GFK-2439

New In Stock!

GE Fanuc Manuals

http://www.pdfsupply.com/automation/ge-fanuc-manuals/rx3i-plc/GFK-2439

rx3i-plc

1-919-535-3180

PACSystems RX3i Ethernet NIU Manual,

www.pdfsupply.com

Email: sales@pdfsupply.com

GFK-2439

New In Stock!

~~GE Fanuc Manuals~~

http://www.pdfsupply.com/automation/ge-fanuc-manuals/rx3i-plc/GFK-2439

rx3i-plc

1-919-535-3180

PACSystems RX3i Ethernet NIU Manual,

www.pdfsupply.com

Email: sales@pdfsupply.com

GE Fanuc Intelligent Platforms
Programmable Control Products

PACSystems® RX3i

Ethernet Network Interface Unit

User's Manual, GFK-2439C

March 2009

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 assumes no obligation of notice to holders of this document with respect to changes subsequently made.

GE Fanuc 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 Intelligent Platforms, Inc.

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

©Copyright 2009 GE Fanuc Intelligent Platforms
All Rights Reserved

Contents

Introduction 1-1

A PACSystems RX3i Ethernet NIU I/O Station 1-4
The PACSystems RX3i Ethernet NIU 1-5
Ethernet NIU Features 1-5
Specifications for IC695NIU001 1-7
The Ethernet Transmitter Module 1-8
Ethernet Transmitter Module Controls and Indicators 1-9
Ethernet Transmitter Module Specifications 1-10
Ethernet Interface Ports 1-10
Station Manager 1-11
Firmware Upgrades 1-11
Ethernet NIU COMMREQ Support 1-11
Modules and Baseplates in the I/O Station 1-12
Controllers on the Network 1-14
Number of Ethernet Interfaces 1-15
Templates for RX3i Ethernet NIUs 1-16
Available Template Sets 1-17
Programmer and Template Versions 1-18
Overview of Operation 1-19
Ethernet Global Data Features 1-20
Ethernet Global Data (EGD) Exchanges 1-20
Planning a New System 1-21

Installation 2-1

Meeting Agency Standards and Requirements 2-2
CE Mark Installation Requirements 2-2
Installation in Hazardous Locations 2-2
Installing the Ethernet NIU 2-3
Backplane Locations for the ENIU 2-3
Serial Ports 2-5
Programmer Connection 2-7
Firmware Upgrades 2-7
Ethernet Connections to the Ethernet Transmitter Module 2-8
Ethernet Cable 2-8
Embedded Switch 2-8
I/O Station Connections with a Single Controller 2-9
I/O Station Connections for Redundant Controllers with Dual LANs 2-10
I/O Station Connections with Redundant Max-On Controllers 2-11
Redundant Ethernet Cable Connections 2-14
Starting Up the Ethernet NIU 2-15

Template Sets for RX3i and RX7i Applications 3-1

Available Template Sets and C Blocks 3-2
Choices in Template Sets 3-2
C Blocks for the Application Program in the Controller 3-4
Downloading a Template Set 3-5
Bringing a Template Set into the Programmer 3-5
Configuring the Controller(s) 3-7
Configuring Inputs for the Controller 3-10
Deleting EGD Exchanges for Extra Ethernet NIUs 3-11
Completing the Controller Configuration 3-12
Checking Input Operation in a Dual Controller Application 3-12
Configuring the Ethernet NIUs 3-13
Input References for Ethernet NIUs 3-14
Adjusting I/O in Dual LAN (and PPS Systems) 3-15
Example Configuration of Inputs from ENIU_01_LANA 3-16
Configuring the Controller Exchange for NO Discrete Inputs 3-17
Configuring the Controller Exchange for NO Analog Inputs 3-19
Configuration Examples for a Single LAN System 3-21
Example 1: Default Input Addressing, Single LAN 3-21
Example 2: Adding Inputs to the Configuration, Single LAN 3-23
Configuration Examples for a Dual LAN System 3-24
Example 3: Default Input Addressing, Dual LAN 3-24
Example 4: Adding Inputs to the Configuration, Dual LAN 3-27
Powerup Operation 3-29

Input Processing by a CPU 4-1

The Input_Processing Block 4-2
Configuring the Input Processing C Block 4-3
Symbolic Variables for the Input Processing Function 4-5
Input Processing – Error Codes 4-6
Redundant Controller, Dual LAN Considerations 4-8
Switching Logic 4-8
Predefined Signals for Custom Switching Logic 4-9
Dedicated Signals 4-11
Point Fault/Data Quality Feature 4-12
Enabling Point Fault References 4-12
Operation of Point Fault References 4-12

Configuring PC-Based Controllers 5-1

Configuring the Ethernet NIU 5-2
Configuring the PC-Based Controller 5-3
Set Up I/O and Control/Feedback Data 5-6

Hardware Configuration 6-1

Adding an Ethernet NIU Target to the Project 6-2
Completing the Hardware Configuration in the ENIU Target 6-4
Configuring the Ethernet NIU 6-5
Memory Tab for the Ethernet NIU 6-6
Faults Tab for the Ethernet NIU 6-6
Port Tabs for the Ethernet NIU 6-7
Configuring the Ethernet Transmitter Module in the I/O Station 6-15
Settings Tab for the Ethernet Transmitter Module 6-15
RS-232 Port Tab for the Ethernet Transmitter Module 6-16

