs, refer to "Timing for Ethernet Global Data Exchanges" in chapter 5 for more information.

The configuration for tha data ranges for the ENIU is correct. Unless some special case exists, it should not need to be adjusted.

Inputs_from_ENIU_xx – This exchange produces inputs from the Ethernet NIU to be consumed by the controller(s). The Exchange is sent to a group address so both controllers can receive it, if the system uses more than one controller.

In the Navigator tree view, in the ENIU section, double-click on Produced Exchanges. Then click on Inputs_from_ENIU_xx

In the Property Inspector, if the Ethernet NIU has multiple Ethernet Transmitter Modules, select the correct Ethernet Transmitter Module that will produce the exchange in the adaptor name property.

Do not modify the Destination Type or Destination parameter.

The Produced Period property is set to 10 milliseconds. For systems with up to five Ethernet NIUs, this is fine. If the system has more than five Ethernet NIUs refer to “Timing for Ethernet Global Data Exchanges” in chapter 5 for more information.

Either double-click or right-click and select Configure on the Inputs_from_ENIU_xx tree item. This will open the configuration page.

For any discrete or analog inputs in the Ethernet NIU, add ranges to send the inputs to the controller(s). Multiple ranges can be added for %I and %AI. Send only the addresses that are configured in this Ethernet NIU. Each ENIU sends the inputs configured in that ENIU. **WARNING** If you send the same inputs from multiple Ethernet NIUs, the controller will be getting multiple values for the same inputs and erratic operation will result.

Note: (Inputs in the exchange will show a duplicate address warning (different color, default is yellow). This is normal as the address is used for an Input module and EGD Exchange.

RCC _response_from_ENIU_xx *– This exchange produces the Remote COMMREQ Call results from the Ethernet NIU to the controller(s). This exchange is sent to a group address so both controllers receive it, if the system includes more than one controller.*

In the Navigator tree view in the ENIU section, , click on RCC_response_from_ENIU_xx. If there are multiple Ethernet NIUs using Remote COMMREQ Calls, change the "xx" to a designation for the ENIU.

In the Property Inspector, If there are multiple ENIUs using RCC the Exchange ID needs to be different for each ENIU. Change the Exchange ID if needed.

If the Ethernet NIU has multiple Ethernet Transmitter Modules, select the correct Ethernet Transmitter Module that will produce the exchange in the adaptor name property.

The Produced Period property is set to 50 milliseconds. For systems with up to five Ethernet NIUs, this is fine. If the system has more than five Ethernet NIUs, refer to "Timing for Ethernet Global Data Exchanges" in chapter 5 for more information.

The configuration for the data ranges for the ENIU is correct. Unless some special case exists, it should not need to be adjusted.

(Refer to list for steps 8 and 9)

Step 10. Add EGD to Controller(s)

Right click on the Controller target in the Navigator Tree View. Select Add Component and then click on Ethernet Global Data.

Step 11. Set the Ethernet Global Data Producer ID in Controller(s)

In the Navigator tree view click on Ethernet Global Data for Controller

In the Property Inspector enter the Local Producer ID, the easier thing to do is to use the IP Address of the Ethernet Transmitter Module as the Local Producer ID. Local producer ID can also be entered as a number. The Local Producer ID should only be entered once, either as the dotted decimal (IP Address type) or as a number. Leave the use configuration server property as false. If the Property inspector is not open, right click on EGD in the navigator and then click on properties to open the Property Inspector.

Step 12. Add EGD Exchanges in Primary Controller

The Exchanges in the controller need to be created to match the exchanges in the Ethernet NIU. The name of the exchange should match the name used in the ENIU.

Inputs_from_ENIU_xx

Create a New Consumed Exchange with the name “Inputs_from_ENIU_xx”. Enter the following field: (see picture below) If a secondary controller is used it will need an identical consumed exchange.

The Status line can be set to any Reference Address in the controller that does not cause an address conflict.

Add a range of Word data type length 10 words. This range will receive status information from the Ethernet NIU.

Add ranges to match the ranges that were added in the Ethernet NIU to send input to the controller. The Reference Address and length must be the same as they are in the ENIU.

After all ranges are added, the length in bytes of this exchange must match the length in bytes of the exchange configuration in the Ethernet NIU.

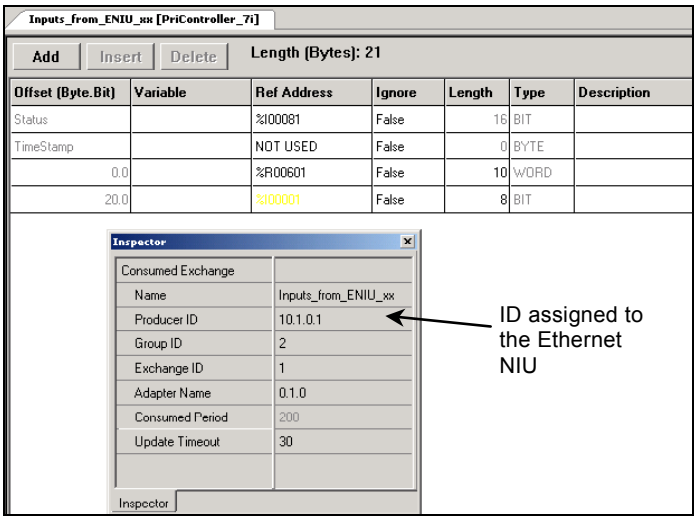
Set the Producer ID to the Producer ID of the Ethernet Transmitter Module.

Set the Group ID to 2, or match the group used in the ENIU if it has been changed.

Set the exchange ID to 1, or match the exchange ID in the ENIU if it has been changed.

Set the adapter name for the Ethernet interface you are using to receive (or consume) the exchange.

Set Update Timeout – 3 times the production Period is recommended.



RCC_response_to ENIU_xx

Create a New Consumed Exchange with the name "RCC_response_to ENIU_xx". Enter the following field: (see picture below) If a secondary controller is used it will need an identical consumed exchange.

The Status line can be set to any reference address in the controller that does not cause an address conflict.

Add one range, it should be in a Word type Reference table and have a length of 200 words. This address will also be an input to the RCCM "C" block in the controller.

In the Property Inspector:

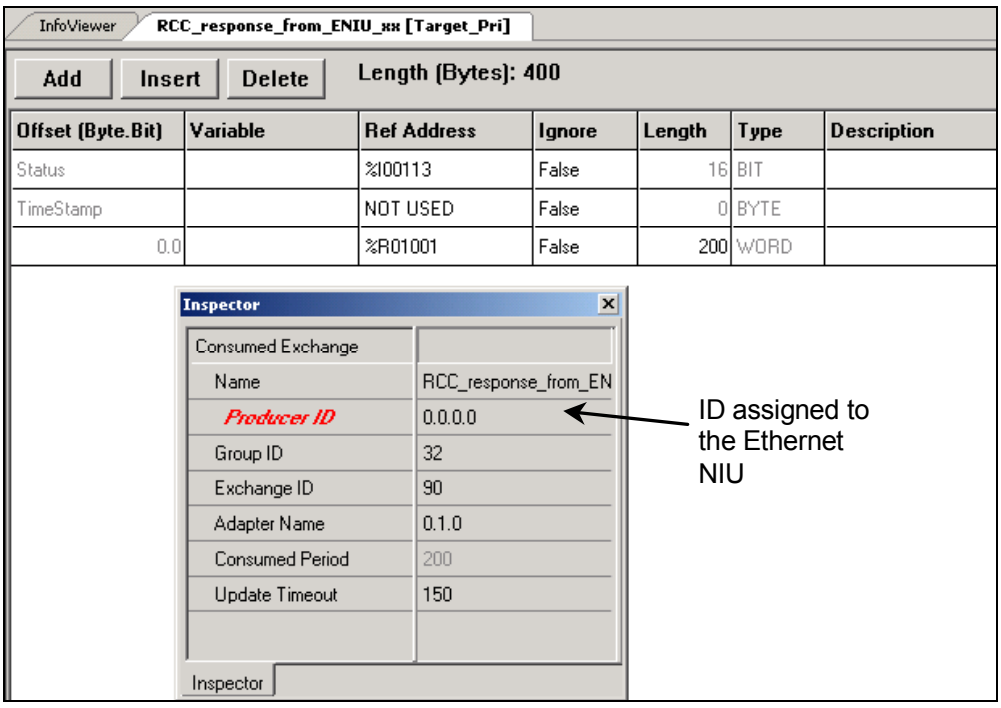
Set the Producer ID to the Producer ID of the ENIU.

Set the Group ID to 32, or match the group used in the ENIU if it has been changed.

Set the exchange ID to 90, or match the exchange ID in the ENIU if it has been changed.

Set the adaptor name for the Ethernet interface you are using to receive (or consume) the exchange.

Set Update Timeout – 3 times the production Period is recommended.



For additional Ethernet NIUs using Remote COMMREQ Calls, the exchange number must change and the name of the Exchange (the "xx") needs to be changed to identify the ENIU.

Output_Pri_to_ENIU

The Produced Exchanges from the Primary Controller to the Ethernet NIU need to be created.

Create a New Produced Exchange with the name “Output_Pri_to_ENIU”. Enter the following field (see picture below):

The Status line can be set to any reference address in the controller that does not cause an address conflict.

Add three ranges. When all three ranges are entered the length of the exchange should be 1300 bytes.

The first range is a 10 Word range used to send supervisory control to the ENIU.

The second Range is the first 2048 %Q from the controller.

The third range is the first 512 %AQ from the controller.

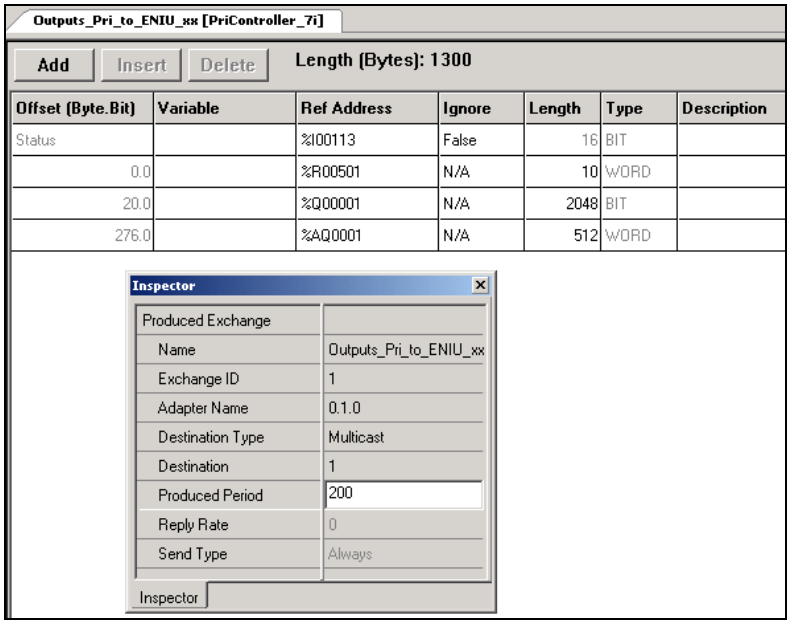
In the Property Inspector:

Set the exchange ID to 1, or match the exchange ID in the ENIU if it has been changed.

Set the adaptor name for the Ethernet interface that will send (produce) the exchange.

Set Destination Type – this should be set to “multicast”.
Set Destination – This should be 1.

Set Produced Period – 10 ms is recommended for systems with five or less ENIUs.



RCC_Pri_request_to ENIU_xx

The Remote COMMREQ Calls produced Exchange from the Primary Controller to the Ethernet NIU needs to be created.

Create a New Produced Exchange with the name "RCC_Pri_request_to_ENIU_xx"..

Enter the following field (see picture below):

The Status line can be set to any reference address in the controller that does not cause an address conflict.

Add one range. It should be in a Word type reference table and have a length of 200 words. This address will also be an input to the "C" block in the controller.

In the Property Inspector:

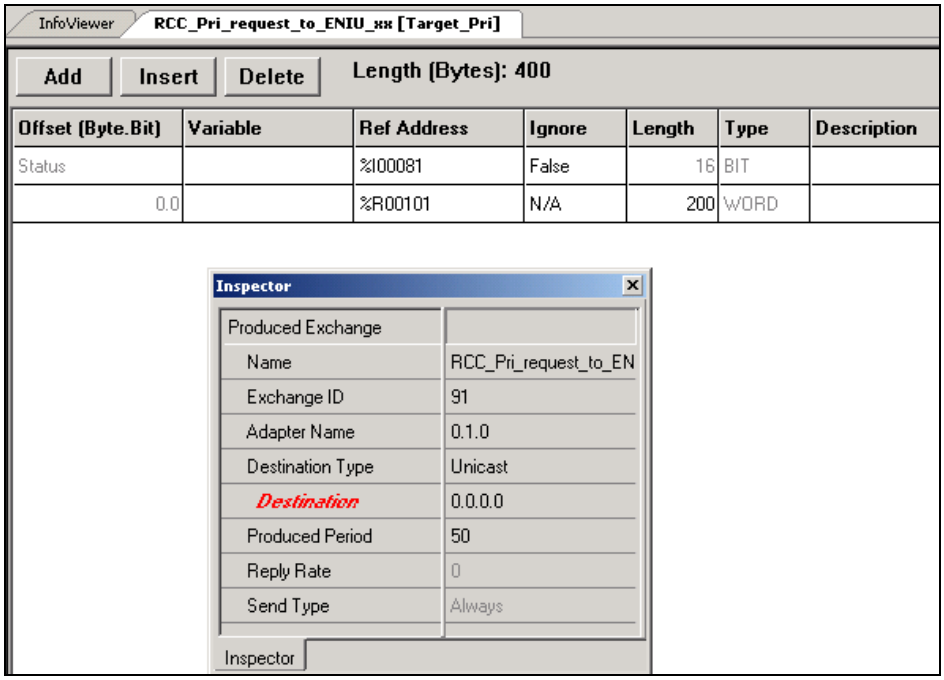
Set the exchange ID to 91, or match the Exchange ID in the ENIU if it has been changed.

Set the adaptor name for the Ethernet interface you are using the exchange will be produced by.

Leave the Destination Type parameter set to Unicast.

Set Destination - this is the IP address of the Ethernet NIU.

Set Produced Period – 50 milliseconds is recommended.



Step 13. Add EGD Exchanges in Secondary Controller

All the exchanges in the secondary controller are the same as the exchanges in the Primary controller except:

- Producer ID of the Secondary Controller
- Names of the exchanges are: "_____ SEC ____" instead of "_____ PRI ____"
- Exchange IDs for Produced Exchanges

Step 14. ADD RCCM "C" Block to Controller (RCCM_xxx)

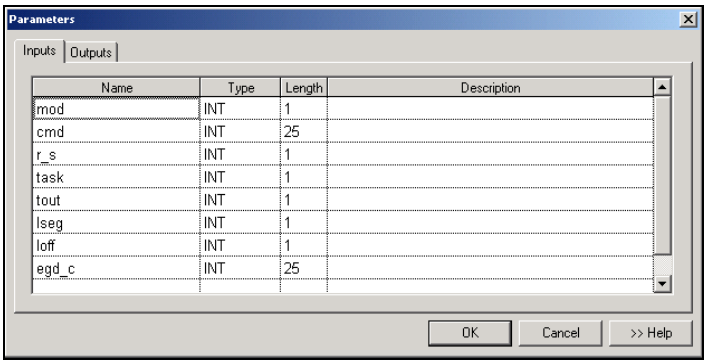
In the navigator tree view, right-click Program Blocks in the logic area of the controller. Click on "add C block". A dialog box to add the "C" block will come up. Browse to the file RCCM_120.gefElf and double click or select and open it to add it to the controller target.

Step 15. Add "C" Parameters

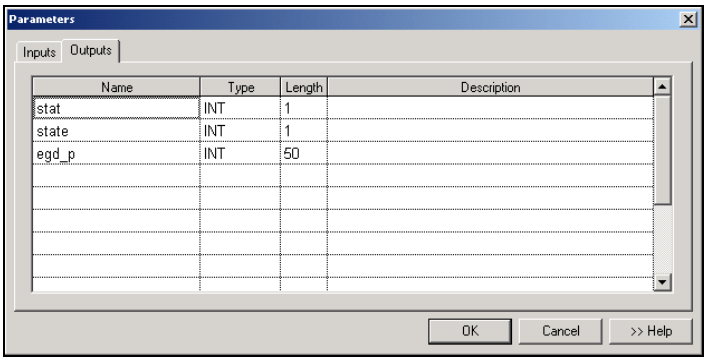
Right-click on the C Block in the Navigator tree view, then click on properties. Click on the parameters line in the Property Inspector to open the parameters dialog.

Enter the parameters as shown below.

On the Input Tab:



On the Output Tab:



Click "OK".

Step 16. Add "C" Block Call to the Controller Logic

Create a LD block called RCC. In _Main add a Call to RCC that is called every scan.

In the RCC block, add a call to RCCM_120 that is called every scan.

The Inputs and Outputs on the "C" block need to be entered. Use the View menu "Adjust Cell Width" to widen the C block if I/O parameters are not readable.

Inputs

For values for the following parameters, see the example that follows, and the details in this manual.

mod *Module type the COMMREQ is being sent to. Enter a Register reference and place the module code in the register.*

cmd *This is the COMMREQ command block Enter the register reference where the block starts. The Register reference must have an array length of 25.*

r_s *Slot number of the module the COMMREQ is being sent to. Enter a Register reference and place the slot number in the register.*

task *Task number that the module uses for COMMREQs it receives. Enter a Register reference and place the task number in the register.*

tout *Timeout for request in milliseconds. Enter a Register reference and place the timeout in the register.*

lseg *Segment selector for a 200 word buffer needed by the "C" block. Enter a constant 8 (%R) or 196 (%W).*

loff *Starting reference number of the buffer. Enter a constant, i.e. 7001.*

egd_c *Pointer to the RCC_response_from_ENIU_xx Exchange. Enter the starting Register reference of the exchange data range. This must have an array length of 25.*

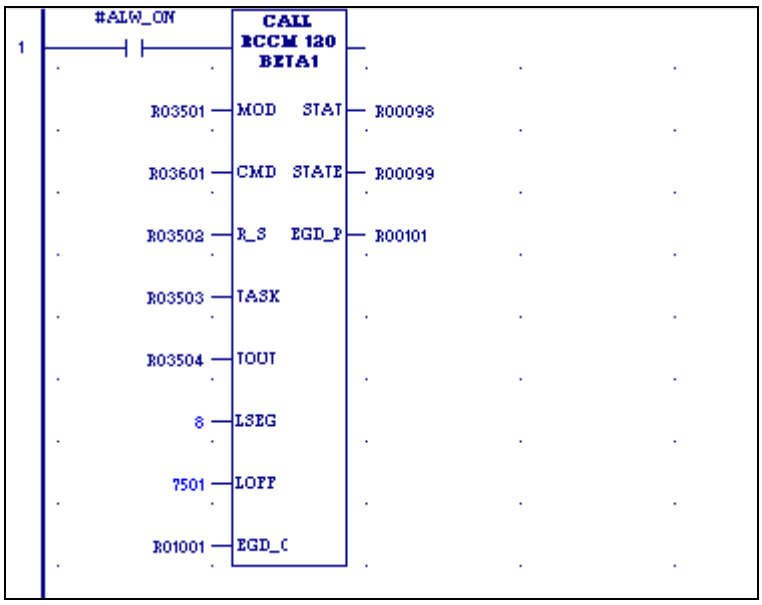
Outputs

Stat *Status of the RCC command. Enter a register reference. This is monitored to determine completion and success of the RCC command.*

State *State of the RCC command. Enter a register reference*

egd_p *Pointer to the RCC_request _to_ENIU_xx Exchange. Enter the starting Register reference of the exchange data range. It must have an array length of 50.*

An example of the call to the "C" block:



Step 17. Add Logic to Sequence RCC Command and to Check the Return Status

A Remote COMMREQ Call command is very much like a COMMREQ.

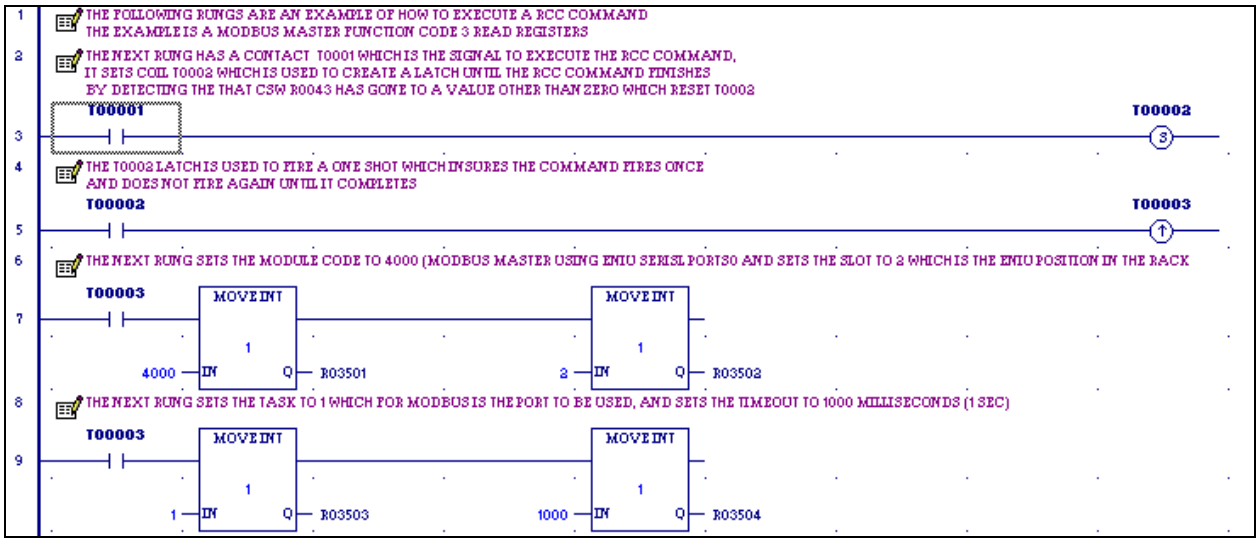
- *Command needs to be executed once and RCC status monitored for completion before command is executed again. The command is executed when the cmd input values are loaded. The "C" block zeros out the seventh register in the array on the cmd input. (Note this is the COMMREQ Command number.)*
- *The cmd input is the COMMREQ command block.*
- *The other inputs are used to route the COMMREQ to the correct module and to set timeout values.*

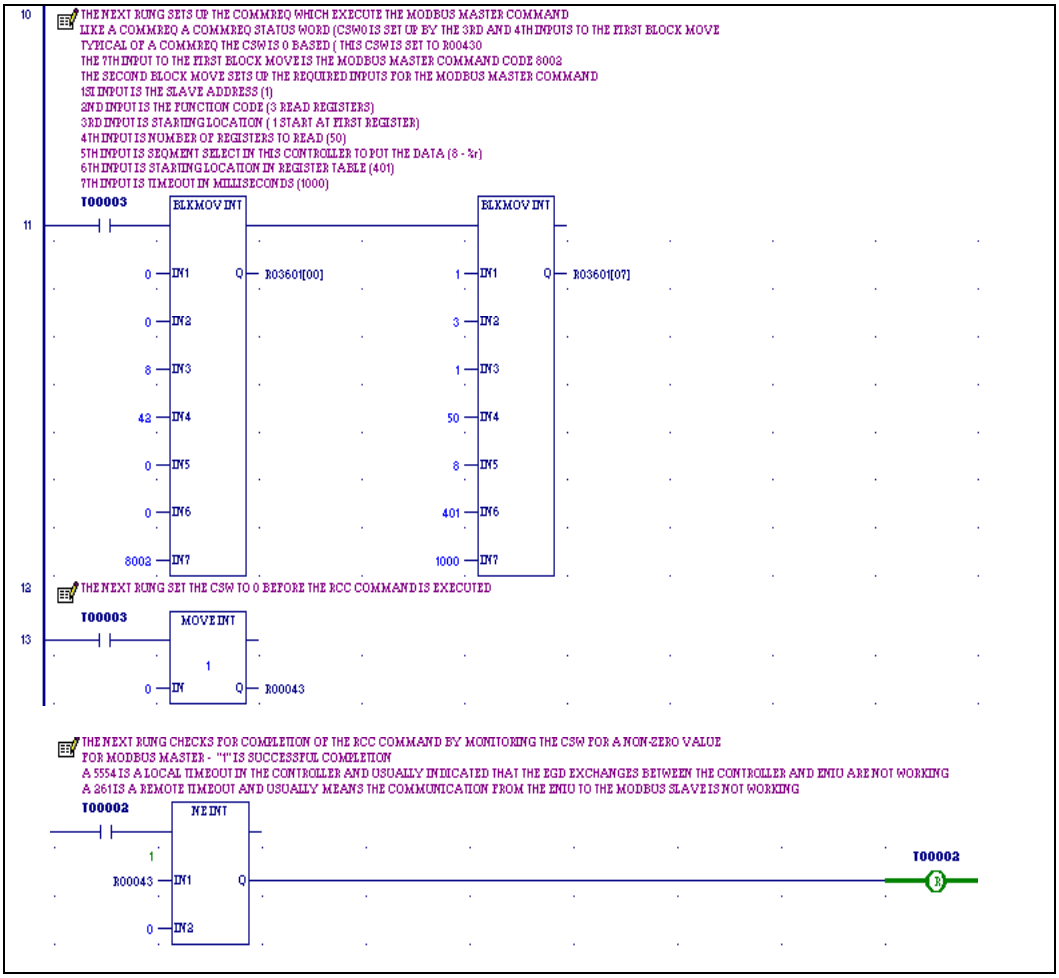
Sample Logic for two Remote COMMREQ Call Commands

Sample logic for Modbus Master is shown first, then sample logic for a Genius Bus Controller "Read Diagnostics" command.

Port 1 of the Ethernet NIU: the Port Mode MUST be changed to Serial I/O and the Baud rate and parity needs to be set to match the Modbus Slave settings. Once the port setup is changed it must be downloaded to the Ethernet NIU.

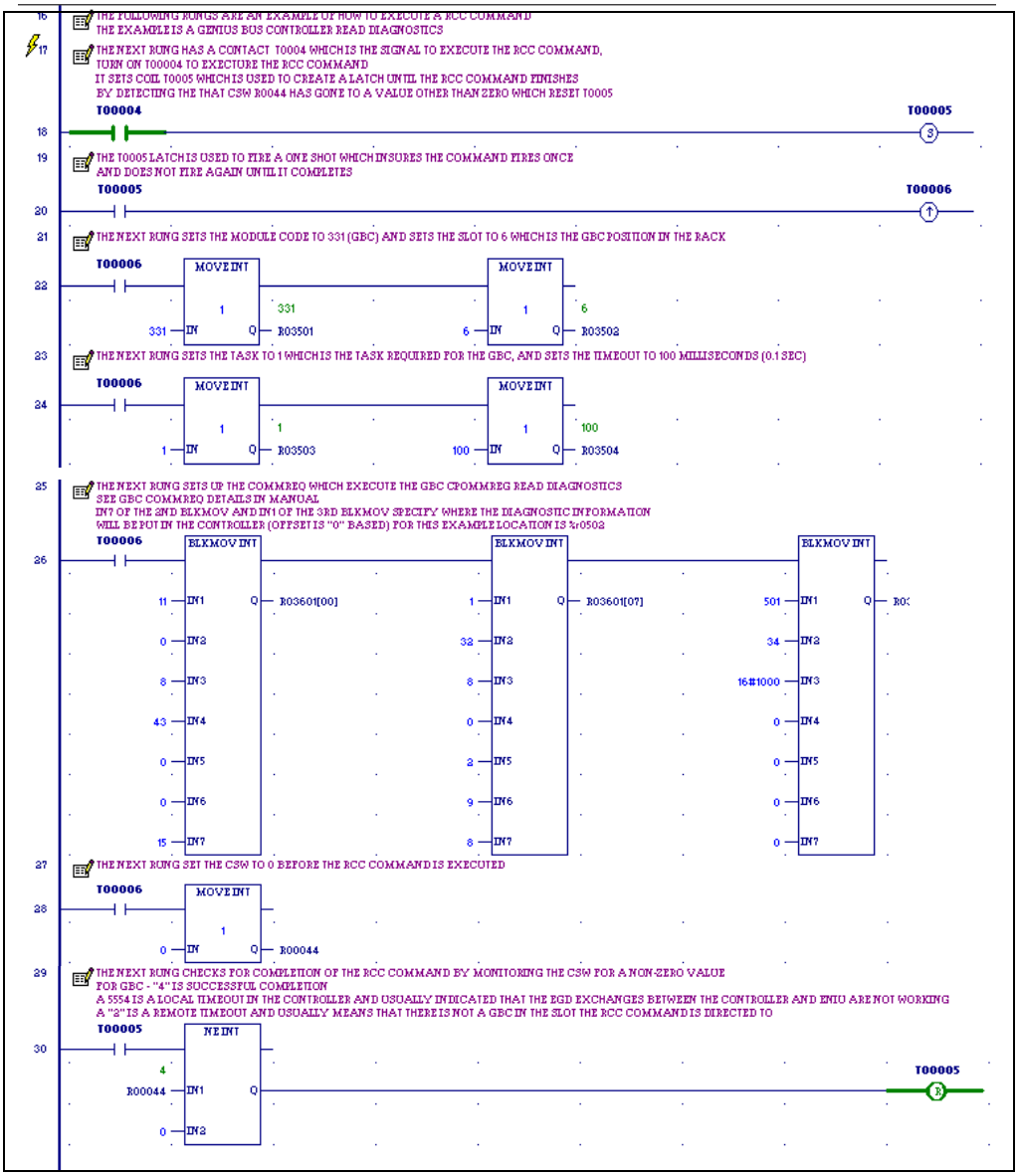
If the port setting are not changed, the example will give an error code of 5379 (1503h).



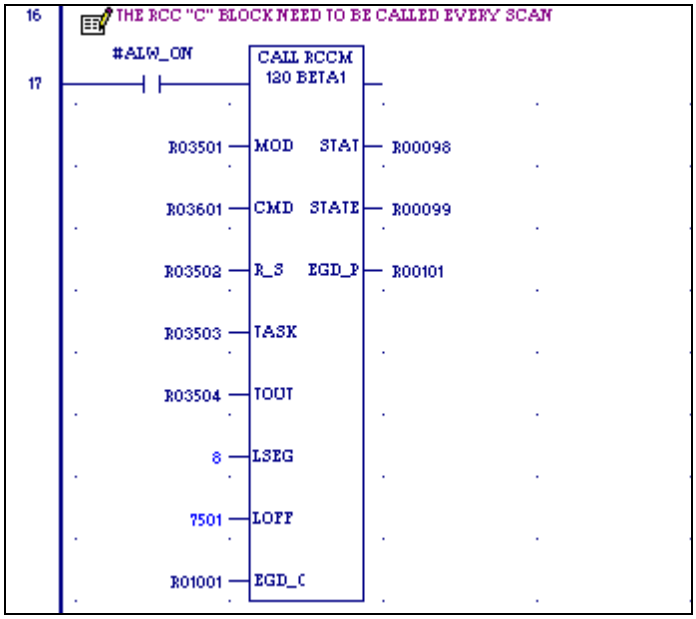

10
THE NEXT RUNG SETS UP THE COMMREQ WHICH EXECUTE THE MODBUS MASTER COMMAND
LIKE A COMMREQ A COMMREQ STATUS WORD (CSW) IS SET UP BY THE 3RD AND 4TH INPUTS TO THE FIRST BLOCK MOVE
TYPICAL OF A COMMREQ THE CSW IS 0 BASED (THIS CSW IS SET TO R00430
THE 7TH INPUT TO THE FIRST BLOCK MOVE IS THE MODBUS MASTER COMMAND CODE 8002
THE SECOND BLOCK MOVE SETS UP THE REQUIRED INPUTS FOR THE MODBUS MASTER COMMAND
1ST INPUT IS THE SLAVE ADDRESS (1)
2ND INPUT IS THE FUNCTION CODE (3 READ REGISTERS)
3RD INPUT IS STARTING LOCATION (1 START AT FIRST REGISTER)
4TH INPUT IS NUMBER OF REGISTERS TO READ (50)
5TH INPUT IS SEGMENT SELECT IN THIS CONTROLLER TO PUT THE DATA (8 - %r)
6TH INPUT IS STARTING LOCATION IN REGISTER TABLE (401)
7TH INPUT IS TIMEOUT IN MILLISECONDS (1000)
T00003
BLKMOV INT
11
0 IN1 Q R03601[00]
0 IN2
8 IN3
42 IN4
0 IN5
0 IN6
8002 IN7
BLKMOV INT
1 IN1 Q R03601[07]
3 IN2
1 IN3
50 IN4
8 IN5
401 IN6
1000 IN7
12
THE NEXT RUNG SET THE CSW TO 0 BEFORE THE RCC COMMAND IS EXECUTED
T00003
MOVE INT
13
1
0 IN Q R00043
THE NEXT RUNG CHECKS FOR COMPLETION OF THE RCC COMMAND BY MONITORING THE CSW FOR A NON-ZERO VALUE
FOR MODBUS MASTER - "1" IS SUCCESSFUL COMPLETION
A 5554 IS A LOCAL TIMEOUT IN THE CONTROLLER AND USUALLY INDICATED THAT THE EGD EXCHANGES BETWEEN THE CONTROLLER AND ENIU ARE NOT WORKING
A 2611 IS A REMOTE TIMEOUT AND USUALLY MEANS THE COMMUNICATION FROM THE ENIU TO THE MODBUS SLAVE IS NOT WORKING
T00002
NE INT
1
R00043 IN1 Q
0 IN2
T00002

Sample Logic for GBC Command Read Diagnostiics

The Genius Bus Controller MUST be configured in slot 6 of the Ethernet NIU.



"C" block used by both sample commands



(Refer to list for step 18)

Step 19. Verify that EGD Exchanges are Working

Outputs %Q1 to %Q2048 are sent from the controller to the Ethernet NIU.

Analog Outputs %AQ1 to %AQ512 are sent from the controller to the Ethernet NIU.

In the controller, turn one or more discrete outputs on and off. Verify that the outputs come on in the Ethernet NIU. This can be done by using Reference View tables in the controller and the Ethernet NIU.

Do the same with analog outputs.

Repeat the same steps to check that the configured inputs from the Ethernet NIU (in "Inputs_from_ENIU_xx" Ethernet Global Data exchange) are being correctly receiver by the controller(s).

Troubleshooting Ethernet I/O

If Outputs and Analog outputs are not changing in the ENIU when they are changed in the controller

- Make sure both the Controller and ENIU are in RUN mode.
- Make sure the Controller and ENIU are both connected to Ethernet.
- Check to see that both Controller and ENIU are sending and receiving EGD exchanges without error. This can be checked by using Station Manager on the Ethernet ports and using a stat g command. Refer to GFK-2225, the *Ethernet TCP/IP for PACSystems Station Manager* manual for details.

Step 20. Verify that Remote COMMREQ Call Commands are Working

Use one or both of the samples above.

Enter the sample RCC code from above.

To test the Modbus example, connect a Modbus Slave to port 1 of the Ethernet NIU. Toggle T0001 ON and then check R0043 for the result. "1" is success. If an error is returned, check the error code section to determine what the error is.

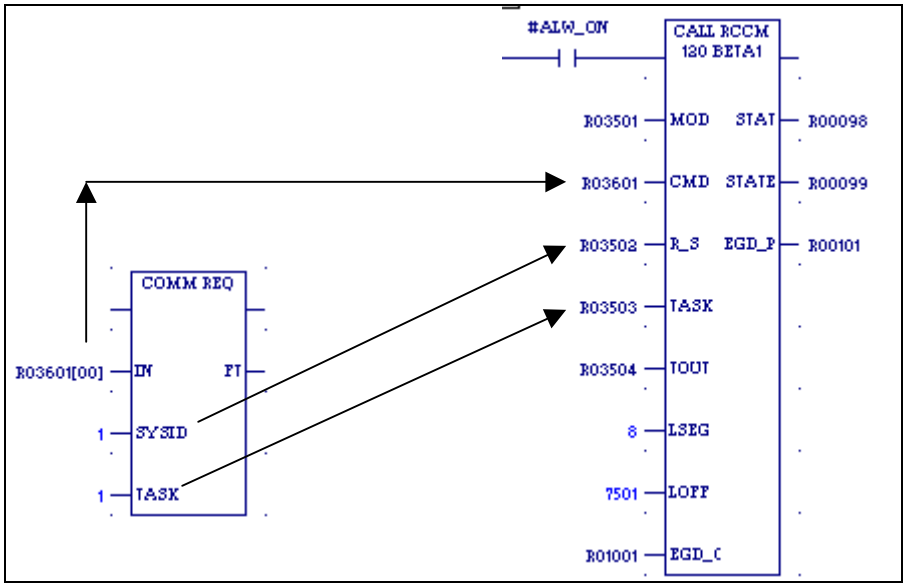
Enter the sample GBC code from above.

To test the Genius Bus Controller example, add a Genius Bus Controller in slot 6 of the Ethernet NIU. Configure the slot and store the folder to the Ethernet NIU. Toggle T0004 ON and then check R0044 for the result. "4" is success. If an error is returned check the error code section to determine what the error is.

Converting a COMMREQ to a Remote COMMREQ Call Command

As the picture below shows:

- The “IN “ of the COMMREQ becomes the “CMD” of the Remote COMMREQ Call “C” block
- The “SYSID” of the COMMREQ becomes the “R_S” of the Remote COMMREQ Call.
- The “TASK” of the COMMREQ is the “TASK” of the Remote COMMREQ Call.
- The COMMREQ Status Word is used the same way in Remote COMMREQ Calls, and is the mechanism for checking for completion just as it is in a COMMREQ.



Chapter 3 Installation

When installing the Ethernet NIU and the modules in its I/O Station, the primary references for installation instructions should be the *PACSystems RX3i System Manual, GFK-2314*, and the *Series 90-30 PLC Installation Manual*, GFK-0356.

This chapter provides additional installation information for the Ethernet NIU and I/O Station.

- Meeting Agency Standards and Requirements
- Installing the Ethernet NIU
  - Backplane Locations for the NIU
  - Programmer Connection
  - Serial Ports
- Ethernet Connections to the Ethernet Transmitter Module
- Starting Up the Ethernet NIU

Meeting Agency Standards and Requirements

Before installing GE Fanuc products in situations where compliance to standards or directives from the Federal Communications Commission, the Canadian Department of Communications, or the European Union is necessary please refer to GE Fanuc's *Installation Requirements for Conformance to Standards*, GFK-1179.

CE Mark Installation Requirements

The following requirements for surge, electrostatic discharge (ESD), and fast transient burst (FTB) protection must be met for applications that require CE Mark listing:

- The I/O Station is considered to be open equipment and should therefore be installed in an enclosure (IP54).
- This equipment is intended for use in typical industrial environments that utilize anti-static materials such as concrete or wood flooring. If the equipment is used in an environment that contains static material, such as carpets, personnel should discharge themselves by touching a safely grounded surface before accessing the equipment.
- If the AC mains are used to provide power for I/O, these lines should be suppressed prior to distribution to the I/O so that immunity levels for the I/O are not exceeded. Suppression for the AC I/O power can be made using line-rated MOVs that are connected line-to-line, as well as line-to-ground. A good high-frequency ground connection must be made to the line-to-ground MOVs.
- AC or DC power sources less than 50V are assumed to be derived locally from the AC mains. The length of the wires between these power sources and the PLC should be less than a maximum of approximately 10 meters.
- Installation must be indoors with primary facility surge protection on the incoming AC power lines.
- In the presence of noise, serial communications could be interrupted.

Installation in Hazardous Locations

- Equipment labeled with reference to Class I, Groups A, B, C & D, Div. 2 Hazardous locations is suitable for use in Class I, Division 2, Groups A, B, C, D or non-hazardous locations only.
- Warning - explosion hazard - substitution of components may impair suitability for Class I, Division 2;
- Warning - explosion hazard - when in hazardous locations, turn off power before replacing or wiring modules; and
- Warning - explosion hazard - do not disconnect equipment unless power has been switched off or the area is known to be non-hazardous.

Installing the Ethernet NIU

It is the responsibility of the OEM, system integrator, or end user to properly install the control system equipment for safe and reliable operation. Installation should not be attempted without referring to the *PACSystems RX3i Hardware and Installation Manual,* GFK-2314.

1. Make sure that backplane power is off.
2. Install the Ethernet NIU module in the I/O Station backplane 0. The NIU requires two slots and can use any slots except the highest numbered (rightmost) slot. It is recommended that the ENIU be located in slots 2 and 3. For more information about choosing a slot for the ENIU, see below.
3. Turn on power. The module should power up. When the NIU has successfully completed initialization, the NIU OK LED stays on and the NIU SCANNING I/O and EN LEDs are off.
4. To save battery life, do not connect the battery for the first time until the ENIU is installed in the backplane and the backplane powered on. The battery may then be attached to either of the two terminals in the battery compartment. Once that is done, the ENIU may be powered down and normal battery back up operation will begin.

Backplane Locations for the ENIU

1. The A/C Power Supply (IC695PSAx40) for the RX3i is a doublewide module whose connector is left-justified as viewed when installed in a backplane. It cannot be located in slot 11 of a 12-slot backplane or slot 15 of a 16-slot backplane. No latch mechanism is provided for the last (rightmost) slot in a backplane, so it is not possible to place the power supply in the second to last slot.
2. The Ethernet NIU is a doublewide module whose connector is right-justified as viewed when installed in a backplane. The ENIU is referenced for configuration and application logic by the leftmost slot occupied by the entire module, not by the slot the physical connector is located in. For example, if the ENIU has its connector inserted in slot 3, the module occupies slots 2 and 3 and the ENIU is referenced as being located in slot 2.
   - The ENIU may be located in slot 0 with its connector in slot 1.
   - The ENIU cannot be located in slot 11 of a 12-slot backplane or in slot 15 of a 16-slot backplane, because its connector cannot be installed in the slot reserved for an expansion module.
3. When migrating a Series 90-30 ENIU system to a PACSystems RX3i ENIU, maintaining the slot 1 location of the ENIU means that only a singlewide power supply may be used in slot 0. Either DC power supply can be used (IC695PSD040 or IC695PSD140). Therefore, if the application must maintain a slot 1 ENIU and uses an AC power-supply, the RX3i system must have the RX3i AC power-supply located in a slot to the right of the RX3i ENIU in slot 1.

Locating the ENIU in a Slot Other than 1

Before deciding to place the ENIU in a slot other than slot 1, it is important to consider the possible application migration issues that could arise, as explained below.

Ethernet Transmitter Module

The Ethernet Transmitter Module in the I/O Station must be installed in slot 4. If there is a second Ethernet Transmitter Module in the I/O Station, it can go in any available slot in the Ethernet NIU backplane.

Application Program

For Service Request #15 (Read Last-Logged Fault Table Entry) and Service Request #20 (Read Fault Tables), the location of ENIU faults is not the standard 0.1 location, but the slot the ENIU is located in (see above). Logic that decodes fault table entries retrieved by these service requests may need updating.

COMMREQs directed to the ENIU (e.g. those directed to the serial ports of the ENIU) will need to be updated with the correct ENIU slot reference.

Series 90 PLCs

Remote Series 90 PLCs that use SRTP Channels COMMREQs expect the ENIU to be in slot 1 or slot 2. To support communications with Series 90 SRTP clients such as Series 90 PLCs using SRTP Channels, the RX3i internally redirects incoming SRTP requests destined for {backplane 0, slot 1} to {backplane 0, slot 2}, provided that the ENIU is located in backplane 0 slot 2 (and the remote client has not issued an SRTP Destination service on the connection to discover the backplane and slot of the ENIU). This special redirection permits Series 90-30 applications that expect the power supply to be located leftmost and the ENIU to be located to the right of the power supply to function. Attempts to establish channels with ENIUs in slots other than 1 or 2 will fail if initiated from Series 90 PLCs.

HMI and External Communication Devices

All external communication devices that interact with the Ethernet NIU should be checked for compatibility with ENIU slot locations other than slot 1. Problems may arise with, but are not limited to, initial connection sequences and fault reporting. Machine Edition View users should select “GE SRTP” as their communications driver – it can communicate with a ENIU in any slot.