Creating an Application without the Templates 7-1

Controller Configuration for Ethernet NIUs 7-2
Ethernet Interface Configuration in the Controller 7-3
Configuring the EGD Exchanges in the Controller Target 7-4
Configuring the Input Data Areas of the Input from ENIU Exchanges 7-4
Configuring the Number of Ethernet NIUs in the Controller Application 7-5
Configure Ethernet Global Data in the Ethernet NIU Target 7-6
Ethernet Global Data Exchange Descriptions 7-7
Setting Up Ethernet Global Data Exchanges 7-9
Viewing and Editing EGD Exchange Properties 7-9
Viewing and Editing the EGD Exchange Data Range Table 7-11
Configuring EGD Outputs from the Controller to Ethernet NIUs 7-12
Configuring the EGD Exchange Inputs_from_ENIU_xx 7-14
Properties and Data Ranges for SVC_Xchg_Pri_to_ENIU_xx 7-16
Properties and Data Ranges for SVC_Xchg_from_ENIU_xx 7-17
Properties and Data Ranges for Outputs_Pri_to_ENIUs_LANB 7-18
Properties and Data Ranges for Outputs_Sec_to_ENIUs_LANB 7-20
Properties and Data Ranges for Inputs_from_ENIU_xx_LANB 7-22
Properties and Data Ranges for SVC_Xchg_Pri_to_ENIU_xx_LANB 7-24
Properties and Data Ranges for SVC_Xchg_Sec_to_ENIU_xx_LANB 7-26
Properties and Data Ranges for SVC_Xchg_from_ENIU_xx_LANB 7-28

Ethernet Global Data 8-1

EGD Exchanges in an Ethernet NIU Target 8-2
EGD Addresses for Multiple Controllers on One LAN 8-2
EGD Exchanges for I/O, Status, and Control Data 8-3
Update Time for the I/O, Status, and Control EGD Exchanges 8-3
Inputs_from_ENIU_xx 8-4
Outputs_xxx_to_ENIUs 8-7
EGD Exchanges for Faults and Remote COMMREQ Calls 8-9
EGD Timing for the Project Templates 8-11
Setting Up SNTP Time Synchronization 8-13
Advantage of Using SNTP Time Synchronization 8-13

Time Synchronization and the AUP File in the Ethernet Transmitter 8-14
Disabling Fault Table SNTP Alarms when SNTP is Not Used 8-14

I/O Data - Control, Status, and I/O Data Formats 9-1

System I/O Data References 9-2
Data Memory in the Ethernet NIU 9-3
Dedicated Data Memory References Used in the Ethernet NIU 9-3
References Assigned in a 1.2x Ethernet NIU Target 9-3
References Assigned in a 1.3x and Higher Ethernet NIU Target 9-4
Discrete and Analog Outputs in the Ethernet NIU 9-5
Number of Outputs 9-5
Systems Requiring Over 2048 Discrete Outputs or 512 Analog Outputs 9-5
Exchanging Data with Two Controllers 9-6
ENIU Operation with Two Controllers and One Ethernet LAN 9-6
ENIU Operation with Two Controllers and Dual Ethernet LANs 9-6
Ethernet NIU Operation if No Data is Received 9-7
ENIU Operation if Data is Received from the Backup Controller 9-7
Control Data Format 9-10
Status Data Format 9-13
Using the Control and Status Data 9-16
Switching Control Back to the Primary Controller 9-16
Commanding Output Operation if Communication is Lost 9-16
Specifying Individual Output Defaults 9-17
Checking for Faults and Clearing Faults 9-17
Using the Optional Application-Specific Command Word 9-18

Diagnostics 10-1

Using the Status and Control Data for Fault Monitoring 10-2
Checking Faults in the Input Status Data 10-2
Clearing Faults in the Output Control Data 10-2
The Ethernet NIU Fault Tables 10-3
Viewing Faults in the Controller PLC Fault Table 10-3
Viewing Extra Fault Data in the Ethernet NIU's PLC Fault Table 10-3
Enhanced Fault Handling 10-5
Disabling Enhanced Fault Handling 10-5
The ENIU_Faults C Block 10-6
Clearing Faults Using SVC_Xchg_to_ENIU_xx 10-6
Symbolic Variables for Fault Handling 10-7
Using the Station Manager 10-8
Checking the IP Address of the Ethernet NIU 10-9
Checking Communications on the Network 10-10
Viewing the Exception Log 10-11
Checking for Stale Ethernet Global Data Status 10-11
Checking Exchanges with the STAT Command 10-12
When the STAT LED is ON 10-13

Testing Communications after Setup 10-14
Verifying that All Ethernet Global Data Exchanges are Working 10-14
Checking the Cable Connections 10-16
Troubleshooting Ethernet I/O 10-17
Checking the Network Connection 10-17
Checking Communications in the Programmer Watch Windows 10-18
Checking the Ethernet Global Data Status 10-18
Watching Ethernet NIU Status with the Controller 10-19
If You Can't Solve the Problem 10-20

Local Program Logic in the Ethernet NIU 11-1

Using the Local Logic Block 11-1
Reference Table Restrictions for User Logic 11-2
Restricted Addresses 11-2
Addresses Written to by EGD Exchanges 11-2
Using COMMREQs in the Local Logic 11-3

Remote COMMREQ Calls 12-1

Using Remote COMMREQ Calls 12-2
Remote COMMREQ Call Functionality in the Ethernet NIU 12-2
Remote COMMREQ Call Functionality in the Controller 12-2
Remote COMMREQ Call Operation 12-4
Configuring EGD Exchanges for Remote COMMREQ Calls 12-5
EGD Exchanges for Remote COMMREQ Calls 12-6
Configuring EGD Exchanges for RCC (Version 1.3x and later ENIU Target) 12-7
Configuring the ENIU's Consumed Exchange to Receive RCC) 12-7
Configuring the ENIU's Produced Exchange for Response to RCC 12-8
Configuring the Controller's Produced Exchange to Send RCC 12-9
Configuring the Controller's Consumed EGD Exchange for RCC Response 12-11
Configuring Exchanges if Multiple ENIUs Will Receive RCC Commands 12-12
Configuring EGD Exchanges for RCC (Version 1.2x ENIU Target) 12-13
Configuring the Controller's Produced Exchange to Send RCC 12-13
Configuring the Controller's Consumed EGD Exchange for RCC Response 12-14
Configuring Exchanges if Multiple ENIUs Will Receive RCC Commands 12-15
Adding the RCC C Block to the Controller Target 12-16
Adding the C Block Call to Controller Logic 12-18
Adding Logic to Sequence RCC Commands and Check Return Status 12-20
Monitoring Remote COMMREQ Calls for Completion 12-23
Diagnostics for Remote COMMREQ Calls 12-24
COMMREQ Status Word 12-24
C Block Status Output – Codes 12-24
C Block State Output – Codes 12-24
Troubleshooting 12-25