Programmer Connection

The programmer can communicate with the NIU via serial port 1, serial port 2, or the backplane-based Ethernet interface. Connecting a programmer via an Ethernet TCP/IP network requires a CAT5 standard Ethernet cable with RJ-45 connectors.

Before connecting the programmer and ENIU to the Ethernet TCP/IP network, set the IP address using the Initial IP Address software tool. After setting the IP address, connect the RX3i and the computer running the programming software to the Ethernet Interface. For detailed information on programmer connection via Ethernet TCP/IP, refer to the *TCP/IP Ethernet Communications for PACSystems User's Manual,* GFK-2224.

Firmware Upgrades

The ENIU uses non-volatile flash memory for storing the operating system firmware. This allows firmware to be updated without disassembling the module or replacing EPROMs.

To install a firmware upgrade, connect WinLoader to the NIU RS-232 or RS-485 serial port. When connecting directly to the NIU, there is no need to specify the Backplane/Slot location. For upgrades to smart modules (the IC695ETM001, for example), which are performed indirectly via the NIU serial port, you must specify a backplane/slot location.

Serial Ports

The NIU has two independent, on-board serial ports, accessed by connectors on the front of the module. These ports provide serial interfaces to external devices.

Protocols Supported

Protocol	*Port 1*	*Port 2*
RTU (slave)	Yes	Yes
SNP Slave	Yes	Yes
Serial I/O *	Yes	Yes
Firmware Upgrade	ENIU in STOP/No I/O mode	
Message Mode (C Runtime Library Functions: serial read, serial write, sscanf, sprintf)	Yes	Yes

* Modbus Master is supported in application code in Serial I/O mode.

Serial Port Baud Rates

Protocol	*Port 1 (RS-232)*	*Port 2 (RS-485)*
Modbus RTU Slave protocol	1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	
Message	1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	
Firmware Upgrade via Winloader	2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	
SNP Slave	1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	
Serial I/O	1200, 2400, 4800, 9600, 19.2K, 38.4K, 57.6K, 115.2K	

Port 1

Port 1 (COM1) is RS-232 compatible. It has a 9-pin, female, D-sub connector with a standard pin out. This is a DCE (data communications equipment) port that allows a simple straight-through cable to connect with a standard AT-style RS-232 port. The COM1 Active LED provides the status of serial port activity.

Port 1 RS-232 Signals

Pin	*Signal*	*Description*
1*	NC	No Connection
2	TXD	Transmit Data
3	RXD	Receive Data
4	DSR	Data Set Ready
5	0V	Signal Ground
6	DTR	Data Terminal Ready
7	CTS	Clear To Send
8	RTS	Request to Send
9	NC	No Connection

* Pin 1 is at the bottom right of the connector as viewed from the front of the module.

Port 2

Port 2 (COM2) is RS-485 compatible. Port 2 has a 15-pin, female D-sub connector. This port supports the RS-485 to RS-232 adapter (IC690ACC901). This is a DCE port. The COM2 Active LED provides the status of serial port activity.

Port 2 RS-485 Signals

Pin	*Signal*	*Description*
1*	Shield	Cable Shield
2	NC	No Connection
3	NC	No Connection
4	NC	No Connection
5	+5VDC	Logic Power**
6	RTS(A)	Differential Request to Send
7	0V	Signal Ground
8	CTS(B‘)	Differential Clear To Send
9***	RT	Resistor Termination
10**	RD(A‘)	Differential Receive Data
11	RD(B‘)	Differential Receive Data
12	SD(A)	Differential Send Data
13	SD(B)	Differential Send Data
14	RTS(B)	Differential Request To Send
15	CTS(A’)	Differential Clear To Send

* Pin 1 is at the bottom right of the connector as viewed from the front of the module.

** Pin 5 provides isolated +5VDC power (300mA maximum) for powering external options.

*** Termination resistance for the RD A’ signal should be connected on units at the end of the line. To make this termination, connect a jumper between pins 9 and 10 inside the 15-pin D-shell.

Serial Cable Lengths and Shielding

The connection from a NIU serial port COM1 to the serial port on a computer or other serial device requires a serial cable. This connection can be made with the IC200CBL001 cable kit or you can build cables to fit the needs of your particular application.

Maximum cable lengths (the total length from the NIU to the last device attached to the serial cable) are:

- Port 1 (RS-232) – 15 meters (50 ft.), shielded cable optional
- Port 2 (RS-485) – 1200 meters (4000 ft.), shielded cable required

Ethernet Connections to the Ethernet Transmitter Module

Ethernet Transmitter Module IC695ETM001 provides the I/O Station's connection to the Ethernet network. The Ethernet Transmitter Module has two Ethernet port connectors, each of which supports both 10Base-T and 100Base-Tx operation using either full duplex or half duplex operation.

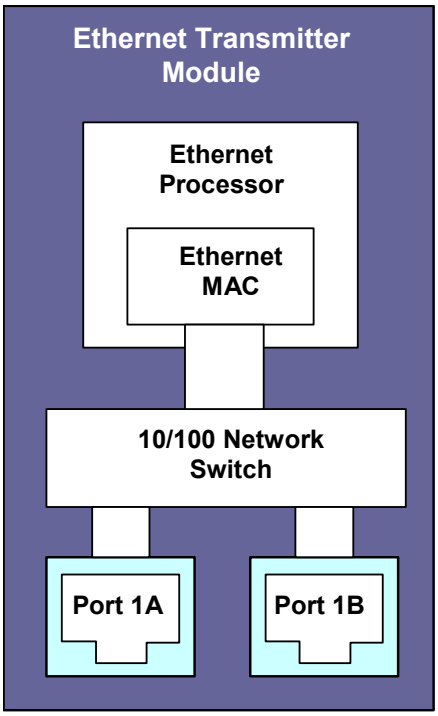
Ethernet Cable

Category 5 cable is required for 100Base-TX operation, and recommended for all installations. 10Base-T / 100Base-TX cables are readily available from commercial distributors. GE Fanuc recommends purchasing rather than making cables. Cables must meet the applicable IEEE 802.3 or 802.3u standard, noted in the table below.

The Ethernet Transmitter Module automatically senses whether it is connected to a 10BaseT or 100BaseTX network, whether communications are half-duplex or full duplex, and automatically determines if a straight through or crossover connection is being used..

Embedded Switch

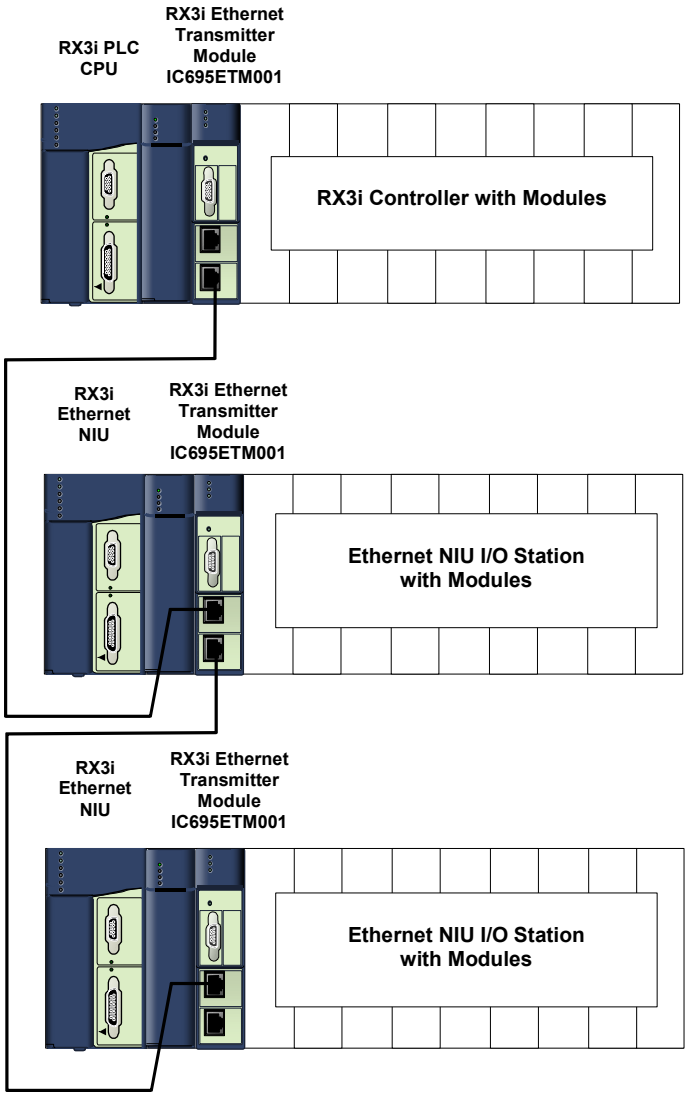
The two Ethernet port connectors on the Ethernet Transmitter Module are controlled by an embedded network switch. The module has only one interface to the network (one Ethernet address and one IP address).



I/O Station Connections with a Single Controller

The two-port embedded switch on the RX3i Ethernet Transmitter Module makes it possible to connect the Ethernet NIU I/O Station to both an upstream controller and an additional downstream I/O Station. A PACSystems RX3i controller is shown below. However, another type of controller with a compatible Ethernet interface, such as a PACSystems RX7i, could be used instead.

The second connector on the Ethernet Transmitter Module in the I/O Station can then be used to further daisy-chain to a third I/O Station, and so on. It is important to remember that when if any I/O Station in the chain is powered down, it disrupts I/O data communication to all subsequent Ethernet Transmitter Modules and the I/O Stations in which they are located. If that type of operation is not acceptable for the application, Ethernet network switch devices should be used.



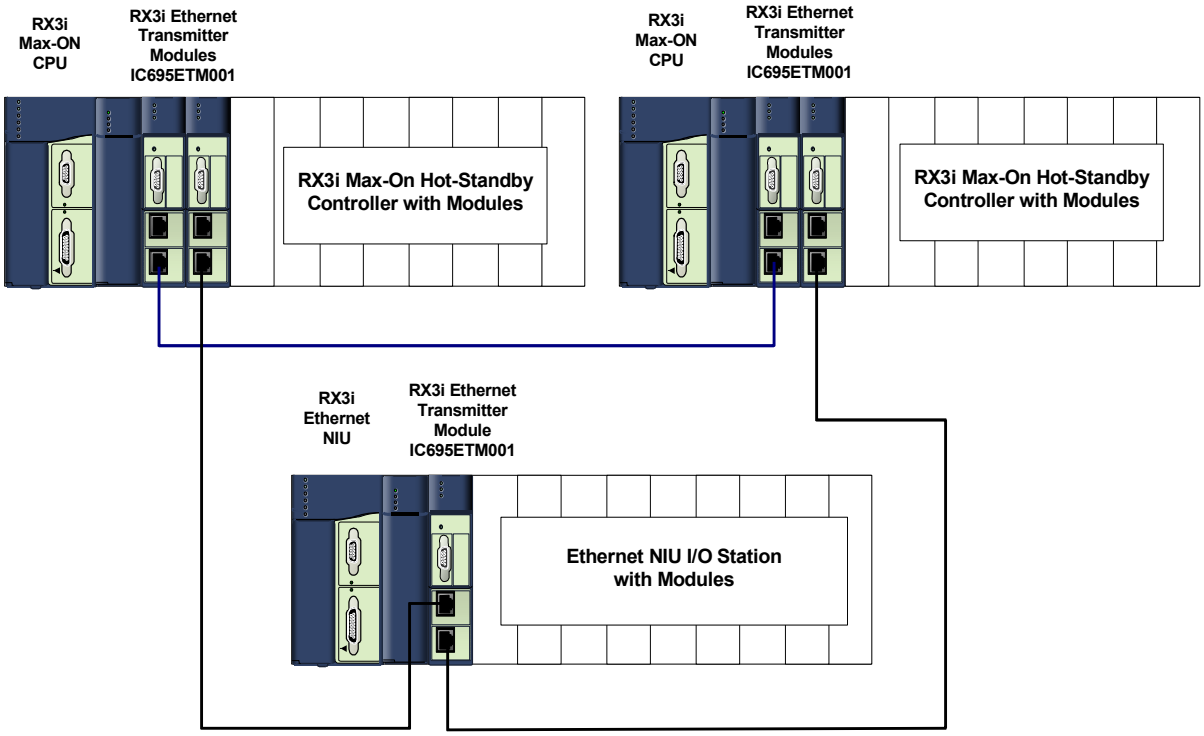
I/O Station Connections with Redundant Controllers

If only one RX3i Ethernet NIU I/O Station is used in a system that includes 2 controllers in a redundant hot standby configuration, the two connectors on the Ethernet Transmitter Module may be used to connect to each of the two redundant CPUs.

Redundant Max-ON CPU Controllers

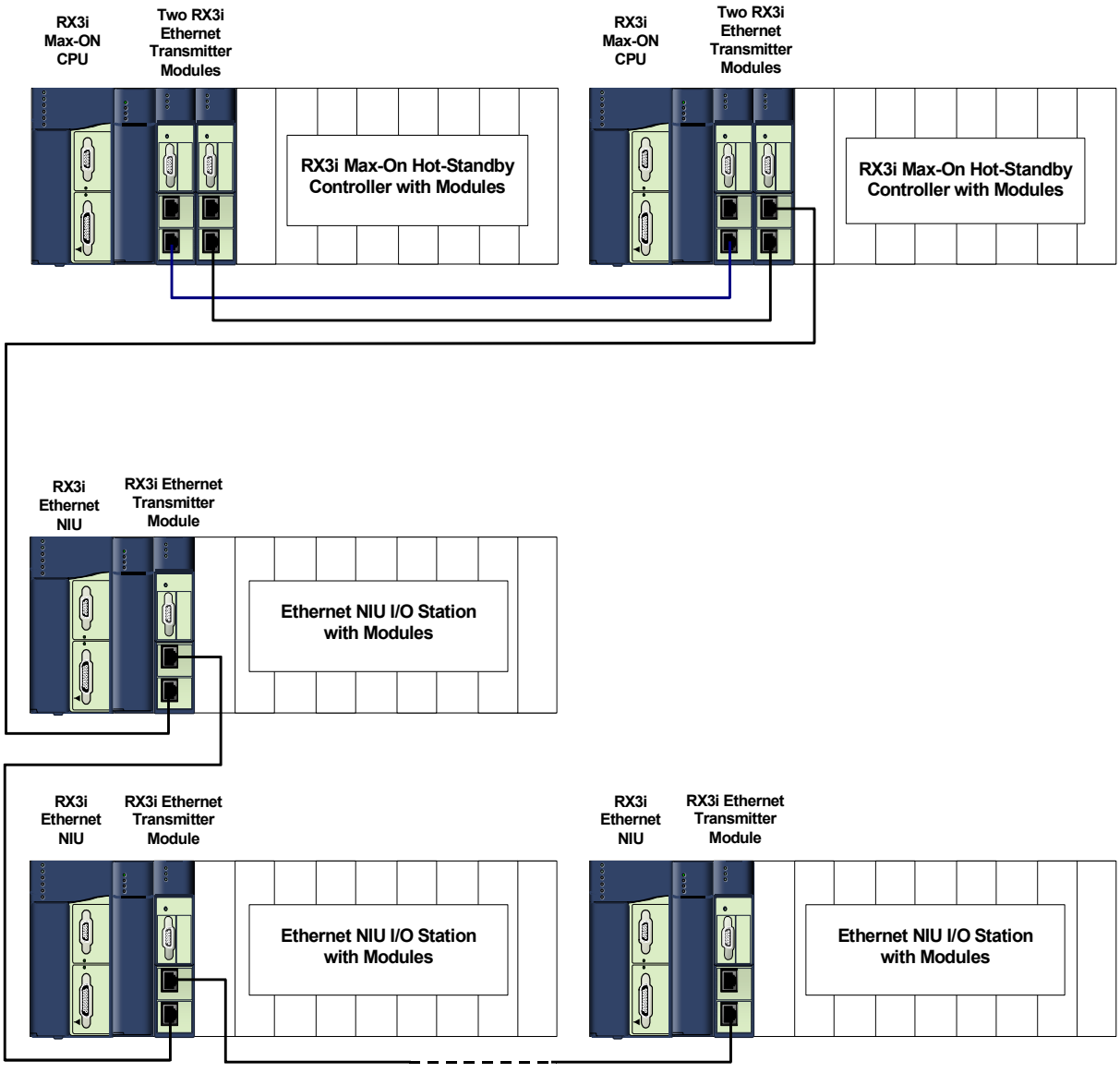
For RX3i Max-ON Hot Standby redundant controllers, there must be a pair of Ethernet Transmitter Modules in each Max-ON controller. The first pair of Ethernet Transmitter Modules in the Max-ON controllers is dedicated to synchronizing application data. To maintain the higher synchronization performance, other devices should not be connected on this synchronization link.

The second pair of Ethernet Transmitter Modules in the Max-ON controllers is connected to the dual port connectors on the Ethernet Transmitter Module in the I/O Station.



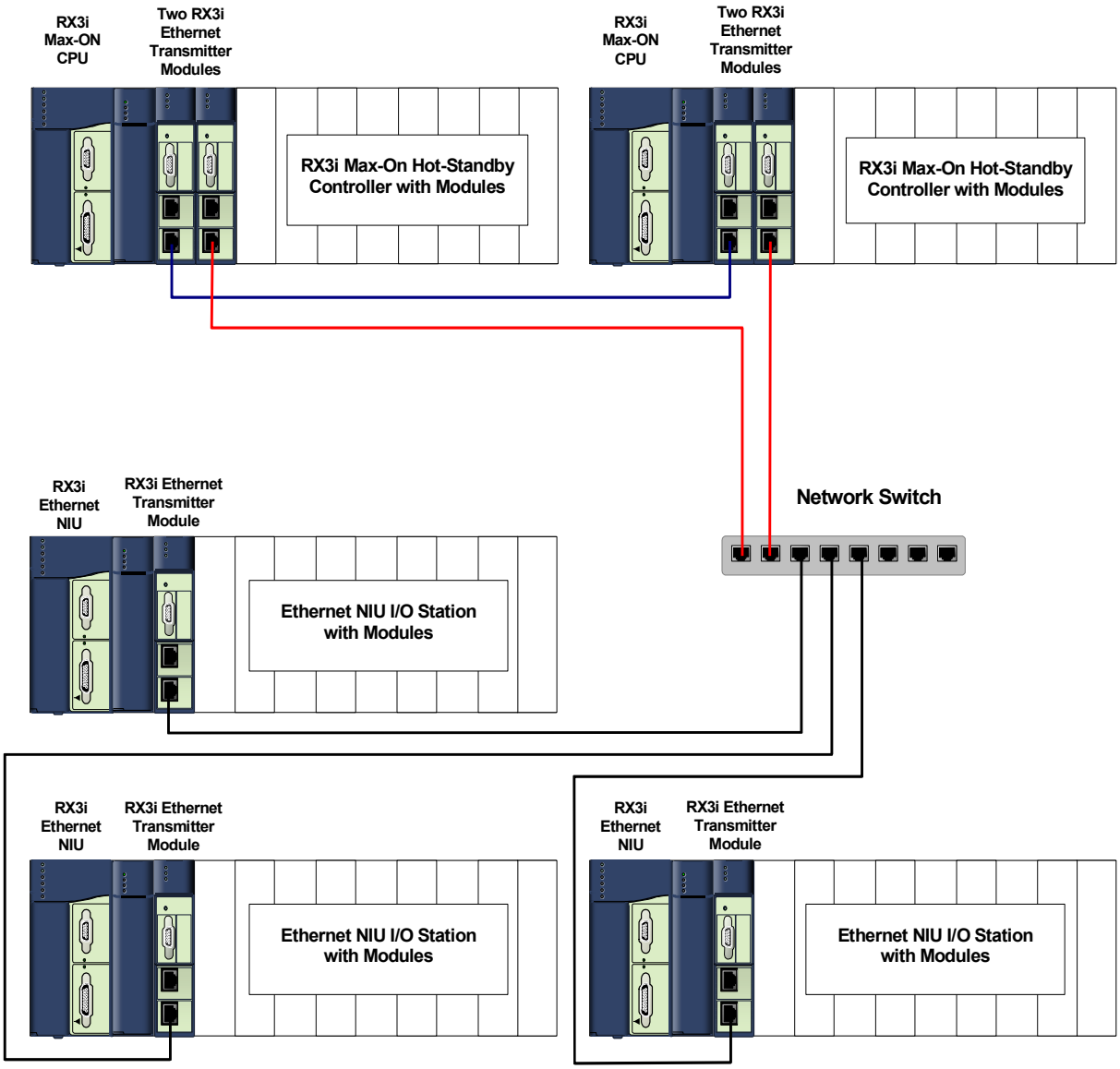
Connections for Redundant Controllers with Multiple I/O Stations

If more than one Ethernet NIU I/O Station will be connected to the redundant controllers, the second connector on one of the controllers can be used to extend the daisy-chain to a second Ethernet NIU I/O Station and so on. It is important to remember that when if any I/O Station in the chain is powered down, it disrupts I/O data communication to all subsequent Ethernet Transmitter Modules and the I/O Stations in which they are located. If that type of operation is not acceptable for the application, Ethernet network switch devices should be used.



Connections for Redundant Controllers using Network Switch Devices

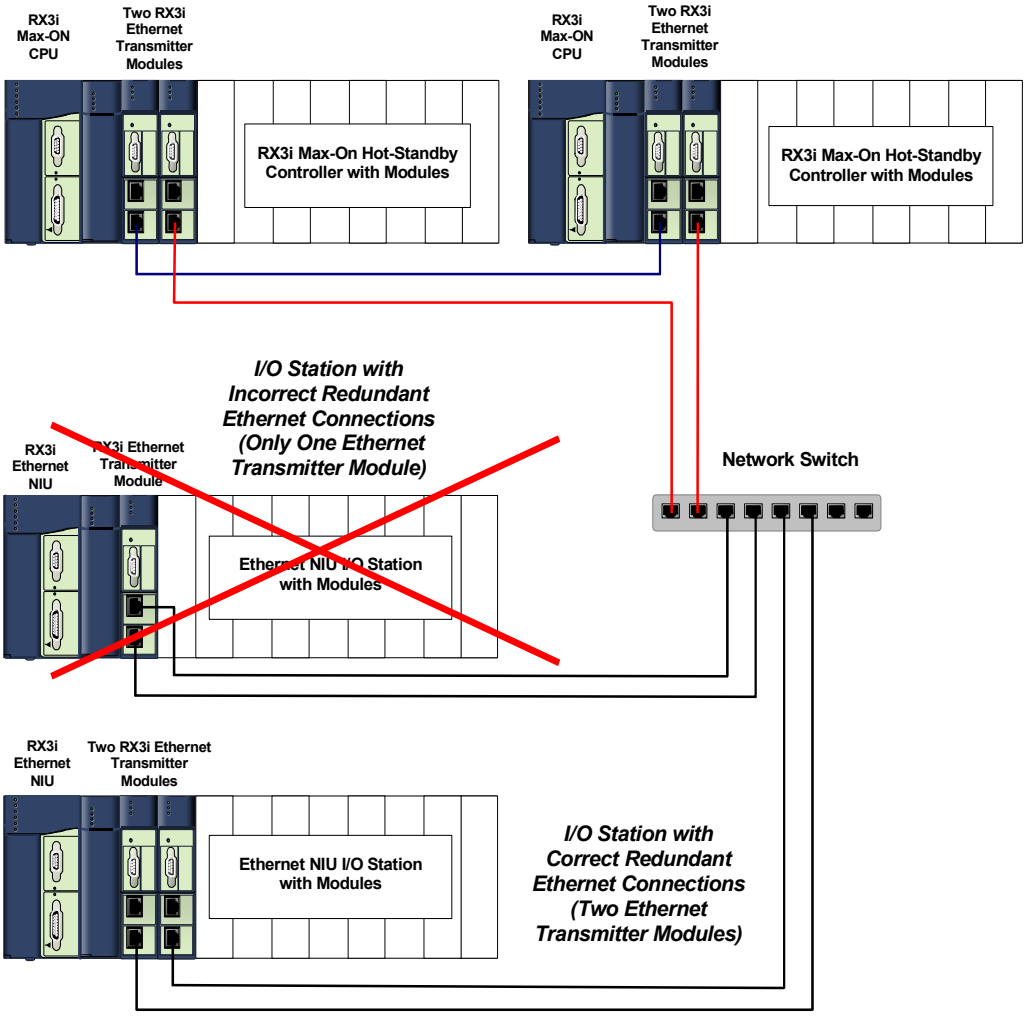
Ethernet switches may be used to facilitate connection of multiple Ethernet NIU I/O Stations as shown. This type of network topology and network connection prevents the disruption of communications to other Ethernet NIU I/O Stations or controllers in the system. When an Ethernet NIU I/O Station or controller is powered down, all other devices can continue Ethernet communications through the network switch(es).



Redundant Ethernet Cable Connections

Generally, having redundant Ethernet cable connections to an Ethernet NIU I/O Station requires installing two Ethernet Transmitter Modules in the I/O Station. This prevents communication loops that will occur if the same network is connected to both connectors (one logical port – one IP address), of one Ethernet Transmitter Module.

Some network switches have STP functionality and can be configured to logically "open" and prevent these loops. However the performance of both STP and RSTP is usually considered unacceptable for real-time I/O and control use. Depending on the complexity of the system, if a redundant connection is lost, STP and RSTP can take several seconds or even minutes to recover and provide a communications path over the redundant connection. For this reason, two RX3i Ethernet Transmitter Modules should be installed in an I/O Station that requires redundant Ethernet connections. That provides two completely separate interfaces, each with its own IP address, preventing the possibility of a communication loop.



Starting Up the Ethernet NIU

Ethernet NIU LED Operation

The following table lists the Ethernet NIU LED functions during normal operation (after initialization sequence is complete).

LED State (On / Blinking / Off)			NIU Operating State
On	**NIU OK**	On	NIU has passed its powerup diagnostics and is functioning properly.
Off	**NIU OK**	Off	NIU problem. RUN and OUTPUTS ENABLED LEDs may be blinking in an error code pattern, which can be used by technical support for troubleshooting. This condition and any error codes should be reported to your technical support representative.
Blinking	**NIU OK**, **OUTPUTS ENABLED, NIU SCANNING I/O**	Blinking in unison	NIU is in boot mode and is waiting for a firmware update through serial port.
On	**NIU SCANNING I/O**	On	NIU is in Run mode
Off	**NIU SCANNING I/O**	Off	NIU is in Stop mode.
On	**OUTPUTS ENABLED**	On	Output scan is enabled.
Off	**OUTPUTS ENABLED**	Off	Output scan is disabled.
On	**I/O FORCE**	On	Override is active on a bit reference.
Blinking	**BATTERY**	Blinking	Battery is low.
On	**BATTERY**	On	Battery is dead or not attached.
On	**SYSTEM FAULT**	On	NIU is in Stop/Faulted or Stop/Halted mode.
Blinking	**COM1** **COM2**	Blinking Blinking	Signal activity on port.

LED States During Power-up

If a problem is occurs during power-up, the Ethernet NIU may not transition directly to the operational state. In that case, check the LED pattern on the module and refer to the following chart and table for corrective action.

Chapter 4 I/O Data - Control, Status, and I/O Data Formats

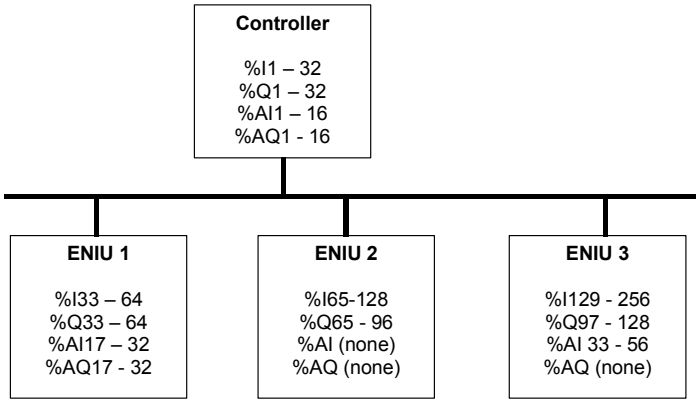
This chapter describes the content of the I/O data exchanged by the Ethernet NIU and the controller.

- System I/O Data References
- Data Memory in the Ethernet NIU
  - References Used in the Ethernet NIU
  - Discrete and Analog Outputs in the Ethernet NIU
- Exchanging Data with One or Two Controllers
  - ENIU Operation with Two Controllers
  - ENIU Operation if No Data is Received
- Control Data Format
- Status Data Format
- Using the Control and Status Data
  - Switching Back to the Primary Controller
  - Setting Up the Output Defaults
  - Checking for Faults and Clearing Faults
  - Using the Optional Application-Specific Command Word

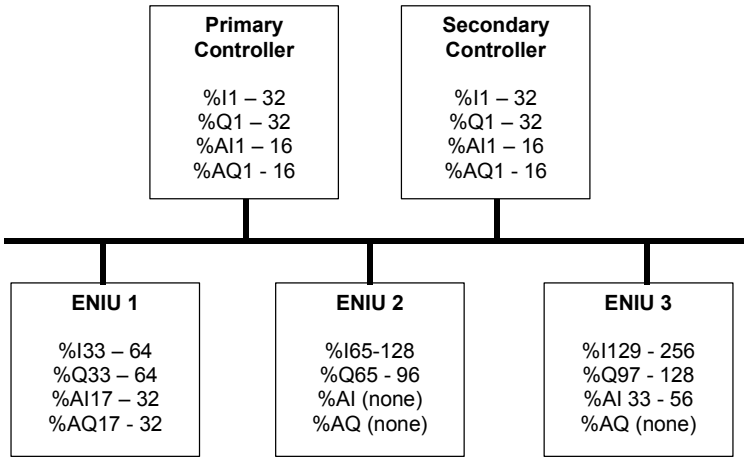
System I/O Data References

I/O modules are added to the Ethernet NIU configuration and their parameters are configured the same way they are configured in a PLC system.

To a controller, the I/O data it exchanges with Ethernet NIUs on the network is part of its overall I/O system. If the same controller serves multiple Ethernet NIUs and their I/O Stations, each I/O Station MUST use a unique set of I/O references, as shown in the simplified example below. Duplicated I/O references for multiple Ethernet NIUs would be overwritten in the controller's memory.



If an I/O Station has two controllers, the local I/O in each controller would use all of the same I/O references. In the illustration below, both controllers use the same local references.



Data Memory in the Ethernet NIU

The Ethernet NIU has the following types of data memory:

Discrete Input Points - %I	32768 (fixed)
Discrete Output Points - %Q	32768 (fixed)
Discrete Global Memory - %G	7680 (fixed)
Internal Coils - %M	32768 (fixed)
Output (Temporary) Coils - %T	1024 bits (fixed)
System Status References - %S	128 bits (%S, %SA, %SB, %SC - 32 bits each) (fixed)
Register Memory - %R	32640 (Default is 9999)
Analog Inputs - %AI	32640 (Default is 2048)
Analog Outputs - %AQ	32640 (Default is 512)

References Used in the Ethernet NIU

The references used by the Ethernet NIU for its I/O, status, and control data are assigned during configuration.

The Ethernet NIU maps data into its internal memory as shown below. *The references shown in italics for status and control data are required for correct operation. These reference addresses are automatically pre-populated in Ethernet Global Data exchanges when the ENIU target is created.*

Type of Data	***Ethernet NIU References***
Discrete Inputs from field devices	%I0001 - %I32768 (bits)*
Discrete Outputs from controller (primary / only)	*Must be %M0001 - %M2048 (bits)*
Discrete Outputs from optional secondary controller	*Must be %M2049 - %M4096 (bits)*
Ethernet Global Data Exchange status (consumed from primary / only controller)	*Must be %T0001 - %T0016 (bits)*
Ethernet Global Data Exchange status (consumed from secondary controller)	*Must be %T0017 - %T0032 (bits)*
Ethernet Global Data Exchange status (produced by ENIU)	*Must be %T0033 - %T0048 (bits)*
Analog Inputs from field devices	%AI001 - %AI32640 (words)*
Analog Outputs from controller (primary / only)	*Must be %R0001 - %R0512 (words)*
Analog Outputs from optional secondary controller	*Must be %R0513 - %R1024 (words)*
ENIU Status data to be sent to controller(s)	*Must be %R1101 - %R1110 (words)*
Control Data (from primary / only controller)	*Must be %R1111 - %R1120 (words)*
Control Data (from secondary controller)	*Must be %R1121 - %R1130 (words)*

* Input and Analog input references are the ranges available. Actual used references are added as ranges in EGD exchange “Inputs_from_ENIU_xx”.

Discrete and Analog Outputs in the Ethernet NIU

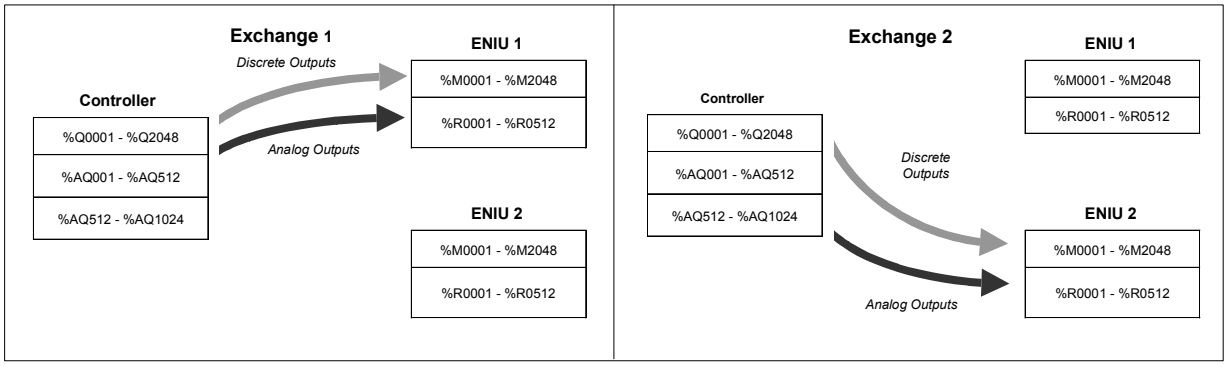
The Ethernet NIU receives discrete Output data and Analog Output Data from a primary controller and optionally from a secondary controller. In order to allow the ENIU to use default values for outputs when communication to the controller(s) is lost, The Output data in the EGD consumed exchange is placed in the %M table for discrete outputs and in %R memory for analog outputs.

The Ethernet NIU moves the discrete and analog output data to the %Q and %AQ tables after determining that data is being received from an active controller. If no active controller is available then the ENIU moves zeros, hold last state, or default values to the %Q and %AQ tables as directed by the control bit from the last active controller.

Typically, the controller sends 2048 discrete outputs and 512 analog outputs. When the data is received, the Ethernet NIU places it in memory beginning at the first reference in each table (for example, %Q0001). However, this is not required. The exchange definitions for both the controller and the Ethernet NIU can be adjusted for improved performance by only transferring the data actually used in the system.

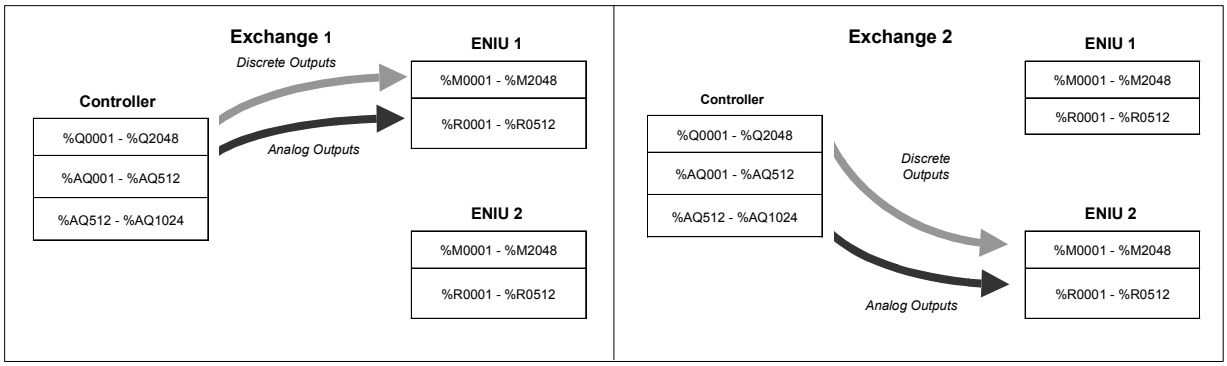
Using Multiple Exchanges for Systems with more than 2048 Discrete Outputs or more than 512 Analog Outputs

In a system with multiple Ethernet NIUs, it is possible for the total amount of discrete output data needed for of all the ENIUs to exceed the 2048 bit limit or Analog output data to exceed the 512 word limit. In either case, the controller must produce multiple exchanges to send all the output data. Each exchange will have different discrete outputs (%Q) and/or analog outputs(%AQ). A bank of Ethernet NIUs will receive each exchange, and each exchange will have different output references. When an Ethernet NIU receives its exchange, it stores the discrete outputs in discrete memory (%M) and analog outputs in (%R) as described above. However, some of the Ethernet NIUs will use different reference addresses for the discrete and/or analog output data than the addresses used in the controller:



Using Multiple Exchanges for Systems with More than 512 Analog Outputs

In a system with multiple Ethernet NIUs, it is possible for the total amount of analog output data of all the ENIUs to exceed the 512 word limit of one ENIU. In that case, the controller must produce multiple exchanges to send all the output data. Each exchange can have the same discrete outputs (%Q), but different analog outputs (%AQ). When an Ethernet NIU receives its exchange, it stores the discrete outputs in discrete memory as described above. However, some of the Ethernet NIUs will use different reference addresses for the analog output data than are used in the controller:



Exchanging Data with One or Two Controllers

In addition to the Ethernet NIU's primary controller, there can also be a secondary controller that provides backup if the primary controller becomes unavailable. Chapter 4 explains how to set up messaging between the ENIU and one or two controllers.

ENIU Operation with Two Controllers

If the system includes a primary controller and a secondary controller, both controllers regularly send output and control data for the I/O Station, and receive the latest input and status data from the Ethernet NIU.

During normal operation, the Ethernet NIU uses the output and control data it receives from its primary controller. However, if the ENIU stops receiving data from the primary controller, the ENIU begins using output and control data from the secondary controller instead.

After the ENIU has started using data from the secondary controller, it keeps using data from the secondary controller until it receives a command from the primary controller (in the control data portion of the output message) telling it to switch back.

The primary controller can also command the Ethernet NIU to switch to the secondary. If the secondary controller is not available, the Ethernet NIU will NOT switch.

ENIU Operation if No Data is Received

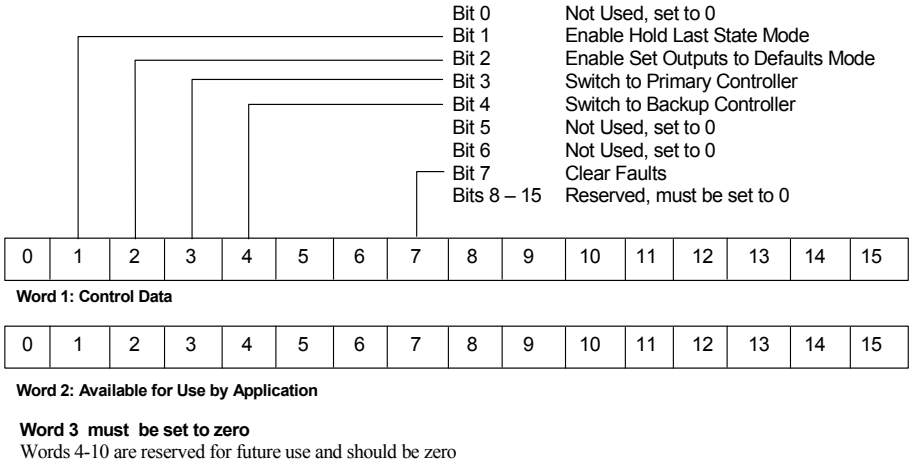
If the Ethernet NIU does not receive output and control data from any controller within the configured timeout period, the ENIU sets the outputs in the I/O Station to their defaults, holds them in their last state, or zeroes the outputs. How outputs will behave if communications are lost is determined by the output control bits (described later in this section).

If the Ethernet NIU has not received output and control data from any controller since the ENIU powered up, the state of the ENIU outputs is normally the default state. It is possible to change this option so that the ENIU outputs are zeroed after powerup if no controller communications have been received. To make this change go, to the variable InitDefaults for the Ethernet NIU in Machine Edition and change the initial value from 1 to 0. Then store to the ENIU. This must be done for each ENIU that is to operate this way.

Control Data Format

The first 10 words of data consumed by the Ethernet NIU are control data. They determine the behavior of outputs if communications are lost, and can be used to clear faults.

In addition, if there are two controllers, the control data determines which of them will supply the I/O Station outputs.



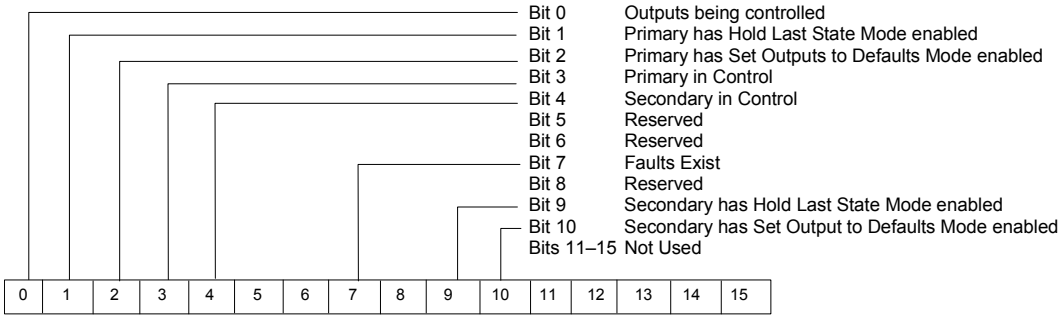
The application program in the controller(s) is responsible for correctly setting the content of this control data as described below. Unused words should be set to zero.

Enable Hold Last State Mode:*	Set this bit if outputs in the I/O Station should hold their last commanded state when communications are lost. For systems with two controllers, this bit should be the same in both the primary and secondary controller exchanges.
Enable Set Outputs to Default Mode: *	Set this bit if outputs in the I/O Station should go to their configured defaults when communications are lost. If this bit is set, bit 1 (Hold Last State) is ignored. For systems with two controllers, this bit should be the same in both the primary and secondary controller exchanges.
Switch to Primary Controller:	If the secondary controller is presently controlling the NIU and providing output data for the I/O Station, the primary controller must set this bit to regain control of the I/O Station. See “Switching Back to the Primary Controller” below for additional steps that are necessary to return to normal operation with the primary controller.
Switch to Secondary Controller:	If the primary controller is presently controlling the NIU and providing output data for in the I/O Station, it can switch control to the secondary by setting this bit. If this bit is set, bit 3 (Switch to Primary) should NOT be set. If the secondary controller is not present, the switch will not occur.
Clear Faults:	Setting this bit clears all faults in ALL Ethernet NIUs that receive the same exchange. In a system with two controllers, only the exchange from the currently-active controller is used to clear faults.
Word 2, Available to Application:	The application program in the controller(s) can optionally use word 2 as described later in this section.

* See the section on setting up the output defaults.

Status Data Format

The 20 bytes of status data sent by the Ethernet NIU provide the controller(s) with information about output and fault status in the format shown below. The application program in the controller(s) should continually monitor this status data from the ENIU.



Word 1: Status and Fault Data

O	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Word 2: Copy of Control Data Word 2 (Application-based) from Primary Controller

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

Word 3: Copy of Control Data Word 2 (Application-based) from Secondary Controller

Words 4 - 10: Reserved

Status Data Definitions

Outputs Being Controlled:	Set if the I/O Station outputs are being controlled from the application program, and are not defaulted or in Hold Last State mode. If this bit is set, bits 1 and 2 should NOT be set.
Controller has Hold Last State Mode Enabled:	The ENIU sets bits 1 and 9 to mirror the present Hold Last State control bit being received from the primary controller and the secondary controller
Controller has Set Outputs to Defaults Mode:	The ENIU sets bits 2 and 10 to mirror the present Outputs Default control bit being received from the primary controller and the secondary controller.
Primary in Control:	Set when the primary controller is presently controlling the NIU and providing output data for the I/O Station. If this bit is set, bit 4 (Secondary in control) should NOT be set.
Secondary in Control:	Set when the secondary controller is presently controlling the NIU and providing output data for in the I/O Station. If this bit is set, bit 3 (Primary in Control) should NOT be set.
Faults Exist:	Set when any fault exists in the Ethernet NIU.
Words 2 & 3, Copy of Optional Control Data	The ENIU mirrors the content of word 2 of the control data in these status words. If the ENIU is receiving outputs from the primary controller, status word 2 has content. If the ENIU is receiving outputs from the secondary controller, status word 3 has content.

Using the Control and Status Data

The application program in the controller(s) should monitor the Ethernet NIU status data, and use the control data to interact with the NIU.

Switching Control Back to the Primary Controller

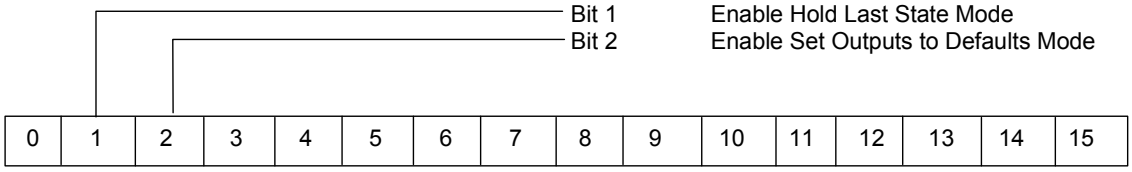
When the Ethernet NIU is using output data from the secondary controller, the application program in the primary controller should follow the steps below to regain control of the ENIU.

The switchover from secondary to primary controller will occur if bit 3 is set. It is recommended that the steps below be followed to synchronize the primary controller with the secondary controller before switching control to the primary.

1. Start up with bit 3 reset.
2. Synchronize the program state with data from the secondary controller.
3. Set output bit 3 (“Switch to Primary Controller”) of the data going to the ENIU.

Setting Up the Output Defaults

If the Ethernet NIU does not receive any communication with the controller(s) within the configured timeout period, it sets the outputs in the I/O Stations to specified states. These output states are determined by commands previously received in the output data control bits.



Word 1: Control Data

If control bit 1 is set to 1, the ENIU will hold the outputs at their last commanded states.

If control bit 2 is set to 1, the ENIU will set outputs to their individual default states (see below).

Bit 2 takes precedence; if both bits 1 and 2 are inadvertently set, the ENIU sets outputs to their default states.

If control bits 1 and 2 are both 0, outputs are set to 0.

When the Ethernet NIU has both primary and secondary controllers, output bits 1 and 2 should be set the same by both. If they are not the same, the Ethernet NIU will use the values it received from the last controller that provided outputs before communications were lost.

Specifying Individual Output Defaults

If the control outputs are set to have the outputs default instead of hold last state, ordinarily all outputs will default to zero. If that is suitable for the application, no further action is needed. However, for some applications taking outputs to a safe state requires setting them to one or forcing analog outputs to individually-specified values.

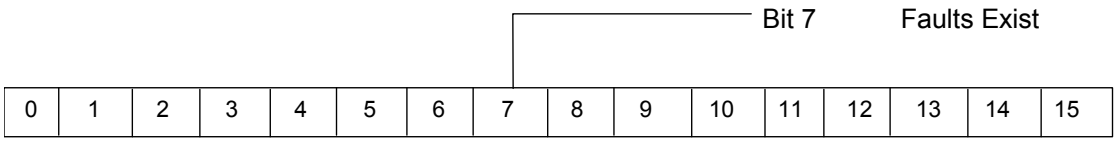
An optional procedure can be used to set up defaults for the Ethernet NIU. Use of this procedure is described in chapter 5.

Checking for Faults and Clearing Faults

The regular exchange of status and control data provides the controller with the ability to check for fault conditions and clear faults.

Checking for Faults

The application program in the controller(s) should monitor Ethernet NIU status bit 7 to check for faults.



Word 1: Status and Fault Data

O	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Word 2: Copy of Control Data Word 2 (Application-based) from Primary Controller

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

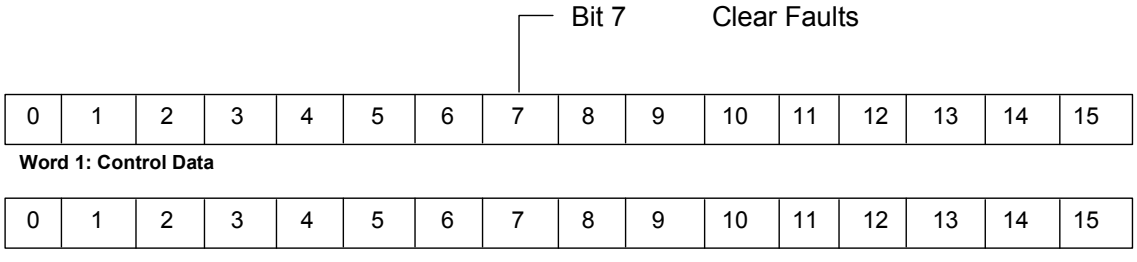
Word 3: Copy of Control Data Word 3 (Application-based) from Secondary Controller

Words 4 - 10: Reserved

If faults exist, they can be viewed using the programming software as described in chapter 6.

Clearing Faults

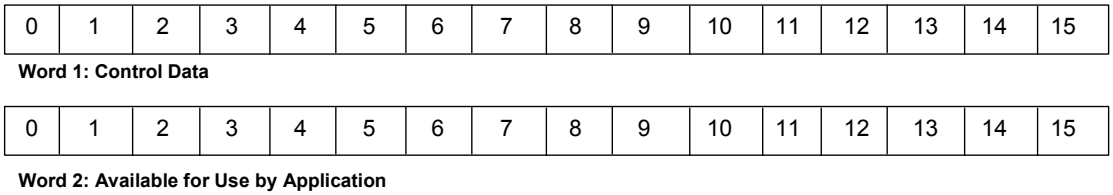
The controller can clear faults by setting bit 7 in the control data portion of its produced exchange. This will clear faults in ALL the Ethernet NIUs that receive the same exchange.



Words 3 – 10 should be set to zero

Using the Optional Application-Specific Command Word

The word 2 of the command data can be used by the controller for several purposes.



Words 3 – 10 should be set to zero

Setting Up a Heartbeat

For example, the controller could use a free-running counter as a heartbeat for the value of this word, then check the incoming Ethernet NIU status block to make sure the ENIU is still running. In redundant applications, each controller could check the other controller's heartbeat to determine whether the other controller is operating.

Sequencing Outputs

This word could also be used to sequence outputs. The controller would set the outputs to a particular state and set the sequence number in the command data. When the Ethernet NIU returns the same sequence number in its status data, the controller knows that the ENIU has received the outputs. The controller can then take the next step in the sequence.

Checking the Status of the Heartbeat / Sequence

The primary controller's heartbeat/sequence ID word is returned in the second word of the ENIU status block. The secondary controller's heartbeat/sequence ID word is returned in the third word of the ENIU status block.

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

Word 1: Status and Fault Data

O	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Word 2: Copy of Control Data Word 2 (Application-based) from Primary Controller

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

Word 3: Copy of Control Data Word 2 (Application-based) from Secondary Controller

Words 4 - 10: Reserved

Chapter 5 I/O Configuration

This chapter explains how an Ethernet NIU and the modules in an I/O Station can be configured. Configuration determines certain characteristics of module operation and also establishes the program references to be used by each module in the system.

- Configuring the Exchanges of the device that will control the Ethernet NIU.
  - Configuring a Controller's Produced Exchange
  - Configuring a Controller's Consumed Exchange
- Configuring the Ethernet NIU
  - Rx3i Ethernet NIU Ethernet Global Data Exchanges are pre-populated with most information when the ENIU target is created. The exchanges only need to be completed as described here.
    - Completing an Ethernet NIU's Produced Exchange Configuration
    - Completing an Ethernet NIU's Consumed Exchange Configuration
    - Completing the Ethernet NIU's Consumed Exchange from a Secondary Controller
- Setting Up Output Defaults
- Programmer Communications with the Ethernet NIU

Configuration of EGD Exchanges for Remote COMMREQ Calls is covered in Chapter 8. The Remote COMMREQ Call exchanges can be deleted if Remote COMMREQ Calls will not be used.

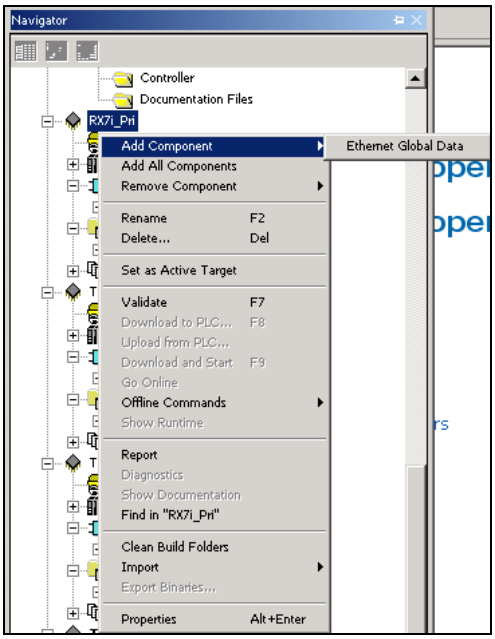
For the RX3i Ethernet NIU, when the Ethernet NIU target is created in Proficy Machine Edition, the Ethernet Global Data exchanges are automatically created but are incomplete. The IP Address, Local Producer ID, Destination IP Address, and data ranges for inputs to be sent to the controller must be entered.

Configuration Overview

Configuring a Controller to Work with the Ethernet NIU.

In addition to any other configuration required for the controller, two basic configuration steps are required to incorporate the Ethernet NIU and its I/O modules into the system:

1. The controller must be set up to enable Ethernet Global Data exchanges. Right-click on the target to add Ethernet Global Data.



   The maximum number of EGD exchanges (produced and consumed) that can be configured in a single controller depends on the controller type. For PACSystems RX7i, RX3i, and Series 90-70 PLCs, it is 255 exchanges. For the Series 90-30 CPU364 and 374, it is 128. Consult the documentation for the control system if you need more information.

2. The EGD exchanges need to be created and defined. Configuring the exchanges assigns I/O references in the controller's memory to the data in the exchange. During operation, the application program in the controller will handle these I/O references in the same way as the references used by local I/O modules. The individual modules in the Ethernet NIU's I/O Station are not explicitly included in the controller configuration.

If the system includes a secondary controller, its EGD exchanges must also be created and configured, and they must match the exchanges of the primary controller, with the exception of the Producer ID and Exchange ID.

Please refer to the controller documentation and the online help for the programmer software for specific configuration instructions.

Timing for Ethernet Global Data Exchanges

When an RX3i Ethernet NIU target is added in Proficy Machine Edition, three I/O-related Ethernet Global Data exchanges are included in the EGD exchanges component of the target configuration. Those exchanges are:

- Outputs_Pri_to_ENIU: Consume output data exchange from the primary controller
- Outputs_Sec_to_ENIU: Consume output data exchange from the secondary controller
- Inputs_from_ENIU_xx: Produce an Input data exchange back to the controller(s)

Each of these Ethernet Global Data Exchanges has a default produced period of 10 milliseconds and a consume update timeout of 32 milliseconds. These default values accommodate almost any I/O mix and corresponding exchange sizes for up to five Ethernet NIU I/O Stations, and allow optional Remote COMMREQ Call exchanges to one or more of the Ethernet NIUs and a single programmer of the Ethernet link used for the Ethernet I/O. These default settings are suitable for most applications.

If the maximum number of I/O Stations and the exchange sizes listed below are not exceeded, the produced periods and consumer update timeouts can be modified as shown.

Most applications require the ability to remove power from an I/O station or CPU (for maintenance or power outage etc.), without disturbing the communications on the rest of the I/O network. For this reason, good-quality recommended Ethernet network switches must be used with multiple I/O Stations (for performance reasons, Ethernet hubs are NOT recommended for use with the RX3i Ethernet NIU). When smaller numbers of I/O stations (10 or less) are used *and* there is no requirement for the I/O network to function under the condition of a powered-down I/O station, then the 2-port switch that is built in to the Ethernet module may be used to daisy-chain from one I/O station to the next without the need for an external network switch.

Please contact your local GE Fanuc application engineer for more information regarding recommended network switches.

Suggested Producer Period and Consumer Update Timeout Settings for Ethernet NIU I/O Station EGD Exchanges*:

ENIU I/O Stations	*Input Exchanges per I/O Station*	*Output Exchanges from Controller(s)*	*Suggested Produced Period (ms)*	*Suggested Consumer Update Timeout (ms)*
Up to 5	1 (220 bytes)	1 (1300 bytes)	6	18
Up to 10	1 (220 bytes)	1 (1300 bytes)	8	24
Up to 20	1 (220 bytes)	2 (1300 bytes)	14	42
Up to 42	1 (220 bytes)	4 (1300 bytes)	25	75

* The table above is based on RX3i Ethernet NIU I/O Stations that consume a 1300-byte "Output Data" Ethernet Global Data exchange and produce a 200-byte "Input Data" EGD

exchange. All I/O data EGD exchanges between Ethernet NIU(s) and Controller(s) in a system should be set to the same producer period and consumer update timeout. For the Ethernet NIU I/O drops requiring much slower update rates, different produced period and consumer update timeout values may be used for the EGD exchanges for that I/O Station.

The suggested produced period and consumer update timeout values are based on RX3i Ethernet NIU I/O Stations with a maximum of 256 I/O consisting of:

160 or 60% Inputs (96 analog inputs and 64 digital inputs)

96 or 40% Outputs (48 analog outputs and 48 digital outputs)

The number and frequency of production of the Ethernet Global Data exchanges has the greatest effect on performance. The size of the exchanges also impacts performance.

The produced period parameter determines how frequently an Ethernet device attempts to send (or produce) the output data. To account for latencies in the interface and any other network devices, as well as the possibility that a packet could be dropped on the network, the guideline for setting the consumer update timeout parameter is to multiply the produced period by 3. If the update timeout period of a consumer (an I/O Station's Ethernet Transmitter Module in the case of an output exchange; or the Controller in the case of an input exchange) is exceeded before a new exchange arrives, the timeout status informs the consumer that new data has not arrived within the expected period. The Ethernet NIU uses this timeout status as an indication that it should default (or zero, or hold last state depending on configuration) the output data. Timeout status is also available to the application program in the Ethernet NIU or controller. Under normal operating conditions, exchanges are received within the consumer update times listed in the previous table. For example, an output exchange in a system (configured as described in the chart above) with five drops will be received in 18 milliseconds or less. This can be considered the one-way (input or output) update rate for that configuration. As normal, for the overall system I/O response time, the controller(s) scan time also needs to be taken into consideration.

For applications that require performance or update rates faster than what is listed in the table above, consider breaking your single I/O network into two separate I/O networks. This will require an additional Ethernet module in the controller(s).

Example: A single I/O network with 18 I/O stations would typically have EGD exchange produce periods of 14 milliseconds and EGD exchange consumer timeout values of 42 milliseconds. The performance of the I/O system can be drastically improved by splitting the single 18 I/O station network into two 9 I/O station networks. Each 9 I/O station network then can be configured for produce periods of 8 milliseconds and EGD exchange consumer timeout values of 24 milliseconds.

Stale Data EGD Status

A stale data status is a non-fatal status. Although an exchange is producing at the correct period, the data in the exchange can be old (stale) if the controller has not yet updated it. If the produced period for an exchange is less than the controller's scan time, the Ethernet device can send the same data in more than one Ethernet Global Data exchange. If the controller has not updated the EGD data before the exchange produced period expires, the Ethernet device sends the same data again.

Stale data status can also occur from an Ethernet NIU if the ENIU uses local logic. Local logic can increase scan time so it is close to or longer than the Ethernet Global Data input data exchange's producer period. Each consumed EGD exchange status word received by the consumer of the exchange provides the indication of stale data. The stale data status is available for use by the application. The occurrence of stale data can also be determined by using the Ethernet Transmitter Module's Station Manager command – "tally G". A count of stale data occurrences for the Ethernet Transmitter Module's produced EGD exchanges is displayed along with other tally G data.

Configuring an Ethernet NIU

Configuring an Ethernet NIU includes:

- Adding I/O modules to the I/O Station hardware configuration. This is done in the same way as adding I/O modules to the hardware configuration of a PLC. This part of the configuration assigns I/O references in the ENIU to each module and sets up any other configurable module parameters.
- The Ethernet Global Data for the RX3i Ethernet NIU is automatically enabled. Default Exchanges are created with correct parameters, but they are incomplete. The EGD exchanges must be completed
- The following EGD Exchanges automatically created in the ENIU and are used by the ENIU (the xx can be replaced with the ENIU number if the system has more than one ENIU):
  - One EGD produced exchange to send inputs to the controller(s), which has the name (Inputs_from_ENIU_xx)
  - One EGD consumed exchange from the primary or only controller to receive outputs from the controller which has the name (Outputs_Pri_to_ENIU).
  - An optional EGD consumed exchange from a secondary controller to receive outputs from the secondary controller which has the name (Outputs_Sec_to_ENIU).
  - One EGD produced exchange to send RCC results back to the controllers(s) which has the name (RCC_Response_from_ENIU_xx).
  - One EGD consumed exchange to receive RCC commands from the primary or only controller. that has the name (RCC_Pri_Request_to_ENIU_xx).
  - An optional EGD consumed exchange to receive RCC commands from the secondary controller that has the name (RCC_Sec_Request_to_ENIU_xx).

Configuring the Ethernet NIU Parameters

Before you can use the Ethernet NIU on the network, its configuration must be set up as summarized below. Please refer to the programming software online help if you need detailed instructions for using the software.

The RX3i Ethernet NIU is a target type in Machine Edition. When you create a project or add a target to an existing project, select GE Fanuc Remote I/O, PACSystems RX3i Ethernet. When the RX3i ENIU target is created, the Ethernet Global Data exchanges are pre-populated and just need to be completed as described in this chapter.

Completing the Ethernet NIU Parameter Configuration

Configuration parameters for the Ethernet NIU are defined on its Settings tab.

Adapter Name: This is set to 0.4 (the I/O Station rack and slot where the Ethernet Transmitter Module is located).

Status Address: The Status Reference Type (location of the LAN interface status) is set to %R4001, Length = 5. It cannot be changed. If a second Ethernet Transmitter Module is used, its LAN interface status is set to %R4006 and cannot be changed.

IP Address, Subnet Mask, and Gateway IP Address: These values should be assigned by the person in charge of your network (the network administrator). If these parameters are incorrect; the Ethernet Transmitter Module may be unable to communicate on the network and/or network operation may be corrupted. It is especially important that each node on the network is assigned a *unique* IP address. For a simple isolated network with no gateways, you can use the following range of values for the assignment of local IP addresses (these are the same IP addresses used in the Quick Start Example in Appendix A):

10.0.0.1	First NIU
10.0.0.2	Second NIU
10.0.0.3	Third NIU
.	.
10.0.0.101	Primary Controller
10.0.0.102	Secondary Controller
	.
	.
10.0.0.255	PLC Programmer TCP or host

Also, in this case set the subnet mask and gateway IP address to 0.0.0.0

Note: If the isolated network is ever connected to another network, IP addresses 10.0.0.1 through 10.0.0.255 must not be used. The subnet mask and gateway IP address must be assigned by the network administrator. The IP addresses must be compatible with the connected network.

Network Time Sync: This is set to None (for no network time synchronization)

Configuring EGD Exchanges in the Controller

These are the basic steps to setting up communications between the Ethernet NIU and the controller that will operate the Ethernet NIU:

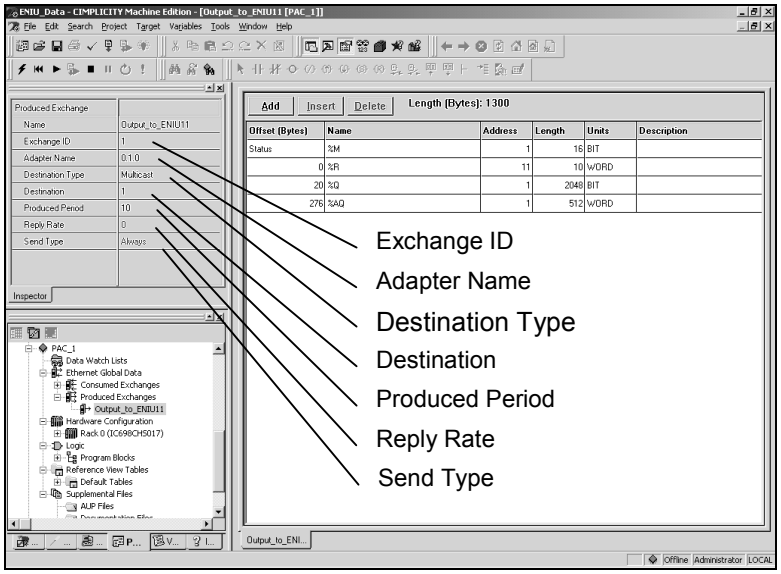
1. Configuring the controller for at least one Ethernet Global Data produced exchange to send outputs and control data to one or more Ethernet NIUs.
   - The simplest method is to configure one Ethernet Global Data produced exchange that contains all the discrete outputs and analog outputs for one or more Ethernet NIUs in the system. This is the recommended method that will produce the best performance for most systems.
   - The controller could also produce multiple Ethernet Global Data exchanges, each of which would send outputs to only some of its Ethernet NIUs. That must be done if the overall amount of I/O Station discrete output data is more than 2048 bits, or if the analog output data is more than 512 words.
   - Alternatively, the controller could produce one or more EGD messages, each containing only a portion of its output data. There is no need to send all of the output data if the Ethernet NIUs don't need it all. Producing multiple EGD exchanges when one exchange could be used **WILL** reduce the performance of the system.
2. Configuring the controller for at least one Ethernet Global Data consumed exchange to receive inputs and status data from each Ethernet NIU. In systems where there are multiple Ethernet NIUs, the controller must be configured to receive a consumed exchange from each ENIU.

These configuration parameters are part of the overall CPU (controller) configuration.

If the system includes both a primary controller and a secondary (backup) controller, both will require this configuration.

Configuring a Controller's Produced Exchange "Outputs_Pri_to_ENIU"

To set up a Produced Ethernet Global Data, configure the parameters and ranges as described below. The controller's produced exchange must match the Ethernet NIU's consumed exchange. Note: the description below is for a PACSystem controller.



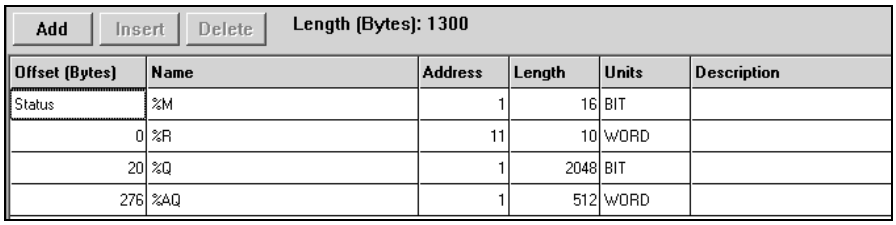
1. The Exchange name should be the same as the consumed exchange in the Ethernet NIU, this helps to track and debug what is configured. The ENIU is automatically set to Outputs_Pri_to_ENIU.
2. The Exchange ID should be 1 (default in ENIU) if the controller will produce only one exchange. If the controller sends more than one exchange (to other devices), each must have a different Exchange ID.
3. Change the Destination Type to Multicast (Default for ENIU). (For Series 90 Controllers this is Group).
4. Any Destination (1 to 32) can be used if it is consistent: The ENIU is defaulted to "1".

Parameter	IP Address
Group 1	224.0.7.1
Group 2	224.0.7.2
:	:
Group 32	224.0.7.32

5. The Produced Period defaults to 200 milliseconds. Change it to match the ENIU. The ENIU is defaulted for operation with a 10-millisecond produced period from the controller. 10 milliseconds is usually a good setting. Do not set it less than 6 milliseconds, as settings less than 6 will cause performance to decrease. For very large configurations or Ethernet segments with a lot of Ethernet traffic, a Produced Period greater than 10 milliseconds may be required. This parameter sets the network production time, and is the main configuration factor in I/O response time.

Setting Up the Data Ranges for a Controller's Produced Exchange

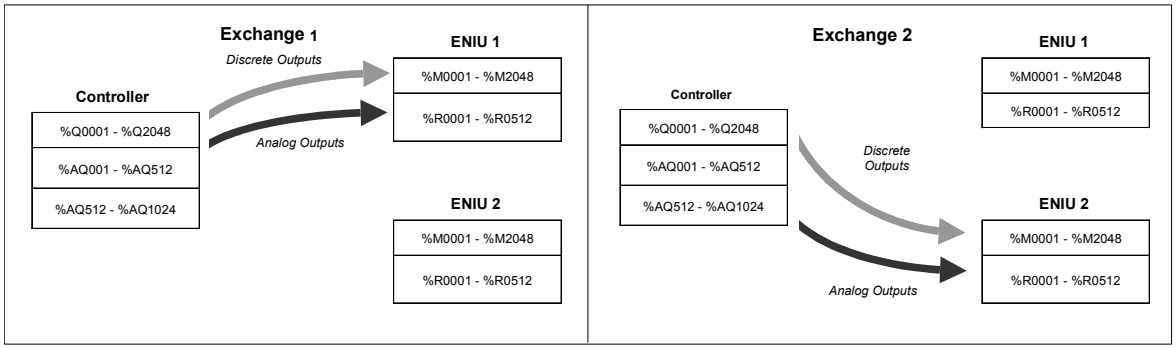
After establishing the parameters for a produced exchange, set up the controller memory ranges for the exchange. An exchange can include up to 100 ranges and/or variables. Click the Add button to add a range.



Offset (Bytes)	Name	Address	Length	Units	Description
Status	%M	1	16	BIT	
0	%R	11	10	WORD	
20	%Q	1	2048	BIT	
276	%AQ	1	512	WORD	

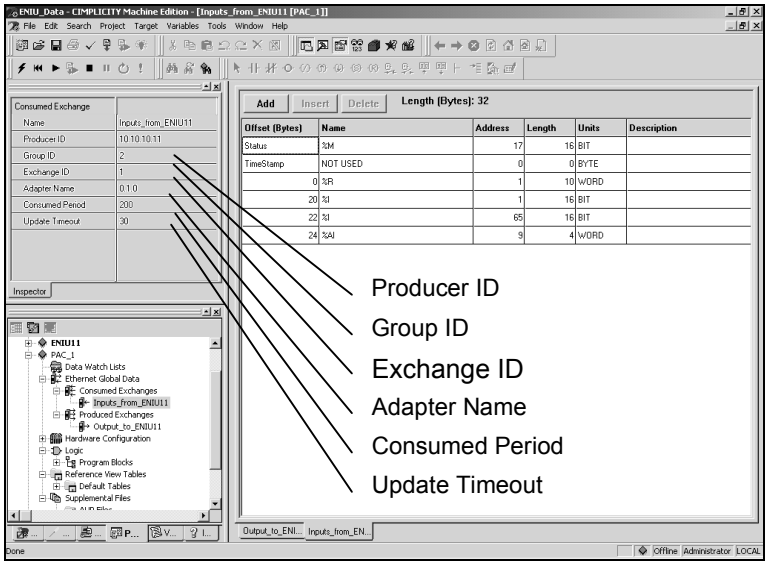
1. The first range is for the status data for the Ethernet Global Data exchange. In the controller, this data can be assigned to any appropriate available reference range. In the example above, it is assigned to 16 bits starting at %M0001. This range is for local status in the controller. It is NOT transmitted to the ENIU.
2. Add a 10-word data range for the control data that will be sent in the exchange. In the example above, the starting reference for this data is %R0011.
3. It is recommended that 2048 discrete Outputs %Q be sent to the ENIU(s). The ENIU is defaulted to 2048 discrete outputs. Each ENIU that receives the exchange will use only the discrete outputs it needs and ignore the rest. If the total number of %Q used in the system is much less than 2048 a smaller range can be configured, but must be changed in the controller(s) and in each ENIU.
4. It is recommended that the first 512 analog outputs %AQ be sent to the ENIU(s). The ENIU is defaulted to 512 analog outputs. Each ENIU will use only the analog outputs it needs and ignore the rest. If the total number of %AQ used in the system is much less than 512 a smaller range can be configured. but must be changed in the controller(s) and in each ENIU.

In a system with multiple Ethernet NIUs, if the total amount of analog output data exceeds the 512 word limit of one ENIU, the controller must produce multiple exchanges to send all the analog outputs to multiple ENIUs. Each exchange can have the same discrete outputs (%Q), but different analog outputs (%AQ). When configuring this type of produced exchange for the controller, enter the actual controller references to be sent. When an Ethernet NIU receives the exchange, it will store the analog outputs beginning at the start of its analog output table. Therefore, in a system with more than 512 analog outputs, some of the analog outputs will have different reference addresses in the ENIU than in the controller as illustrated below.



Configuring a Controller's Consumed Exchange

Set up a Consumed EGD exchange in the controller for each Ethernet NIU in the system. The controller's consumed exchange must match the ENIU produced exchange. Even if an I/O Station has only output modules, its controller(s) must be configured for a consumed exchange to receive the ENIU's status data. Note: the description below is for a PACSystem controller.



For each exchange:

1. The Exchange name should be the same as the produced exchange in the ENIU, this helps to track and debug what is configured. The ENIU Exchange name is automatically set to "Inputs_from ENIU_xx".
2. Producer ID should be the IP address of the Ethernet NIU that produced the exchange. In this example, it is 10.10.10.11.
3. Set the Group ID to 2 this is the ENIU default. For systems with Multiple ENIUs, all consumed exchanges from Ethernet NIUs in the controller are set for Group 2. Do not forget to set the Group ID. The default is 0, which means do not use a Group but use just the producer ID.
4. The Exchange ID should be 1 because each ENIU produces only 1 exchange. This is the ENIU default.
5. The Adapter Name of the CPU was configured previously in the CPU configuration window. If multiple Ethernet interfaces are in the controller, select the Ethernet interface which receives the EGD exchange.
6. The Update Timeout parameter is related to produced period of the ENIU, which should be set to approximately 3 times the Produced Period of the ENIU. The ENIU defaults are set for10 millisecond production. The recommended setting for Update Timeout is 32 milliseconds. The Update Timeout defaults to "0" which means don't enforce a timeout. This must be set to detect loss of communication. If the CPU does not receive an

exchange from the Ethernet NIU within this time period, it will indicate a refresh error in the exchange status word. This parameter can be adjusted for best performance.

Setting Up the Data Ranges for a Controller's Consumed Exchange

After establishing the parameters for a consumed exchange, set up the data ranges in controller memory for the exchange. Click the Add Range button to add a range.

Add | Insert | Delete | Length (Bytes): 32

Offset (Bytes)	Name	Address	Length	Units	Description
Status	%M	17	16	BIT	
TimeStamp	NOT USED	0	0	BYTE	
0	%R	1	10	WORD	
20	%I	1	16	BIT	
22	%I	65	16	BIT	
24	%AI	9	4	WORD	

1. Set up a 16-bit reference in controller memory (%M00017 above) to store the status of the Ethernet Global Data exchange. It can be any CPU memory location. This status is local to the controller and is NOT data received from the ENIU.
2. Add a range to store the 10 words of status data that will be sent by the Ethernet NIU. The format of this data is shown in chapter 4.
3. Add one or more ranges for any discrete input data (%I) that will be received in the exchange. These discrete inputs must not duplicate or overlap any discrete inputs in other exchanges or in the controller, as explained in chapter 3. The ranges entered will correspond to discrete inputs that are configured in the hardware configuration of the ENIU. In most applications, the controller input references should match the references configured in the Ethernet NIU. It is not necessary to add a separate range for each input module in the I/O Station. Contiguous inputs can be grouped into ranges that include data from multiple modules. If there is a gap in the reference assignments, separate ranges must be configured as shown above.
4. Add one of more ranges for any analog input data (%AI) that will be included in the exchange. These analog inputs must not duplicate or overlap any analog inputs in other exchanges or in the controller. The ranges entered will correspond to analog inputs that are configured in the hardware configuration of the ENIU. Like the discrete input assignments, analog inputs can be assigned in ranges that include multiple modules as long as care is taken to assure that the ranges match the module inputs.

Configuring the Ethernet NIU

The Ethernet NIU and I/O Station must be configured using Proficy Machine Edition Logic Developer version 5.50 SIM1 or later.

Configuring ENIU Network Parameters

ENIU Network configuration establishes the basic operating characteristics of the Network Interface Unit. If the ENIU will be communicating with devices on other networks, the parameters in the following table must be set appropriately. These values should be assigned by the person in charge of your network (the network administrator).

Feature	Description	Config. Default	Choices
IP Address	The IP Address is the unique address of the Ethernet interface as a node on the network.	0.0.0.0	A valid Class A, B, or C address
Subnet Mask	Subnet mask of the ENIU used to identify the section of the overall network the ENIU is on.	0.0.0.0	A valid dotted-notation mask.
Gateway IP Address	IP address of the default gateway (router) device to be used when the ENIU is unable to locate the desired remote device on the local sub-network.	0.0.0.0	A valid Class A, B, or C address in the same subnet as the ENIU.
ENET Status	First Ethernet Interface	%R4001	Fixed at %R4001.
ENET Status	Second (optional) Ethernet Interface	%R4006	Fixed at %R4006.

Configuring the Ethernet NIU's Produced Exchange

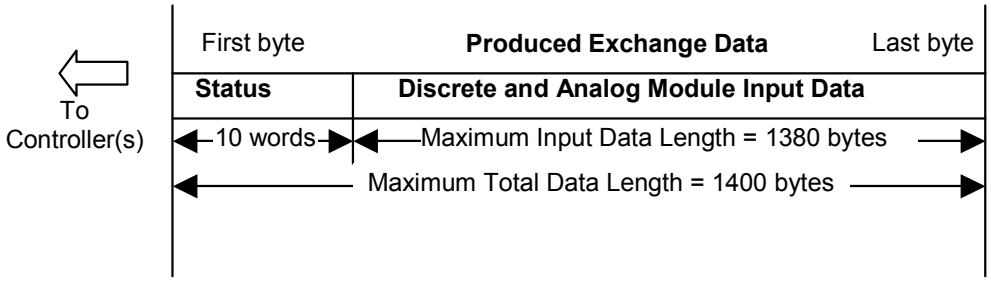
The RX3i Ethernet NIU's produced exchange is automatically pre-populated with settings that are required. The following settings must be added to the configuation for the produced exchange.

Setting Up the References for the ENIU's Produced Exchange

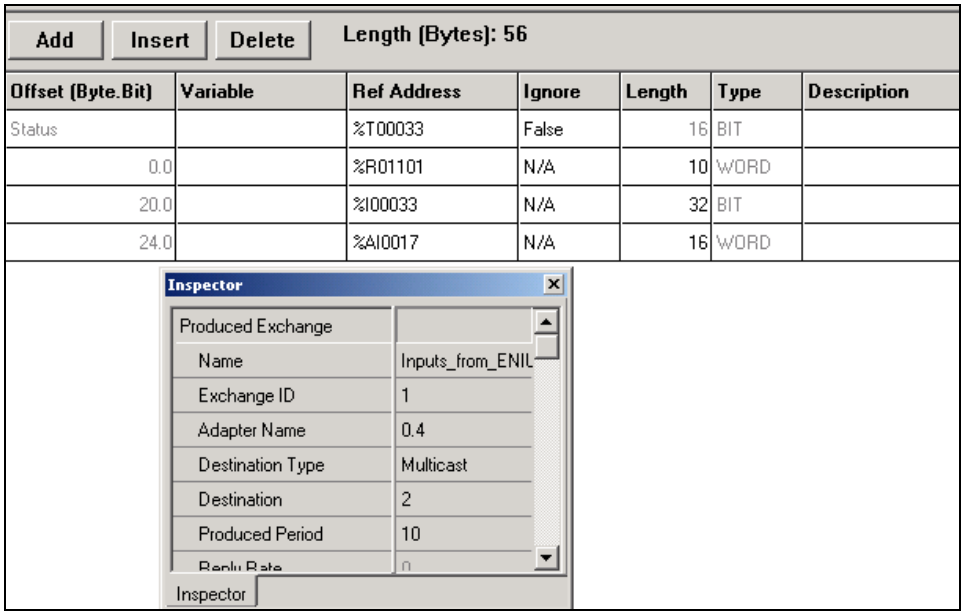
1. The Status location is automatically set up and must not be changed. The produced EGD exchange status must be assigned 16 bits of %T memory, from %T0033 to %T0048 as shown (see chapter 4 for more information about required references in the Ethernet NIU).
2. The 10 words of the I/O Station status data that the Ethernet NIU will send to the controller(s) is set up automatically, and must not be changed. The required range for this data is %R1101 to %R1110.
3. Add ranges as needed to configure the I/O Station discrete and analog inputs. These inputs must not duplicate or overlap any inputs in other NIUs or in the local CPU, as explained earlier in this chapter. The ranges entered will correspond to inputs %I and %AI which are configured in the hardware configuration of the Ethernet NIU. In most applications, the controller input references should match the references configured in the Ethernet NIU. It is not necessary to add a separate range for each input module in the I/O Station. Contiguous inputs can be grouped into ranges that include data from multiple modules. If there is a gap in the reference assignments, separate ranges must be configured.

The parameters for the produced Ethernet Global Data exchange for the inputs the Ethernet NIU will send to the controller(s) are set up automatically. If the parameters of the produced EGD exchange are changed, corresponding changes must also be made in the controller(s).

The produced EGD exchange includes all the discrete and analog inputs in the I/O Station. Even if the I/O Station has no input modules, it still sends an input message to report its status data.



An example configuration screen for a produced EGD exchange is shown below:



Setting Up the Parameters of the ENIU's Produced Exchange

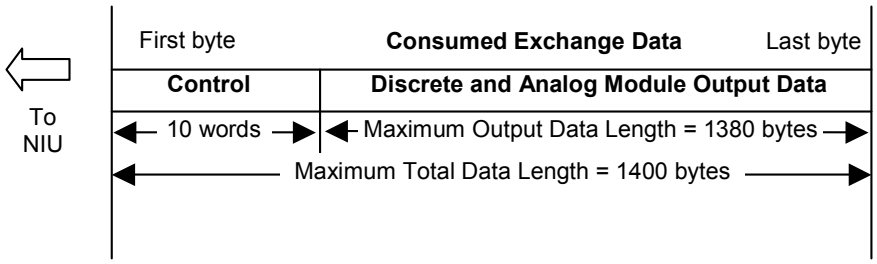
1. Enter the Exchange number. "1" is recommended. This is pre-populated.
2. The Cons Type should be "Group ID". This is pre-populated.
3. Group ID should be 2. This is pre-populated.
4. The Produced Period should typically be 10 milliseconds The default is 10 milliseconds. For small exchanges this can be made shorter, but it should not be less than 6 milliseconds. For very large exchanges, the produced period may need to be longer. This parameter should be tuned for best performance.

Configuring the Ethernet NIU's Consumed Exchange

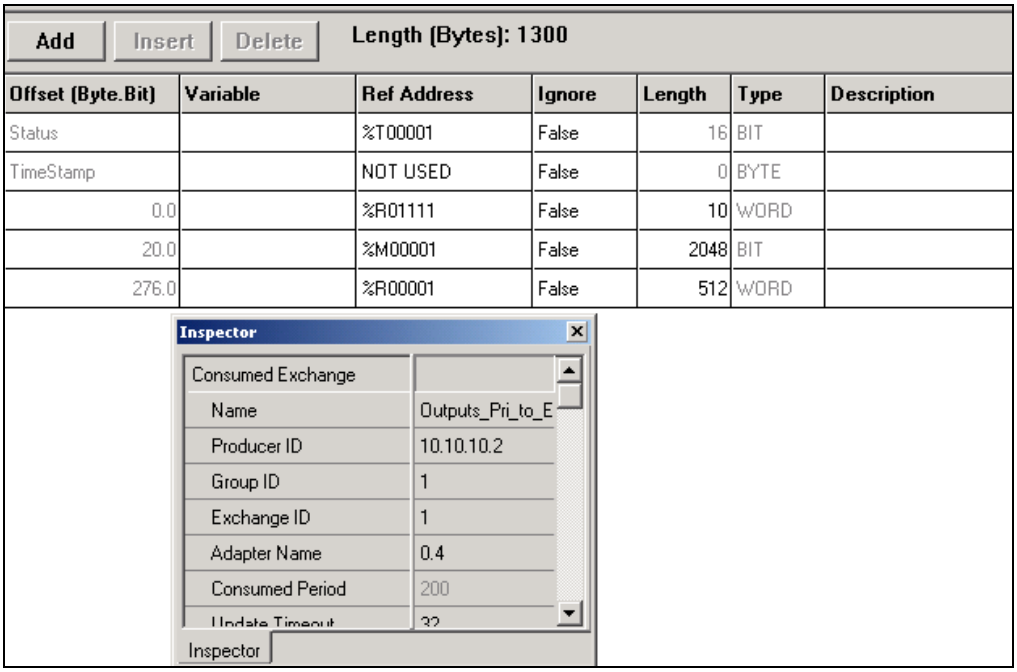
The exchange "Outputs_Pri _to_ENIU" is created when the Ethernet NIU is created in Machine Edition. Most of the parameters for the exchange are pre-populated.

Set up an Ethernet Global Data consumed exchange for the output data the Ethernet NIU will receive from the controller, or from the primary controller in a two-controller system. In a two-controller system, another consumed exchange must be configured in the Ethernet NIU for the secondary controller.

Even if the I/O Station does not have any output modules, it must consume an exchange from its controller(s) containing the control outputs.



An example configuration screen for the consumed EGD exchange is shown below:



Setting Up the Parameters of the Consumed Exchange

On the Consumed Exchanges tab:

1. The Exchange number must be “1”). This is pre-populated
2. The Adapter Name is 0.4 (this represents slot 0, rack 4).). This is pre-populated
3. Producer ID is the IP address of the primary controller. In this example, it is: 10.10.10.2 **This field MUST be entered.**
4. The Group ID should be “1”.). This is pre-populated.
5. The consumed period is not used and can be left at 200.
6. The update timeout for this example is 32 milliseconds.). This is pre-populated. This parameter should be tuned for best performance. It should be 3 times greater than the Produced Period of the controller. If the system has a very large configuration or if the controller has a high volume of Ethernet traffic, the update timeout may need to be increased. If occasional timeouts are occurring on the Ethernet NIU communications, increase the update timeout to 5 times the produced period of the controller.

Setting Up the Ranges for the ENIU's Consumed Exchange

The I/O references portion of the configuration screen lists the assigned references and gives the offset of each from the start of the Ethernet NIU's consumed data exchange.

Offset	Reference	Low Point	Hi Point	Description
Status	%T	1	16	Status: This is where the PLC is to pu
Time Stamp	NOT USED			Time Stamp: Optional place for the PL
0.0	%R	1111	1120	
20.0	%M	1	2048	
276.0	%R	1	512	

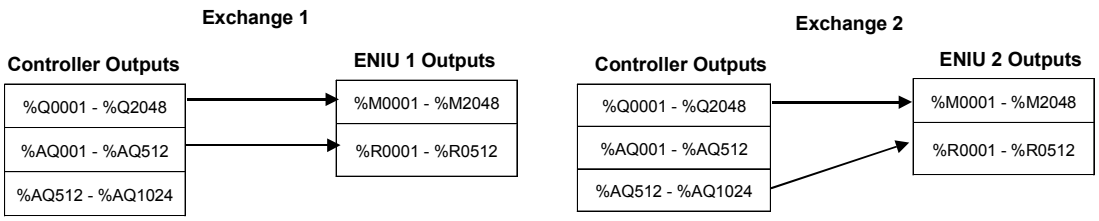
1. The Status reference is the location in the Ethernet NIU's internal memory where the status of the Ethernet Global Data exchange will be stored. This pre-populated field MUST assign 16 bits of %T memory, from %T0001 to %T0016. The Ethernet NIU firmware requires these specific addresses to operate.