Remote COMMREQ Calls in a Redundancy System 12-26
Read RCC Command at Switchover 12-27

COMMREQs for Remote COMMREQ Calls 13-1

COMMREQS Supported by Remote COMMREQ Calls 13-2
COMMREQs for DeviceNet Master Modules 13-3
DeviceNet Master Modules, COMMREQ 1: Send Device Explicit 13-3
DeviceNet Master Modules, COMMREQ 4: Get Detailed Device Status 13-7
DeviceNet Master Modules, COMMREQ 5: Get Status Information 13-9
DeviceNet Modules, COMMREQ 6: Get Input Status from a Device 13-11
DeviceNet Modules COMMREQ 7: Send Device Explicit Extended 13-13
DeviceNet Master Modules, COMMREQ 9: Read Module Header 13-16
Read Module Header, COMMREQ Example 13-16
Read Module Header, Reply Data Format 13-17
COMMREQs for Genius Bus Controller Modules 13-20
Genius Bus Controller Modules, COMMREQ 8: Enable/Disable Outputs 13-20
Genius Bus Controller Modules, COMMREQ 13: Dequeue Datagram 13-21
Genius Bus Controllers, COMMREQ 14: Send Datagram Command 13-24
Genius Bus Controllers, COMMREQ 15: Request Datagram Reply 13-25
COMMREQs for RX3i Analog Modules with HART Communications 13-26
COMMREQ 1, Get HART Device Information 13-27
COMMREQ 2, Send HART Pass-Thru Command 13-29
COMMREQs for an RX3i Profibus Master Module 13-32
Profibus Master Module, COMMREQ 1: Get Device Status 13-32
Profibus Master Module, COMMREQ 2: Get Master Status 13-34
RX3i Profibus Master Module, COMMREQ 4 : Get Device Diagnostics 13-37
Profibus Master Module, COMMREQ 5: Read Module Header 13-38
Profibus Master Module, COMMREQ 6: Clear Counters 13-40
COMMREQ for RX3i and Series 90-30 Motion Controller Modules 13-41
Motion Controller Modules, COMMREQ E501: Parameter Load 13-41
COMMREQ for High-Speed Counter Modules 13-42
High-Speed Counter Modules, COMMREQ E201: Send Data Command 13-42
COMMREQs for MODBUS RTU Master on the RX3i ENIU Serial Ports 13-43
MODBUS Master COMMREQs Command Block- All Function Codes 13-43
COMMREQ Error Codes by Module Type 13-45
PACSystems RX3i and Series 90-30 DeviceNet Modules 13-45
PACSystems RX3i and Series 90-30 Genius Bus Controllers 13-46
PACSystems RX3i Analog Modules with HART Communications 13-47
PACSystems RX3i Profibus Master Module 13-48
PACSystems and Series 90-30 Motion Controllers 13-48
PACSystems RX3i and Series 90-30 High Speed Counter Modules 13-49
Status Values for MODBUS Master Communications 13-50

Generic Remote COMMREQ Calls 14-1

Generic Remote COMMREQ C Block 14-2
RCC C Block Generic COMMREQ Input Parameters 14-2
Using Generic COMMREQs 14-3

Modbus Master for the Ethernet NIU 15-1

MODBUS Master for the Ethernet NIU 15-2
CPU or Ethernet NIU Control of MODBUS Master Communications 15-3
Hardware Configuration for MODBUS Master 15-3
Software Function Blocks for MODBUS Master Communications 15-4
Version of C Software Function Block 15-4
Setting Up the C Function Block MB2_xxx for MODBUS Master 15-4
Input and Output Parameters of the C Block 15-6
Operation of the C Block 15-10
Execution of the MODBUS Master Function Codes 15-11
MODBUS Communications Status Codes 15-14
MODBUS Communication State 15-14
Programming Examples 15-15
Example 1: MODBUS Master Using Local User Logic 15-15
Troubleshooting Tips 15-18
Example 2: MODBUS Master Using RCC Communications 15-19

Upgrading a Release 1.2x Ethernet NIU Application 16-1

Determining the Ethernet NIU Application Needed 16-2
Update the Application if Fault Reporting will be Used 16-3
Update the Application if Generic RCC will be Used 16-3
Update the Application if Dual LANs will be Used 16-3
Saving the Existing Application Target Data 16-4
Save the Local_User_Logic Block 16-4
Save the Hardware Configuration 16-4
Save the Ethernet Global Data Configuration 16-5
Creating a New Version Target 16-6
Replace the Local Logic Block 16-6
Replace the Hardware Configuration 16-7
Replace the Ethernet Global Data Configuration 16-8
Adding Enhanced Fault Reporting and/or Generic RCC to a Version 1.2x ENIU Application 16-9
Existing EGD Exchanges 16-9
Adding/Modifying Ethernet Global Data Exchanges in the ENIU 16-9
Adding Fault Logic Blocks for Fault Reporting and/or RCC in the ENIU Application 16-12
Adding Enhanced Fault Reporting and/or Generic RCC to the Controller Application. 16-13
Symbolic Variables for Remote COMMREQ Calls 16-13
Symbolic Variables for the Enhanced Fault Reporting Block 16-13
Adding/Modifying Ethernet Global Data Exchanges in the Controller 16-14
Controller Program Blocks 16-17

Using Fault Reporting in the Control Program 16-18
Using Remote COMMREQ Calls in the Control Program 16-19
Adding a New Target to a Version 1.2x Application 16-22
Modification to Hardware Configuration 16-22
Modification to Ethernet Global Data Exchanges 16-23

Configuration Worksheets A-1

Primary Controller A-2
Secondary Controller A-3
Ethernet NIU (Complete for Each ENIU I/O Station) A-4
I/O Modules in the Ethernet NIU I/O Station A-5
Inputs_from_ENIU A-6
Outputs_Pri_to_ENIUs A-8

Chapter 1 Introduction

This manual describes operation and setup of an Ethernet NIU. It also includes information on Proficy Machine Edition projects that contain multiple Ethernet NIU targets and PACSystems controller targets that are pre-configured for various controller and LAN architectures.