2. The pre-populated range of 10 words will contain command information from the controller, as explained in chapter 4. This range MUST be assigned references %R1111 to %R1120.

3. The default output range shown above will receive 2048 discrete outputs and puts them in %M1 through %M2048. If the controller will send less output data, this can be edited to match the actual output range. These discrete outputs must go into the %M reference table.

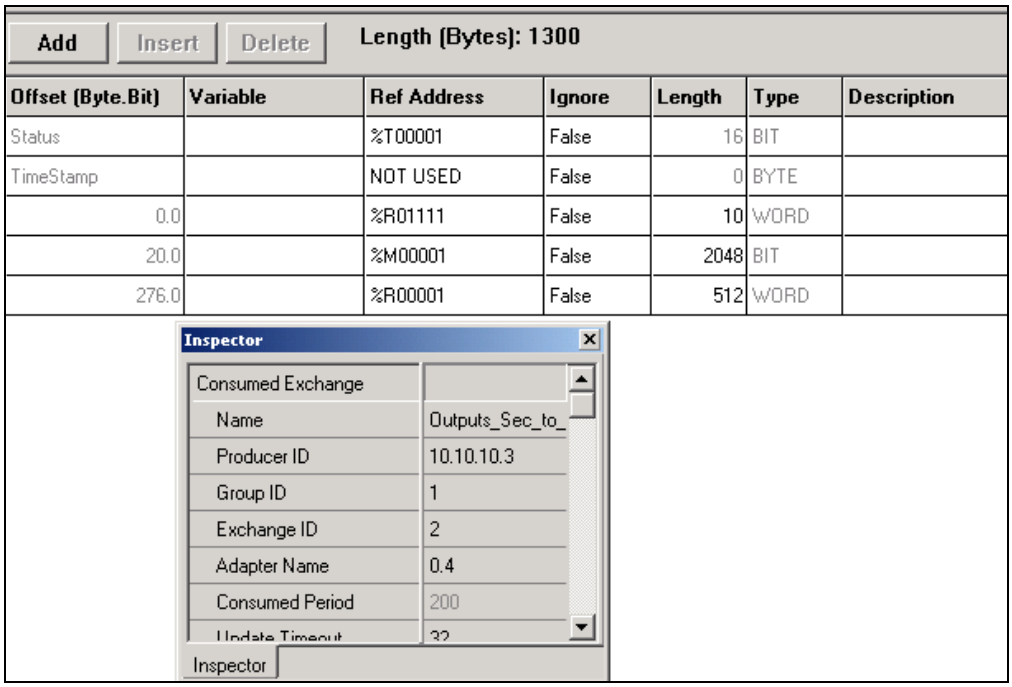
4. By default, the Ethernet NIU's consumed exchange is set up to receive 512 analog outputs and puts them in %R1 through %R512. If the controller will not send 512 analog outputs, edit this range to match the analog outputs sent by the controller. These analog outputs must go into the %R reference table.

   The Ethernet NIU can accommodate up to 512 analog outputs. If the control system includes multiple Ethernet NIUs and a total of more than 512 analog outputs, the controller must use separate (and separately-configured) exchanges to send the analog outputs. Therefore, some of the ENIUs must be configured with low and high points in %AQ memory that do not match the reference offsets used for the controller. Each exchange will send the same range of discrete outputs to all Ethernet NIUs as shown below.



Configuring the Ethernet NIU's Consumed Exchange from a Secondary Controller

In a two-controller system, a consumed exchange must also be configured from the secondary controller. The exchange "Outputs_Sec _to_ENIU" is created when the Ethernet NIU is created in Machine Edition. Most fields are pre-populated. The parameters and ranges for this exchange must correspond to the configuration of the consumed exchange from the primary controller, with the exceptions described below:



Setting Up the Parameters of the Consumed Exchange

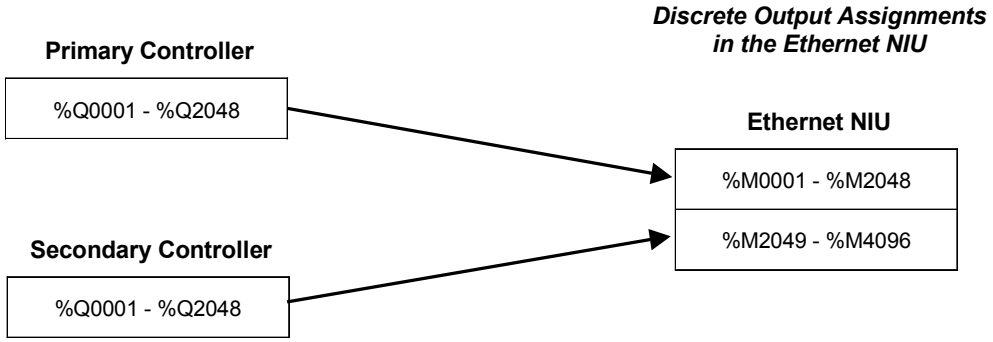
On the Consumed Exchanges tab:

1. The Exchange number MUST be "1".
2. The Adapter Name defaults to 0.4 (slot 0, rack 4), which is the required location of the Ethernet Transmitter Module in the I/O Station. However, exchanges from the secondary controller could go through another Ethernet Transmitter Module in the I/O Station. In that case, the adapter name must be changed to the actual position of that module.
3. Producer ID is the IP address of the secondary controller. In this example, it is: 10.10.10.3 **This must be entered**
4. The Group ID should be left at the default setting of "1".
5. The consumed period field is not used and can be ignored.
6. The update timeout defaults to 32 milliseconds. This parameter should be tuned for best performance. It should be 3 times greater than the produced period of the controller.

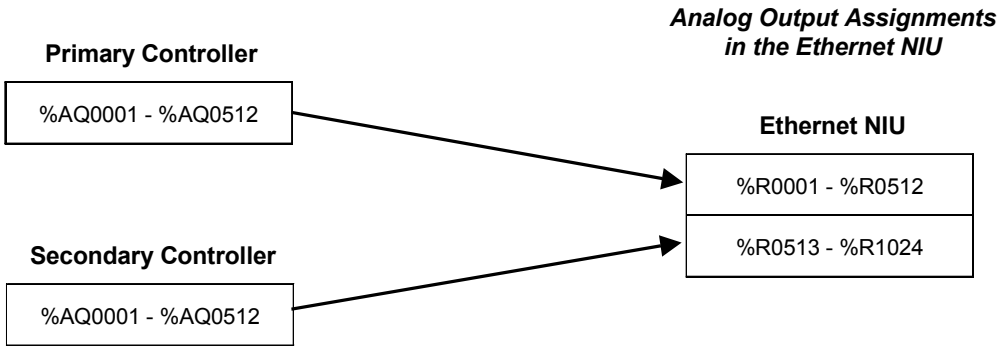
Setting Up the Ranges for the ENIU's Consumed Exchange

The I/O references portion of the configuration screen lists the assigned references, and gives the offset of each from the start of the Ethernet NIU's consumed data exchange.

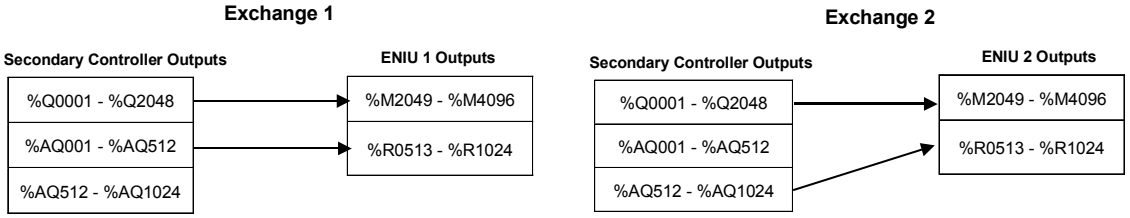
1. Status is the location in the Ethernet NIU's internal memory where the status of the Ethernet Global Data exchange will be stored. This MUST be assigned to the default of 16 bits of %T memory, from %T0017 to %T0032. The Ethernet NIU firmware requires these specific addresses to operate.
2. The range of 10 words will be used for command information from the secondary controller, as explained in chapter 4. This range MUST be assigned to the default references %R1121 to %R1130.
3. Default references are supplied for the discrete outputs for the I/O Station. The Ethernet NIU stores discrete outputs from the primary controller in %Q memory, but it stores discrete outputs from the secondary controller in %M memory, as shown below.



4. Default I/O ranges are provided for the analog outputs to the I/O Station The Ethernet NIU stores analog outputs from the primary controller in %AQ memory, but it stores analog outputs from the secondary controller in %R memory as shown below.



As mentioned earlier, the Ethernet NIU can accommodate up to 512 analog outputs. If the control system includes multiple Ethernet NIUs that have a total of more than 512 analog outputs, the controller must use separate (and separately configured) exchanges to send the analog outputs. When configuring the exchange from the secondary controller, some of the ENIUs must be configured with low and high points in %R memory that do not match the reference offsets used for the controller. Each exchange will send the same range of discrete outputs to all Ethernet NIUs as shown below.



Setting Up Output Defaults

This section explains how to establish output defaults for applications where defaults are needed:

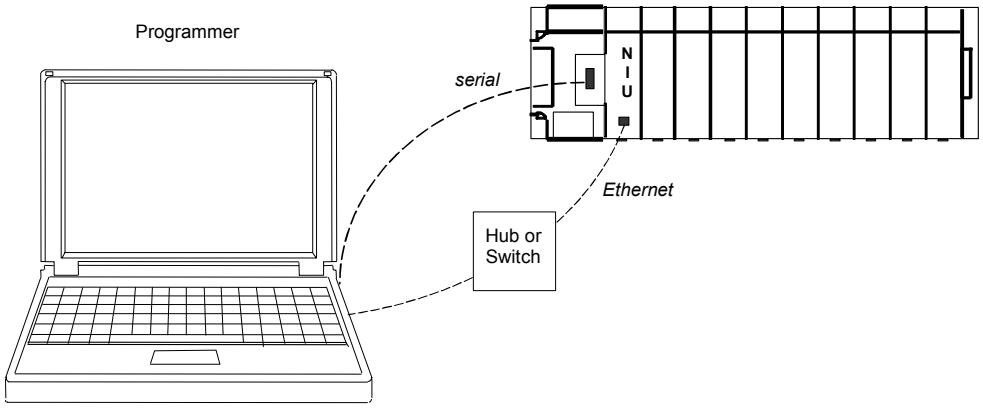
1. In Machine Edition, select the Variable Table and locate the Ethernet NIU variables. The variables are in a table of the form <Devicename><variable name>
2. In the Ethernet NIU section, locate the particular output (Qxxxx or AQxxxx) that is to be given a default value. If there is no variable with the current reference address, create a new variable and give it the desired address. A range of new variables with sequential addresses can be generated using the Duplicate command available by right-click.
3. When creating output variables (Qxxxx), set the Retentive property to True or the default value will not be stored properly.
4. Do not execute the command to delete unused variable as this will delete your added variables and initial values.
5. In the properties of the selected variable, change the Initial Value to the desired default value.
6. Download to the Ethernet NIU. The initial values will be downloaded and also stored to flash. Default values are loaded into a holding buffer from flash when the Ethernet NIU starts up.

Note: For systems with more than 2048 discrete outputs, discrete outputs 1-2048 in the Ethernet NIU are controller by higher discrete outputs in the controller. The range in the produced Ethernet Global Data exchange in the controller is set to the higher outputs and the range in the consumed exchange in the Ethernet NIU is set to 1-2048.

For systems with more than 512 analog outputs, analog outputs 1-512 in the Ethernet NIU are controller by higher analog outputs in the controller. The range in the produced EGD exchange in the Controller is set to the higher analog outputs and the range in the consumed exchange in the Ethernet NIU is set to 1-512. In setting the default values in the ENIU this address mapping must be accounted for.

Programmer Communications with the Ethernet NIU

After completing the configuration, it is stored from the programmer to the ENIU. A serial connection can be used to store the initial configuration to the ENIU.



After establishing the IP Address of the ENIU in the initial configuration, an Ethernet connection can be used for subsequent communications between the programmer and the ENIU.

For serial communications, the computer can be connected to the 9 Pin RS232 port or the 15-pin RS-485 compatible serial port on the RX3i ENIU.

After completing the configuration as described on the following pages, the programmer can be used to:

- *Store* the configuration to the Ethernet NIU.
- *Load* a previously-stored configuration from the Ethernet NIU back to the programmer.
- Compare (*Verify)* a configuration file in the programmer with a configuration that was previously stored to the Ethernet NIU.
- *Clear* a previously-stored configuration from the ENIU. After a *Clear* function, the ENIU will remain at the same IP address with the same subnet mask and gateway IP address.

Chapter 6 I/O Diagnostics

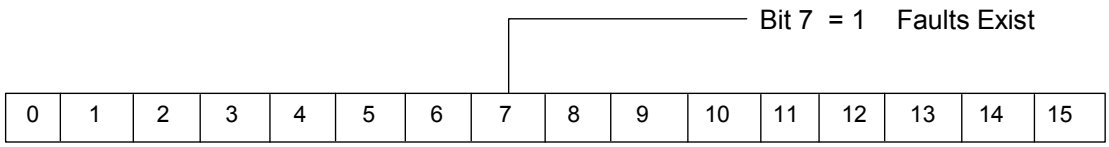
This chapter describes:

- Using the Status and Control Data for Fault Monitoring
- Viewing the Fault Tables in the Ethernet NIU
- Using the Station Manager
  - Checking the IP Address of the Ethernet NIU
  - Testing communications on the network
  - Viewing the Exception Log
  - Checking the Network Connection
- Stale Ethernet Global Data Status
- What to do if you can’t solve the problem

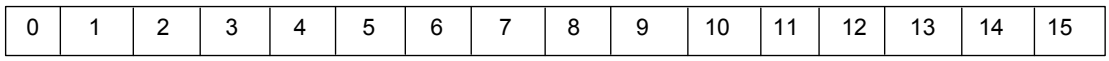
Using the Status and Control Data for Fault Monitoring

During system operation, the controller(s) should routinely monitor the status portion of each EGD consumed exchange to check for faults in the Ethernet NIUs in the system.

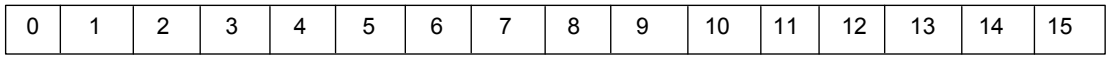
If bit 7 of a consumed exchange is set to 1, the fault should be investigated and corrected as described in this section.



Word 1: Status and Fault Data



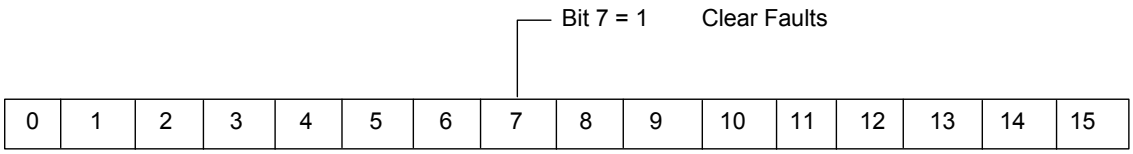
Word 2: Copy of Control Data Word 2 (Application-based) from Primary Controller



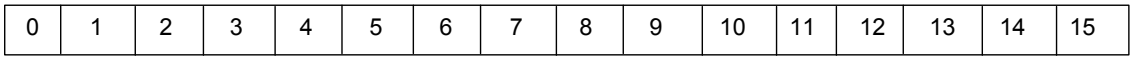
Word 3: Copy of Control Data Word 3 (Application-based) from Secondary Controller

Words 4 - 10: Reserved

The programmer can be used to view the Fault Table in the Ethernet NIU. After the condition that caused the fault condition has been corrected, the programmer can be used to clear the Fault Table or the application program in the controller can set bit 7 in the control data portion of its produced EGD exchange to clear the fault report. Setting this bit clears faults in ALL Ethernet NIUs that consume the same EGD exchange. If the system has a primary and secondary controller, only the exchange from the currently-active controller can be used to clear faults.



Word 1: Control Data



Word 2: Available for Use by Application

Words 3 – 10 should be set to zero

Viewing the Fault Tables in the Ethernet NIU

The Ethernet NIU maintains two fault tables that can be accessed with a connected programmer.

The *I/O Fault Table* lists faults associated with modules in the I/O Station, such as the loss or addition of a module.

The *PLC Fault Table* lists faults associated with the ENIU itself. Fault descriptions are listed on the next page. For each fault, see the “User Action” column for information about correcting the fault.

Viewing Extra Fault Data

In the PLC Fault Table, double-clicking the entry displays additional fault data as shown below:

28000100030005000000000000000000000000000000000000

The Station Manager utility can be used to view more detailed information about specific faults.

For the Ethernet NIU, the leftmost 14 digits of fault extra data (underlined in the example above) show the corresponding Log Event (2 digits) and Entries 2, 3, and 4 (in that order, 4 digits each). The example above is reporting an Event 28, Entry 2=1, Entry 3=3, and Entry 4=5.

PLC Fault Table Descriptions

PLC Fault	User Action
Backplane communications with PLC fault; lost request	If problem persists, contact GE Fanuc.
Bad local application request; discarded request	If problem persists, contact GE Fanuc.
Bad remote application request; discarded request	Try to validate the operation of the remote node. *
Can't locate remote node; discarded request	Error reported when message received where IP/MAC address cannot be resolved. Error may indicate that remote host is not operational on the network.
Comm_req - Bad task ID programmed	Internal request for unknown Ethernet Interface task.
Comm_req - Wait mode not allowed	Internal request error.
Config'd gateway addr bad; can't talk off local net	Error in configuration. Verify IP address, Subnetwork Mask, and default Gateway IP address are correct.
Connection to remote node failed; resuming without it	Underlying communications software detects error transferring data; resuming. If persistent error, check connection to LAN and operation of remote node.
LAN controller fault; restart LAN I/F	Hardware fault, perform power cycle. *
LAN controller Tx underflow; attempt recovery	Internal system error. *
LAN controller underrun/overrun; resuming	Internal system error. *
LAN data memory exhausted - check parms; resuming	The Ethernet NIU does not have free memory to process communications. *
LAN duplicate MAC Address; resuming	A frame was received in which the source MAC Address was the same as this station's MAC Address. Immediately isolate the offending station; it may be necessary to turn it off or disconnect it from the network. This station remains Online unless you intervene to take it Offline.
LAN I/F can't init - check parms; running soft Sw utl	Internal system error. *
LAN I/F capacity exceeded; discarded request	Verify that connection limits are not being exceeded.
LAN interface hardware failure; switched off network	Replace Ethernet NIU.
LAN network problem exists; performance degraded	Backlog of transmission requests due to excessive traffic on the network. For a sustained period the MAC was unable to send frames as quickly as requested. *
LAN severe network problem; attempting recovery	External condition prevented transmission of frame in specified time. Could be busy network or network problem. Check transceiver to make sure it is securely attached to the network. Check for unterminated trunk cable.
LAN system-software fault; aborted connection resuming	Internal system error. *
LAN system-software fault; restarted LAN I/F	Internal system error. *
LAN system-software fault; resuming	Internal system error. *
LAN transceiver fault; OFF network until fixed	Transceiver or transceiver cable failed or became disconnected. Reattach the cable or replace the transceiver cable. Check SQE test switch if present on transceiver.
Local request to send was rejected; discarded request	Internal error. Check that the Ethernet NIU is online.*
Memory backup fault; may lose config/log on restart	Internal error accessing FLASH device. * May need to replace Ethernet NIU.
Module software corrupted; requesting reload	Catastrophic internal system error. *
Module state doesn't permit Comm_req; discarded	Ethernet NIU cannot process request. Make sure Ethernet NIU is configured and online.
Unsupported feature in configuration	Attempt has been made to configure a feature not supported by the Ethernet NIU version.

- If this problem persists, contact GE Fanuc.

Using the Station Manager

The built-in Station Manager function of the Ethernet NIU provides additional tools for troubleshooting that are particularly useful during system startup.

Use of the Station Manager requires an operator interface device, either a computer running terminal emulation software or an ASCII terminal. The commands that can be used with the Station Manager are described in *the Station Manager User's Manual*. For PACSystems controllers, this manual is catalog number GFK-2225. For Series 90 systems, it is GFK-1186. Both manuals are available online at GEFanuc.com.

The Station Manager can be used to:

- Check the IP Address of the local Ethernet NIU.
- Make sure the IP Address is unique on the network.
- Display additional information about a node, such as its data rate and parity.
- Test communications on the network.
- View the Exception log, which lists the same types of faults as the PLC Fault Table.
- View communications errors with the TALLY command.
- Check Status of Exchanges with the Stat command.
- View Details of individual Exchanges with the Xchange command.

Checking the IP Address of the Ethernet NIU

With the terminal connected directly to the Station Manager port on the Ethernet NIU, issue the NODE command:

```
> node
IC695 Peripheral Ethernet Interface
Copyright (c) 2003-2005.  All rights reserved.
Version 3.60 (35A1) TCP/IP
Version 2.50 (20A1) Loader
IP Address = 10.0.0.2        Subnet Mask = 255.255.0.0
Gateway = 0.0.0.0
MAC Address = <<080019010203>>
SNTP Not Configured

Station Manager Port:
  Data Rate = 9600,  Parity = NONE,  Flow Control = NONE

Source of Soft Switches: PLC Configuration
Source of IP Address:    Configuration

Oct 24, 2005  16:33:31.8
Date/time initialized from PLC CPU
```

The NODE command also displays other identifying information about the Ethernet NIU as shown above.

Verifying that the IP Address of the Ethernet NIU is Unique

Make sure the Ethernet NIU does not have the same IP address as another node.

1. Disconnect the LAN cable from the Ethernet NIU.
2. Log on to another device on the network
3. From the other device, ping the IP address assigned to the Ethernet NIU.

If you get an answer to the ping, it means the chosen IP address is already in use by another node. You *must* correct this situation by assigning unique IP addresses.

Testing Communications on the Network

During system setup, use the Station Manager to test each installed Ethernet device to be sure that each is operational and configured with proper TCP/IP parameters. To do that:

1. Enter the LOGIN command:

   ```
   login
   ```

   The password prompt appears:

   ```
   Password:
   ```

2. The factory default password is:

   system (lower case).

   Enter the default password, or other password if it has been changed.

3. If the password matches the current password for the Modify level, the Modify prompt appears:

   ```
   =
   ```

4. Use the PING command to test the ability to reach individual nodes. The test works by sending an ICMP echo request message to a specific destination and waiting for a reply. Most nodes on TCP/IP networks implement *ping*.

   PING can reach remote IP networks through gateways.

   Enter the PING command using the IP address for the destination to be tested. A typical PING command is shown below:

   ```
   = ping 10.0.0.2 10
   Ping initiated

   <<< Ping Results >>>
   Command:  ping  10.0.0.2  10  100  64
   Sent = 10, Received = 10, No Timely Response = 0
   Late/Stray Responses = 0
   Round-trip (ms)  min/avg/max  0/1/10
   ```

Viewing the Exception Log

When the Ethernet NIU detects an unusual condition, it records information about the condition in its *exception log*. The exception log can be viewed using the Station Manager LOG command. For example:

```
> log
<<< Extended Exception Log >>>
IC695 Peripheral Ethernet Interface version 3.60 (35A1)
Log displayed 24-OCT-2005  16:39:32.5
Log initialized using valid RAM information
Log last cleared 21-OCT-2005  09:33:46.9                                        Remote IP Addr:Port
  Date         Time         Event Count  Entry 2 through Entry 6         Scode   or Producer ID:Exchg   Local IP Addr: Port
  24-OCT-2005  16:38:52.9    1H    1H   0000H 0001H 0000H 0000H 0000H
  24-OCT-2005  14:01:22.2   20H    1H   0001H 0000H 0000H 0001H 0117H
->24-OCT-2005  09:33:47.2   2aH    1H   0004H 0000H 0000H 0004H 0192H
```

Each new (not repeating) log event is also sent to the PLC Fault Table, where it can be viewed using the programming software.

The Station Manager LOG command returns the time/date of each exception event, a hexadecimal code that identifies the fault type (for example, 28H for an Ethernet Global Data fault), a count, and additional data in entries 2 through 6. When an error occurs, this information may pinpoint the cause more precisely than the PLC Fault Table display.

Checking the Network Connection

If the LAN LED is off, the Ethernet NIU is not able to send or receive on the network. The usual cause is some type of hardware problem. If this occurs, follow the procedure below.

1. Check to be sure that the network cables are securely fastened to the Ethernet NIU and to the network connection device (hub, switch, etc.).
2. Use the Station Manager to check the network interface task using a TALLY L command. The TALLY L command displays a list of tallies for all network interface tasks, and will identify specific communications errors that may be occurring.

If the Ethernet NIU is the only device experiencing problems:

1. Be sure the network cable is properly connected to the Ethernet NIU and to the network connection device.
2. Verify that the network connection device is operating properly on the network. (Are other devices operating on the same network segment?)
3. Make sure the Ethernet NIU is seated and secured properly.
4. Replace the network cable with a known good cable.
5. Verify that the system power supply is properly grounded.

If all stations are experiencing the problem, the network is probably at fault. Contact the network administrator.

Checking Exchanges with the STAT Command

The existence and correct operation of Exchanges can be checked using the STAT command

Using the Station Manager, type STAT G

The Station Manager will show the configured exchanges for this device, show their status and indicate the number of exchanges that have occurred.

```
> stat g
<<< EGD Status >>>   24-OCT-2005  16:46:05.0
                                                          Transfers
Ndx     Producer ID     Exchange ID  Mode      State      Completed
---     -------------   ----------- --------   ---------- -----------
 0H       10.10.10.3             1  CONSUMER      ACTIVE(00H)    1379368
 1H       10.10.10.2             1  CONSUMER      ACTIVE(00H)    1447992
 2H      10.10.10.11             1  PRODUCER      ACTIVE(01H)    1399605
```

The State column indicates whether the Exchange is active or idle and gives a code in hexadecimal that indicates the status.

For Produced Exchanges, (01H) indicates the exchange is being sent

For Consumed exchanges:

- 00H and 01H indicate the exchange is being received properly and on time
- 05H indicates the exchange is being received properly and on time, but the data is stale. The PLC has not updated the data since the last exchange was received. It is normal to receive Stale indications if the PLC Scan is longer than the EGD Production Period.
- 06H indicates the exchange is not being received.
- 0eH indicates the exchange is being received but the number of bytes received is different than expected. The exchange is not being used due to the length error.

Individual Exchange setups can be viewed by using the Xchange command in station manager.

Type "Xchange <producer ID> <exchange ID>

When the STAT LED is ON

Sometimes problems can occur even when the STAT LED is on, indicating normal operation. In that case, check if the LAN LED is steadily on, indicating that the Ethernet NIU is successfully attached to the Ethernet network, but there is no network activity.

To find out whether the Ethernet interface component in the Ethernet NIU can access the module's CPU, issue successive TALLY C commands. If the *PlcSweep* tally is not increasing, there are no windows being provided by the CPU. If any of the following tallies: *PlcAbt*, *MyAbt*, or *Timeout* are incrementing, there may be a hardware problem with the backplane interface. Check the PLC Fault Table entries.

Stale Ethernet Global Data Status

A stale data status is a non-fatal status. Although an Ethernet Global Data exchange is producing at the correct period, the data in the exchange can be old (stale) if the controller has not yet updated it. If the produced period for an exchange is less than the controller's scan time, the Ethernet device can send the same data in more than one Ethernet Global Data exchange. If the controller has not updated the EGD data before the exchange produced period expires, the Ethernet device sends the same data again.

Stale data status can also occur from an Ethernet NIU if the ENIU uses local logic. The use of local logic can increase scan time to become close to or larger than the Ethernet Global Data input data exchange's producer period. Each consumed EGD exchange status word received by the consumer of the exchange provides the indication of stale data. The stale data status is available for use by the application. The occurrence of stale data can also be determined by using the Ethernet Transmitter Module's Station Manager command – "tally G". A count of stale data occurrences for the Ethernet Transmitter Module's produced EGD exchanges is displayed along with other tally G data.

If You Can't Solve the Problem

If you are not able to solve the problem, call GE Fanuc Automation. Please have the following information available when you call.

1. The name and catalog number marked on the module
2. Description of symptoms of problem. Depending on the problem, you may also be asked for the following information:
   - The application program and the PLC sweep time at the time the problem occurred.
   - A list of the configuration parameters for the Ethernet device that failed.
   - A list of reported errors. This can be the contents of the Ethernet exception log, the contents of the PLC Fault Table, or both.
   - A description of the network configuration. This should include the following:

- The number of systems accessing the network
- The type of network cable used (for example, twisted pair, fiber optic, etc.)
- The length of network cable
- The manufacturer and quantity of hubs and network switches.

Chapter 7 Local Program Logic in the Ethernet NIU

This section describes the Local Logic feature of the Ethernet NIU.

- Using the Local Logic Block
- Reference Table Restrictions for Local Logic
- Using COMMREQs in the Local Logic

Using the Local Logic Block

The RX3i Ethernet NIU allows the addition of up to 20K bytes of logic to be executed locally in the I/O Station. When the RX3i Ethernet NIU target is created in Machine Edition, an empty LD logic block named “Local User Logic” is created and is called from Main. This block can be changed to ST or FBD by deleting the LC block “LocalUserLogic” and the creating a new block with the type of ST or FBD.

NOTE: Even if Local_User_Logic is not used, the block MUST be left in the program. Deleting the block will cause a store to the Ethernet NIU to fail.

Reference Table Restrictions for User Logic

Restricted Addresses

I/O operation and the Remote COMMREQ Calls feature use reference table addresses in the RX3i Ethernet NIU. The following reference table addresses are used for I/O operation and RCC functionality and MUST NOT be written to by Local_User_Logic:

%R00001 to %R09999

%M00001 to %M04096

%T0001 to %T0512

Addresses written to by EGD Exchanges

The Ethernet Global Data exchanges “Outputs_Pri _to_ENIU” and “Outputs_Sec_to_ENIU” write to %M and %R addresses (which are in the Restricted Addresses listed above) and the Ethernet NIU then writes to the following addresses (listed below) every scan of the ENIU:

%Q00001 to %Q02048

%AQ0001 to %AQ0512

If Local User Logic writes to these addresses, the Ethernet NIU functionality overwrites the values and it appears that the local logic is not working.

Using COMMREQs in the Local Logic

COMMREQs can be used in Local_User_Logic. There are two items that the Local_User_Logic must be sure to take into account.

1. The COMMREQ is local to the Ethernet NIU. That means the COMMREQ status word address, data source address, and data response addresses are also local to the RX3i Ethernet NIU. If the result of a COMMREQ in the local logic must be provided to the controller(s) associated with the Ethernet NIU, special consideration must be made. The COMMREQ will put the data in the Ethernet NIU memory. The Ethernet NIU then needs to send the data to the controller(s). This can be done by:
   - including the memory in the EGD exchange “Inputs_from_ENIU_xx” , or
   - creating a new EGD exchange to send the memory, or
   - using an SRTP channel or EGD Command COMMREQ to send the data to the controller(s).
2. If the Remote COMMREQ Calls feature is used in the controller(s), and COMMREQs are used in local logic, be careful not to inadvertently issue a COMMREQ from both Remote COMMREQ Calls and local logic to the same module at the same time. If a module receives a second COMMREQ before the first COMMREQ completes, the module responds with a busy error code to the second Commreq.

Chapter 8 Remote COMMREQ Calls

This chapter describes the Remote COMMREQ Call (RCC) feature that allows PACSystems RX7i and RX3i controllers to pass COMMREQs to modules in an I/O Station via the Ethernet NIU. This capability is not available with other types of system controllers.

- Using Remote COMMREQ Calls
- The Remote COMMREQ Call “C” Block
- Configuring Ethernet Global Data Exchanges for Remote COMMREQ Calls
- Diagnostics for Remote COMMREQ Calls
- Remote COMMREQ Calls in a Redundancy System

Using Remote COMMREQ Calls (RCC)

The Remote COMMREQ Call feature utilizes of a pair of dedicated Ethernet Global Data exchanges between the controller and an RX3i Ethernet NIU. One exchange sends standard COMMREQs from the controller to the Ethernet NIU. The other sends the result of the COMMREQ from the RX3i Ethernet NIU to the controller.

In a system with redundant controllers, there is an RCC exchange from each controller to the Ethernet NIU. The RCC exchange from the RX3i ENIU is sent to a group destination so that both controllers can receive it.

Chapter 9 describes the COMMREQs that the controller(s) can send to an RX3i ENIU I/O Station using Remote COMMREQ Calls.

RCC Functionality in the Ethernet NIU

Remote COMMREQ Call functionality is built into the RX3i Ethernet NIU. In order to use it in a system, the Ethernet Global Data exchanges must first be configured as described in this chapter. The EGD exchanges are prepopulated with required fields to facilitate the configuration.

RCC Functionality in the Controller

Remote COMMREQ Call functionality is only available in PACSystem RX7i or RX3i controllers. The functionality is provided by two EGD exchanges and a parameterized “C” block. The “C” block is named “RCCM_xxx_yy”. In this name, xxx is a revision code. The yy portion of the name is used if multiple copies of the “C” block are needed for multiple Ethernet NIUs that have Remote COMMREQ Calls. The controller needs a separate “C” block for each RX3i ENIU which will receive RCC commands. Each “C” block must have a unique name.

Like a COMMREQ instruction, the “C” block in the controller has input parameters. The application program must include additional logic to sequence the commands to the “C” block and to monitor the status for completion of the Remote COMMREQ Call. The “C” block drives the EGD exchange (RCC_Pri_request_to ENIU_xx) that sends the COMMREQ to the ENIU. The Ethernet NIU sends the result back to the controller using the Ethernet Global Data exchange: RCC_response_from_ENIU_xx. The “C” block in the controller puts the results into the status and data areas that were specified in the Remote COMMREQ Call request.

Remote COMMREQ Call Operation

A PACSystems controller with the RCC “C” block sends COMMREQs to the RX3i Ethernet NIU in an Ethernet Global Data exchange named RCC_Pri_request_to_ENIU_xx.

The RX3i Ethernet NIU executes the COMMREQ, then sends the result back to the controller in an Ethernet Global Data exchange named RCC_response_from_ENIU_xx.

The “C” block in the controller does the following:

- Takes the inputs to the “C” block and loads the Ethernet Global Data exchange with the information required for the Ethernet NIU to execute the COMMREQ.
- Sends the COMMREQ information in the RCC_Pri_request_to_ENIU_xx exchange and adds a sequence number for checking.
- Monitors for a response, returns the COMMREQ Status Word, and returns data if a response is expected and the COMMREQ was successful.
- Does a timeout if the Ethernet NIU does not respond.
- Detects a powerup of the Ethernet NIU and provides a power up status to the controller.

Configuring EGD Exchanges for Remote COMMREQ Calls

The Remote COMMREQ Call (RCC) Communications function uses additional Ethernet Global Data exchanges that communicate COMMREQ commands from the controller to an intelligent module in an Ethernet NIU I/O Station. Remote COMMREQ Calls also communicate the responses from the Ethernet NIU back to the controller. When an RX3i Ethernet NIU target is added in Machine Edition, three RCC-related Ethernet Global Data exchanges are included in the EGD exchanges component of the target configuration. Those exchanges are:

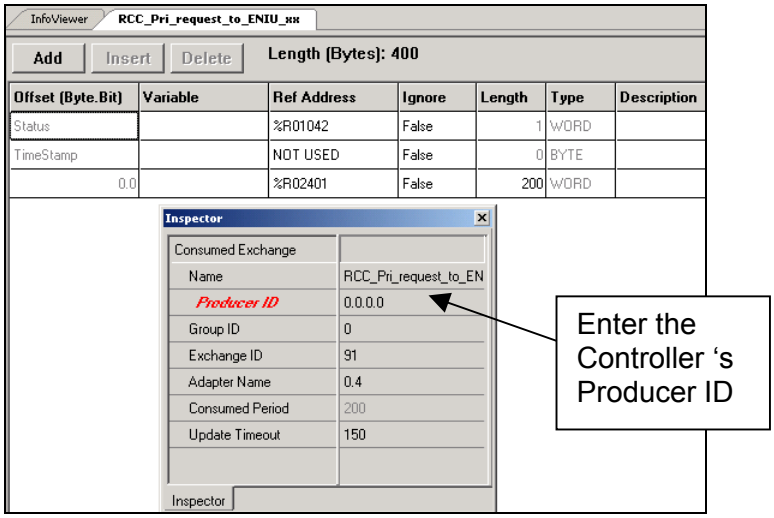
- RCC_Pri_request_to_ENIU_xx: Consume a RCC request from the primary controller
- RCC_Sec_request_to_ENIU_xx: Consume a RCC request from the secondary controller
- RCC_response_from_ENIU_xx: Produce a RCC response back to the controller(s)

Each of these EGD Exchanges has a default produced period of 50 milliseconds and a consumer update timeout of 150 milliseconds. The suggested produced periods and consumer update timeouts listed in chapter 5 can be maintained even with the use of RCC exchanges with a single Ethernet NIU. If additional RCC exchanges are added for more Ethernet NIUs, the suggested produced periods and consumer update timeouts in chapter 5 may need to be modified.

Configuring the ENIU's Consumed Exchange to Receive RCC

"RCC_Pri_request_to_ENIU_xx" is the Ethernet Global Data exchange that delivers the COMMREQ request from the primary controller to the Ethernet NIU.

In the Rx3i Ethernet NIU configuration, this exchange is pre-populated, so only the Producer ID of the controller needs to be entered as shown below.

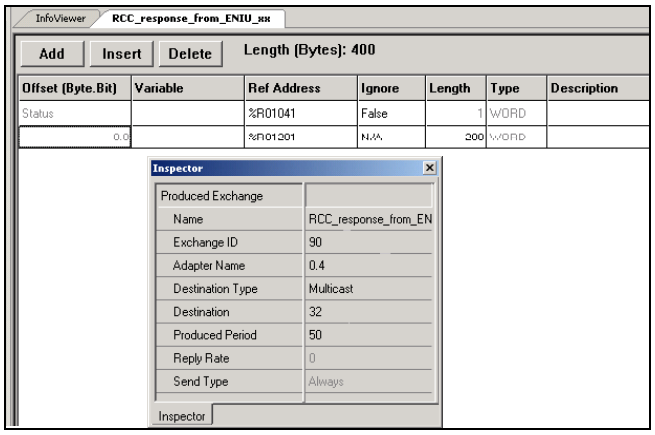


If a secondary controller is used, its EGD configuration is also pre-populated, so only the Producer ID of the secondary controller needs to be configured.

Configuring the ENIU's Produced Exchange for Response to RCC

RCC_response_from_ENIU_xx is the Ethernet Global Data exchange that delivers the COMMREQ request from the controller to the Ethernet NIU.

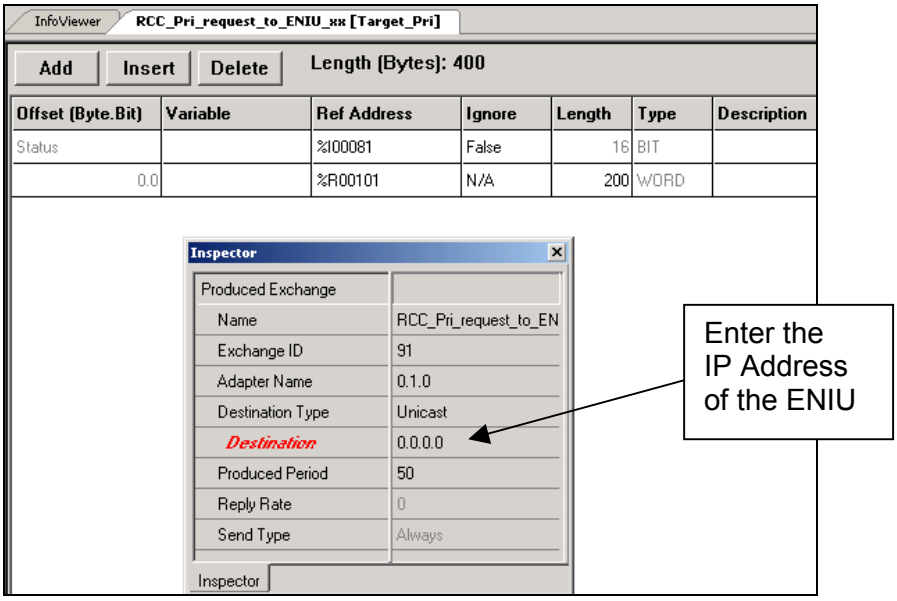
In the Ethernet NIU, the exchange is pre-populated and does not need changing.



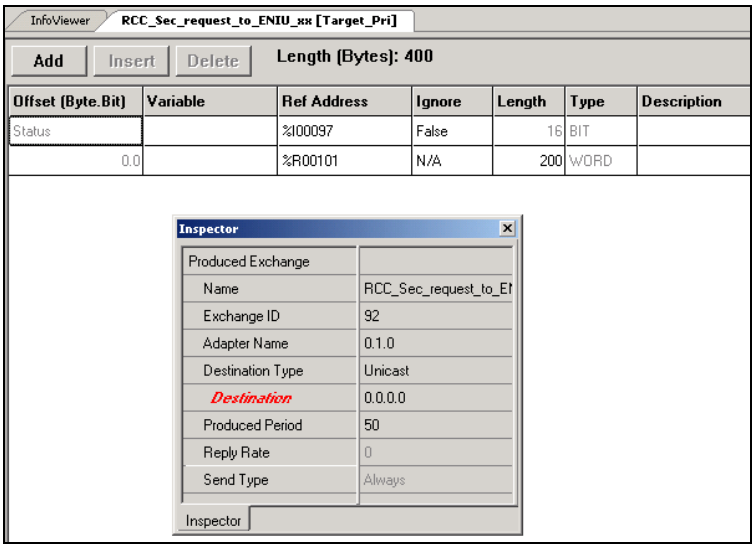
Configuring the Controller's Produced Exchange to Send RCC

The request exchange (RCC_Pri_request_to_ENIU_xx) needs to be configured in the primary or only controller as shown below.

- The Exchange ID is 91, if multiple ENIUs are to receive RCC command each ENIU will need a separate exchange and each exchange will need a unique Exchange ID.
- The Adapter Name identifies the Ethernet module that is sending the EGD Exchange.
- The Destination Type is Unicast.
- The Destination is the IP Address of the Ethernet NIU.
- The Produced period should be 50 milliseconds to match the ENIU default.



If there is a secondary controller, the request exchange needs to be configured in the secondary controller as shown below.

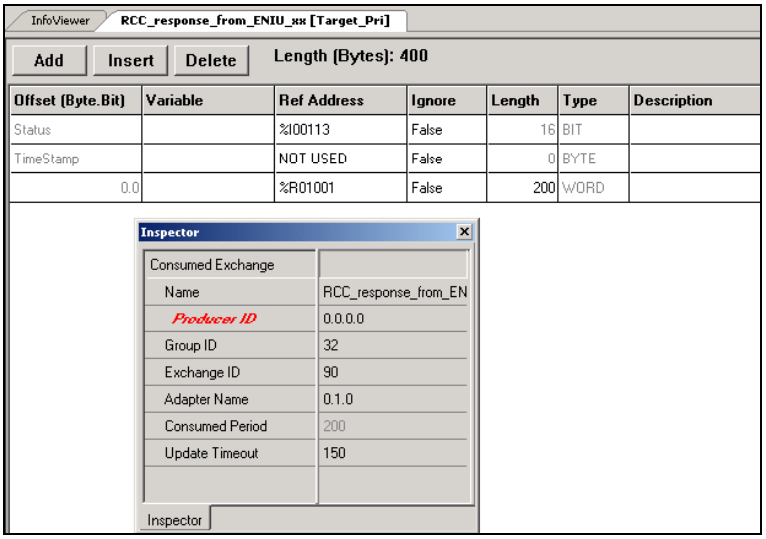


Note that the Exchange ID is 92

All the other parameters are the same as in the exchange from the primary controller.