Chapter 1: Introduction, describes the PACSystems RX3i Ethernet Network Interface Unit and other equipment in the I/O Station. This chapter also provides an overview of the features of the Ethernet NIU.

Chapter 2: Installation, provides additional installation information for the Ethernet NIU and I/O Station. 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*.

Chapter 3: Templates for RX3i and RX7i Applications, describes and explains how to use the preconfigured application templates for RX3i and RX7i controller systems.

Chapter 4: Configuring Input Arbitration for a Dual LAN System and for PPS Applications, explains how to configure and set up input arbitration in a Proficy Machine Edition dual LAN system that was not developed using the application templates, or for applications using Proficy Process Systems controllers.

Chapter 5: Configuring for PC-Based Controllers, describes additional configuration steps for a PC-based controller such as QuickPanel Control or PC Control.

Chapter 6: Module Configuration, describes the basic steps for configuring Ethernet NIUs and Ethernet Transmitter Modules.

Chapter 7: Creating an Application without the Templates, describes configuration steps for a system that is not developed using the application templates.

Chapter 8: Ethernet Global Data, provides additional details about Ethernet Global Data that can be used to customize the EGD exchanges in the project templates. This information can also be used as a reference when setting up a system that does not use the project templates.

Chapter 9: Control, Status, and I/O Data Formats, describes the data that an Ethernet NIU regularly exchanges with its controller(s).

Chapter10: Diagnostics, explains how faults are handled, how faults can be viewed, how to use the Station Manager to check communications, and troubleshooting steps.

Chapter 11: Local Program Logic in the Ethernet NIU, describes addressing requirements for local logic, and discusses considerations for including COMMREQs in the local logic.

Chapter 12: Remote COMMREQ Calls, explains how to set up the Remote COMMREQ Call (RCC) feature of PACSystems RX7i and RX3i controllers. The controller can issue Remote COMMREQ calls to pass a predefined set of COMMREQs to intelligent modules in an I/O Station via the Ethernet NIU.

Chapter 13: COMMREQs for Remote COMMREQ Calls, explains how PACSystems RX7i and RX3i controllers can use Remote COMMREQ Calls to send the COMMREQs to intelligent modules in the I/O Station.

Chapter 14: Generic Remote COMMREQ Calls, explains how additional types of COMMREQs can be sent to Ethernet NIUs.

Chapter 15: Modbus Master, 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.

Chapter 16: Upgrading a Version 1.2x Ethernet NIU Application, describes upgrading a version 1.2x Ethernet NIU application to include the additional features that are available in later versions.

Appendix A: Configuration Worksheets, provides two sample configuration worksheets that can be used to record configuration parameters of Ethernet Global Data Exchanges that will transfer input and output data between the controller(s) and Ethernet NIU(s).

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:

PACSystems Hot Standby CPU Redundancy User's Guide, GFK-2308. While this manual illustrates the Genius I/O LAN, the concepts are the same as used for systems based on the Ethernet NIU.

***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.

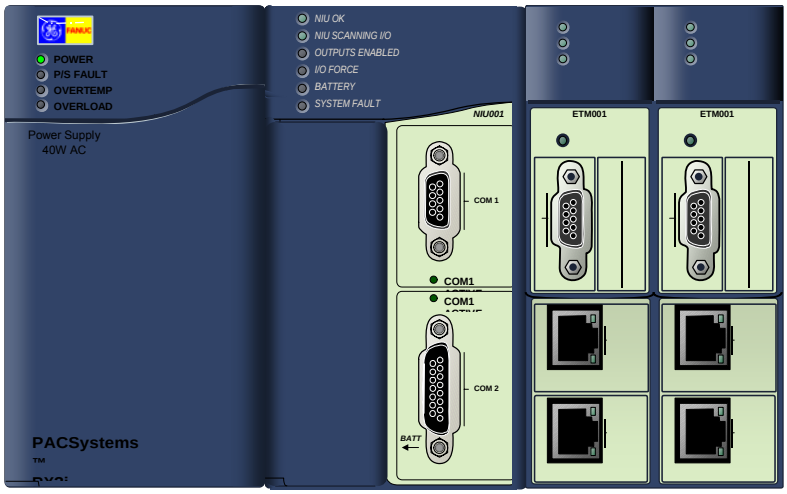
***Series 90-30 Module* Specifications*, GFK-0898*.** This manual is a collection of detailed module datasheets.

These user manuals, module datasheets, and other important product documents are available online at www.support.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 power supply (IC695PSxxxx)
- the RX3i Ethernet NIU (IC695NIU001)
- one or more RX3i Ethernet Transmitter modules (IC695ETM001), which interface the I/O Station and Ethernet NIU to the Ethernet network and to the controller.



- an RX3i Universal Backplane (IC695CHS0xx)
- 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 modules 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.