Configuring the Controller's Consumed EGD Exchange for RCC Response

The RCC_response_from_ENIU_xx request exchange needs to be configured in the primary or only controller as shown below. The Producer ID is the Produced ID of the RX3i Ethernet NIU.



The request exchange also needs to be configured in the optional secondary controller. The parameters must be identical to the configuration in the primary controller.

Configuring Exchanges if Multiple ENIUs Will Receive RCC Commands

If multiple Ethernet NIUs will receive Remote COMMREQ Call commands, the produced exchange from the controller(s) to the Ethernet NIU must each have a different Exchange ID. That makes each Ethernet Global Data Exchange unique on the Ethernet media. Exchanges in the Ethernet NIUs and controllers must be adjusted for every Ethernet NIU after the first, in order not to duplicate Ethernet Global Data exchanges on the network.

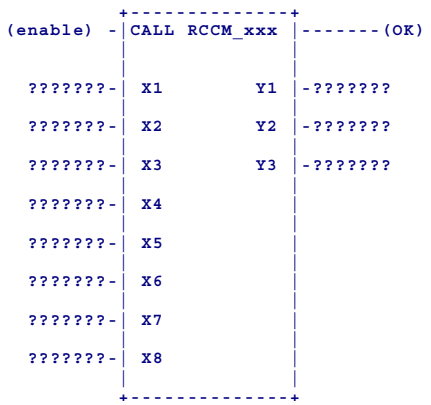
The Ethernet Global Data produced exchange from the primary controller to the first Ethernet NIU uses Exchange ID 91. The secondary controller uses Exchange ID 92. For each additional ENIU the Exchange ID should be incremented by 2.

TheExchange ID needs to be incremented on both ends: at the produced exchange in the controllers, and at the consumed exchange in the Ethernet NIU.

Adding the RCC "C" Block to the Controller Logic

In the Machine Edition navigator tree view, right-click Program Blocks in the logic area of the controller. Click on "add C block". A dialog box to add the "C" block will come up. Browse to the file RCCM_120.gefElf. To add it to the controller target, double-click or select and open it.

The "C" block has this form:

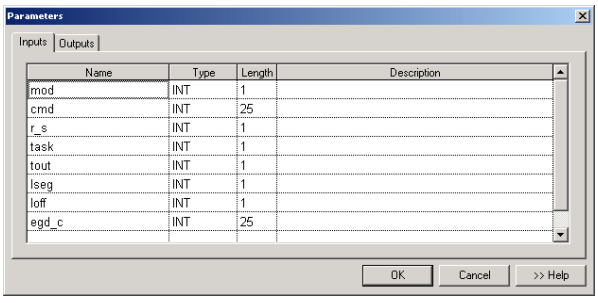


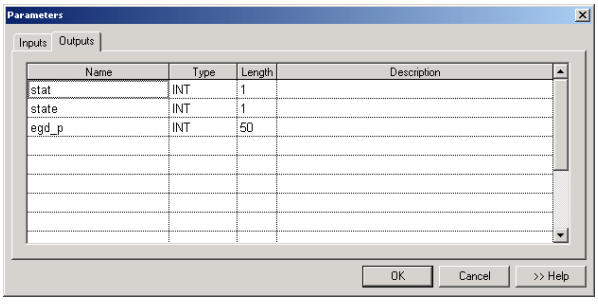
The inputs and outputs for the "C" block have the following labels:

	Name	Type	Length	Description
X1	mod	int	1	Module Type
X2	cmd	int	25	COMMREQ Data block
X3	r_s	int	1	Rack/Slot Module COMMREQ is directed to
X4	task	int	1	COMMREQ Task
X5	tout	int	1	COMMREQ Timeout in milliseconds
X6	lseg	int	1	Seg Sel for a 200 word buffer in the controller
X7	loff	int	1	Offset of 200 word buffer in the controller
X8	egd_con	int	25	Address of RCC Consumed Exchange data range
Y1	stat	int	1	status of Remote COMMREQ Call
Y2	statt	int	1	state of Remote COMMREQ Call
Y3	egd_pro	int	25	Address of RCC Produced Exchange data range

Adding the "C" Block Parameters

Right-click on the C Block in the Navigator tree view, then click on Properties. Click on the parameters line in the property inspector to open the parameters dialog. Enter the parameters as shown below.





Adding the "C" Block Call to Controller Logic

Create a LD block called RCC. In _Main add a Call to RCC that is called every scan.

In the RCC block, add a call to RCC_120_xx that is called every scan.

The inputs and outputs of the "C" block need to be entered as described below.

Inputs for the C Block

mod Module type the COMMREQ is being sent to. Enter a Register reference and place the module code in the register.

cmd This is the COMMREQ command block. Enter the register reference where the block starts. The register reference must have an array length of 25.

r_s Slot number of the module the COMMREQ is being sent to. Enter a Register reference and place the slot number in the register.

task Task number that the module uses for COMMREQs it receives. Enter a register reference and place the task number in the register.

tout Timeout for the request in milliseconds. Enter a register reference and place the timeout in the register.

lseg Segment selector for a 200-word buffer needed by the "C" block. Enter a constant (8 for %R, or 196 for %W).

loff Starting reference number of the buffer. Enter a constant, i.e. 7001.

egd_c Pointer to the "RCC_response_from_ENIU_xx" Exchange. Enter the starting register reference of the exchange data range. It must have an array length of 25.

Outputs of the C Block

Stat Status of the Remote COMMREQ Call command. Enter a register reference. This is monitored to determine completion and success of the Remote COMMREQ Call command.

State State of the Remote COMMREQ Call command. Enter a register reference Tell the intermediate step the "C" block is on.

egd_p Pointer to the "RCC_request _to_ENIU_xx" exchange. Enter the starting register reference of the exchange data. The range must have an array length of 50.

List of Module Type Codes

The RCCM “C” block needs the module type to be specified on the first input parameters. Each module type is represented by a numeric code, as listed below:

Module	*Code*
Genius Bus Controller	331
Profibus Master	300
DeviceNet Master	200
Motion Module (DSM314)	314
Motion Module (DSM324)	324
High Speed Counter	3000
Modbus Master	4000
Hart	5000
ENIU (Read last COMMREQ)	6000

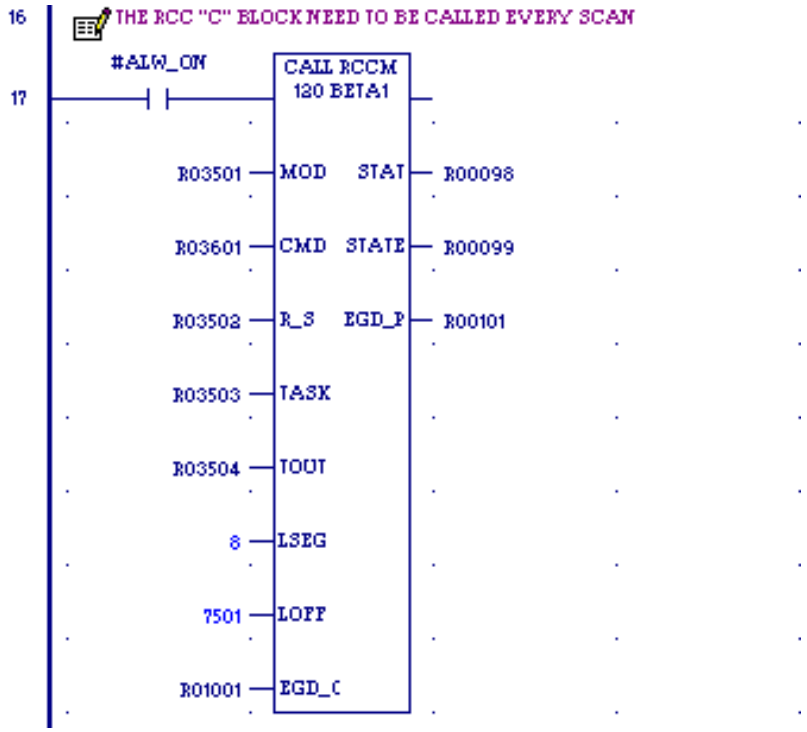
Adding Logic to Sequence RCC Commands and Check Return Status

A Remote COMMREQ Command works like a COMMREQ.

- The command needs to be executed once and the Remote COMMREQ Call status needs to be monitored for completion before the command is executed again.
- The command executes when the CMD input values are loaded. The “C” block zeros out the seventh register in the array on the CMD input. (This is the COMMREQ Command number.)
- The CMD input is the COMMREQ command block.
- The other inputs are used to route the COMMREQ to the correct module and to set timeout values.

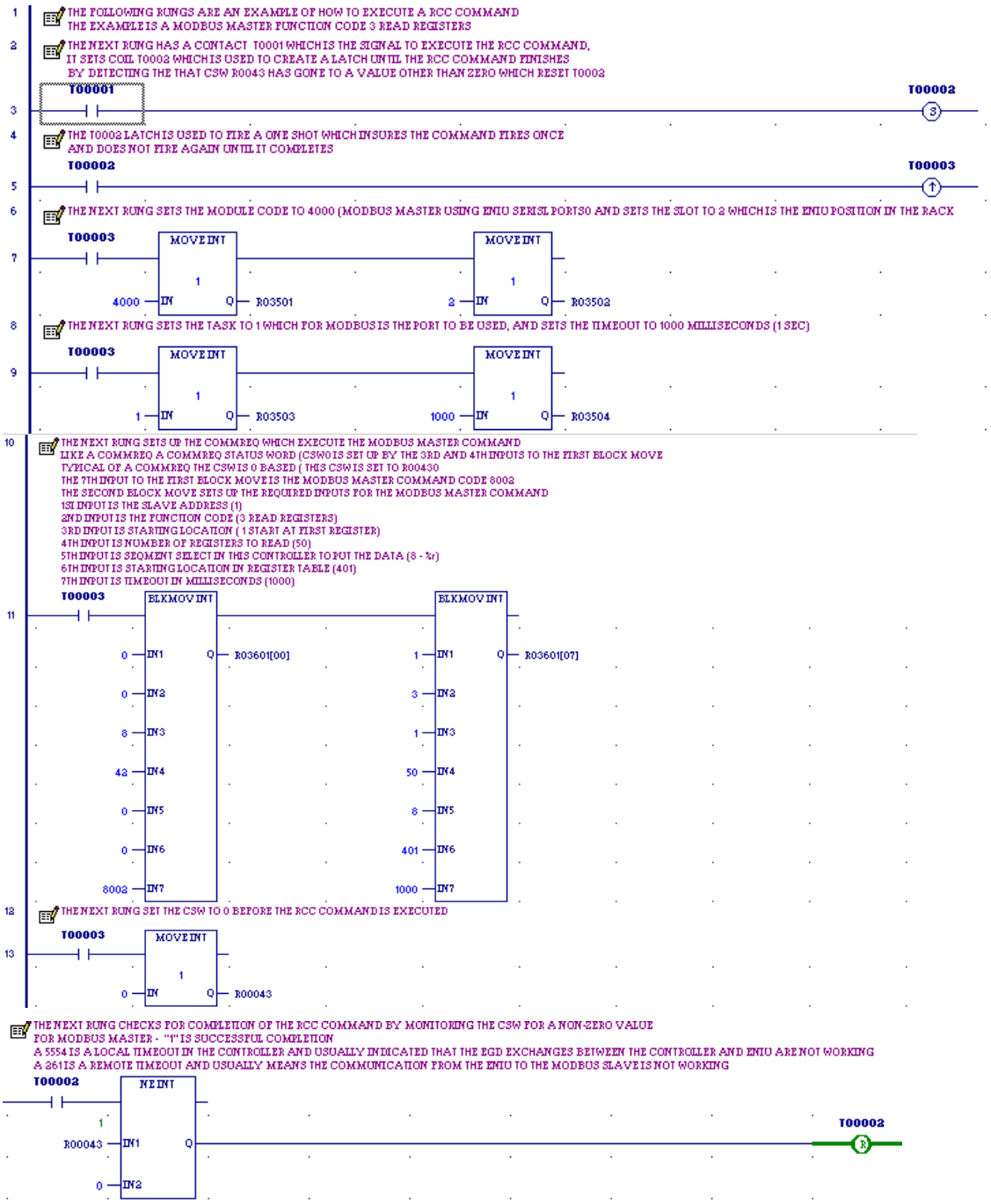
Sample Logic

Example logic for two commands is shown on the following pages. Both commands use the same “C” block, which is shown below.



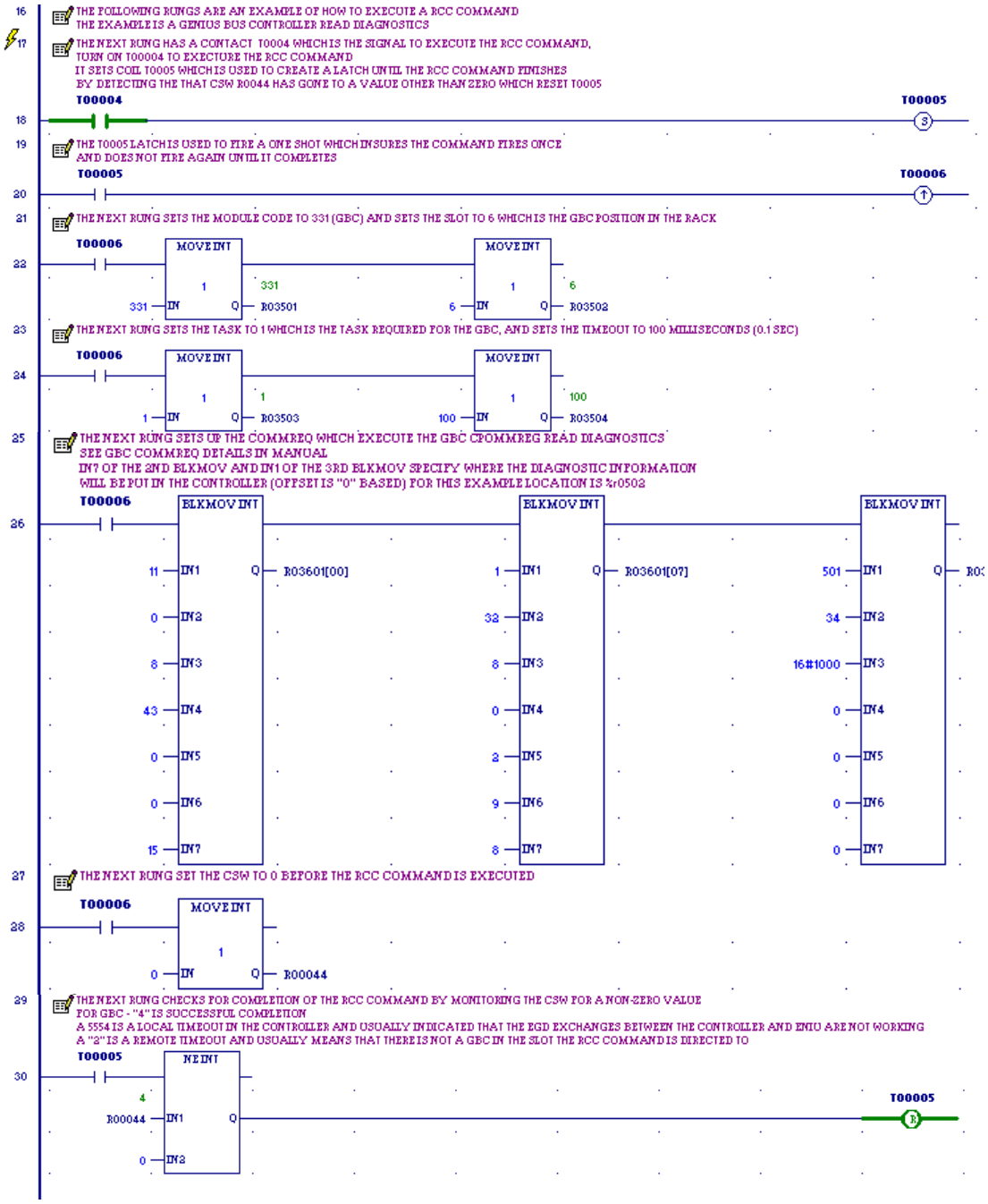
Sample Logic for Modbus Master

Port 1 of the Ethernet NIU the Port Mode MUST be changed to Serial I/O and the Baud rate and parity need to be set to match the Modbus Slave settings. Once the port setup is changed it must be downloaded to the Ethernet NIU. If the port setting are not changed, the example will give an error code of 5379 (1503h).



Sample Logic for GBC Command Read Diagnostics

The Genius Bus Controller MUST be configured in slot 6 of the Ethernet NIU I/O Station.



Monitoring Remote COMMREQ Calls for Completion

The RCCM "C" block will supply a result for the requested RCC command. The application in the controller needs to monitor it for completion.

The RCCM "C" block does the following checks in the order listed:

- The COMMREQ status word address is checked for validity. If the COMMREQ status word is not valid, the RCCM block places an error code on the Status output. An error code on the Status output should only happen during application development and not in a running system. Two exceptions to this are:

  1. After power is cycled to the Ethernet NIU. A Status output value of 9999 indicates that the ENIU has powered up.
  2. (For redundant controllers only), a code of 2040 indicates the Ethernet NIU switched to the other controller during the execution of a Remote COMMREQ Call.

- The requested RCC command is checked to be sure that:

  A. The requested COMMREQ is one that is supported by the Remote COMMREQ Call feature.

  B. The addresses in the controller for source and destination data are valid.

     If either check fails, an error code is placed in the COMMREQ status word and the Remote COMMREQ Call is NOT sent to the Ethernet NIU.

  C. The RCCM block times out on receiving a response to the Remote COMMREQ Call command from the Ethernet NIU (this is not a check that the EGD exchanges are being received). This is a local timeout and a local timeout error code is placed in the COMMREQ Status Word.

- The Ethernet NIU responds with a completion code, indicating success or an error condition. The RCCM "C" block places this result code in the COMMREQ Status Word. Note that different modules use different values in the COMMREQ Status Word for successful completion. Most modules use "1" to indicate success, but the Genius Bus Controller uses a "4" for success.

Diagnostics for Remote COMMREQ Calls

COMMREQ Status Word

The “cmd” input to the “C” block in the controller is the data block that is used for a COMMREQ. The third and fourth words in the data block specify a Reference Table and offset (0-based) location for the COMMREQ status word. When a Remote COMMREQ Call command completes, a value is placed in the COMMREQ status word that indicates the status of the COMMREQ execution. Chapter 9 lists the COMMREQ status word values for all COMMREQs that can be sent using a Remote COMMREQ Call.

“C” Block Status Output – Codes

0	idle
2	Command active (in progress)
2040	ENIU detect controller switch before command completed
9999	ENIU Powered Up

“C” Block State Output – Codes

111	
333	
222	
47	Initial value

Troubleshooting

1. Verify that I/O and the Ethernet Global Data exchanges for Remote COMMREQ Calls are working. Connect the Station Manager to the controller Ethernet port and to the Ethernet Transmitter Module in the I/O Station, and type in “stat g”. This should return a list of the Ethernet Global Data exchanges that are configured for the Ethernet interface. All the Ethernet Global Data exchanges both I/O and Remote COMMREQ Calls should be listed as Active and have a status of (00h), (01h), or (05h).

   Any exchanges that are not listed or Active must be fixed.

   A status of (06h) means timeouts are occurring on the exchange. A status of (0Eh) means the size of the exchange does not match on the two ends. The xchange command in station manager can be used to check details of an exchange, such as the size.

2. Check the COMMREQ status word. The COMMREQ status word shows the result of the Remote COMMREQ Call command. COMMREQ status word values are listed in chapter 9.

   Typical operation is to zero the COMMREQ status word and monitor for a successful result. If the COMMREQ status word remains “0” either the COMMREQ status word location was specified incorrectly or the “C” block did not execute.

   Check the Status output of the “C” block . 8802 and 8803 indicate the value for the COMMREQ status word is incorrect.

Remote COMMREQ Calls in a Redundancy System

When the Ethernet NIU is used with redundant controllers, the Ethernet NIU is configured to receive I/O data and Remote COMMREQ Call requests from both controllers. The Ethernet NIU sends I/O data and Remote COMMREQ Call responses to both controllers.

The Ethernet NIU responds only to Remote COMMREQ Call requests from the controller that has control of the I/O. The status words that are returned in the “Inputs_from_ENIU_xx” exchange have bits that indicate which controller has control of the I/O. Bit 3 indicates that the Primary is in control. Bit 4 indicates that the secondary is in control. Only the controller that is controlling the I/O should send Remote COMMREQ Call commands. This makes the switchover logic easier and reduces the load on the Ethernet interfaces in the controller and I/O Station.

When control switches from one controller to the other, control of the I/O also automatically changes from one controller to the other. However, switching the Remote COMMREQ Call operation depends on the state that Remote COMMREQ Calls are in when the switchover occurred, which can be:

- Remote COMMREQ Calls are idle, with no RCC activity. In the idle state, Remote COMMREQ Call commands just move from one controller to the other.
- The controller issued a Remote COMMREQ Call command, but the switchover occurred before the Ethernet NIU received it, so the Ethernet NIU never saw the command. This can happen because the Remote COMMREQ Call command is sent to the Ethernet NIU in a Ethernet Global Data exchange, which introduces a delay of one production period in sending the command.
- The Ethernet NIU received a Remote COMMREQ Call command from the controller, but the switchover occurred before the Ethernet NIU could send its response. If that happens, the Ethernet NIU returns a status code of 2040. The new controller can detect the 2040 code in the state output of the “C” block. The application program in the new controller can use a “Read RCC command at switchover” command to retrieve the command that was sent to the RCC.
- The Ethernet NIU has queued a Remote COMMREQ Call response, but the switchover occurred before the controller received the response. The response is received by both controllers, but the new controller does not recognize the response, and does not process it. The response will be in the 200 Word Range specified in the Exchange “RCC_response_from_ENIU_xx”.

How the redundant system handles these four cases will depend on the redundant system and the needs of the application.

Read RCC Command at Switchover

If a 2040 response code is detected at the controller which has just become the active controller, it can retrieve the last command that was issued by the other controller. This should be done before any other Remote COMMREQ Call command is issued, or the information from the last command information will be lost.

The response to the last issued command is in the 200 Word Range specified in the exchange “RCC_response_from_ENIU_xx”. The response should be saved to another location before a command to retrieve last issues command is executed.

READ LAST COMMAND FROM ENIU

Inputs to “C” block RCCM:

Mod – 6000 code to tell RCCM block what type of module -

Cmd – Command block

0 – always zero

0 – always zero

8 – seg selector for CSW

48 – offset for CSW (zero based)

0 – always zero

0 – always zero

2040 – command code (Read last COMMREQ Command)

196 – seq select – where to put response

6601 – offset – where to put response

6601

24 – length number of words to read

8 – seg select of data (must be 8)

4123 – offset of data (must be 4123)

1000 – timeout in milliseconds

r_s – 2 Slot module is located in (CPU)

task – 1

tout – 25 timeout in milliseconds

egd_c – R01001 starting address of Produced Exchange for RCC

Response (for example above)

The first six words are:

R6601 – 0 This is Seq Number of RCC command but it gets zeroed

R6602 – 0 always zero

R6603 – 8 seg select for CSW in ENIU (“C” block sets this to 8 always)

R6604 – 4998 offset for CSW (“C” block sets this to 4998)

R6605 – 0 always zero

R6606 – 0 always zero

The relevant information in this response is:

R6607 – 8802 Module type code (8002 is Modbus)

R6608 – Command words follow in this and following registers

Status Values Generated by the RCCM_xxx "C" Block

COMMREQ Status Word

The "Cmd" input to the "C" block in the controller is the data block used for a COMMREQ. The third and fourth words in the data block specify a reference table type and offset (0-based) location for the COMMREQ status word. When a Remote COMMREQ Call command completes, the result code is placed in the COMMREQ Status Word.

Incorrect COMMREQ Status Word Location

If the COMMREQ Status Word location specified is wrong, the "C" block status output contains an error code that indicates which part of the address location is incorrect:

8802 Bad CSW Segment Selector

8803 Bad CSW Offset (either less than 1 or after end of reference table)

These errors usually occur during application development and checkout, not in a completed application. When an incorrect COMMREQ Status Word location has been specified, the controller will not send a Remote COMMREQ Call command to the Ethernet NIU.

COMMREQ Status Word Error Codes

COMMREQ Status Word error codes can be generated by the “C’ block in the controller or by the Ethernet NIU when it tries to execute the COMMREQ. When an error code is generated by the “C” block in the controller, the controller does not send the Remote COMMREQ Call command to the Ethernet NIU. The one exception to this is a timeout in the “C” block. If there is a timeout of the “C” block, a command may or may not have been sent to the Ethernet NIU.

COMMREQ Status Word Error codes generated by the “C” block are:

5554 Timeout detected in Controller “C” block
5555 Unsupported Module
5556 Unsupported
5557 Unsupported Ref Table
5558 Unsupported Command
6602 Bad Segment selector for Data Source in Controller
6603 Bad Offset for Data Source in Controller
7702 Bad Segment selector for Response location in controller
7703 Bad Offset for Response location in controller

COMMREQ Status Word error codes generated in the Ethernet NIU are the same error codes that would be generated if the COMMREQ were executed locally in a controller. The Ethernet NIU passes the error code back to the controller and the “C” block puts the error code in the designed COMMREQ Status Word location.

Chapter 9 COMMREQs for Remote COMMREQ Calls

The PACSystems RX3i Ethernet NIU supports selected COMMREQs received from a “C” block application in a Rx7i or RX3i controller. Ladder code in the RX7i or RX3i interfaces to the “C” block. The “C” block sends COMMREQ commands in an Ethernet Global Data Exchange to the Ethernet NIU. The Ethernet NIU executes the COMMREQ and sends the results back to the RX7i or RX3i with another Ethernet Global Data exchange.

This feature uses standard COMMREQ Command Data blocks. The standard COMMREQ Command Data block goes on the RCCM “C” block input “cmd”.

This chapter summarizes the standard COMMREQ Command Data blocks supported by the Ethernet NIU.

COMMREQS Supported by Remote COMMREQ Calls

The Ethernet NIU supports the following COMMREQs in Remote COMMREQ Calls:

Supported Devices	***COMMREQ Numbers***	***COMMREQ Descriptions***
PACSystems RX3i DeviceNet Master Module IC694DNM200 Series 90-30 DeviceNet Master Module: IC693DNM200	1	Send Device Explicit
	4	Get Detailed Device Status
	5	Get Detailed Server Status
	6	Get Status Information
	7	Send Device Explicit Extended
	9	Read Module Header:
PACSystems RX3i Ethernet NIU		Read last COMMREQ (Redundant systems only)
PACSystems RX3i Genius Bus Controller: IC694BEM331, Series 90-30 Genius Bus Controller: IC693BEM331	8	Enable/Disable Outputs Command
	13	Dequeue Datagram Command
	14	Send Datagram Command (switch BSM, clear fault, clear all faults, assign monitor, read diagnostic)
	15	Request Datagram Reply Command
PACSystems RX3i Analog Modules with HART Communications: IC695ALG26, 628, 728	1	Get HART Device Information
	2	Send HART Pass-Thru Command
PACSystems RX3i Profibus Master Module, IC695PBM300	1	Get Device Status
	2	Get Master Status
	4	Get Device Diagnostics
	5	Read Module Header
	6	Clear Counters
PACSystems RX3i and Series 90-30 Motion Controller Modules: IC693/694DSM314, DSM324	E501	Parameter Load
PACSystems RX3i High Speed Counter: IC694APU300, Series 90-30 High Speed Counter: IC693APU300	E201	Send Data Commands
PACSystems ENIU Serial ports: Modbus RTU Master	1	Read Outputs
	2	Read Inputs
	3	Read Holding Registers
	4	Read Input Registers
	5	Set/Clear One Coil
	6	Preset One Register
	7	Read Exception Status
	15	Write Multiple Coils
	16	Write Multiple Registers
	17	Report Device ID

COMMREQs for DeviceNet Master Modules

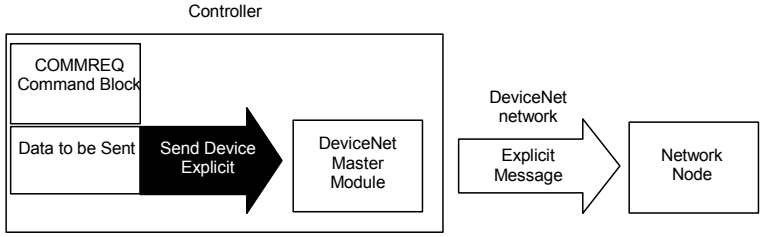
The controller can use a Remote COMMREQ Call to send the following COMMREQs to a PACSystems RX3i DeviceNet Master Module IC694DNM200 or Series 90-30 DeviceNet Master Module IC693DNM200 in the I/O Station:

1	Send Device Explicit
4	Get Detailed Device Status
5	Get Detailed Server Status
6	Get Status Information
7	Send Device Explicit Extended
9	Read Module Header:

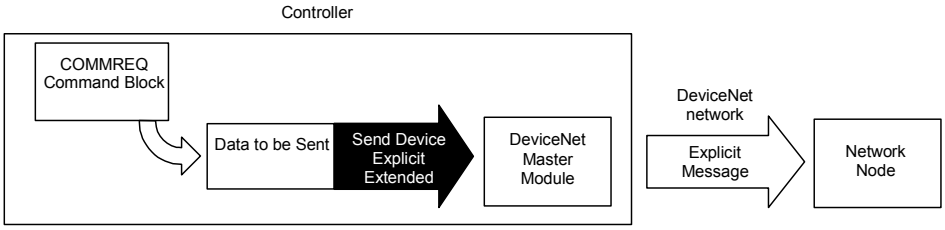
DeviceNet Master Modules, COMMREQ 1: Send Device Explicit

COMMREQ 1 commands the Master to send a DeviceNet explicit message to a specified device on the DeviceNet network. The message can be up to 238 bytes long. The reply data is limited to 2048 bytes maximum. To send more than 238 bytes of data or to use a separate data memory area in the PLC, use COMMREQ 7, Send Device Explicit Extended instead.

The difference between Send Device Explicit and Send Device Explicit Extended is how they store the data that will be sent in PLC memory. For Send Device Explicit, the data to be sent is located in the same memory area as the COMMREQ command block.



For Send Device Explicit Extended, the data to be sent is located in a separate memory area, which is indicated by a pointer in the COMMREQ command block. This makes it possible to store and send more data or to have the data separate from the command memory.



The addressed device must be configured for an explicit message connection in the controller configuration of the DeviceNet Master Module, and sufficient buffer memory must be configured to contain the largest message produced by the COMMREQ or the largest reply produced by the device. If the device was not configured for explicit messaging or if the number of bytes configured is not enough for the command, the COMMREQ fails with a code of 8 in the COMMREQ Status Word.

Send Device Explicit, COMMREQ Example

The Send Device Explicit COMMREQ command block contains the data to be sent in the explicit message (the data may optionally be offset from the end of the command block as explained below). For this example, there are multiple channels in the VersaPoint analog module to configure. The application program can repeat the message with a different instance [another channel]. Having the PLC application check the COMMREQ status is important even if there is only one COMMREQ, to be sure it has worked. For example, the VersaPoint DeviceNet NIU may be offline when the command is sent. When sequencing multiple commands to the same device (MAC ID), it is critical to test for successful command completion prior to executing a subsequent command.

The example COMMREQ below does the following:

- Sends an explicit message to device # 4 (a VersaPoint DeviceNet NIU)
- Returns the COMMREQ Status Words to %R10-%R13
- Sets Analog Input 1 to the 4-20mA range.

Word	*Dec*	*(Hex)*	*Description*
1	00012	(000C)	**Length of command:** Length of the command block for this COMMREQ. For the Send Device Explicit (command 1) the command length is 10 words plus the number of words of Service Data. The COMMREQ header (words 1–6) is not counted in the command length. Note: Service Data is in bytes, divide by 2 and round up for words. Service data length will vary depending on message executed; consult vendor documentation of the addressed server device. For this example: 12 words = Service Data is 3 bytes (rounded up to 2 Words) + 10 words command length.
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status Segment Select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
4	00009	(0009)	**Status Memory Offset:** COMMREQ status words start address minus 1. (%R10 for this example)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00001	(0001)	**Command Code:** Send Device Explicit command number (1)
8	00008	(0008)	**Reply Segment Select:** Memory type for the reply data. (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
9	00250	(00FA)	**Reply Memory Offset:** Offset within the memory type for the reply minus 1. Word offset for memory types: 8, 10, 12; byte offset for memory types: 16, 18, 20, 22. For this example, it is %R251.
10	00006	(0006)	**Reply Memory Size:** Maximum size required to hold the reply to the command: in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22) For command 1 the size must be 10 bytes (5 words) or more, or an error will be reported in the COMMREQ status word and the request will be ignored. Note: The size needed for the reply depends on the service used and the instance accessed. Consult the server device documentation. Add 10 bytes (5

Word	Dec	(Hex)	Description
			words) to the server reply data for the reply header. The reply memory size can be larger than the reply data of a particular message it must not be smaller.
11	00004	(0004)	**MAC ID:** of the device to send the message to (0 - 63). For this example, the VersaPoint Network Interface Unit uses MAC ID #4.
12	00002	(0002)	**Service Data Size:** Number of Service Data bytes being sent. This needs to be determined from the documentation of the DeviceNet server to which the message is being sent. For the example, 2 bytes = 1 attribute byte + 1 bytes data. Note: For service codes 0x10 or 0x0E the attribute byte is contained in the service data at byte zero.
13	00016	(0010)	**DeviceNet service code:** See the vendor documentation for the server device. In this example, the Service Code for the VersaPoint DeviceNet NIU is 0x10 (Set Attribute Single Service) to write data. Another service code often used is 0x0E (Get Attribute Single Service) to read data.
14	00010	(000A)	**Class/Object:** The object class to which this is requested. See the vendor documentation for the server device. For this example, the object class is 0x0A (Analog Input Point Object).
15	00001	(0001)	**Instance:** The specific instance of the object class to which this request is directed. See vendor documentation for the server device. For the example the instance represents which VersaPoint analog channel to set.
16	00001	(0001)	**Service Data Byte Offset:** If the offset is 0, then the service data is located immediately after this data word in memory (at word 17, see below). The value entered here is the number of bytes between this word and the beginning of the service data. For example, if the offset were 2, then two bytes would be "skipped" and the service data would begin at word 18.
17	1792	(00)	**Skipped byte(s):** This byte is skipped because the data byte offset in word 16 is a “1” for this example. Multiple bytes may be skipped. Data in skipped bytes is ignored.
		(07)	**Service Data byte 0, Attribute:** Attribute is used in service code 0x10 and 0x0E messages. The attribute is a one-byte field always at byte zero of the service data when used. The “Attribute” field is not used by other message services. This byte is the actual beginning of the service data since the data byte offset caused a one-byte skip. For the example attribute 7 is the VersaPoint, Analog Input Point Object, Range setting.
18	00003	(0003)	**Service Data:** Offset of the start of this data depends on entry for Service Data Byte Offset. Service data to is limited to 238 bytes maximum for command 1. For the example “Range” 3 is the vendor code for the VersaPoint Analog Input 4-20ma setting and the data type is USINT (2 bytes). Note: It is important to know the type of the data to properly calculate the length setting of word 1 and word 12 of the COMMREQ.
19 to end			**Service Data:** Additional service data as required by the message. In the example this is unused space.

Send Device Explicit (and Extended), Reply Data Format

Word	Description			
1	Command code that this data block is replying to. (1 or 7)			
2	Status of the explicit message. Bits 0 and 1 should both be 0.			
	bit 0	1 = Explicit message response truncated to fit in shared memory buffer. The configured size of the explicit buffer of the device is too small.		
	bit 1	1 = Explicit message response truncated to fit in Reply Memory. The reply buffer allocated by the COMMREQ is too small.		
	bits 2 - 15	Reserved, should be ignored.		
3	MAC ID of the device producing this reply.			
4	Number of reply data bytes consumed. Note: if allocated buffers are not large enough this value should indicate the actual size of the reply data. Allocate reply size at least 10 bytes (for reply words 1-5) larger than the service data.			
5	DeviceNet service code / internal result code. Values less than 0xFF: The service code low byte, in explicit message replies contains the same service that is returned on the DeviceNet network. Since the message is in reply to the explicit service issued by the COMMREQ, the high bit of the low byte is set to a 1. For example: GET_ATTRIBUTE_SINGLE is service code 0x0E. The DeviceNet response will have the high bit set: 0x8E SET_ATTRIBUTE_SINGLE is service code 0x10. With the high bit set on response: 0x90 DeviceNet errors use service code 0x14, and since errors are responses, the high bit will be set: 0x94. For example: GET_ATTRIBUTE_SINGLE: 0x0E, DeviceNet error response: 0x94 (With following bytes of main code and additional code)			
	Value	*Error*	*Value*	*Error*
	0x00 - 01	Reserved	0x12	Reserved
	0x02	Resource needed for the object to perform the requested service not available.	0x13	The service did not supply enough data to perform the requested service
	0x03 - 07	Reserved	0x14	Attribute specified in the request is not supported
	0x08	Requested service not implemented or not defined for the object class/instance	0x15	The service supplied more data than was expected
	0x09	invalid attribute data detected	0x16	The specified object does not exist in the device
	0x0A	Reserved	0x17	Reserved
	0x0B	Object is already in requested mode or state requested by the service	0x18	Attribute data of the object was not stored prior to the requested service
	0x0C	Object cannot perform the requested service in its current mode / state	0x19	Attribute data of this object not saved by the object
	0x0D	Reserved	0x1A - 1E	Reserved by DeviceNet
	0x0E	Request to modify a non-modifiable attribute was received	0x1F	Vendor specific error
	0x0F	Permission/privilege check failed	0x20	Invalid parameter
	0x10	Device's current mode or state prohibits the requested service	0x21 - CF	Reserved
	0x11	Data to be transmitted is larger than the allocated response buffer	0xD) - FF	Vendor specific object and class errors
	Values above 0xFF are internal Series 90-30 DeviceNet Master Module codes (see below).			
	0x0100	Explicit connection is not established		
	0x0101	Explicit body format cannot represent requested class. (i.e. class > 255 and connection body format is 8/8 or 8/16)		
	0x0102	Explicit body format cannot represent requested instance. (i.e. instance > 255 and connection body format is 8/8 or 8/16)		
	0x0103	Resources not available to send explicit message		
	0x0104 - FFFF	Reserved		
6 - end	Optional data as required by the service. The size of this data is indicated by word 4			

DeviceNet Master Modules, COMMREQ 4: Get Detailed Device Status

The controller can send COMMREQ 4 to an RX3i or Series 90-30 DeviceNet Master in the I/O Station, to read the following information about a device on the DeviceNet network. (This command does not generate a DeviceNet network message).

- whether the network device is included in the master's list of configured devices
- whether it is being scanned
- configuration error status (invalid id, device type, product code, I/O connections, etc)
- its connection 1 and connection 2 input states

Get Detailed Device Status, Example COMMREQ

The example COMMREQ below does the following:

- Gets the Device Status of the slave with MAC ID #4 from the DeviceNet Master Module.
- Returns the COMMREQ Status to %R10-%R13
- Returns the Device Status to %R251-%R260.

Word	*Dec*	*(Hex)*	*Description*
1	00005	(0005)	**Length of command Data Block:** For the Get Detailed Device Status COMMREQ, always 5
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
4	00009	(0009)	**Status memory offset:** COMMREQ status words address minus 1 (%R10)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00004	(0004)	**Command code:** Get Detailed Device Status command number 4
8	00008	(0008)	**Reply segment select:** Memory type for the reply data (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
9	00250	(00FA)	**Reply memory offset:** Offset within the memory type for the response minus 1. For this example %R251. (Word offset for memory types: 8, 10, 12; byte offset for memory types: 16, 18, 20, 22)
10	00009	(0009)	**Reply memory size:** Maximum size for the reply (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Maximum 2048 bytes. Note: For command 9 must be 18 bytes (9 words) or more, or an error will be returned in the COMMREQ status and the command will be ignored.
11	00004	(0004)	**MAC ID:** of the network device. For this example, 4.

Get Detailed Device Status, Reply Data Format

Upon receiving COMMREQ 4 from the PLC CPU, the DeviceNet Master Module generates a reply containing the status data it currently has stored for the specified MAC ID.

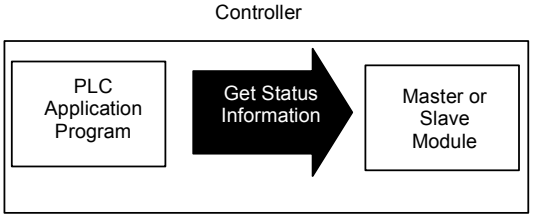
Word	***Description***			
1	Command number that this data block is replying to. (4)			
2 low byte	Status Code: Number indicating the status of the client connection to the device.			
	Status	*Meaning*	*Status*	*Meaning*
	0x 00	Device not in device list	0x 0D	Invalid I/O connection 1 input size
	0x 01	Device idle (not being scanned)	0x 0E	Error reading I/O connection 1 input size
	0x 02	Device being scanned	0x 0F	Invalid I/O connection 1 output size
	0x 03	Device timed-out	0x 10	Error reading I/O connection 1 output size
	0x 04	UCMM connection error	0x 11	Invalid I/O connection 2 input size
	0x 05	Master/Slave connection set is busy	0x 12	Error reading I/O connection 2 input size
	0x 06	Error allocating Master/Slave connection set	0x 13	Invalid I/O connection 2 output size
	0x 07	Invalid vendor id	0x 14	Error reading I/O connection 2 output size
	0x 08	Error reading vendor id	0x 15	Error setting I/O connection 1 packet rate
	0x 09	Invalid device type	0x 16	Error setting I/O connection 2 packet rate
	0x 0A	Error reading device type	0x 17	M/S connection set sync fault
	0x 0B	Invalid product code	0x 18	Error setting Production Inhibit Time
	0x 0C	Error reading product code	0x 19 - FF	Reserved
2 high byte	Status flags: Bits indicating the connection states of the slave's connection 1 and connection 2 inputs.			
	bits 0-4	Reserved, should be ignored		
	bit 5	1 = Input area 1 receive idle condition		
	bit 6	1 = Input area 2 receive idle condition		
	bit 7	Reserved, should be ignored		
3 to 9	Reserved, should be ignored			

DeviceNet Master Modules, COMMREQ 5: Get Status Information

The controller can send COMMREQ 5 to an RX3i or Series 90-30 DeviceNet Master in the I/O Station that is operating in server mode. The command will retrieve the following status information:

- whether the module is set up for slave operation (its network settings are configured)
- the module's output connection states
- whether the module has sent a DeviceNet explicit message (previously commanded by a Send Server Response COMMREQ).
- how the module's I/O messaging settings are configured.

This function is internal to the PLC system; it does not generate a DeviceNet message.



Get Detailed Server Status, COMMREQ Example

In this example, the application program sends a Get Detailed Served Status COMMREQ to a DeviceNet Master Module that is configured for slave operation.

Word	Dec	Hex	Description
1	00004	(0004)	**Length of command Data Block.** Always 4 words for this command.
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
4	00009	(0009)	**Status memory offset:** COMMREQ status words address minus 1 (%R10 for this example)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00005	(0005)	**Command code:** Get Detailed Server Status command (5)
8	00008	(0008)	**Reply segment select:** Memory type for the reply data. (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
9	00250	(00FA)	**Reply memory offset:** Offset within the memory type for the reply minus 1. For this example, it is %R251.
10	00009	(0009)	**Reply memory size:** Maximum size for the reply (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Maximum 2048 bytes. Note: For command 5 must be 18 bytes (9 words) or more, or an error is returned in the COMMREQ status and the command is ignored.

Get Detailed Server Status, Reply Data Format

The response to a Get Detailed Server Status COMMREQ supplies details of the module's configured Network Settings. It also shows whether the module has sent (on the DeviceNet network) a previously-commanded Send Server Explicit message.

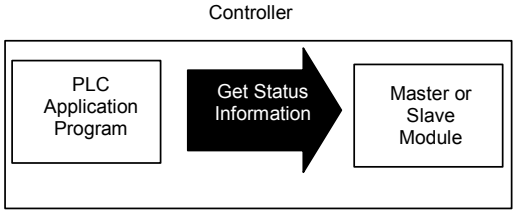
Word	*Dec/Bin*	*Hex*	*Description*	
1	00005	(0005)	Command number that this data block is replying to. (5)	
2 low byte	00000001	(01)	Indicates whether the module is set up for slave operation (Network Settings configured). For this example, the module is being scanned.	
			0x00	Idle (Group 2 master/slave connection is not allocated, the slave server is not active, no master is scanning).
			0x01	Active (Group 2 master/slave connection allocated, the slave server is active and is being scanned by a master device).
			0x02-0xFF	Reserved, these bits should be ignored.
2 high byte	Bits indicating the various connection states. In this example, the module's output 1 and 2 connections are both configured for receive idle, and it has an UCMM connection.			
	11100000	(E0)	*bits 0 - 4*	Reserved, these bits should be ignored.
			bit 5	1 = Output connection 1 receive idle condition
			bit 6	1 = Output connection 2 receive idle condition
			bit 7	1 = Group 3 UCMM connection(s) allocated.
3	00000	(0000)	Reserved, these bits should be ignored.	
4 low byte	Bits indicating explicit message status since the last Get Detailed Server Status. These bits are automatically cleared by this COMMREQ in preparation for the next call. For this example, the module has sent the explicit message response on the network.			
	00001	(0001)	*bit 0*	1 = Explicit response sent. Set when the scanner has submitted the explicit response from a Send Server Explicit message, for transmission on the network.
			bits 1 - 7: Reserved	
4 high byte	Bits showing the configured features of the module. For this example, the DeviceNet module slave server is set up for explicit messaging and polled I/O operation.			
	00000011	(03)	*bit 0*	1 = Explicit connection allocated
			bit 1	1 = Polled I/O connection allocated
			bit 2	1 = Bit-strobed I/O connection allocated
			bit 3	Not used
			bit 4	1 = Change of State I/O connection allocated
			bit 5	1 = Cyclic I/O connection allocated
			bit 6	1 = Acknowledge Suppress Enabled
			bit 7	Not used
5 to 9			Reserved, these bits should be ignored.	

DeviceNet Modules, COMMREQ 6: Get Input Status from a Device

The controller can send COMMREQ 6 to an RX3i or Series 90-30 DeviceNet Master in the I/O Station, to read the information that is normally mapped to the DeviceNet Master Module's 64 device status bits. The module responds to the command with the following information:

- the network activity status of each MAC ID on the network
- the module's own configured Network Settings.
- the module's current network status.
- the module's firmware ID.

The information read by this command comes from the module; this command does not generate a DeviceNet message.



Get Status Information, COMMREQ Example

- The example COMMREQ below does the following:
- Gets status information from the DeviceNet master.
- Returns the COMMREQ Status Words to %R10-%R13.
- Returns the Device Status to %R251-%R260.

Word	Dec	(Hex)	Description
1	00004	(0004)	**Length of command Data Block:** For Get Status Information, the length is 4 words (8 bytes).
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status words (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
4	00009	(0009)	**Status memory offset:** COMMREQ status words start address minus 1 (%R10 for this example)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00006	(0006)	**Command code:** Get Status Info command number (6)
8	00008	(0008)	**Reply segment select:** Memory type for the reply data. (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
9	00250	(00FA)	**Reply memory offset:** Offset within the memory type for the reply (0-based). For this example, it is %R251. (Word offset for memory types: 8, 10, 12; byte offset for memory types: 16, 18, 20, 22).
10	00008	(0008)	**Reply memory size:** Maximum size for the reply (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Maximum 2048 bytes. Note: For command 6 must be16 bytes (8 words) or more, or an error will be returned in the COMMREQ status and the command will be ignored.

Get Status Information, Reply Data Format

Word	Description
1	Command code that this data block is replying to. (6)
2 - 5	**Device Status**. Each bit corresponds to an individual device MAC ID. The state of that bit indicates the device's status: 0 = Device is not active (not configured, faulted, etc…), 1 = Device is active, being scanned. For the master's own MAC ID, the status bit is always 0.

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
byte 0	7	6	5	4	3	2	1	0
byte 1	15	14	13	12	11	10	9	8
byte 2	23	22	21	20	19	18	17	16
byte 3	31	30	29	28	27	26	25	24
byte 4	39	38	37	36	35	34	33	32
byte 5	47	46	45	44	43	42	41	40
byte 6	55	54	53	52	51	50	49	48
byte 7	63	62	61	60	59	58	57	56

Server Status

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
byte 0	res.	AKS	CYC	COS	res.	ST	P	EX
byte 1	reserved			SERA	IDLE2	IDLE1	G3	

Group 2 only I/O connections	AKS	Acknowledge suppress enabled
	CYC	Cyclic I/O connection allocated
	COS	Change-of-state I/O connection allocated
	ST	Bit Strobed I/O connection allocated
	P	Polled I/O connection allocated
Group 2 Explicit Connections	EX	Explicit connection allocated
Group 3 Connection	G3	At least one Group 3 (UCMM) connection allocated
Status Bits	IDLE1	Output area 1 receive idle status bit.
	IDLE2	Output area 2 receive idle status bit
	SERA	Server Explicit Request Available. Use Receive server explicit command to retrieve the request

CAN Network Status.

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
byte 0	ML	RO	TO	TA	A	BO	BW	OL
byte 1	SA	O5	O2	O1	RE	reserved	BP	ER

Application Specific Flags

SA	Scanner Active (at least one connection established)
O5	Online at 500 Kbaud
O2	Online at 250 Kbaud
O1	Online at 125 Kbaud
RE	Firmware is resetting so DeviceNet I/O data is not valid

Common Flags

BP	Bus power present (zero if power sense not supported)
ER	CAN communication error
ML	Message lost (CAN controller / receive ISR)
RO	Receive buffer overrun (host app. too slow emptying receive queue)
TO	Transmit failed due to timeout (flooded network)
TA	Transmit failed due to ack error (no other nodes connected)
A	Network activity detected (messages received or transmitted)
BO	Bus off (this node has been disconnected due to excessive errors)
BW	Bus warning (this node is experiencing a large number of errors)
OL	Online, CAN interface has been initialized

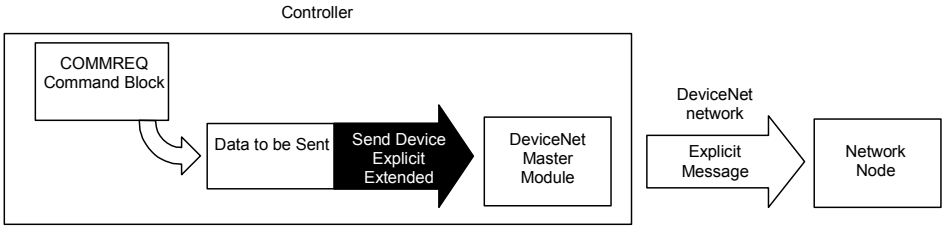
8	Firmware ID, Minor revision:	In BCD four hex digits. For example, revision 1.10 = 01 10 hex.
	Firmware ID, Major revision:	See above.

DeviceNet Modules COMMREQ 7: Send Device Explicit Extended

The controller can send COMMREQ 7 to an RX3i or Series 90-30 DeviceNet Master in the I/O Station, to send more than 238 bytes of data on the DeviceNet network. This command can also be used to send data that should be mapped to a separate data memory area in the PLC as explained below. The reply is limited to 2048 bytes maximum. This command is similar to COMMREQ 1, Send DeviceNet Explicit. See the description of DeviceNet COMMREQ 1 for more information.

Both of these COMMREQs command the DeviceNet Master Module to send a DeviceNet explicit message on the network.

For Send Device Explicit Extended, the data to be sent is located in a separate memory area, which is indicated by a pointer in the COMMREQ command block. This makes it possible to store and send more data or to have the data separate from the command memory.



The addressed device must be configured for an explicit message connection in the controller configuration of the DeviceNet Master Module and sufficient buffer memory must be configured to contain the largest message produced by the COMMREQ or the largest reply produced by the device. If the device was not configured for explicit messaging or if the number of bytes configured is not enough for the command, the COMMREQ fails with a code of 8 in the COMMREQ Status Word.

Send Device Explicit Extended, COMMREQ Example

The Send Device Explicit Extended COMMREQ command block contains a pointer to the data to be sent in the explicit message. The programmer can use this functionality to point to different stored messages without recalculating command length each time. Command 7 additionally avoids the 238-byte service data limit of command 1 by increasing the maximum size for the service data.

This example COMMREQ sends an explicit message to Mac ID 4 (a GE Fanuc S2K DeviceNet Motion Controller), returns the COMMREQ Status Words to %R10-%R13, and sets (writes) an array of data (32 DINT) variables to the S2K integer memory (VI registers).

	Word	*Value*	*Description*
COMMREQ Header	1	00007	**Command Length:** Length of the command block for the Send Device Explicit Extended command . Always 7 words.
	2	00000	Always 0 (no-wait mode request)
	3	00008	**Status Memory Select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
	4	00009	**Status Memory Offset:** COMMREQ status words start address minus 1 (%R10 for this example)
	5		Reserved
	6		Reserved
COMMREQ Command	7	00007	**Command Code:** Send Device Explicit Extended command number (7)
	8	00008	**Reply Segment Select:** Memory type for the reply data. (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
	9	00250	**Reply Memory Offset:** Offset within the memory type for the reply minus 1. For this example, it is %R251. (Word offset for memory types: 8, 10, 12; byte offset for memory types: 16, 18, 20, 22).
	10	00005	**Reply Memory Size:** Maximum size required to hold the reply for the command: (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Add 10 bytes to expected reply size. Note: must be 10 bytes (5 words) or more, or an error will be reported in the COMMREQ status word and the request will be ignored. Actual length needed will vary depending on which message is sent; consult vendor information for the target device. Maximum 2048 bytes.
	11	00008	**Data Segment Select:** Memory type for the service data. (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
	12	00300	**Data Memory Offset:** Offset within the specified memory type for the service data start address minus 1. (Word offset for memory types: 8, 10, 12; byte offset for memory types: 16, 18, 20, 22). For this example, it is %R301.
	13	00071	**Data Memory Size:** Size of the data to be sent, in units of the selected type (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Must be large enough to contain the entire explicit data block. The entire data block calculation is; the service data header 12 bytes (6 words) + skipped bytes (specified in word 6 of the service data header + the service data. Note: It is important to know the type of the data used in the service to calculate the minimum length accurately. The attribute byte when used is always byte 0 of the service data and must be added to the data length. Round size up as needed. For this example we have 71 words; 6 service data header words + 1 skipped byte + 1 attribute byte + 32 DINT data (64 words) service data.

Send Device Explicit Extended, Data Block Format

The following data must be placed in the PLC memory location specified in the command by the data memory offset.

One use of the data byte offset (see below) would be to “point “ to a start location within a large array of data in the PLC memory. In the following example the data byte offset is used to maintain word boundary location of the data within the PLC memory even though we require the service data to contain the attribute value.

	Word	*(Hex)*	*Description*
Service Data Header	1	(0004)	**MAC ID:** Address of the device to send the message to (0 - 63).
	2	(0081)	**Number of Service Data bytes:** This needs to be determined from the vendor documentation of the DeviceNet server to which the message is being executed. For the example Service Data 0x81 (129 bytes) = 1 byte attribute + 128 bytes (32 DINT) of data.
	3	(0010)	**DeviceNet service code:** See the vendor documentation for the server device. In this example, the Service is 0x10 (Set Attribute Single Service) to write data.
	4	(0004)	**Object Class:** to which this is requested. See the documentation for the server device. For this example, the object class is 0x04 (S2K Assembly Object).
	5	(0300)	**Instance:** of the object class to which this request is directed. See documentation for the server. In this example Instance 768 decimal (0300h) points to VI001 in the S2K as the first of 32 DINT variables to write.
	6	(0001)	**Data Byte Offset:** The number of bytes between this word and the beginning of the service data to be sent. If the offset is 0, the service data is located immediately after this data word (at word 7, see below). For example, if the offset were 2, then two bytes would be "skipped" and the data would begin at word 8.
	7	(00)	**LSB: Skipped** - Least significant byte “skipped” because of setting in word 6.
Service Data		(03)	**Service Data Byte 0, Attribute** – An attribute is used in service 0x10 and 0x0E messages. See documentation of targeted server device for meaning of specific attributes. Since word 6 “skipped” a byte this is the actual beginning of the service data. Locate data for messages without an attribute to start data here. May be at a different location depending on the value of word 6.
	8, 9	DINT	**Service Data:** May be located at a different offset based on word 6. Using the offset in word 6 allowed, in this example, the DINT data to be aligned on a word boundary.
	10 to end	-	**Service Data:** For this example the end of the service data is located at word 71 [6 header words + 1 skipped byte + 1 attribute byte + 64 data words].

Send Device Explicit Extended, Reply Data Format

Reply Data format for the Send Device Explicit Extended COMMREQ is the same as that shown for Send Device Explicit (DeviceNet COMMREQ 1).

DeviceNet Master Modules, COMMREQ 9: Read Module Header

The controller can send COMMREQ 9 to an RX3i or Series 90-30 DeviceNet Master in the I/O Station to read the following information:

- Module Type, Module ID, Module revision
- CAN Kernel identification and revision
- DeviceNet serial number
- Error codes for any existing fault
- CAN Network status

Upon detecting an error, the PLC application program can send this COMMREQ to the module. Unless the error prevents normal backplane operation, the module returns information about the fault in the reply data. Error codes are listed in this section. This command reads data from the DeviceNet module's internal memory; no message is sent on the DeviceNet network.

Read Module Header, COMMREQ Example

The example COMMREQ below does the following:

- Gets the Module Header Data
- Returns the COMMREQ Status Words to %R10-%R13
- Returns the Device Status to %R251-%R283.

Word	*Dec*	*(Hex)*	*Description*
1	00004	(0004)	**Length of command Block:** Always 8 bytes (4 words) for command 9.
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type for COMMREQ status words (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
4	00009	(0009)	**Status memory offset:** status words starting address minus 1. (%R10 for this example)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00009	(0009)	**Command Code:** Read Module Header; command 9
8	00008	(0008)	**Reply segment select:** Memory type for the reply data (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M)
9	00250	(00FA)	**Reply memory offset:** Offset within the memory type for the response minus 1. (%R251 for this example).
10	00065	(0041)	**Reply memory size:** Maximum size for the reply (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Maximum 2048 bytes. Note: For command 9 must be 130 bytes (65 words) or more, or an error will be returned in the COMMREQ status and the command will be ignored.

Read Module Header, Reply Data Format

<table>
<tr><th>Word</th><th>Description</th></tr>
<tr><td>1</td><td>Command Code. Echo of Command Code that this data block is replying to (0x0009)</td></tr>
<tr><td>2</td><td>Module Type. Contains "DN" (0x444E) or "ER" (0x4552) if a fatal error is detected</td></tr>
<tr><td>3</td><td>Window size: Indicates host interface window size. 0 = 16K, 1 = 32K, 2 = 64K, 3=128K</td></tr>
<tr><td>4</td><td>Reserved</td></tr>
<tr><td>5</td><td>Kernel identification. 0x0001 = CAN 2.0A kernel</td></tr>
<tr><td>6</td><td>Kernel revision</td></tr>
<tr><td>7</td><td>Module ID, 0x0017</td></tr>
<tr><td>8</td><td>Module revision in binary coded decimal (BCD), 4 hex digits XX.XX
(i.e. rev 1.0 = 0x0100, rev 1.10 = 0x0110)</td></tr>
<tr><td>9,10</td><td>DeviceNet serial number</td></tr>
<tr><td>11 - 18</td><td>Module type</td></tr>
<tr><td>19 - 22</td><td>Module serial number (i.e. "9409001")</td></tr>
<tr><td>23, 24</td><td>Reserved</td></tr>
<tr><td>25</td><td>Main Application Error Code. See the error code listings on the following pages.</td></tr>
<tr><td>26</td><td>CAN Network Status word
<table>
<tr><th></th><th>bit 7</th><th>bit 6</th><th>bit 5</th><th>bit 4</th><th>bit 3</th><th>bit 2</th><th>bit 1</th><th>bit 0</th></tr>
<tr><td>byte 0</td><td>ML</td><td>RO</td><td>TO</td><td>TA</td><td>A</td><td>BO</td><td>BW</td><td>OL</td></tr>
<tr><td>byte 1</td><td>SA</td><td>O5</td><td>O2</td><td>O1</td><td>RE</td><td></td><td>BP</td><td>ER</td></tr>
</table>
Application –Specific Flags
SA Scanner Active (at least one connection established)
O5 Online at 500 Kbaud
O2 Online at 250 Kbaud
O1 Online at 125 Kbaud
RE Firmware is performing DeviceNet reset, I/O data is not valid
Common Flags
BP Bus power present (zero if power sense not supported)
ER CAN communication error
ML Message lost (CAN controller / receive ISR)
RO Receive buffer overrun (host app. too slow emptying receive queue)
TO Transmit failed due to timeout (flooded network)
TA Transmit failed due to ack error (no other nodes connected)
A Network activity detected (messages received or transmitted)
BO Bus off (this node has been disconnected due to excessive errors)
BW Bus warning (this node is experiencing a large number of errors)
OL Online, CAN interface has been initialized</td></tr>
<tr><td>27</td><td>CAN transmit counter. Incremented when messages are submitted to the CAN controller.</td></tr>
<tr><td>28</td><td>CAN acknowledgment error counter. Increments if a transmit message is terminated due to lack of acknowledgment from other stations. When this counter is incremented, the CAN transmit counter (word 27) is decremented to compensate for a message not actually transmitted.</td></tr>
<tr><td>29</td><td>CAN receive counter. Increments when messages are received. Messages that fail the receive filter still increment this counter.</td></tr>
<tr><td>30</td><td>CAN communication error counter. Increments if a CAN frame error is detected.</td></tr>
<tr><td>31</td><td>CAN lost messages counter. Increments if a CAN message is received before the previous message is placed into the receive queue.</td></tr>
<tr><td>32</td><td>CAN receive queue overrun counter. Increments if a CAN message is lost due to a full receive queue.</td></tr>
<tr><td>33</td><td>Additional Application Error Code. See the error code listings on the following pages.</td></tr>
<tr><td>34 - 63</td><td>When Module Type in word 2 is "DN", contains the module identification string. For example:
“DeviceNet Module 1.00.00\n(C) 2002 GE Fanuc Automation.”
The format is: major rev.minor rev.build When Module Type is "ER”, contains the kernel error string.</td></tr>
<tr><td>64</td><td>Major Tick Interval (equivalent of system time base)</td></tr>
<tr><td>65</td><td>Number of minor ticks per major tick interval</td></tr>
</table>

Runtime Error Codes in the Module Header

After a runtime error [word 2 of the Read module header, reply data = "DN" (0x444E)], the Main Error Code [word 25] and Additional Code [word 33] fields of the reply data describe the runtime error. A zero value in the main error code word indicates no error.

Category Name	*Main Code*	*Error Name*	*Additional Code*	*Description*
No Error	0x0000	-	0x0000	Zero in Main Code indicates no error
Config File Error	0x0001	Unknown Version	0x0001	Incorrect / unsupported version
Init File Error	0x0002	Unknown Version	0x0001	Incorrect / unsupported version
		Unknown Header Id	0x0002	Init file's header ID not recognized or invalid
		Invalid Block Definition Count	0x0003	Block's definition count is invalid
		Unknown Block Type	0x0004	Block type not recognized
		Invalid Block Checksum	0x0005	Block's checksum is invalid
		Invalid Shared Memory Offset	0x0006	Shared memory offset not in the 0x1000 to 0x3FFF range.
		Unknown9030IoType	0x0007	I/O Type code not recognized
		Invalid Mac Id	0x0008	Mac Id in the Data Pointer list not in the range of 0 to 64, inclusive, or 255.
		Unknown Memory Area Type	0x0009	Memory area type code not recognized
		Invalid Block Size	0x000A	Size of block in INIT file did not match what was expected, or larger than the maximum size supported by the firmware.
		Invalid Block Offset	0x000B	Offset in Block Definition Record points beyond maximum INIT file size.
		Duplicate Block	0x000C	Block of with same type code already exists in INIT file.
		Data Pointer Out Of Range	0x000D	Data pointer refers to a location outside of shared memory.
		Missing Block	0x000E	One or more of required blocks 1, 3, and 4 are missing from the INIT file.
Add Device Error	0x0003	Duplicate Device	0x0005	Device already in scan list.
		Invalid Shared Memory Offset	0x000D	Shared memory offset not in the 0x1000 to 0x3FFF range.
		Invalid Connection Flags	0x000F	The combination of bits in the Flags field is invalid.
		Invalid Explicit Buffer Size	0x0010	Explicit buffer size is invalid.
		Invalid Strobe Buffer Size	0x0011	Strobe buffer size is invalid. Note that the output size must be 1.
		Invalid Path Buffer	0x0012	Path buffer is not initialized.
Online Error	0x0004	Invalid Mac Id	0x0002	MacId in the Server Config block is not in the range of 0 to 63 inclusive.
		Invalid Baud Rate	0x0003	Baud rate not set to 0, 1, or 2 (i.e. 125K, 250K, or 500K)
		Duplicate Mac Id Failure	0x0004	DUP MacId check failed while attempting to go online.
		Bus Not Offline	0x0009	Bus is already online. Internal firmware error; report to manufacturer.
		Bus Off	0x000E	Bus fault detected.
		Invalid Connection Flags	0x000F	Combination of bits in the Flags field of the Server Config block is invalid.
		Invalid Explicit Buffer Size	0x0010	Explicit buffer size is invalid.
		Invalid Strobe Buffer Size	0x0011	Strobe buffer size is invalid. Note that the output size must be 1.
		Invalid Path Buffer	0x0012	Path buffer is not initialized.
		Ack Fault	0x0013	No CAN acknowledge received during Duplicate MacId Sequence.
Start Scan	0x0005	Bus Offline	0x0007	Bus is not online yet
		Scanner Running	0x000A	Scanner is already started
		Scanner Stopping	0x000C	Scanner is stopping

Fatal Error Codes in the Module Header

If the module is capable of executing and reporting an error after a fatal error, the Module Type field of the Module Header data contains the value “ER” (0x4552). For specific errors listed below that do not have any source code information available, the value 0xFFFF is placed in the Main Code and one of the listed error numbers is placed in the Additional Code field. Fatal errors that can report source code location store the source file number in Main Code and the source line number in Additional Code.

Additional Error Code	*Error*	*Description*
1	RAM data test failed	An error occurred during testing of the RAM data bus.
2	RAM address test failed	An error occurred during testing of the RAM address bus.
3	RAM A16 address test failed	An error occurred during testing of the RAM A16 signal.
4	RAM A17 address test failed	An error occurred during testing of the RAM A17 signal.
5	Module checksum is invalid	The most likely cause of this error is an undetected memory failure. If this error occurs with more than one application module, the module should be returned for repair.
6	CAN reset flag failed to clear	An error occurred testing the CAN controller.
7	CAN data test failed	An error occurred testing the CAN controller data bus.
8	CAN address test failed	An error occurred testing the CAN controller address bus.
9	Invalid NVRAM data	The module's non-volatile memory contains invalid information.
10	Execution permission denied	This module has not been configured to execute the application module.
11	Application initialization error	An error occurred initializing the application module.
12	Unknown application initialization code	An error occurred initializing the application module.
13	Application terminated	The application module terminated (abnormal condition).
14	Application fatal error	A fatal runtime error occurred.
15 - 21	XXX interrupt	An unexpected interrupt was detected.
22	Event queue overflow	This error should be reported to the vendor of the application module. Make note of the circumstances that caused this error.
23	Nested user timer interrupt	
24	Invalid CAN interrupt	
25	Nested system timer interrupt	
26	Imperfect interrupt	This error should be reported to the vendor of the application module. This error is caused by an incorrectly generated interrupt from the host bus adapter to the module.
27	Stack Overflow	This error should be reported to the vendor of the application module.
99	Unexpected condition encountered	A fatal runtime error occurred.

COMMREQs for Genius Bus Controller Modules

The controller can use a Remote COMMREQ Call to send the following COMMREQs to a PACSystems RX3i Genius Bus Controller module IC694BEM331 or Series 90-30 Genius Bus Controller module IC693BEM331 in the I/O Station:

8	Enable/Disable Outputs Command
13	Dequeue Datagram Command
14	Send Datagram Command: switch BSM clear fault clear all faults assign monitor read diagnostic
15	Request Datagram Reply Command

Genius Bus Controller Modules, COMMREQ 8: Enable/Disable Outputs

COMMREQ 8 can be used to enable or disable outputs on one device or on all devices on the Bus Controller's Genius bus.

Command Block for the Enable/Disable Outputs Command

Word	*Dec*	*(Hex)*	*Description*
1	00003	(0003)	**Length of command Data Block:** For the Enable/Disable Outputs command, always 3
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 70 = I, 72 = Q)
4	00009	(0009)	**Status memory offset:** COMMREQ status words address minus 1 (%R10)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00008	(0008)	**Command code:** Enable/Disable Outputs is command number 8.
8			**Device Number:** Enter 0-31 to enable or disable outputs to one block. To enable or disable outputs to ALL devices on the bus, enter the number 255.
9			**Enable/Disable Command:** To disable outputs to the device(s) specified in address +7, enter 0. To enable outputs, enter 1.

This COMMREQ overrides the configuration parameter **outputs enable/disable at start**. For example, if outputs were initially disabled to all blocks during configuration, this COMMREQ could be used to enable outputs to specific devices or to all devices.

Genius Bus Controller Modules, COMMREQ 13: Dequeue Datagram

COMMREQ 13 commands a Genius Bus Controller to transfer incoming datagrams to the CPU. In an RX3i Ethernet NIU I/O Station, this command would be used to retrieve the reply to a Read Diagnostics datagram that was sent using COMMREQ 14, Send Datagram. COMMREQ 13 is not needed if the Read Diagnostics datagram has been sent using COMMREQ 15, Request Datagram Reply. For that command, the reply is returned automatically.

Command Block for the Dequeue Datagram Command

Word	Dec	(Hex)	Description
1	00007	(0007)	**Length of command Data Block:** For the Dequeue Datagram command, always 7.
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 70 = I, 72 = Q)
4	00009	(0009)	**Status memory offset:** COMMREQ status words address minus 1 (%R10)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00013	(000D)	**Command code:** Dequeue Datagram is command number 13
8			**Maximum data memory length:** Enter bit or word value (depends on the memory type selected below). This entry tells the CPU how much memory will be needed to store all the data. If the length of data returned by the device exceeds this length, the GBC writes as much data as possible to the PLC CPU and returns a data error to the COMMREQ status location.
9			**Memory Type:** Enter the number that represents the location where the GBC will place the data in the CPU: 70 (%I), 72 (%Q), 8 (%R), 10 (%AI), or 12 (%AQ).
10 – 13			**Not used.**

Number of Dequeue Datagram Commands Needed

One Dequeue Datagram command is needed for each incoming datagram. If multiple incoming datagrams are expected during one CPU sweep, it will be necessary to place multiple Dequeue Datagram commands in the program to assure their efficient transfer to the CPU.

The number of Dequeue Datagram commands needed depends on whether the datagrams have been sent using Normal or High Priority, and the relative lengths of the CPU sweep time and the scan time of the bus, as explained below.

If the Bus Scan Time is Greater than the CPU Sweep Time

If all datagrams on the bus are sent with Normal Priority, there is a limit of one incoming datagram per CPU sweep. Therefore, only one Dequeue Datagram command per sweep will be needed to handle incoming datagrams. If all datagrams on the bus are sent with High Priority, the Genius Bus Controller can potentially receive one Datagram from each transmitting device during a scan. The program should include the same number of Dequeue Datagram commands as incoming datagrams.

If the Bus Scan Time is Less than the CPU Sweep Time

If the bus scan time is significantly shorter than the CPU sweep time, you can estimate the number of Dequeue Datagram commands that must be sent to the GBC to accommodate incoming datagrams on that bus. First, determine how many scans can occur in one CPU sweep. For example, if the bus scan were 20mS and the CPU sweep were 90mS, the ratio between them would be 4.5 to 1. This should be rounded upward to 5.

This is the maximum number of normal-priority datagrams that might be received in a single CPU sweep. Plan to have the same number of Dequeue Datagram commands to that Genius Bus Controller in the program to handle the incoming datagrams.

For high-priority datagrams, multiply the number found above by the total number of devices on the bus that might send a high-priority datagram to the Bus Controller in one bus scan. This is the total number of incoming datagrams from that bus the program might have to handle in a single CPU sweep. Plan on this number of Dequeue Datagram commands to the Bus Controller.

Additional Logic for Incoming Datagrams

The Genius Bus Controller can place up to 16 datagrams into an internal queue. These include any unsolicited reply–type datagrams. Program logic in the controller should be used to assure that no datagrams are accidentally written over. This might be done by copying each datagram to another memory location, or by changing the data memory location specified in the Command Block after each incoming datagram is received.

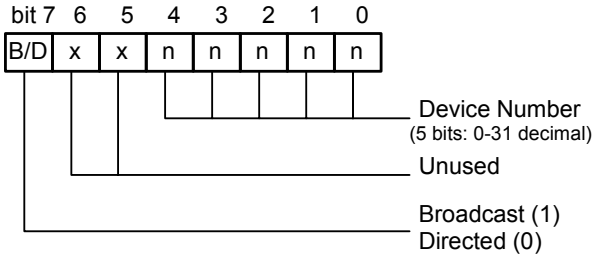
Note that the Dequeue Datagram queue is operated as a first–in–first–out (FIFO) queue. Specific datagrams within the queue cannot be dequeued without first dequeueing datagrams received earlier.

Format of Returned Data

The Dequeue Datagram returns data in the following format.

Location	*High Byte*	*Low Byte*
Memory Address	Data Length	Status byte
Memory address +1	Subfunction code	Function code
Memory address +2	Data byte 2	Data byte 1
.	.	.
.	.	.
.	.	.
Memory address +69	Data byte 134	Data byte 133

Returned Data items are explained below.

Status Byte	The status byte reports the Device Number of the device that sent the datagram. It also indicates whether the message was broadcast or directed by the other device. 
Data Length	The number (0 to 134) of data bytes after the subfunction code.
Function Code	The function code of the received message: 0 to 111 decimal or 0 to 6F hex.
Subfunction Code	The subfunction code of the received message: 0 to 255 decimal or 0 to FF hex.

Genius Bus Controllers, COMMREQ 14: Send Datagram Command

The controller can use COMMREQ 14 (Send Datagram) to send the following datagrams to an RX3i or Series 90-30 Genius Bus Controller in an Ethernet NIU I/O Station:

- Assign Monitor
- Read Diagnostics
- Clear Circuit Faults
- Clear All Circuit Faults
- Switch BSM

If COMMREQ 14 is used to send a Read Diagnostics datagram, which has a reply, then COMMREQ 13 (Dequeue Datagram) must be used to obtain the reply from the Bus Controller. It is easier to use COMMREQ 15, Request Datagram Reply to send a Read Diagnostics datagram.

Command Block for the Send Datagram Command

Word	*Dec*	*(Hex)*	*Description*	
1	00007	(0007)	**Length of command Data Block:** 6 - 70. Enter the number of words from Word 7 to the end of the data block.	
2	00000	(0000)	Always 0 (no-wait mode request)	
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 70 = I, 72 = Q)	
4	00009	(0009)	**Status memory offset:** COMMREQ status words address minus 1 (%R10)	
5	00000	(0000)	Reserved	
6	00000	(0000)	Reserved	
7	00014	(000E)	**Command code:** Send Datagram is command number 14	
8			**Device Number of device to receive the message:** Enter 0-31, or 255 to broadcast the message.	
9	0032	0020	**Function Code**.	
10			**Subfunction Code:** see below	
			Subfunction Code(hex)	*Datagram*
			05	Assign Monitor
			08	Read Diagnostics
			12	Clear Circuit Faults
			13	Clear All Circuit Faults
			1C	Switch BSM
11			**Priority:** 0 for normal priority, or 1 for high priority.	
12			**Datagram length in bytes:** Enter the actual length of the Datagram, beginning at word 13.	
13 to end			**Datagram content:** Enter the entire datagram as part of the Command Block. The *Genius I/O System User's Manual* shows datagram structures.	

If the Genius bus is used for I/O block control, normal-priority datagrams are recommended to allow other messages such as fault reports to get through. If there are I/O blocks on the bus, use high priority only if the datagram transmission cannot be delayed.

The application program should include logic that verifies successful completion of earlier datagrams before requesting new ones.

Genius Bus Controllers, COMMREQ 15: Request Datagram Reply

The controller can use COMMREQ 15 to send a Read Diagnostics datagram to an RX3i or Series 90-30 Genius Bus Controller in the I/O Station. The Genius Bus Controller automatically transfers replies to COMMREQ 15 to the CPU, so no separate Dequeue Datagram command is needed to obtain the diagnostics data. (A Read Diagnostics datagram could also be sent using COMMREQ 14, Send Datagram, then COMMREQ 13, Dequeue Datagram).

Command Block for the Request Datagram Reply Command

Word	*Dec*	*(Hex)*	*Description*
1			**Length of command Data Block:** 10 - 78. Enter the number of words from Word 7 to the end of the data block.
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 70 = I, 72 = Q)
4	00009	(0009)	**Status memory offset:** COMMREQ status words address minus 1 (%R10)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00015	(000F)	**Command code:** Request Datagram Reply is command number 15
8			**Device Number of device to receive the message:** Enter 0-31.
9	0032	0020	**Function Code**.
10	0008	0008	**Subfunction Code:** Read Diagnostics is 8.
11			**Priority:** 0 for normal priority, or 1 for high priority.
12			**Datagram length in bytes:** Enter the actual length of the Datagram, beginning at word 17.
13	0009	0009	**Subfunction Code of the Reply:** Read Diagnostics Reply is 9.
14			**Memory Type for the Reply:** 8 (%R), 10 (%AI), or 12 (%AQ)
15			**Memory Offset for the Reply:** Starting address within this memory type.
16			**Maximum Data Memory Length Needed:** Enter a value in words. The length depends on the message and device type. Message formats are shown in the *Genius I/O System User's Manua*l. When all the data has been received, the Bus Controller transfers it to the CPU and sets the COMMREQ status to 4 (Done).If the length of the memory is smaller than the amount of reply data received, the extra portion of the data will be lost, and a data error (16) will be returned to the status location.
17 to end			**Datagram Content:** Enter the entire datagram as shown in the *Genius I/O System User's Manual.*

Returned data format is the same as for the Dequeue Datagram.

COMMREQs for RX3i Analog Modules with HART Communications

The controller can use a Remote COMMREQ Call to send the following COMMREQs to a PACSystems RX3i Analog Module with HART Communications, IC695ALG626, ALG628, or ALG728 in the I/O Station:

1	Get HART Device Information
2	Send HART Pass-Thru Command

RX3i Analog Modules with HART: COMMREQ 1, Get HART Device Information

The controller can send COMMREQ 1 to an RX3i Analog module with HART Communications to read information about an installed HART device.

Get HART Device Information, COMMREQ 1 Command Block

Word	Dec	(Hex)	Description
1	0008	(0008)	**Length of command Data Block:** in words beginning at Word 7.
2	0000	(0000)	Always 0 (no-wait mode request)
3			**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 70 = I (bit), 72 = Q (bit), 196 = W)
4			**Status memory offset:** COMMREQ status words address minus 1 (%R10)
5	0000	(0000)	Reserved
6	0000	(0000)	Reserved
7	0001	(0001)	**Command code**: for the COMMREQ to be executed. Get HART Device Information = 1.
8	0001	(0001)	**Number of Response Reference areas:** that follow (does not include COMMREQ status word). Always 1.
9			**Memory type for the reply data:** (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 20 = T, 22 = M, 196 = W) (Words 9—12 specify the starting address where the response will be written.)
10	0000	(0000)	**Bit Offset:** (must be 0 for all requests).
11			**0-based offset:** (low word).Starting address to which the response will be written. The offset from the beginning of PLC memory for the memory type specified in Word 9. This offset is in bytes or words depending on the memory type specified. Valid ranges of values depend on the PLC's memory ranges. Example: If Words 9 and 11 contain values of 8 and 250 respectively, the response will be written to %R251.
12			**0-based offset:** (high word). Value = 0 for most memory types. High word is non-zero only on if %W memory is used.
13	Words: 90	(005A)	**Maximum size of response area:**. Must be 90 if word memory type is used; 180 if discrete memory type is used.
	Bytes: 180	(00B4)	
14			**Channel Number**: 1-16 (valid range depends on module channel count and single-ended versus differential mode)

COMMREQ Status Word

Content of the COMMREQ status word for this command is shown later in this chapter.

Get HART Device Information, COMMREQ 1: Reply Data Format

The response to a Get HART Device Information COMMREQ is written to the PLC memory location specified in words 9-12 of the COMMREQ.

Byte	*Name*	*Description*
1, 2	Command Code	Echo of Command code. (0x0001)
3, 4	Channel Number	Echo of Channel Number
5-8	HART Primary Variable	CMD#3, Bytes 5-8. Type: REAL
9-12	HART Secondary Variable	CMD#3, Bytes 10-13 Type: REAL
13-16	HART Tertiary Variable	CMD#3, Bytes 15-18. Type: REAL
17-20	HART Fourth Variable	CMD#3, Bytes 20-23. Type: REAL
21-24	Slot 0 value	CMD#33, Bytes 2-5. Type: BYTE
25-28	Slot 1 value	CMD#33, Bytes 8-11. Type: BYTE
29-32	Slot 2 value	CMD#33, Bytes 14-17. Type: BYTE
33-36	Slot 3 value	CMD#33, Bytes 20-23. Type: BYTE
37	HART communication status byte from the last HART command response.	
38	HART device status byte from the last HART command response.	
39-40	Spare for alignment.	Type: BYTE
41	HART device Manufacturer ID. CMD#0, Byte 1	Type: BYTE
42	HART device type code. CMD#0, Byte 2	Type: BYTE
43	Minimum number of preambles device requires	CMD#0, Byte 3. Type: BYTE
44	HART Universal command code	CMD#0, Byte 4. Type: BYTE
45	HART Transmitter specific revision	CMD#0, Byte 5 Type: BYTE
46	HART device software revision number	CMD#0, Byte 6 Type: BYTE
7	HART device hardware revision number	CMD#0, Byte 7 Type: BYTE
48	HART flags	CMD#0, Byte 8 Type: BYTE
49-52	HART device ID number	CMD#0, Byte 9-11 Type: 4 BYTEs
53-60	8 character device tag.	CMD#13, Type: 8 BYTEs. Bytes 0-5 are unpacked ASCII
61-76	Device Descriptor	CMD#13, TYPE: 16 BYTEs. Bytes 6-17 are unpacked ASCII
77	HART Primary Variable Units	CMD#3, Byte 4. Type: BYTE
78	HART Secondary Variable Units	CMD#3, Byte 9, 0 if not present. Type: BYTE
79	HART Tertiary Variable Units	CMD#3, Byte 14, 0 if not present. Type: BYTE
80	HART Fourth Variable Units	CMD#3, Byte 19, 0 if not present. Type: BYTE
81	HART Primary Variable Code	CMD#50, Byte 0 Type: BYTE
82	HART Secondary Variable Code	CMD#50, Byte 1 Type: BYTE
83	HART Tertiary Variable Code	CMD#50, Byte 2 Type: BYTE
84	HART Fourth Variable Code	CMD#50, Byte 3 Type: BYTE
85	Units code for range parameter	CMD#15, Byte 2 Type: BYTE
86-88	Spare for alignment	3 BYTEs
89-92	Low transmitter range for analog signal in engineering units	CMD#15, Bytes 3-6 Type: REAL
93-96	High transmitter range for analog signal in engineering units	CMD#15, Bytes 7-10 Type: REAL
97	Slot 0 units code	CMD#33, Byte 1 Type: REAL
98	Slot 1 units code	CMD#33, Byte 7 Type: REAL
99	Slot 2 units code	CMD#33, Byte 13 Type: REAL
100	Slot 3 units code	CMD#33, Byte 19 Type: REAL
101	Slot 0 variable code	CMD#33, Byte 0 Type: REAL
102	Slot 1 variable code	CMD#33, Byte 6 Type: REAL
103	Slot 2 variable code	CMD#33, Byte 12 Type: REAL
104	Slot 3 variable code	CMD#33, Byte 18 Type: REAL
105-136	32 character message	CMD#12, Bytes 0-23 unpacked ASCII. Type: 32 BYTEs
137-140	Stored date in the field device	CMD#13, Bytes 18-20. Type 4 BYTEs
141-144	Number identifying the field device's material and electronics	CMD#16, Bytes 0-2. Type 4 BYTEs
145-169	The extended status returned by HART command 48.	Type: 25 BYTEs
170-180	Spare for alignment	Type: 11 BYTEs

RX3i Analog Modules with HART, COMMREQ 2: Send HART Pass-Thru Command

The controller can use COMMREQ 2 to send HART Pass-Thru commands to an RX3i Analog Module with HART Communications. A list of Pass-Thru commands is included in the *PACSystems RX3i System Manual,* GFK-2314C or later. The RX3i HART module then passes the command to the intended HART input or output device. Responses to HART Pass-Thru commands are available to the application program in the COMMREQ replies.

The Send HART Pass-Thru Command COMMREQ automatically fills in the Start Character, Address, Byte Count, Status, and the checksum. The RX3i HART module waits until the data from the HART device is available before it replies to this command, so the application program does not have to query the module for the response. The application program must check the COMMREQ Status word to determine when the reply data is available. The reply is returned between 750mS and 8 seconds later. The reply time depends on the number of channels enabled, the pass thru rate selected, and whether other pass-thru operations are occurring at the same time.

Only one application program Pass-Thru command per channel is allowed at a time. If another request is made on a channel that has a Pass-Thru in-progress, the module returns a COMMREQ Status Word = 0x0002 (module busy).

HART Pass-Thru Command Block, COMMREQ 2

Word	Dec	Hex	Description
1	10+x	000A + x	**Length of command Data Block:** in words beginning at Word 7
2	0	0000	Always 0 (no-wait mode request)
3			**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 70 = I (bit), 72 = Q (bit), 196 = W)
4			**Status memory offset:** COMMREQ status words address minus 1 (%R10
5	0	(0000)	Reserved
6	0	(0000)	Reserved
7	0002	(0002)	**Command code:** for the COMMREQ to be executed. HART Pass-Thru Command = 2
8	1	(0001)	**Number of Response Reference areas**: that follow (does not include COMMREQ status word). Always 1
9			**Memory type for the reply data:** (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 20 = T, 22 = M, 196 = W)
10	0	0000	**Bit Offset:** (must be 0 for all requests)
11			**0-based offset:** (low word).Starting address to which the response will be written. The offset from the beginning of PLC memory for the memory type specified in Word 9. This offset is in bytes or words depending on the memory type specified. Valid ranges of values depend on the PLC's memory ranges. Example: If Words 9 and 11 contain values of 8 and 250 respectively, the response will be written to %R251.
12			**0-based offset:** (high word). Value = 0 for most memory types. High word is non-zero only on if %W memory is used.
13			**Maximum size of response area:** Size in bytes if discrete memory type used for response. Size in words if word type used
14			**Channel Number**: 1-16 (valid range depends on module channel count and single-ended versus differential mode)
15			**HART Pass-Thru Command type:** HART Pass-Thru Commands (0x0 – 0xff) that can be sent to an RX3i HART module are listed in the *PACSystems RX3i System Manual,* GFK-2314D or later..
16			**Command Data byte count**: Size in bytes of command data that follows
…	…		…
Word 16+x	…		**HART Command Data:** Request data must be byte-packed and in big-endian format.