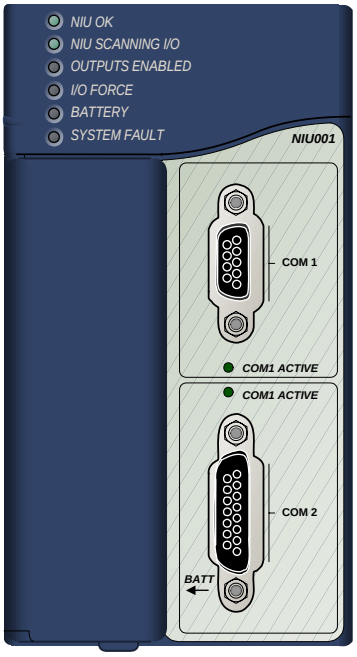
The Ethernet NIU automatically provides the controller with status information in each exchange. The application program logic in the controller can monitor this status data, and issue appropriate commands to the Ethernet NIU.

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.

The Ethernet NIU can access one block of program logic called the Local Logic Block, which can be up to 20K bytes in size. The Proficy programming software automatically includes the proprietary logic blocks needed for the Ethernet NIU application.

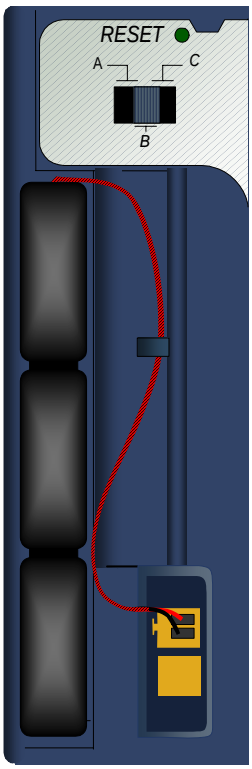
Ethernet NIU Features

- 20Kbytes of optional local logic. Supports all languages except C programming.
- 10 Mbytes of built-in flash memory for local user data storage.
- Battery-backed calendar clock and fault tables.
- 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 and redundant (dual) LANs
- Fault Reporting to controller(s)
- Remote COMMREQ Execution



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.



If a new battery is installed when no battery is currently installed, the new battery must be installed while the CPU has power. Otherwise, the CPU may not power up. If that happens, remove the battery, power-cycle the CPU, then reinstall the battery.

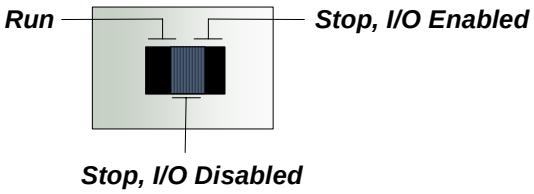
Disposal of lithium batteries must be done in accordance with federal, state, and local regulations. Be sure to consult with the appropriate regulatory agencies before disposing of batteries.

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.

Switches

The Ethernet NIU has two switches that are located in the battery area behind the protective door, as shown above. The Reset switch is not used.

The 3-position switch labeled A-B-C is the Run/Stop switch. Unlike the Run/Stop on a CPU module, on an Ethernet NIU the use of this switch is disabled by default. If the switch is to be used to control the Run or Stop mode operation of the NIU, clear faults, and/or prevent writing to program memory or configuration, its functionality must be enabled on the Settings Tab of the Ethernet NIU configuration in the folder, and stored to the ENIU.



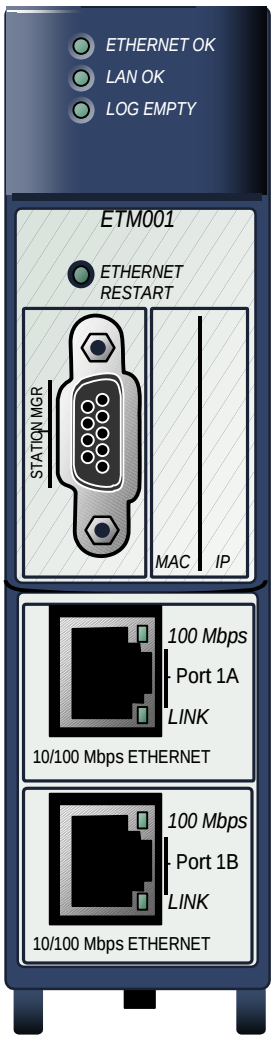
Specifications for IC695NIU001

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 and with host computers running 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:

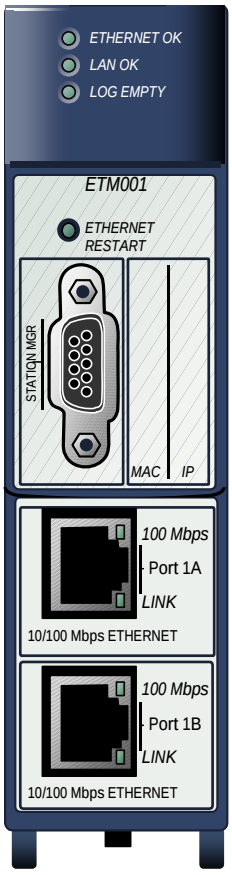
- Implements EGD Class 1 and Class 2 capabilities.
- Firmware upgrades using the WinLoader software utility.
- Periodic data exchange using Ethernet Global Data (EGD).
- EGD Commands to read and write PLC and EGD exchange memory over the network.
- TCP/IP communication services using SRTP.
- Support 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.
- Version 5.61 or higher firmware is required for use in an Rx3i Ethernet NIU I/O Station. Higher versions are required to use specific features of the ENIU application templates.

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 Support

The Ethernet NIU supports 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 in the RX7i or RX3i CPU interfaces to the C block. The C block sends COMMREQ commands to the Ethernet NIU in an Ethernet Global Data Exchange. The Ethernet NIU executes the COMMREQ and sends the results back to the RX7i or RX3i using another EGD exchange. The following COMMREQs can be sent in this way:

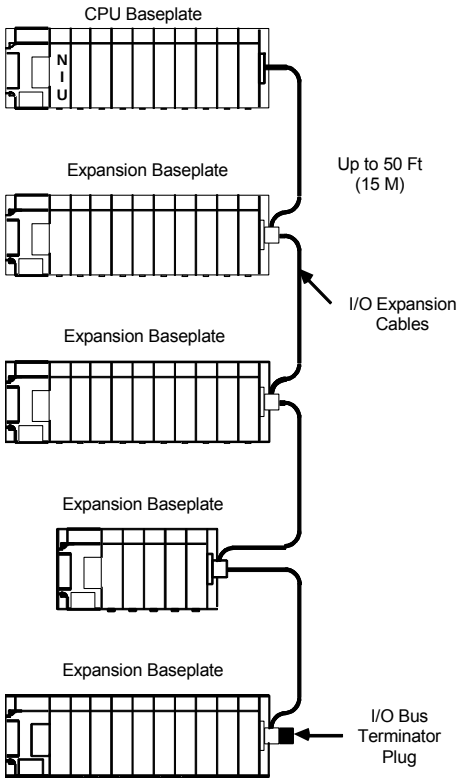
- Modbus Master – function codes 1, 2, 3, 4, 5, 6, 7, 15, 16, 17
- Genius – enable/disable outputs, switch BSM, clear fault, clear all faults, assign monitor, read diagnostic
- PROFIBUS Master – COMMREQs 1, 2, 4, 5, 6
- Motion (DSM314/DSM324) – load parameters
- High Speed Counter – Data command
- DeviceNet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

See chapters 12 and 13 for more information.

In addition, any COMMREQ supported by a module in the Ethernet NIU can be sent as a Generic COMMREQ, with the exception of DeviceNet Master Send Extended Explicit Message. See chapter 14 for information on Generic COMMREQs.

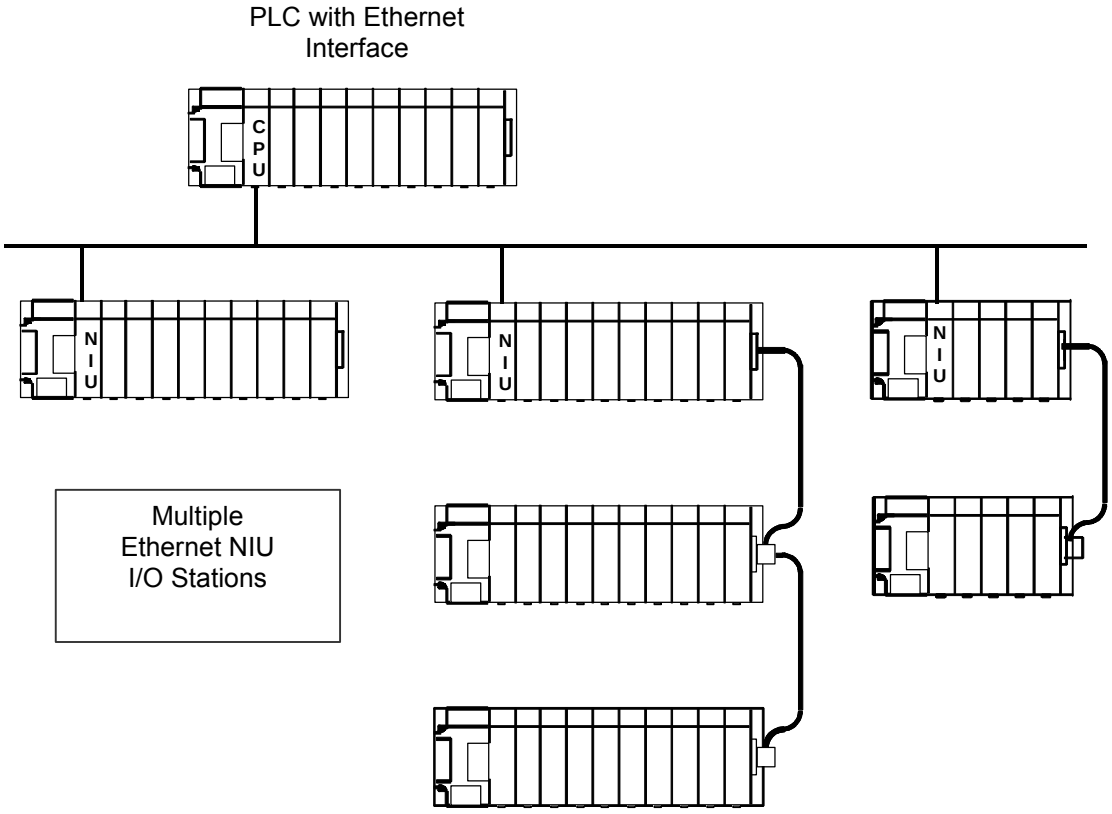
Modules and Baseplates in the I/O Station

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 I/O modules with up to 2048 discrete inputs, 2048 discrete outputs, 1268 analog inputs and 512 analog outputs in the I/O station. 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 seven. 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 seven 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.