HART Pass-Thru Reply Data Format

The RX3i HART module returns the response data below to the CPU memory location specified by words 9-12 of the COMMREQ. Data beginning at Word 7 of the reply is byte-packed and in big-endian format. PLC CPU format is little-endian, so some commands may require swapping of fields from big-endian to little-endian format as described in the *PACSystems RX3i System Manual*. This is usually needed for floating point data.

Word	Name	Description
1	Command Code	Echo of Command code (0x0002)
2	Channel Number	Echo of Channel Number (same as request)
3	HART command	Echo of HART Pass-Thru Command type. See the tables in this section.
4	HART Status	Low byte is HART Comm Status and high byte is HART Dev Status from HART device response.
5	Spare	Spare for future use. User logic should not check this value because future module revisions may make this non-zero.
6	Response Byte Count (x)	Size in bytes of the response data that follows.
7L	Data Low	First response data byte from device.
7H	Data High	Second response data byte from device.
…	…	…
7+(x-1)/2 L	Data Low	….
7+(x-1)/2 H	Data High	Last response data byte from device.

COMMREQ Status Word

The values that can be returned in the COMMREQ status word are defined later in this chapter.

This status information relates to the execution of the COMMREQ function, not to the status of the HART communications. HART communications status is provided in the response data, as shown previously.

COMMREQs for an RX3i Profibus Master Module

The controller can use a Remote COMMREQ Call to send the following COMMREQs to a PACSystems RX3i Profibus Master Module IC695PBM300 in the I/O Station:

1	Get Device Status
2	Get Master Status
4	Get Device Diagnostics
5	Read Module Header
6	Clear Counters

Profibus Master Module, COMMREQ 1: Get Device Status

The controller can send COMMREQ 1, Get Device Status, to a PACSystems RX3i Profibus Master module in the I/O Station to retrieve detailed status information for a specified device on the Profibus network. This request retrieves diagnostics directly from the slave device using a Profibus network request. If network scan time is critical, the network impact should be considered when using this command.

Get Device Status Command Block

Word	*Dec*	*(Hex)*	*Description*
1	0005	(0005)	**Length of command Data Block:** in words beginning at Word 7.
2	0000	(0000)	Always 0 (no-wait mode request).
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 20 = T (byte), 22 = M byte, 196 =W)
4			COMMREQ status word address minus 1. Example: If Words 3 and 4 contain values of 8 and 9 respectively, the status word will be written to %R10.
5	0	(0000)	Reserved
6	0	(0000)	Reserved
7	0001	(0001)	**Command code**: for the COMMREQ to be executed. Get Device Status = 1.
8			**Memory type for the reply data**. See Word 3 above.
9			**Starting address to which the response will be written**. The value entered is the 0-based offset from the beginning of PLC memory for the memory type specified in Word 8. This offset will be in bits, bytes, or words depending on the memory type specified. Valid ranges of values depend on the PLC's memory ranges.
10	Words: 0009	(0009)	**Maximum size of response area** in words. Must be 9 if word memory type is used; 18 if discrete memory type is used.
	Bytes: 0018	(0012)	
11	0 to 125	(0 to 007D)	**The address of the device** the COMMREQ is to retrieve device status from. If the address of the master or a slave that is not on the bus is entered, a COMMREQ Status Word response of 4 will be returned.

Get Device Status Reply Data Format – Response written to location specified by Words 8 & 9

Word	*Name*	*Description*
1	Command Code	Echo of Command Code that this data block is replying to (0x0001).
2	Device Status 1	Code indicating the status of the slave device. See tables below.
3	Device Status 2	Code indicating the status of the slave device. See tables below.
4	Device Status 3	Code indicating the status of the slave device. See tables below.
5	Master Address	The address of the master connected to this slave. If the slave is not parameterized this value will be 255 (0x00FF).
6	Ident Number	The Ident Number of the slave.
7 …9	Reserved for future use.	Word 10 of the Get Device Status command block should specify a minimum of 9 words (18 bytes) to accommodate possible future use of this space.

Device Status 1 – Word 2

Bit	*Name*	*Description*
0	Sta._Non_Exist	No response from slave device. The station is non-existent.
1	Sta._Not_Ready	Slave not ready.
2	Cfg_Fault	Slave has incorrect parameterization.
3	Ext_Diag	The extended diagnostics area is used.
4	Not_Supp	Unknown command is detected by the slave.
5	Inv._Slv_Res	Invalid slave response.
6	Prm_Fault	Last parameterization telegram was faulty.
7	Master_Lock	Slave is controlled by another master.
8 … 15	RFU	Reserved for further use

If this status word is zero, the slave device has no errors. The non-zero values, which are errors, are defined in the following table.

Device Status 2 – Word 3

Bit	*Name*	*Description*
0	Prm_Req	Slave must be parameterized.
1	Stat_Diag	This bit remains active until all diagnostic data has been retrieved from the slave.
2	1	Always a value of one.
3	WD_On	Slave watchdog is activated.
4	Freeze_Mode	Freeze command active.
5	Sync_Mode	Sync command active
6	Reserved	Reserved.
7	Deactivated	Slave not active.
8 … 15	RFU	Reserved for further use

Device Status 3 – Word 4

The Device Status 3 word has only one active meaning. If this word is set to 0x0080 then the slave has an Extended Diagnostic data overflow. This means that the slave has a large amount of diagnostic data and cannot send it all.

Profibus Master Module, COMMREQ 2: Get Master Status

The controller can send COMMREQ 2 to an RX3i Profibus Master module in the I/O Station to obtain detailed status information about the module.

Warning

If a Get Master Status COMMREQ is called on the first scan of the PLC, the COMMREQ may return a false positive status. The Get Master Status COMMREQ should not be called or relied upon for any data during the first scan of the PLC.

Get Master Status Command Block

Word	Dec	(Hex)	Description
1	0004	(0004)	**Length of command Data Block:** For Get Master Status the length is 4 words (8 bytes).
2	0000	(0000)	Always 0 (no-wait mode request)
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q byte, 20 = T byte, 22 = M byte, 196 =W)
4	00009	(0009)	**Status memory offset:** COMMREQ status word address minus 1 (%R10 for this example).
5	0000	(0000)	Reserved
6	0000	(0000)	Reserved
7	0002	(0002)	**Command code:** Command code for the COMMREQ to be executed. Get Master Status = 2.
8			**Reply segment select:** Memory type for the reply data. (See Word 3 above).
9			**Reply memory offset:** Offset within the memory type for the reply (0-based). Starting address to which the response will be written. The value entered is the offset from the beginning of PLC memory for the memory type specified in Word 8. This offset will be in bits, bytes, or words depending on the memory type specified. Valid ranges of values depend on the PLC's memory ranges.
10	Words: 0009	(0009)	**Reply memory size:** Maximum size of response area. Must be 9 if word memory type is used; 18 if discrete memory type is used.
	Bytes: 0018	(0012)	

Get Master Status Reply Data Format

Word	Name	Description
1	Command Code	Echo of Command code that this data block is replying to. (0x0002)
2	Global State Bits	Bits indicating the global state of the master. See "Global State Bits."
3	DPM State	Control state of the Dual Port Memory in the master. See "DPM State".
4L	Error Remote Address	Remote address of device with error. See "Error Remote Address".
4H	Error Event	Error code response to the Error Remote address. See "Error Event".
5 … 9	Reserved	Word 10 of the Get Master Status command block should specify a minimum of 9 words to accommodate possible future use of this space.

Global State Bits

The Profibus master's global state is reported in Word 2 of the Get Master Status reply data and the low byte of Word 4 in the Read Module Header reply data. If there are no errors reported by the master, all bits in this word have a value of zero. The following table provides definitions for bits with a value of 1.

Bit	Name	Description
0	CTRL	CONTROL-ERROR: Parameterization error.
1	ACLR	AUTO-CLEAR-ERROR: Master has stopped communications to all slaves and reached the auto-clear end state.
2	NEXC	NON-EXCHANGE-ERROR: At least one slave has not reached the data exchange state and no process data is being exchanged with it.
3	FAT	FATAL-ERROR: Because of major network fault, no further bus communication is possible.
4	EVE	EVENT-ERROR: The master has detected bus short circuits. The number of detected events is reported in Word 6, BusErrorCnt, of the Read Module Header reply. The bit is set only when the first event is detected.
5	NRDY	HOST-NOT-READY-NOTIFICATION: If this bit is set, the HOST program is not ready to communicate.
6	TOUT	TIMEOUT-ERROR: The timeout supervision time has been exceeded because of rejected PROFIBUS telegrams. This error indicates bus short circuits that cause the master to interrupt communications. The number of detected timeouts is reported in Word 7, TimeOutCnt, of the Read Module Header reply. The bit is set only when the first timeout is detected.
7	NA	Reserved.

DPM State

The current control state of the Dual Port Memory in the master. DPM State is reported in Word 3 of the Get Master Status reply data and the high byte of Word 4 of the Read Module Header reply data. The following table provides definitions of the possible values.

Value	DPM Master State	Description
0x00	OFFLINE	The master system has been switched on, but there is no data transfer on the bus.
0x40	STOP	The master loads bus parameters and initializes the diagnostic buffer. No data transfer takes place.
0x80	CLEAR	The master parameterizes and configures the slaves through the bus. It reads the input data, but retains the output data.
0xC0	OPERATE	User data transfer is active. New output data is transmitted cyclically and the latest input data is read.

Error Remote Address (Low Byte Word 4)

The Error Remote Address field contains the physical address of a device that has caused an error. If the master is the source of the error, this byte contains the value 255. If the error was detected at or reported by a network device, the byte contains the source station address and has a range from 0 to 125. If this field is contains an address, the Error Event byte will contain a code that identifies the error.

Error Event (High Byte Word 4)

The Error Event byte contains the error code of the device identified in the Error Remote Address field. This error code is also reported in the high byte of Word 5 of the Read Module Header reply data.

Error Event Codes for Profibus Master (Error Remote Address is 255)

Code	*Indication*	*Source*	*Corrective Action*
0	No errors are present.	None	None.
50	USR_INTF-Task not found.	Master	Firmware is invalid. Update module.
51	No global data-field.	Master	Firmware is invalid. Update module.
52	FDL-Task not found.	Master	Firmware is invalid. Update module.
53	PLC-Task not found.	Master	Firmware is invalid. Update module.
54	Non-existing master parameters.	Master	Download hardware configuration.
55	Faulty parameter value in the master parameters	Configuration	Firmware is invalid. Update module.
56	Non-existing slave parameters.	Configuration	Download hardware configuration.
57	Faulty parameter value in a slave parameters data file.	Configuration	Check GSD file for possible incorrect slave parameterization values.
58	Duplicate slave address.	Configuration	Check configured slave addresses in project.
59	Configured send process data offset address of a slave is outside the allowable range of 0—255.	Configuration	Check slave configuration in project.
60	Configured receive process data offset address of a slave is outside the allowable range of 0—255.	Configuration	Check slave configuration in project.
61	Data areas of slaves overlapping in the send process data.	Configuration	Check slave configuration in project.
62	Data areas of slaves are overlapping in the receive process data.	Configuration	Check slave configuration in project.
63	Unknown process data handshake.	Master	Problem with master's startup parameters.
64	Free RAM exceeded.	Master	Master has a hardware issue.
65	Faulty slave parameter dataset.	Configuration	Check GSD file for possible incorrect slave parameterization datasets.
202	No memory segment free.	Master	Master has a hardware issue.
212	Faulty reading of a database.	Configuration	Execute download of configuration database again.
213	Structure used by the operating system is faulty.	Master	Master has a hardware issue.
220	Software Watchdog error.	Host	Firmware watchdog has an error.
221	No Data Acknowledge in process data handshake.	Host	Firmware is having trouble with Host acknowledgement.
222	Master in Auto Clear mode	Slave Device	The auto clear mode was activated, because one slave is missing during runtime.
225	No further segments.	Master	Master has a hardware issue.

Error Event Codes for Slave Devices (Error Remote Address Not Equal to 255)

Code	*Indication*	*Source*	*Corrective Action*
0	No errors	NA	NA
2	Slave station reports data overflow.	Master Telegram	Check length of configured slave parameter or configuration data.
3	Master is requesting a function that is not supported in the slave.	Master Telegram	Check if slave is PROFIBUS-DP norm compatible.
9	No answering data, although the slave must respond with data.	Slave	Check configuration data of the slave and compare it with the physical I/O data length.
17	No response from the slave.	Slave	Check bus cable and bus address of slave.
18	Master not in the logical token ring.	Master	Check FDL-Address of master or highest station address of other master systems. Examine bus cabling for bus short circuits.
21	Faulty parameter in request.	Master Telegram	Master has a firmware issue.

RX3i Profibus Master Module, COMMREQ 4 : Get Device Diagnostics

The controller can send COMMREQ 4, Get Device Diagnostics, to a Profibus Master Module to retrieve detailed status information for the device.

Get Device Diagnostics Command Block

Word	Dec	(Hex)	Description
1	00005	(0004)	**Length of command Data Block:** For Get Device Diagnostics, the length is 4 words (8 bytes).
2	00000	(0000)	Always 0 (no-wait mode request)
3	00008	(0008)	**Status segment select:** Memory type of COMMREQ status words (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I, 18 = Q, 20 = T, 22 = M, 196 =W)
4	00009	(0009)	**Status memory offset:** COMMREQ status words start address minus 1 (%R10 for this example)
5	00000	(0000)	Reserved
6	00000	(0000)	Reserved
7	00004	(0004)	**Command code:** Get Device Diagnostics Info command number (4)
8	00008	(0008)	**Reply segment select:** Memory type for the reply data. See Word 3 above.
9	00250	(00FA)	**Reply memory offset:** Offset within the memory type for the reply (0-based). For this example, it is %R251. (Word offset for memory types: 8, 10, 12; byte offset for memory types: 16, 18, 20, 22).
10	00009	(0009)	**Reply memory size:** Maximum size for the reply (in words for memory types: 8, 10, 12; in bytes for memory types: 16, 18, 20, 22). Maximum 2048 bytes. Note: For command 4 must be 18 bytes (9 words) or more, or an error will be returned in the COMMREQ status and the command will be ignored.
11	0 to 125	(0 to 007D)	**Address of the device the COMMREQ is retrieving device status from**. If the address of the master or a slave that is not on the bus is entered, a COMMREQ Status Word response of 4 will be returned.

Get Device Diagnostics Reply Data Format – Response written to location specified by Words 8 & 9

Word	Name	Description
1	Command Code	Echo of the Command Code = 4.
2	Size x of Diagnostics Received	Size in bytes of the Extended Diagnostics received.
3	Diag 0 (Low Byte) Diag 1 (High Byte)	Extended diagnostic data bytes.
4	Diag 2 (Low Byte) Diag 3 (High Byte)	Extended diagnostic data bytes.
…	…	…
2 + (x/2)	Diag ….x (Low Byte) Diag ….x+1 (High Byte)	Extended diagnostic data bytes.

Profibus Master Module, COMMREQ 5: Read Module Header

The controller can send COMMREQ 5, Read Module Header, to a Profibus Master module to retrieve Network Diagnostic Information and statistics.

Read Module Header Command Block

Word	Dec	(Hex)	Description
1	0004	(0004)	**Length of command Data Block:** in words beginning at Word 7.
2	0000	(0000)	Always 0 (no-wait mode request).
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 20 = T (byte), 22 = M byte, 196 =W)
4			COMMREQ status word address minus 1. Example: If Words 3 and 4 contain values of 8 and 9 respectively, the status word will be written to %R10.
5	0000	(0000)	Reserved
6	0000	(0000)	Reserved
7	0005	(0005)	**Command code**: for the COMMREQ to be executed. Read Module Header = 5.
8			**Memory type for the reply data**. See Word 3 above.
9			**Starting address to which the response will be written**. The value entered is the 0-based offset from the beginning of PLC memory for the memory type specified in Word 8. This offset will be in bits, bytes, or words depending on the memory type specified. Valid ranges of values depend on the PLC's memory ranges.
10	Words: 0020	(0014)	**Maximum size of response area** in words. Must be 20 if word memory type is used; 40 if discrete memory type is used.
	Bytes: 0040	(0028)	

Read Module Header Reply Data Format

Word	Name	Description
1	Command Code	Echo of the Command Code = 5.
2	Interface Type	2 if the interface is a master. 1 if the interface is a slave.
3	Firmware Revision	Indicates the current firmware revision: high byte is major version number; low byte is minor version number.
4L	Global State Bits (Low Byte)	Indicates the global state of the master. See “Global State Bits”.
4H	DPM State (High Byte)	Dual Port Memory control state of the master. See “DPM State”.
5L	Error Remote Address	The physical address of a device that has caused an error. ■ If the master is the source of the error, this byte contains the value 255. ■ If the error was detected at or reported by a network device, the byte contains the source station address and has a range from 0 to 125. If this field is non-zero, the Error Event byte will contain a code that identifies the error.
5H	Error Event	Error code response to the Error Remote address. See “Error Event”.
6	BusErrorCnt	Number of major bus error, for example bus short circuits.
7	TimeOutCnt	Number of rejected PROFIBUS telegrams.
8	SlaveDiagReq	Number of slave diagnostics requests.
9	GlobalConReq	Number of global control requests.
10	DataExReq	Number of data exchange cycles.
11	DataExReqPos	Number of positive data exchange cycles.
12	DataExReqNeg	Number of negative data exchange cycles.
13	DataExAllReq	Number of all active data exchange cycles.
14	DataExAllReqPos	Number of data exchange cycles (all positive requests).
15	DataExAllReqNeg	Number of data exchange cycles (all negative requests.).
16	SlavesFound	Number of slaves found on bus. **Note:** Only the slaves on the network that do not belong to another master are counted as SlavesFound.
17	SlavesConfigured	Number of configured slaves on the bus.
18	SlavesActive	Number of slaves active in data exchange mode.
19	DataControlTime	Time (in ms) of the data exchange.
20	Reserved	Reserved for future use

Profibus Master Module, COMMREQ 6: Clear Counters

The controller can send COMMREQ 6 to a Profibus Master module to set its counters to zero. For a list of counters, see words 6 through 18 of the "Read Module Header Reply Data Format" on page 9-39.

Clear Counters Command Block

Word	Dec	(Hex)	Description
1	0004	(0004)	**Length of command Data Block:** in words beginning at Word 7.
2	0000	(0000)	Always 0 (no-wait mode request).
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 20 = T (byte), 22 = M byte, 196 =W)
4			**COMMREQ status word address** minus 1. Example: If Words 3 and 4 contain values of 8 and 9 respectively, the status word will be written to %R10.
5	0000	(0000)	Reserved
6	0000	(0000)	Reserved
7	0006	(0006)	**Command code**: for the COMMREQ to be executed. Clear Counters = 6.
8			**Memory type for the reply data**. See Word 3 above.
9			**Starting address to which the response will be written**. The value entered is the 0-based offset from the beginning of PLC memory for the memory type specified in Word 8. This offset will be in bits, bytes, or words depending on the memory type specified. Valid ranges of values depend on the PLC's memory ranges.
10	Words: 0002	(0002)	**Maximum size of response area** in words. Must be 2 if word memory type is used; 4 if discrete memory type is used.
	Bytes: 0004	(0004)	

Series Clear Counters Reply Data Format

Word	Name	Description
1	CommandCode	Echo of Command code that this data block is replying to. (0x0006)
2	StatusCode	Reports 1 for success and 0 for failure.

COMMREQ for RX3i and Series 90-30 Motion Controller Modules

The controller can use a Remote COMMREQ Call to send the following COMMREQ to a PACSystems RX3i Motion Controller Module IC694DSM314 or DSM324 or to a Series 90-30 Motion Controller Module IC693DSM314 or DSM324 in the I/O Station:

E501	Parameter Load

Motion Controller Modules, COMMREQ E501: Parameter Load

DSM Parameter Load COMMREQ Command Block

Word	*Dec*	*(Hex)*	*Description*
1	0004	(0004)	**Length of command Data Block:** in words beginning at Word 7.
2	0000	(0000)	Always 0 (no-wait mode request).
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 70 = I (bit), 72 = Q (bit), 20 = T (byte))
4			**COMMREQ status word** address minus 1. Example: If Words 3 and 4 contain values of 8 and 9 respectively, the status word will be written to %R10.
5	0	(0000)	Reserved
6	0	(0000)	Reserved
7		(E501)	**Command code**: for the COMMREQ to be executed. Parameter Loads = E501.
8	0068	(0044)	**Parameter Data Block Size**, in bytes (includes 4 bytes for Parameter Specifier Words)* Always set to 68 (44 hex)
9			**Parameter Data Memory Type** (for Word 11). See Word 3 above.
10	Words: 0002	(0002)	**Parameter Data Offset** (for Word 11) To load all 16 parameters, the value must be 240 or less.
11			**Starting parameter number** (0 - 255) in DSM Parameter Table
12	0016	(0010)	**Number of parameters to load.** Always set to 16 (10 hex)
13, 14			**1st parameter data** (2 words per parameter)
15, 16			**2nd parameter data**
. . .			**. . .**
Word 43, Word 44			**16th parameter data** (4 bytes)

* Parameter Data Block size equals 4 bytes for the Parameter Specifier Words plus 4 bytes for each Parameter

COMMREQ for High-Speed Counter Modules

The controller can use a Remote COMMREQ Call to send the following COMMREQ to a PACSystems RX3i High-speed Counter Module IC694APU300 or Series 90-30 High-speed Counter Module IC69APU300 in the I/O Station:

E201	Send Data

High-Speed Counter Modules, COMMREQ E201: Send Data Command

High-speed Counter COMMREQ Command Block

Word	*Dec*	*(Hex)*	*Description*
1	0004	(0004)	**Length of command Data Block:** in words beginning at Word 7.
2	0000	(0000)	Always 0 (no-wait mode request).
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 70 = I (bit), 72 = Q (bit), 20 = T (byte))
4			**COMMREQ status word address** minus 1. Example: If Words 3 and 4 contain values of 8 and 9 respectively, the status word will be written to %R10.
5	0	(0000)	Reserved
6	0	(0000)	Reserved
7		(E201)	**Command code**: for the COMMREQ to be executed. (E2 - message ID for 6 byte Data Command to High Speed Counter) and Command Parameter (1 = write).
8	0006	(0006)	**Byte length of data to High Speed Counter**
9	0008	(0008)	**Parameter Data Memory Type** (for Word 11). See Word 3 above.
10	0010	(000A)	**Parameter Data Offset** (for Word 11) To load all 16 parameters, the value must be 240 or less.
11			**Starting parameter number** (0 - 255) in DSM Parameter Table
12	0016	(0010)	**Number of parameters to load.** Always set to 16 (10 hex)
13			LS data word
14			MS data word

COMMREQs for Modbus RTU Master on the RX3i ENIU Serial Ports

Modbus RTU Master COMMREQs Command Block- All Function Codes

Word	*Dec*	*(Hex)*	*Description*			
1	0000	(0000)	**Length of command Data Block:** always 0			
2	0000	(0000)	Always 0 (no-wait mode request).			
3	0008	(0008)	**Status segment select:** Memory type of COMMREQ status word (%R for this example). (8 = R, 10 = AI, 12 = AQ, 16 = I (byte), 18 = Q (byte), 196 = W)			
4			COMMREQ status word address minus 1. Example: If Words 3 and 4 contain values of 8 and 9 respectively, the status word will be written to %R10.			
5	0000	(0000)	Reserved			
6	0000	(0000)	Reserved			
7	8002	(1F42)	**Command code**: for the Modbus Master			
8	0001	(0001)	**Address of Modbus Slave:** 1- 247, 0 = broadcast (broadcast is for Function Codes 5, 6, 15, 16 only)			
9			**Modbus Function Code***:*			
			1	Read Outputs	**6**	Preset One Register
			2	Read Inputs	**7**	Read Exception Status
			3	Read Holding Registers	**15**	Write Multiple Coils
			4	Read Input Registers	**16**	Write Multiple Outputs
			5	Set/Clear One Coil	**17**	Report Device Type
10			*Use and Value depends on Function Code*			
			1	Starting address for read	**6**	Register number
			2	Starting address for read	**7**	Always 0
			3	Starting address for read	**15**	Starting address for write
			4	Starting address for read	**16**	Starting address for write
			5	Coil number	**17**	Always 0
11			*Value depends on Function Code*			
			1	Number of items	**6**	Value to write to register
			2	Number of items	**7**	Not used, must be 0
			3	Number of items	**15**	Number of items
			4	Number of items	**16**	Number of items
			5	0 turn coil OFF, 1 turn coil ON	**17**	Not used, must be 0

Word	Dec	(Hex)	Description			
12			*Value depends on Function Code*			
			1	Reference table for response: %I = 16, %Q = 18, %T = 20, %M = 22, %AI = 10, %AQ = 12, %R = 8, %W =196	**6**	Not used, must be 0
			2		**7**	Reference table for response: %I = 16, %Q = 18, %T = 20, %M = 22, %AI = 10, %AQ = 12, %R = 8, %W =196
			3	Reference table for response: %AI = 10, %AQ = 12, %R = 8, %W =196	**15**	Reference table for data source: %Q= 18, %R = 8, %W =196
			4		**16**	Reference table for data source: %AI = 10, %AQ = 12, %R = 8, %W =196
			5	Not used, must be 0	**17**	Reference table for response: %AI = 10, %AQ = 12, %R = 8, %W =196
13			*Value depends on Function Code*			
			1	Offset in Reference table to put response	**6**	Not used, must be 0
			2		**7**	Offset in Reference table to get data
			3		**15**	
			4		**16**	
			5	Not used, must be 0	**17**	Offset in Reference table to put response
14	0100	(01F4)	**Timeout in Milliseconds:** Range 0 to 10,000			

COMMREQ Error Codes by Module Type

PACSystems RX3i and Series 90-30 DeviceNet Modules

DeviceNet modules IC694DNM200 and IC693DNM200 return the four-word COMMREQ status block shown below..

Word	*Name*	*Description*	
1	**State:** The state of the current COMMREQ request		
	0x00	Module has not yet processed the COMMREQ	
	0x01	Command Complete - this status does not necessarily mean success. Some commands have reply data that must also be checked.	
	0x02	Busy – Command is being processed and has not completed Note: It is not guaranteed that the status will transition to busy before complete or terminated.	
	0x03	Command Terminated – invalid command	
	0x04	Command Terminated – invalid command data	
	0x05	Command Terminated – not enough data	
	0x06	Reserved	
	0x07	Command Terminated – not enough memory in reply area The command did not specify sufficient PLC memory for the reply. Command will be ignored.	
	0x08	Command Terminated – command-specific error. See Error Code and Additional Code in the Status Block for more information.	
	0x09	Command Terminated – invalid COMMREQ	
	0x0A	Command Terminated – specific segment selector for COMMREQ reply is not supported	
	0x0B	Command Terminated – reply failed to write PLC memory	
	0x0C	Command Terminated - specific segment selector for COMMREQ data is not supported	
	0x0D	Command Terminated – failed to read PLC memory	
	0x0E to 0xFF	Reserved	
2	**Lost Command**	Command code of the last command lost. Set to 0 if no command was lost.	
3	**Error code:** Meaning Depends on the Command number		
	Command	***Error Code***	
		0	Reserved
	1,3,7,8	1	Explicit data too large for shared memory buffer. Additional Code word holds the real size of the shared memory buffer.
	1, 4, 7	2	Invalid MacID specified.
	1,2,3,7,8	3	Explicit connection not configured.
	2	4	Explicit request not available.
4	**Additional code:** for error reporting.		

PACSystems RX3i and Series 90-30 Genius Bus Controllers

Genius Bus Controllers IC694BEM331 and IC693BEM331 return the following values in their COMMREQ status word:

Value	Description
0	Device has not yet processed the COMMREQ.
1	Command not accepted, GBC busy with previous request
4	Command completed successfully
8	Command terminated due to syntax error
16	Command terminated due to data error
32	Command terminated due to suspended activity on bus
64	No data to transfer
128	Command not supported by target device
256	Only No Wait commands may be sent to the target device
512	Maximum Comms. Time must be greater than or equal to 5mS
1024	Text buffer invalid in wait mode
2048	Device did not accept the message, or timed out.

The upper word of the status location provides additional status information. Not all of these values are relevant for the set of COMMREQS that can sent using Remote COMMREQ Calls.

Value	Description
11	Non-discrete block specified for Pulse Test
21	Non-I/O device specified for Read Configuration
51	Invalid circuit number
71	Non-controller device specified for Assign Monitor
101	Switch BSM - device not BSM
102	Switch BSM - bus position greater than 1
121	P and L access not available
141	Function code greater than 111
142	Sub function code greater than 255
143	Priority greater than 1
144	Datagram length greater than 134
201	Invalid Device Number (greater than 31, but not 255)
202	Incorrect length for the command type
203	Device Number not configured or not active
204	Previous No Wait command in progress; current No Wait command not accepted
205	Invalid status pointer location specified
206	Command number is out of range
207	Subcommand code is out of range
208	Only partial data transferred
209	Device Number 255 not allowed for this command
210	Command specified is not valid for GBC
211	Command specified is only valid for controller devices
212	Command specified is not supported by the device to which it was sent
213	Invalid Alarm Enable/Disable mask

PACSystems RX3i Analog Modules with HART Communications

PACSystems RX3i Analog Modules with HART Communications (IC695ALG626, IC695ALG628, and IC695ALG728) return the following values in their COMMREQ status word.

Value		Description
Dec	(Hex)	
0	(0000)	Device has not yet processed the COMMREQ.
1	(0001)	Command Complete - this status does not necessarily mean success. The command reply data that must also be checked.
2	(0002)	Command Terminated – module busy
3	(0003)	Command Terminated – invalid command
4	(0004)	Command Terminated – invalid command data
5	(0005)	Command Terminated – not enough data
6	(0006)	Not used
7	(0007)	Command Terminated – The command did not specify sufficient PLC memory for the reply. The command will be ignored.
8	(0008)	Command Terminated – command-specific error.
265	(0109)	Error, Hart device not connected
521	(0209)	Error, Channel not HART-enabled
777	(0309)	Error, Analog Output Module, No field power
1033	(0409)	Error. HART command now allowed
1289	(0509)	Error. Invalid HART command
1545	(0609)	Error. Device did not respond
1801	(0709)	Error, HART data count too large

This status information relates to the execution of the COMMREQ function only, not to the status of the HART communications.

PACSystems RX3i Profibus Master Module

PACSystems RX3i Profibus Master Module IC695PBM300 returns the following values in its COMMREQ status word.

Value		Description
Dec	**(Hex)**	
0	(0000)	Device has not yet processed the COMMREQ.
1	(0001)	Command Complete -- this status does not necessarily mean success. Some commands have reply data that must also be checked.
2	(0002)	Command Terminated – module busy
3	(0003)	Command Terminated – invalid command
4	(0004)	Command Terminated – invalid command data
5	(0005)	Command Terminated – not enough data
6	(0006)	Not used
7	(0007)	Command Terminated – the command did not specify sufficient PLC memory for the reply. Command will be ignored.
8	(0008)	Command Terminated – command-specific error.

PACSystems and Series 90-30 Motion Controllers

Motion Controller Modules IC694DSM314, IC694DSM324, IC693DSM314, and IC694DSM324 return the COMMREQ Status Words shown below:

Value	Name	Description
1	IOB_SUCCESS	All communications proceeded normally.
-1	IOB_PARITY_ERR	A parity error occurred while communicating with an expansion rack.
-2	IOB_NOT_COMPL	After the communication was over, the module did not indicate that it was complete.
-3	IOB_MOD_ABORT	The module aborted the communication.
-4	IOB_MOD_SYNTAX	The module indicated that the data sent was not in the correct sequence.
-5	IOB_NOT_RDY	The RDY bit in the module's status was not active.
-6	IOB_TIMEOUT	The maximum response time elapsed without receiving a response from the module.
-7	IOB_BAD_PARAM	One of the parameters passed was invalid.
-8	IOB_BAD_CSUM	The checksum received from the DMA protocol module did not match the data received.
-9	IOB_OUT_LEN_CHGD	The output length for the module was changed, therefore normal processing of the reply record should not be performed.

PACSystems RX3i and Series 90-30 High Speed Counter Modules

High-Speed Counter modules IC694APU300 and IC693APU300 return the following values in their COMMREQ status word.

Value	Name	Description
0	IOB_BUSY	Module is reconfiguring
1	IOB_SUCCESS	All communications proceeded normally.
-1	IOB_PARITY_ERR	A parity error occurred while communicating with an expansion rack.
-2	IOB_NOT_COMPL	After the communication was over, the module did not indicate that it was complete.
-3	IOB_MOD_ABORT	For some reason, the module aborted the communication.
-4	IOB_MOD_SYNTAX	The module indicated that the data sent was not in the correct sequence.
-5	IOB_NOT_RDY	The RDY bit in the module's status was not active.
-6	IOB_TIMEOUT	The maximum response time elapsed without receiving a response from the module.
-7	IOB_BAD_PARAM	One of the parameters passed was invalid.
-8	IOB_BAD_CSUM	The checksum received from the DMA protocol module did not match the data received.
-9	IOB_OUT_LEN_CHGD	The output length for the module was changed, so normal processing of the reply record should not be performed.

Status Values for Modbus Master Communications

For Modbus Master, status values have a Major code and a Minor Code. The Major code is in the low-order byte, and the Minor code is in the high-order byte. Status values are expressed in hexadecimal, and are most easily viewed in hexadecimal format.

Minor	***Major***	***Description***
0	0	In Process (or no Modbus Query attempt since power-up)
0	1	Success
1	1	Broadcast Timeout (this is success for a Query to broadcast ID (0)).
5	3	Bad Port Number – Port number must be 1 or 2 (19 & 20 will also work).
6	3	Bad Slave ID – Slave ID must be in the range 0-247
7	3	Bad Function Code - Function Code must be 1,2,3,4,5,6,7,8,15,16,17
8	3	Bad Broadcast Function Code – Only FC 5, 6, 8 sub 4,15,16 support broadcast
A	3	Start Address is Zero – start address (command word 3) must be > 0
B	3	Too Many Items - – number of items must be > 0
C	3	Bad Local Seg Selector – See Function Code Chart to see supported segments
21	3	Bad Local Seg Selector – See Function Code Chart to see supported segments
22	3	Bad Local Address Offset – Start Addr + num of items > size of segments
23	3	Bad set/Clear Coil Value - must be (0 = Clear, 1 = Set, FF = Set)
15	3	Bad Port Type – Hardware configuration of port must be “Serial I/O”
16	3	Bad Rack Slot - CPU is not in slot specified in “C” block
17	3	Bad COMMREQ command number – Command word 1 must by 8002
19	3	Bad Command Code – Command word 1 must by 8002
20	3	Unexpected State, call Tech Support
1	4	Parity Error received
2	4	Framing Error received
3	4	Bad CRC received
5	4	Overflow Error received
7	4	Multiple UART Errors
1	5	Timeout – response was not received within timeout period
2	5	Transmit Timeout – Query was not sent check CTS signal
10	6	Bad Buffer Seg Select – “C” block Input 4 must by 8, %R or 196 %W
11	6	Bad Buffer offset – “C” block Input 5 not a good value need space for 150
x	8	Exception response received. x is the number of the exception.

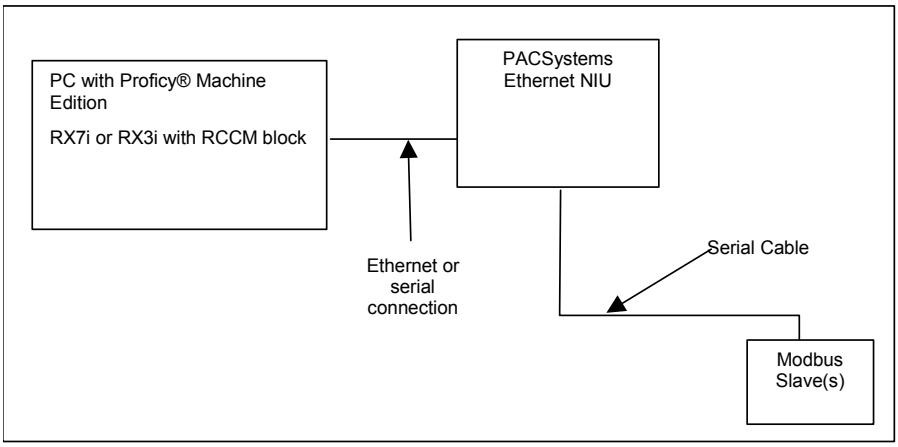
Chapter 10 Modbus Master for the Ethernet NIU

This chapter explains how to implement Modbus Master communications between the PACSystems RX3i Ethernet NIU and Modbus Slaves, using one or both of the Ethernet NIU's serial ports.

- Modbus Master for the Ethernet NIU
- CPU or Ethernet NIU Control of Modbus Master Communications
  - Hardware Configuration
- Software Function Blocks for Modbus Master Communications
  - Revision of the "C" Software Function Block
  - Setting Up the "C" Function Block for Modbus Master
  - Input and Output Parameters of the "C" Block
- Operation of the "C" Block in the Local User Logic
  - Execution of the Modbus Master Function Codes
  - Modbus Communications Status Codes
- Programming Examples
  - Modbus Master Using Local User Logic
  - Modbus Master Using Remote COMMREQ Call Communications

Modbus Master for the Ethernet NIU

The GE Fanuc PACSystems RX3i Ethernet NIU has two serial ports; one is an RS-232 port, and the other is an RS-485 port. Either or both of these ports can be used for Modbus Master protocol.



Recommended Media

- RS-232 point to point
- RS-485, 2-wire and 4-wire, point to point and multi-drop
- Telephone modem connection using SixNet VT modems

Not Verified

- Radio Modems
- Cellular Phone or Modem

Supported Modbus Master Function Codes	Function Code 1 – Read Outputs Function Code 2 – Read Inputs Function Code 3 – Read Holding Registers Function Code 4 – Read Input Registers Function Code 5 – Set/Clear One Coil Function Code 6 – Write One Register Function Code 7 – Read Exception Status Function Code 8 -- Subfunction 0: Loopback Function Code 8 – Subfunction 1: Restart Communication Function Code 8 – Subfunction 4: Enter Listen Only Mode Function Code 15 – Write Multiple Coils Function Code 16 – Write Multiple Registers Function Code 17 – Report Device Type
Unsupported Function Codes	Function Code 65 – Read Scratchpad Function Codes to Read/Write 32 bit Registers

CPU or Ethernet NIU Control of Modbus Master Communications

Modbus Master communications can be used in two ways in the PACSystems RX3i Ethernet NIU:

- Remote COMMREQ Call commands can be sent from the PACSystems RX7i or RX3i controller to tell the ENIU to do Modbus communications. In this case, the data source address and data destination address are addresses in the controller.
- Local User Logic in the Ethernet NIU can call a “C” block (built into the Ethernet NIU) to do Modbus communication. In this case, the addresses for the data to write to the Modbus Slave or the data Read from the Modbus Slave are addresses in the Ethernet ENIU, not in the controller. If the data needs to come from or go to the controller, the Local User Logic needs to move the data from/to the controller.

Modbus Master communication can be done with either of these methods on either Ethernet NIU serial port, *but not with both methods on the same port*. If both methods were used on one port, the “C” block would be called from two different places. In that event, the response to the Modbus Master communication could be returned to either place where the “C” block is called, resulting in loss of data.

Remote COMMREQ Call communications always use the “C” block MB_P1 and can be used for either or both ports.

Local User Logic should use MB_P2.

Hardware Configuration for Modbus Master

Hardware configuration is done with Proficy® Machine Edition LD-PLC. Any RX3i Ethernet NIU serial port to be used for Modbus Master protocol must be configured for Serial I/O with appropriate communication parameters.

Parameter	*Required Setting*	*Choices*
Port Mode	Serial I/O	
Data Rate		1200, 2400, 4800, 9600, 19200, Higher data rates may be possible, but are not assured.
Data Bits	8	
Flow Control		None (recommended), hardware.
Parity		None, odd, even. Slave / modems must match
Stop Bits		1, 2
Physical Interface		2-wire, 4-wire
Stop Mode		Any setting

Note – If both ports are used, the run/stop switch MUST be enabled

Software Function Blocks for Modbus Master Communications

If both ports are to be used simultaneously for Modbus Master Protocol, two separate “C” .gefElf files are needed. The two gefElf files are provided: MB_P1.gefElf and MB_P2.gefElf. The two files are identical except for their names.

Revision of “C” Software Function Block

The revision of the software function block is encoded in the block. The revision can be checked by looking at the hexadecimal value in the first register specified by inputs X4(seq) and X5(seq_off) of the Call to the “C” block. The “C” block places the revision code in that register on the first scan of the controller. The low-order byte contains the major revision code. The high-order byte contains the minor revision code. For example, version 1.01 appears as 0x0101 in the revision code.

Setting Up the “C” Function Block for Modbus Master

The input parameters of the “C” function block used for Modbus Master communications must be correctly set up for:

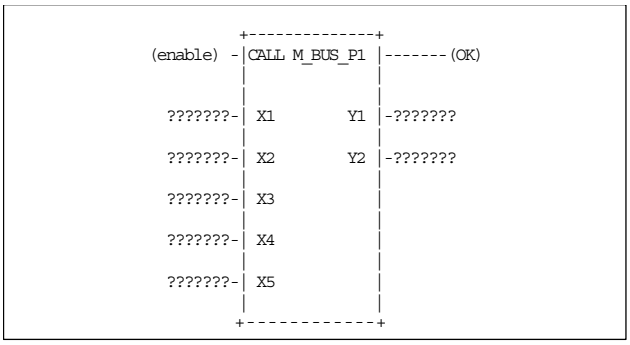
- the slot the ENIU is located in
- the port to be used for Modbus Master communications
- a required buffer area in %R or %W memory (150 registers in size). Note: if both ports are used, each “C” block needs a separate buffer area.
- the particular settings for an individual Modbus Message

Setting up RCC commands in the controller includes supplying the “C” input parameters.

If Modbus Master is done from Local User Logic, the “C” block’s input parameters are set up in the Local User Logic.

Supplying the Input Parameters for the "C" Block

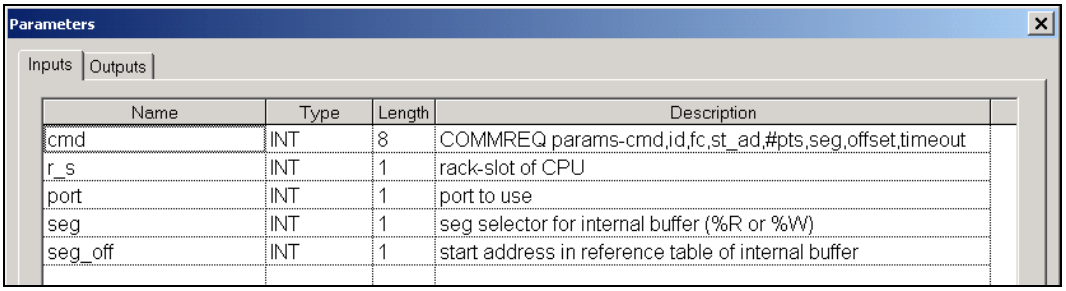
The "C" block has the following parameters:

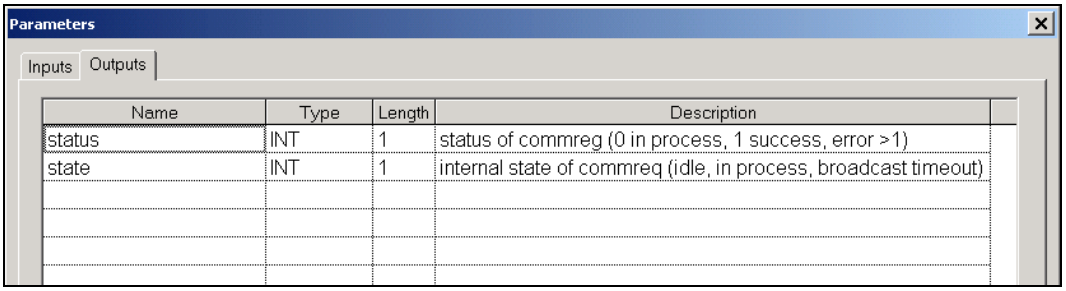


The inputs and outputs for the "C" block are:

X1 = cmd
X2 = r_s of the NIU
X3 = port
X4 = seq
X5 = seq_off
Y1 = status
Y2 = state

The configuration of the parameters for the block is already set up as shown below.