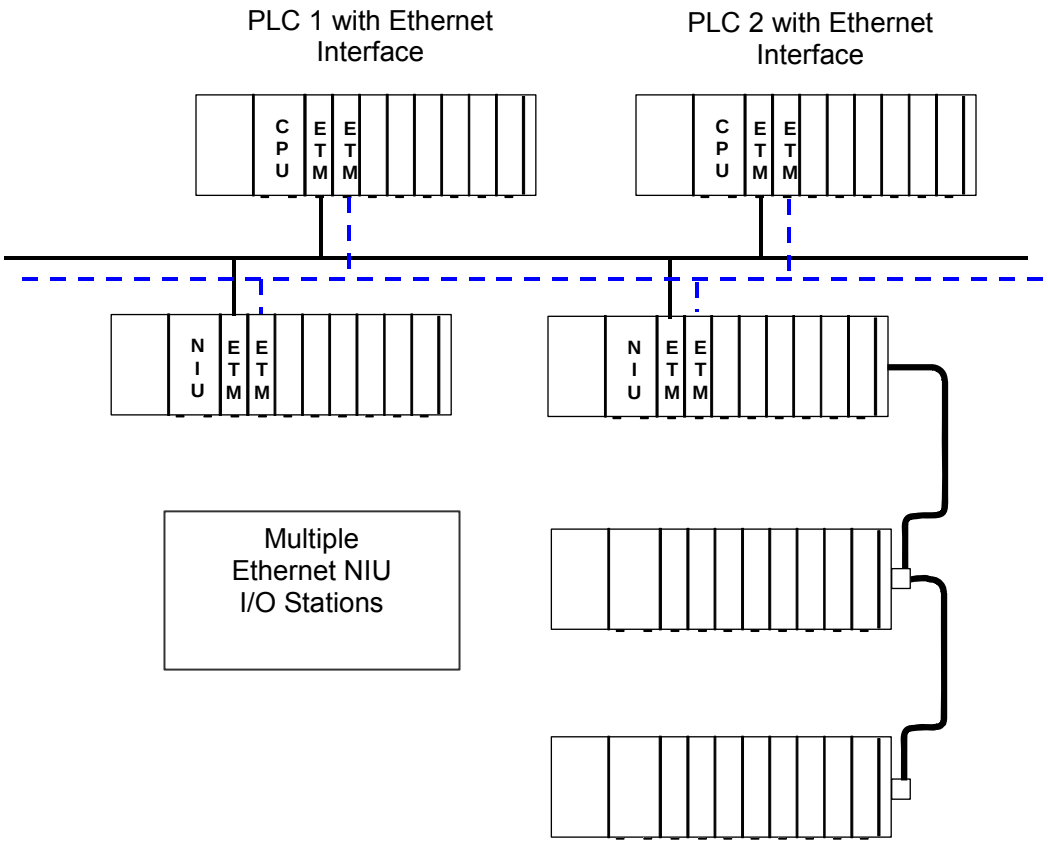


The Ethernet Interface in the master PLC sees all of the modules on the network without regard to their location in a specific I/O Station. That means each module must be assigned *unique* I/O references during configuration. The application program in the PLC sends output data on the Ethernet network, 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. It is also possible to have dual Ethernet LANs in each controller and Ethernet NIU. 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.

Number of Ethernet Interfaces

It is highly recommended that the Ethernet LAN between the controller and Ethernet NIUs be used only for communication between the controller and the Ethernet NIUs. A single programmer can be added to this LAN without causing any noticeable performance impact. For an Ethernet I/O system using a single Ethernet LAN, the controller(s) should have an Ethernet Interface to connect to the Ethernet NIUs.

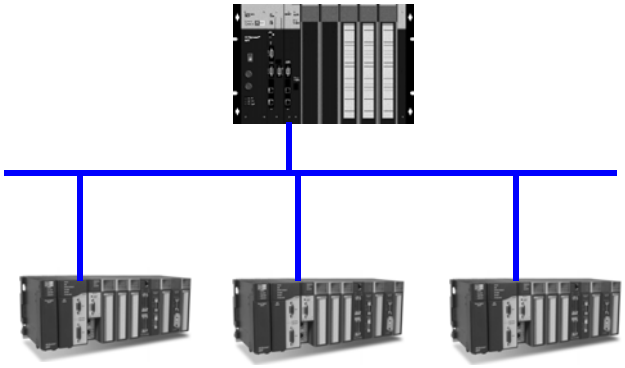
If other Ethernet devices such as HMIs, operator interfaces, operator panels, or level 2 controllers need to communicate with the controller(s), separate Ethernet interface(s) should be used.

For an Ethernet I/O system using dual Ethernet LANs, the controllers should have two Ethernet Interfaces to connect to the Ethernet NIUs - one for each LAN. If other Ethernet devices such as HMIs, operator interfaces, operator panels, or level 2 controllers need to communicate with the controller(s), separate Ethernet interfaces should be used.

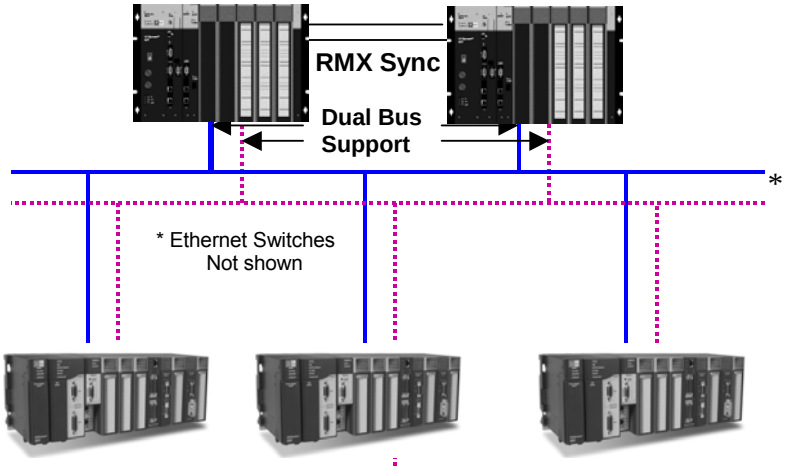
Templates for RX3i Ethernet NIUs

Preconfigured template sets are available to simplify the process of setting up an RX7i or RX3i controller system with several RX3i Ethernet NIUs, in the system configurations shown below. Template sets are not available for other controller types.

Simplex Controller – Single LAN: RX7i or RX3i Controller



Duplex Controller with Dual LAN: RX7i or RX3i Controllers



Available Template Sets

Preconfigured template sets are available for use with Proficy Machine Edition (PME) and Proficy Process Systems (PPS) programmers. The templates are already set up with coordinated references and coordinated parameters for 10, 20, or 24 Ethernet NIUs. For systems with other numbers of Ethernet NIUs, select the template with the next larger number of Ethernet NIUs and delete the extra Ethernet NIUs.