Both "C" blocks have the same input and output parameter setups.

Input and Output Parameters of the "C" Block

This section describes the input parameter values that are required to properly execute the "C" block, and the output parameters that are returned.

Enable Input

The block is executed if "Enable" receives power flow. The "C " block must be called every scan of the PLC.

Input Parameter (X1)(CMD): Input X1 (CMD) is a pointer to a memory location that contains the command parameters. The content of the data in the CMD memory location is shown next in this section.

Format	Int (i.e. %R100). Do not use a constant.
Length	8 X1 must be configured as Type: Int; Len:8
Usage	Modbus Query Parameters

Input Parameter (X2)(r_s): This input is a pointer.

Format	Int (i.e. %R211), example uses a constant (2)
Length	1 Note X2 must be configured as Type: Int; Len:1
Usage	Rack/Slot location of the Ethernet NIU.

Input Parameter (X3)(port): This input is a pointer.

Format	Int (i.e. %R212) , example uses a constant (1)
Length	1 Note X3 must be configured as Type: Int; Len:1
Usage	Port. Valid numbers 1, 2 Note 19, 20 (task numbers will also work)

Input Parameter (X4)(seq): This input is a pointer.

Format	Int (i.e. %R213) , example uses a constant (8 - %R)
Length	1 Note X4 must be configured as Type: Int; Len:1
Usage	Local Buffer Segment Selector. A 150 register buffer is required for the "C" block, this buffer must not be used by the user program. Valid numbers are: 8, 196 (%R, %W)

Input Parameter (X5)(seq_off): This input is a pointer.

Format	Int (i.e. %R214) , example uses a constant (9000)
Length	1 Note X5 must be configured as Type: Int; Len:1
Usage	Local Buffer Offset: Valid numbers 1 to 125 less then end of segment

OK Output

The OK output is on if the command is properly formed and a query is sent. The OK output is off if the command has a problem that stops a query from being sent.

Output Parameter (Y1)(status): This output is a pointer. Y1 must be an address where the "C" block can write results.

Format	Word (i.e. %R98). Do not use a constant for Y1.
Length	1 Note Y1 must be configured as Type: Int; Len:1
Usage	Status Parameter
First word	Completion Code

Output Parameter (Y2)(state): This output is a pointer. Y2 must be an address where the "C" block can write results.

Format	Int (i.e. %R99). Do not use a constant for Y2.
Length	1 Note Y2 must be configured as Type: Int; Len:1
Usage	State Parameter – used by "C" block to keep track of state of query.

Content of the Command Block

The CMD input parameter points to the following additional parameters for the command. Like a COMMREQ, the CMD input is set up by Block Moves or Data Inits.

Word	***Contains***	***Description***	
First word	Modbus Query Command: 8002	The "C" block sets this input parameter to "0" receiving the 8002 command. The input parameter must not be loaded with 8002 again until the "C" block returns a value greater than "0" in the Output Parameter (Y1). If 8002 is loaded again, it is ignored until the "C" block returns a value greater than "0", then the new 8002 command is executed.	
	Modbus Query Command: 8006	Use this command number instead of 8002 for Function Codes 3, 4, and 16, if the slave device(s) use word-swapped registers. Operation is the same as described above for command 8002. Number of Registers is the number of reals or dwords	
Second word	Modbus Slave ID	Valid numbers: 0 – 247, 0 is used for Broadcast	
Third word	Modbus Function Code	1	Read Outputs
		2	Read Inputs
		3	Read Holding Registers
		4	Read Input Registers
		5	Set / Clear One Coil (broadcast allowed)
		6	Write One Register (broadcast allowed)
		7	Read Exception Status
		8	Subfunction 0: Loopback Subfunction 1: Restart Communication (broadcast allowed) Subfunction 4: Enter Listen-only Mode (broadcast allowed)
		15	Write Multiple Coils (broadcast allowed)
		16	Write Multiple Registers (broadcast allowed). Use this Function Code with either 8002 (non-word-swapped) or 8006 (word-swapped) data. DO NOT broadcast this command to slaves that may have different data formats.
		17	Report Device Type
Fourth word	Function Code-dependent, as listed at right	1, 2, 3, 4	Start address in Slave for Read
		5	Coil Number in Slave
		6	Register Number in Slave
		7	not used
		8	Subfunction 0: 0 (must be zero) Subfunction 1: 1 (must be one) Subfunction 4: 4 (must be 4)
		15, 16	Start address in Slave for Write
		17	not used

Word	Contains	Description	
Fifth word	Function Code-dependent, as listed at right	1	Numbers of Output points to read (bits); max = 2000
		2	Numbers of Input points to read (bits); max = 2000
		3, 4	For command 8002: this is the number of Registers to read (16 bit words); maximum = 125. For command 8006, this is the number of 32-bit reals or dwords to read, maximum = 62.
		5	1 = set coil, 0 = clear coil
		6	Value to write to Register
		7	not used
		8	Subfunction 0: data pattern used in loopback message (any value) Subfunction 1 and : 0 (must be 0) Subfunction 4: not used
		15	Numbers of Output points to write (bits); max = 19200
		16	For command 8002, this is the number of Registers to write (16 bit words); maximum = 120. For command 8006, this is the number of 32-bit reals or dwords to write; maximum = 60.
		17	not used
Sixth word	Local Address Segment selector. Numeric values for memory types are: R = 8 AI = 10 AQ = 12 I = 16 Q = 18 T = 20 M = 22 G = 56 W = 196	1, 2	Reference Table in this master to put the values that are read. Valid Reference tables are: I, Q, T, M, G, AI, AQ, R, W
		3, 4	Reference Table in this master to put the values that are read. Valid Reference tables are: AI, AQ, R, W
		5, 6	not used
		7	Reference Table in this master to put the returned exception status Valid Reference tables are: I, Q, T, M, G, AI, AQ, R, W
		8	Not used
		15	Reference Table in this master to get the values that are written: Q, R, W
		16	Reference Table in this master to get the values that are written: AI, AQ, R, W
		17	Reference table in this master to put Device Type information: AI, AQ, R, W
Seventh word	Local Address offset. Function code-dependent, as listed at right.	1, 2, 3, 4	Location in Reference table to put read values
		5, 6	not used
		7	Location in Reference table to put read values
		8	not used
		15, 16	Location in Reference table to get write values
		17	Location in Reference table to put Device Type info
Eighth word	Timeout	Valid numbers – 0 to 10,000 milliseconds (10sec)	

Operation of the "C" Block

This chapter focuses on Local User Logic use of Modbus Master. Remote COMMREQ Calls are described in chapter 8. Most of the setup information for Modbus Master is the same for both Remote COMMREQ Call and Local User Logic operation. The setup of the CMD input for the "C" block MB_Px is the same for the Remote COMMREQ Call CMD input, except that for the Remote COMMREQ Call CMD input, it starts with the seventh word. That is the word that corresponds to the COMMREQ command number in all the other Remote COMMREQ Call commands.

The "C" block executes Serial I/O COMMREQs to do the Modbus Communication. The sequence to execute a Modbus Query or a series of Modbus Queries is as follows:

1. Use a one-shot to set up a communication command block of 8 words (as shown previously in this chapter). This data will be the CMD (X1) input to the "C" block.
2. When the "C" block sees 8002 or 8006 in the first word of the command block, the "C" block performs a Modbus Query based on the rest of the values in the command block.
3. The "C" block validates the command block inputs and then:
    i. Writes a zero back into the first word of the command block
    ii. Writes "2" into the Output Y1 (Status) of the "C" block if the command block is good and a communication is started. If there is an error in the command block or port setup, an error code is written into Output Y1 (Status) and a "0" is written into the first word of the command block.
4. If the command block was correct, the Modbus Query is sent and the "C" block returns a Success (1) or Error Code to the Output Y1 Status when the communication finishes or a timeout occurs.
5. If the value 8002 (or 8006) is in the first word of the command block, the "C" block starts the process again for another Modbus Query.
6. If both ports are being used, each port must have its own "C" block. Each one works independently of the other.

Execution of the Modbus Master Function Codes

Execution of Function Codes 1 and 2 (Read Outputs, Read Inputs)

If the Local Address Segment selector specified in the Command Block is for a discrete memory type (%I, %Q, %T, %M, %G), the Ethernet NIU retrieves the exact number of points requested, and places them in local memory starting at the exact memory offset specified. Only the exact number of bits specified is written to local memory.

If the Local Address Segment selector is for a word memory type (%R, %W, %AI, %AQ), the Ethernet NIU retrieves the number of points requested from the slave. The points are packed into local word memory starting at the low bit of the first specified word. If the number of bits retrieved is not a multiple of 16, the extra bits of the last word are filled with zeros.

Execution of Function Codes 3 (Read Holding Registers) and 4 (Read Input Registers)

Function codes 3 and 4 can be used to read register or Real/dword data from individual Modbus slaves. These commands cannot be broadcast.

To read 16-bit register data from a slave, Modbus Query 8002 should be used to send function code 3 or 4.

However, when reading data types that are Reals or Dwords from some types of Modbus slaves, the two registers that form the 32-bit data type may be reversed. For slaves that have that data format, Modbus Query 8006 should be used instead of 8002 to send function code 3 or 4. When the Ethernet NIU receives Modbus Query 8006, the ENIU executes the requested Modbus function code, automatically swapping the two 16-bit words of the data within each Real/dword value. In the Command Block for command 8006, the Number of Items is the actual number of Reals or Dwords the command should operate on. This is different than command 8002, where the number of items is the number of registers.

Execution of Function Code 7 (Read Exception Status)

If the Local Address Segment selector specified in the Command Block is for a discrete memory type (%I, %Q, %T, %M, %G), the Ethernet NIU writes the slave's Exception status into 8 bits of local memory starting at the exact memory offset specified.

If the Local Address Segment selector is for a word memory type (%R, %W, %AI, %AQ), the Reception status is written into the specified word.

Execution of Function Code 8, Subfunction 0 (Loopback)

No data is returned for Function Code 8, Subfunction 0. If the Loopback succeeds, the Status is set to success (1). If the Loopback fails, either a Timeout or Loopback Fail error code (returned data does not match sent data) is returned.

Execution of Function Code 8, Subfunction 1 (Restart Communication Interface)

The result of Function Code 8, Subfunction 1 depends on whether Function Code 8, Subfunction 4 (Listen-only Mode) was previously sent to the slave.

- If the Slave is in Listen-only mode, the Function Code 8, Subfunction 1 Query results in a timeout. The Slave will restart communications. If another Function Code 8, Subfunction 1 is sent, a status of success should occur.
- If the slave was not in Listen Only mode when a Function Code 8, Subfunction 1 is sent , a status of success is returned.

Execution of Function Code 8, Subfunction 4 (Enter Listen Only Mode)

Function Code 8, Subfunction 4 puts the slave in Listen Only Mode. No response is sent to Function Code 8, Subfunction 4. Function Code 8, Subfunction 4 returns a status code of "Broadcast Timeout". The status code returned is a Broadcast Timeout if Function Code 8, Subfunction 4 is sent to a single slave or is broadcast to all slaves. The status code Broadcast Timeout is not returned until the time specified in command block word 8 has expired.

Execution of Function Code 15 (Write Multiple Coils)

If the Local Address Segment selector specified in the Command Block is for a discrete memory type (%Q), the Ethernet NIU sends the exact number of points requested to the slave.

If the Local Address Segment selector is for a word memory type (%R, %W), the Ethernet NIU retrieves the appropriate number of words required to provide the requested number of bits, starting with the word specified in command block word 6. If the number of points is not a multiple of 16, the appropriate number of bytes is sent in the Query to accommodate the number of points. Any extra points in the last byte contain the value read. See details on the Modbus slave to see how it deals with these extra bits.

Execution of Function Code 16 (Write Multiple Registers)

Function code 16 can be used to write multiple register or real/dword values to Modbus slaves. Although Function Code 16 can be broadcast, that should NOT be done if some slaves have word-swapped data and others do not, as explained below.

To write multiple 16-bit registers to a slave, Modbus Query 8002 should be used to send Function Code 16.

When writing data types that are Reals or Dwords to some types of Modbus slaves, the two registers that form the 32-bit data type must be reversed. For slaves with that data format, command 8006 should be used to execute function code 16. When the Ethernet NIU receives command 8006, the ENIU automatically swaps the words of data before sending it to the slave.

Execution of Function Code 17 (Report Device Type)

For Report Device Type, the Ethernet NIU writes the slave information into the 5 consecutive words specified in command word 6. Only the low-order byte in each word is meaningful. The high-order byte in each word is set to zero. For the meaning of each word consult the documentation for the Slave device.

For a GE Fanuc PLC, Modbus Slave the meaning is:

Word 1 – Device Type - PLC family
Word 2 – Run Status of Slave 0 = stopped, 256 = running
Word 3 - Device Model – CPU type within PLC family
Word 4 – zero
Word 5 – zero

Modbus Communications Status Codes

The Y1 output of the "C" block gives the status code for the Modbus communication. This value that should be monitored to determine when a communication is complete and whether it succeeded or failed.

Error codes have a Major code and a Minor Code. The Major code is in the low-order byte, and the Minor code is in the high-order byte. Error codes are expressed in Hexadecimal, and are most easily viewed in that format.

Minor	Major	Description
0	0	In Process (or no Modbus Query attempt since power-up)
0	1	Success
1	1	Broadcast Timeout (this is success for a Query to broadcast ID (0)).
5	3	Bad Port Number – Port number must be 1 or 2 (19 & 20 will also work).
6	3	Bad Slave ID – Slave ID must be in the range 0-247
7	3	Bad Function Code - Function Code must be 1,2,3,4,5,6,7,8,15,16,17
8	3	Bad Broadcast Func Code – Only Function Codes 5,6,8 subfunction 4,15,16 support broadcast
a	3	Start Address is Zero – start address (command word 3) must be more than 0
b	3	Too Many Items - – num of items must be more than 0
c	3	Bad Local Seg Selector
21	3	Bad Local Seg Selector
22	3	Bad Local Address Offset – Start Addr + num of items more than size of segment
23	3	Bad set/Clear Coil Value - must be (0 = Clear, 1 = Set, FF = Set)
15	3	Bad Port Type – Hardware configuration of port must be "Serial I/O"
16	3	Bad Rack Slot - NIU is not in slot specified in "C" block
17	3	Bad COMMREQ command number – Command word 1 must be 8002 or 8006
19	3	Bad Command Code – Command word 1 must be 8002 or 8006
20	3	Unexpected State –call Technical Support
1	4	Parity Error received
2	4	Framing Error received
3	4	Bad CRC received
5	4	Overflow Error received
7	4	Multiple UART Errors
1	5	Timeout – response was not received within timeout period
2	5	Transmit Timeout – Query was not sent check CTS signal
10	6	Bad Buffer Seg Select – "C" block Input 4 must be 8, %R or 196 %W
11	6	Bad Buffer offset – "C" block Input 5 not a good value; need space for 150
x	8	Exception response received. x is the number of the exception.

Modbus Communication State

The Y2 output of the "C" block gives the state of the Modbus communication:

0 = Idle

2 = Waiting for Response

3 = Timing out after sending a Broadcast message

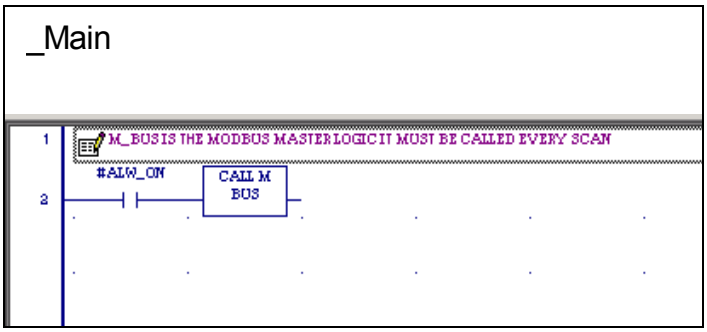
Programming Examples

This section describes application logic to implement Modbus Master communications from:

- Local User Logic in the Ethernet NIU
- Remote COMMREQ Call Communications from the Master

Example 1: Modbus Master Using Local User Logic

The Modbus Master Ladder code is in the Ladder block M_BUS. This block must be called every scan. The example below shows the Ladder block M_Bus being called from _Main.



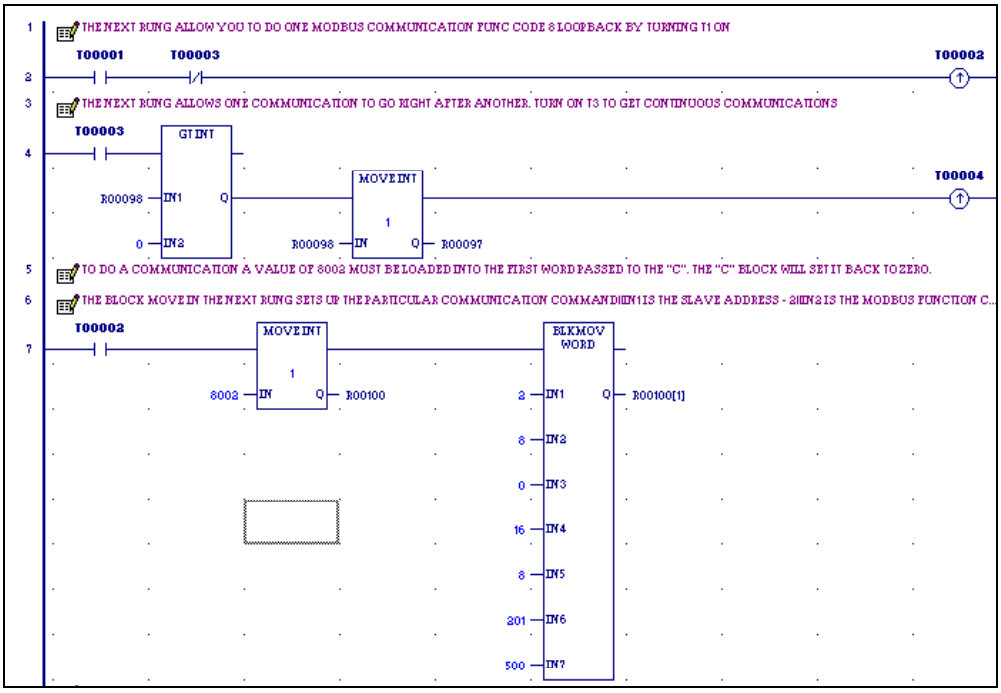
The call in _Main should be at either the beginning or the end of _Main.

Rung 1: An individual Modbus message to do Function Code 8 Loopback can be sent by toggling %T1. A message to Read Registers can be sent continuously, and as quickly as possible, by turning on %T3.

Rung 4 checks for the message to be complete with the GT_INT instruction. The move instruction in Rung 4 puts the completion status in %R00097. Rung 4 then activates a one-shot to start another communication. To do different Modbus Query messages or to do error recovery (retries), the one-shot %T0004 would need to be used to load different parameters into the BLKMOV in Rung 9.

The BLKMOV in rung 7 sets up a Modbus Query to:

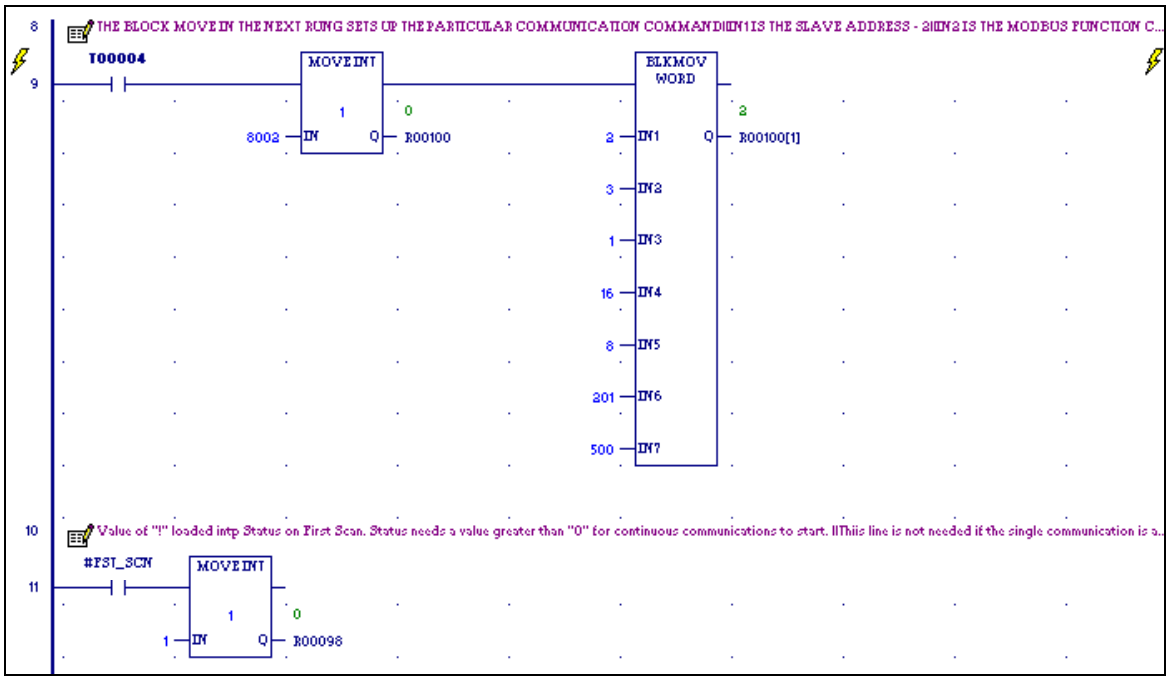
- Slave #2,
- Function Code 8 Loopback,
- Timeout is 500 milliseconds.



The BLKMOV in rung 9 sets up a Modbus Query to:

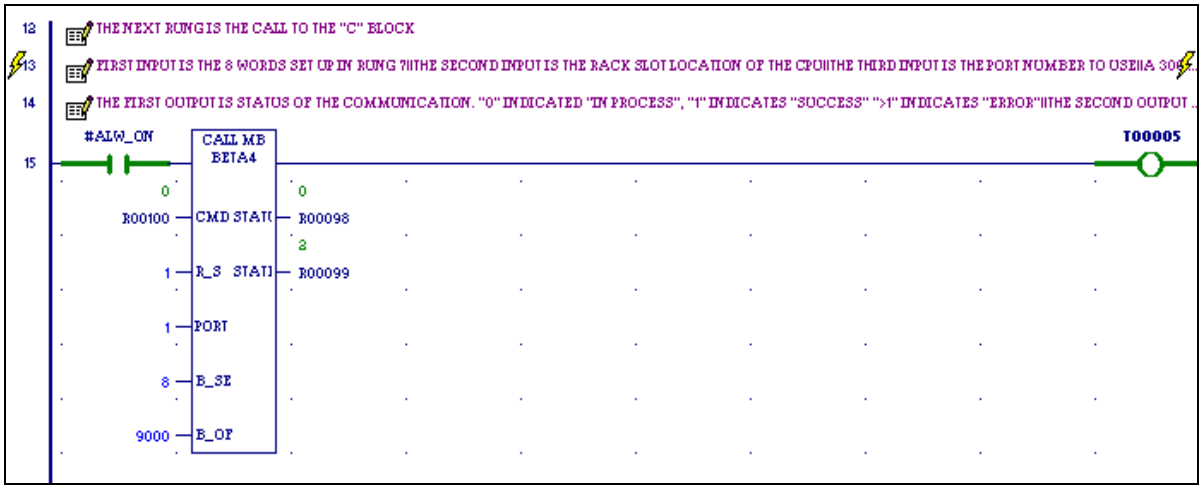
- Slave #2,
- Function Code 3 Read Registers,
- Starting at address 400001 (first Register),
- Read 16 Registers,
- Put the data in %R registers
- Starting at %R201,
- Timeout is 500 milliseconds.

Rung 11 is used to initialize the Status value (%R0098) to “1”. This is allows the continuous communication to start if the single communication is not done first.



Rung 15 of the example logic calls the “C” block. Setup for the example “C” block is:

- Command block of 8 registers starting at %R0100
- NIU located in Slot 1 (Must be rack 0)
- Communication to use Port 1 (RS232 port) of the NIU.
- Internal 150 Register buffer in %R memory
- Starting at Register %R9000



Error-checking for the Modbus communication is done by monitoring %R00098 for the result of a Modbus Query. (0 in process, 1 success, error > 1).

Troubleshooting Tips

- If using both ports, the Run/Stop Switch must be enabled or a store of hardware configuration to RX3i will fail.
- If using %W make sure you configure %W, as it defaults to length of 0.

Example 2: Modbus Master Using RCC Communications

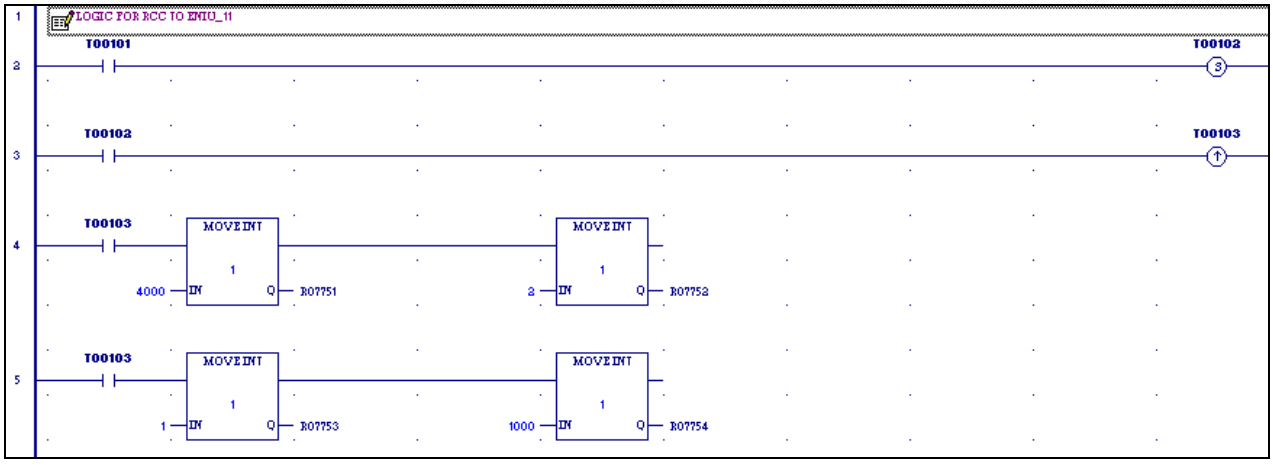
This example uses a Modbus Master command to Port 1 of the Ethernet NIU (Function Code 3 Read 24 Registers from Slave #1 starting at the first Register and put the data in %R101).

See the RCC call at the end of the example.

Rung 2 – Turning on %T101 sets %T102. That fires the one-shot %T103 in Rung 3. %T102 will be used to check for completion.

Rung 4 loads 4000 (the Module code for Modbus Master) into %R7751, and loads 2 (the slot number of the ENIU) into %R7752.

Rung 5 loads 1 (the port number) into %R7753, and loads 1000 (the timeout at the controller in milliseconds) into %R7754.

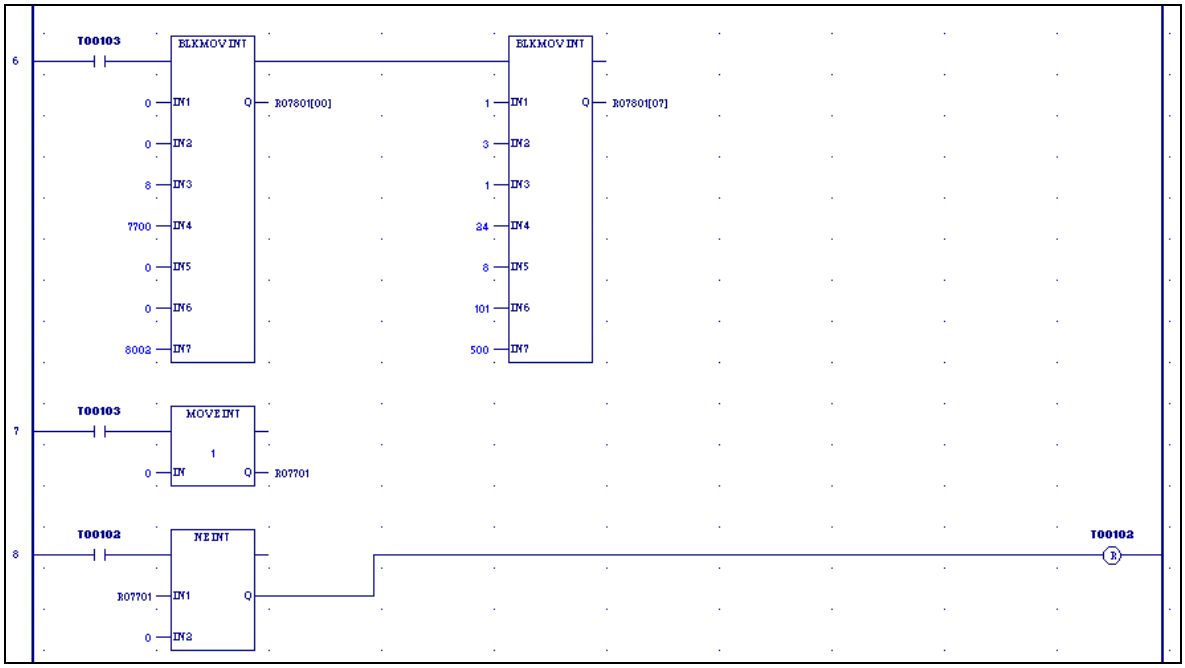


Rung 6 loads the RCC command into %R7801 thru %R7814

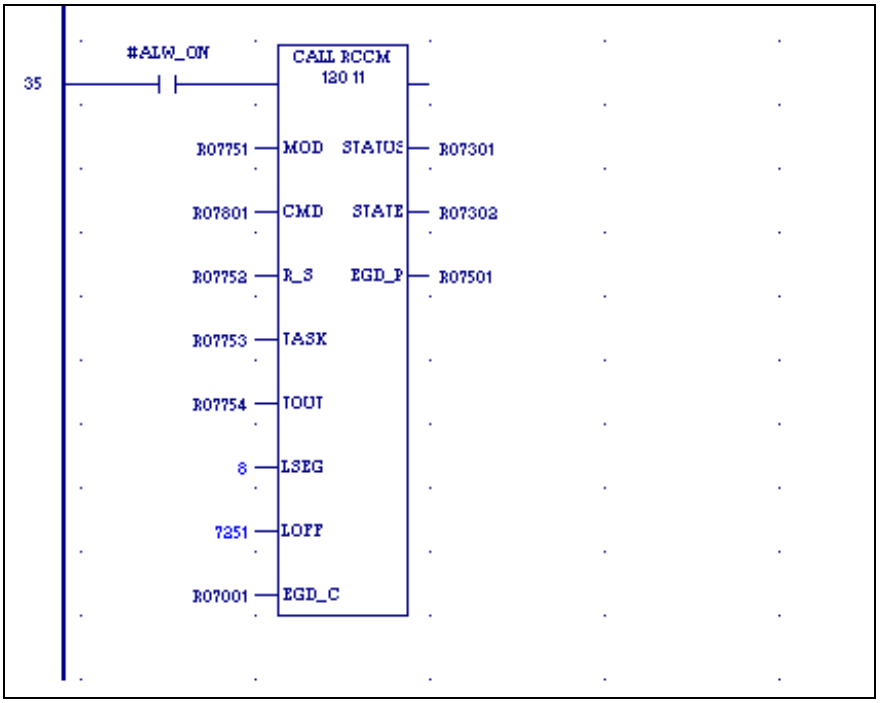
%R7801 – always 0 for Modbus Master
%R7802 – always 0 for Modbus Master
%R7803 - CSW segment selector 8 for %R memory
%R7804 – CSW offset (0 based) %R7701 is CSW in this example
%R7805 – always 0
%R7806 – always 0
%R7807 – COMMREQ Command number 8002 for Modbus Master
%R7808 – Slave ID (Slave #1)
%R7809 – Function Code (3: Read Registers)
%R7810 – Starting Address (“First Register 40001”)
%R7811 – Number of items (Read 24 Registers
%R7812 – Location to put data Segment Select (8 for %R memory)
%R7813 – Location to put data (offset %R101)
%R7814 – timeout in Ethernet NIU (500 milliseconds)

Rung 7 zeros the COMMREQ Status Word (%R7701) to allow checking for compeltion

Rung 8 checks for COMMREQ Status Word going non-zero (Completion). The end of the rung has a Reset coil %T102 (not shown). “1” is success.



Rung 35 executes the Remote COMMREQ Communications call:

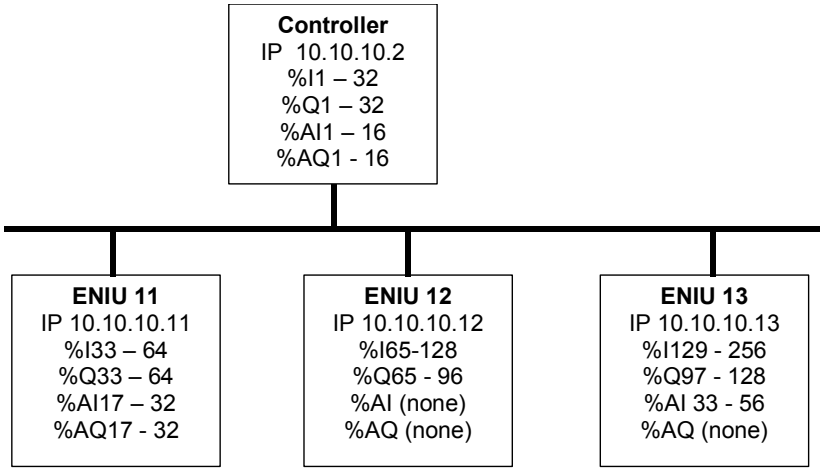


Appendix A I/O Quick Start Guide

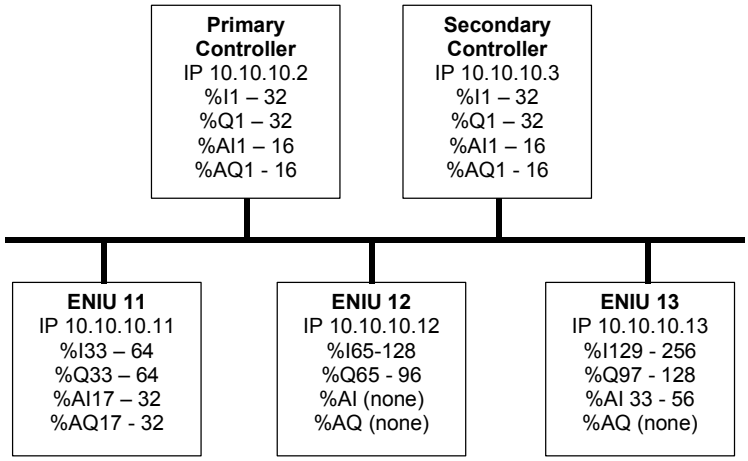
This appendix uses an example system with either one or two controllers to give an overview of the steps needed to set up an Ethernet NIU application.

1. Create List of ENIUs and I/O like the example shown below. The list should include:
   - the Controller(s) with IP addresses and local I/O.
   - Each ENIU with IP Address, and the I/O for the ENIU. Leave expansion space for additional I/O if the system is likely to change or grow.

Example System with One Controller (sample folder uses this IO mapping)



Example System with Redundant Controllers (sample folder uses this IO mapping)



A set of Machine Edition backup folders to start from is provided online at www.gefanuc.com.

2. Choose the folder that best matches your system configuration:
   - 3iENIU_Quick_Start_One_RX7i – Primary Controller and 3 ENIUs
   - 3iENIU_Quick_Start_Two_RX7i – Primary & Secondary Controller and 3 ENIUs
3. Restore the chosen folder.
   - The Quick Start project is set up with controllers named Primary Controller and Secondary Controller (two only) and 3 ENIUs named ENIU_11, ENIU_12, ENIU_13. You can rename the folder and the devices as appropriate. If you need fewer ENIUs, delete the ones you don't need. If you need more ENIUs, add a target with the type of Remote I/O PACSystems RX3i Ethernet You will need to set up the new ENIUs for proper operation.
   - The sample folder should be tried with the existing IP Addresses. Change the IP address of your computer and use an isolated network with your PC, the controller(s), and ENIUs to gain familiarity with using the equipment. Then change the IP addresses to what will be used in your system.
4. Using the list you created in step 1,
   - Check the CPU model for the controller and if necessary change it to match the controller you are using.
   - If the IO mapping is different than in the sample folder, the EGD exchanges will need to be adjusted. NOTE: the EGD exchange (Outputs_from_Primary/ Secondary_to ENIU) should NOT be changed. The controller sends out %Q 1 to 2028 and %AQ 1 to 512. All ENIUs receive these outputs but only the ENIU that has these reference configured will use them. The exchange (Inputs_from_ENIUxx) will need to be changed in both the

controller and ENIU if the %I and/or %AI from a ENIU to the controller is different than in the sample folder. NOTE: Inputs must be mapped directly from an ENIU to the controller ie if controller ENIU_11 uses %I33 to %I64, the exchange in the controller must map these inputs to %I33 to %I64 in the controller. NOTE: Do not duplicate IO reference address in multiple ENIUs as unpredictable results will occur.

5. If you are changing the IP addresses of the devices, you need to change the following items:
   - the IP address of each device. This must be done in two places: in the properties of the target (how programmer connects) and in the Ethernet settings in hardware configuration (CPU TAB for controller and Ethernet module TAB in ENIU.
   - the Subnet mask of each device (if required).
   - the Gateway IP address of each device (if required).
   - the Local Producer ID of each device and verify it is the IP address, in the Tree view select Ethernet Global Data of a target, the Local Producer ID is in the Property Inspector.
   - For Consumed Exchanges, change the Producer ID of the Exchange. In the tree view select the Consumed exchange, The Producer ID is in the Property Inspector
6. Set default values for variables in the ENIU(s). Default values are used (if selected): on power up before a controller is detected; and if communication is lost with the controller(s).
7. Download configurations to the hardware (controller(s) and ENIU(s)
8. When downloading to new or unknown hardware, first set the physical port property of the device in the programmer to a serial com port (com1) and connect via a serial cable to the power supply port. After the initial store of the configuration sets the IP Address, the physical port property can be set to Ethernet and the IP address entered. This will allow connection of the programmer via Ethernet.
9. Put the Controller(s) and ENIU(s) in “Run” if they are not running. I/O updates should now be occurring.

Checking I/O Operation

Inputs from the ENIUs should now appear in the Controller, and Outputs turned on in the controller should appear in the ENIU(s).

If the I/O is not updating, the status of the EGD exchanges should be checked. This is done using the Ethernet Station Manager. There are two versions (serial and Ethernet). The serial version is always available and can be used by connecting a cable from the Ethernet Station manager port on the Controller or ENIU to a COM port on your PC. Hyperterminal (found in accessories in Windows) can be used to access Station Manager. Set up a Hyperterminal session for the appropriate COM port with the following characteristics (9600 baud, 8 data bits, no parity, 1 stop bit, no flow control). Commands are typed in Hyperterminal and the Ethernet interface will respond.

1. Type in “node” <Enter>, to get a description on the device you are connected to.
2. To check the EGD communication type in “stat g”<Enter> which returns the status of the EGD exchanges.

   The status of the exchanges should be “Active (00H) or Active (01H) or Active (05H)” and the number of exchanges should increase if stat g is done twice.

   Stat g should be done on the controller and the ENIU to make sure both ends are correctly receiving exchanges.

   If a consumed exchange has a status of “Active (06H)” the exchange is not being received. The EGD Producer should be checked.

   If a consumed exchange has a status of “Active (0eH), the received length of the exchange does not match the configured length. Check both ends and fix the EGD exchange configuration and download the new configuration.

Sample RCC command (Modbus RTU Master – Read Registers)

1. This sequence will allow you to try sending out a Modbus RTU Query from Port 1 of ENIU_11.
2. In the hardware configuration of ENIU_11, set port 1 to Serial I/O, baud
3. The serial communications parameters (baud rate, parity, flow control, Slave ID number should be set to match the Modbus Slave being used.
4. To Test from port 1 to port 2 the settings would be 19,200 baud, odd parity, no flow control, Slave Id “1”.
5. Download the configuration to ENIU_11.
6. The appropriate cable is needed. Port 1 transmits on Pin 2 and receives on Pin 3. Typically a null modem cable is needed.
7. To test from port 1 to port 2, the Rs485 to Rs232 adapter can be placed on port 2 and a null modem adapter is used on the connection to port 1.
8. In the block “RCC” of the controller, turn on %T101. This will cause the controller to tell the ENIU to execute the Modbus Master query “Read Registers 1 to 24 from Slave 1 and put the data in Register 101 in the controller.
9. When you turn on %T101 the Activity LED on ENIU_11 Port 1 should blink.
10. The status of the RCC command is returned in %R14401 in the controller.

Appendix B Configuration Worksheets

This appendix consists of two sample configuration worksheets. Printed copies of these worksheets can be used to record configuration parameters.

- Inputs_from_ENIU
- Outputs_Pri_to_ENIU

Inputs_from_ENIU

This worksheet can be used to record parameters of the *Inputs_from_ENIU* exchange, the Ethernet Global Data exchange for the inputs sent from the Ethernet NIU to the controller(s).

Parameters of the Ethernet NIU's Produced Exchange

Ethernet Transmitter Module IP Address:		IP Address of the Ethernet Transmitter Module is recommended.
Local Producer ID:		
Exchange Property Inspector		
Exchange ID:	(1)	Match the Exchange ID of the Ethernet Transmitter Module if it has been changed.
Adapter Name:	(0.4)	Rack/slot location of the Ethernet Transmitter Module that will produce the exchange.
Destination Type:	Multicast	Do not change.
Destination:	(2)	Do not change unless group used for Ethernet Transmitter Module is changed.
Produced Period:	(10)	Increase this default If the system has more than 5 Ethernet NIUs.

Parameters of the Controller's Consumed Exchange

Controller IP Address:		IP Address of the Ethernet Transmitter Module.
Local Producer ID:		
Exchange Property Inspector		
Producer ID:		Ethernet NIU Producer ID.
Group ID:	2	Default is 0. Set to 2 for system with multiple Ethernet NIUs.
Exchange ID:	1	Do not change.
Adapter Name:	(0.1.0)	From CPU configuration.
Update Timeout:	32	Default is 0.

Parameters of the Exchange

Offset	*Variable*	*Ref Address*	*Ignore*	*Length*	*Type*	*Description*

Outputs_Pri_to_ENIUs

This worksheet can be used to record parameters of the *Outputs_Pri_to_ENIUs* exchange, the Ethernet Global Data exchange for the outputs sent to the Ethernet NIU from the primary controller.

Parameters of the Ethernet NIU's Consumed Exchange

Ethernet Transmitter Module IP Address:		
Local Producer ID:		IP address of the Ethernet Transmitter Module is recommended.

Exchange Property Inspector

Producer ID:		(Producer ID of the primary controller, usually the controller's IP address.)
Group ID:	1	Leave at default.
Exchange ID:	1	Leave at default.
Adapter Name:	(0.4)	Rack/slot location of the Ethernet Transmitter Module that will consume the exchange.
Update Timeout:	(32)	Should be 3 to 5 times the controller's Produced Period.

Parameters of the Controller's Produced Exchange

Controller IP Address:		
Local Producer IP Address:		(use of the same IP Address is recommended)

Exchange Property Inspector

Exchange ID:	(1)	Change only if the controller will produce more than one exchange).
Adapter Name:	0.1.0	
Destination Type:	Multicast	For Series 90 controller, this is Group.
Destination:	(1)	
Produced Period:	(10)	Defaults to 200. Do not set to less than 6ms,

Parameters of the Exchange

Offset	*Variable*	*Ref Address*	*Ignore*	*Length*	*Type*	*Description*
Status				16	Bit	
0.0				10	Word	
20.0		%Q0001		2048	Bit	
276.0		%AQ0001		512	Word	

H

I

L

M

N

O

P

Q

R

S

T