Each template set contains a target for the controller(s) and multiple targets for the Ethernet NIUs.

Use of the templates to develop a new application is strongly recommended. The templates are available online at the GE Fanuc Support website: *http://support.gefanuc.com*

See chapter 3 for instructions on how to download and set up the templates.

Programmer and Template Versions

The table below lists the Proficy Machine Edition and Proficy Process Systems versions required for module configuration, and for different releases of the application templates.

	Proficy Machine Edition Minimum Version	***Proficy Process Systems Minimum Version***	***New Features***
Ethernet NIU Module Configuration	5.5 SIM1	--	--
Release 1.2x templates	5.60	--	Remote COMMREQ Calls
Release 1.3x templates	5.60; can add only one RX3i ENIU target 5.70	1.00	Generic Remote COMMREQs Support for dual LANs Enhanced Fault Reporting
Release 1.4x templates	5.80 SIM 10 or 5.90 SIM 2	1.50 SIM 10	Ability to update output points from only the active controller

Version 1.3x of the Ethernet NIU programmer target added the following capabilities:

- Dual Ethernet LAN connections to redundant controllers using a second Ethernet interface in the Ethernet NIU(s) and the controllers.
- Automatic Ethernet NIU reporting of non-fatal faults to the controller PLC Fault Table. This feature is only available for PACSystems RX7i and RX3i controllers.
- Ability to send any COMMREQ supported by modules in the RX3i Ethernet NIU station. This feature is only available for PACSystems RX7i and RX3i controllers.

Version 1.4 provides greater flexibility in defaulting outputs, and changes the order of preference used by the Ethernet NIU when accepting data from redundant controllers. See chapter 3 for more information about the version 1,4 templates

Overview of Operation

The Ethernet NIU automatically provides the controller with status information in EGD exchanges sent to the controller (see chapter 9). EGD exchanges received by the Ethernet NIU can provide appropriate commands to the Ethernet NIU.

The Ethernet NIU works in systems with a single controller, redundant controllers, or redundant RX7i CRE or RX3i CRU controllers with redundant (dual) LANs. When used with redundant controllers, the Ethernet NIU automatically switches to the standby controller if the active controller becomes unavailable. When used with redundant RX7i CRE or RX3i CRU controllers with dual LANs, the Ethernet NIU will switch to the communication path that requires the least change in control of the application.

The Ethernet NIU can send faults to be entered in the PLC Fault Table of the controller(s).

The Ethernet NIU can also receive commands to execute COMMREQs to intelligent modules in the Ethernet NIU's I/O Station. Only PACSystems RX7i and RX3i controllers can provide the capability to enter Ethernet NIU faults in the controller fault table and send Remote COMMREQ Calls (RCC) to the Ethernet NIU to take advantage of this COMMREQ capability.

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

Ethernet Global Data Features

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.

Ethernet Global Data (EGD) Exchanges

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.

Planning a New System

Follow the steps below to begin developing a new RX3i Ethernet NIU application. (For information about upgrading an existing Ethernet NIU application, see chapter 16.)

1. **Determine System Settings and Operation.** Determine the number of Ethernet NIUs in the system and define the network addressing:

    a. IP Address and Subnet Mask for both Ethernet interfaces in the controller that will communicate with the Ethernet NIUs. Note: Redundant IP is not used for communicating with Ethernet NIUs.

    b. IP Address for any additional Ethernet interfaces in the controller(s), including IP Address, Subnet Mask, Gateway Address, Redundant IP Address (if used) and any other Ethernet options that may be used.

    c. IP Address and Subnet Mask for each Ethernet NIU.

    Ethernet NIUs should be on a separate LAN without HMIs or connection to a plant-wide network. A Proficy programmer can be connected to the I/O LAN without impacting the performance of the Ethernet I/O, if the EGD Exchange production and timeout guidelines in chapter 8 are followed.

2. **Decide how faults in the Ethernet NIUs will be handled.** By default, applications created using Ethernet NIU version 1.3x and later targets send non-fatal faults to the controller(s) and the C block *Ethernet NIU_Faults* in the controller puts the faults in the controller's PLC Fault Table. Optionally, this Enhanced Fault Reporting can be disabled, and the faults can be viewed and cleared using the Ethernet NIU's status and control data. See chapter 10 for information about diagnostics.

3. **Decide whether Remote COMMREQ Call commands will be sent to the Ethernet NIUs.** The controller can send COMMREQs to intelligent modules in the I/O Station via the Ethernet NIU. This feature is enabled in Ethernet NIU targets by default. See chapters 12 and 13 for information about Remote COMMREQ Calls. Applications created using version 1.3x and later Ethernet NIU targets provide additional COMMREQ functionality. See chapter 14.

4. **Select a template set as a starting point to create the application.** Template sets that provide the framework for setting up an application are available for downloading from the GE Fanuc support website. Use of the templates is strongly recommended. See chapter 3 for instructions.

Chapter 2 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 Ethernet 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 connect or 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 Ethernet NIU 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 Ethernet NIU 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 Ethernet NIU 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 Ethernet NIU 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 Ethernet NIU has its connector inserted in slot 3, the module occupies slots 2 and 3 and the Ethernet NIU is referenced as being located in slot 2.
   - The Ethernet NIU may be located in slot 0 with its connector in slot 1.
   - The Ethernet NIU 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 Ethernet NIU system to a PACSystems RX3i ENIU, maintaining the slot 1 location of the Ethernet NIU 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 Ethernet NIU 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 Ethernet NIU in slot 1.

Locating the Ethernet NIU in a Slot Other than 2 & 3 of the I/O Station

Before deciding to place the Ethernet NIU in a slot other than slot 2 and 3, it is important to consider the possible application migration issues that could arise, as explained below.

Fault Handling

The automatic sending of faults to a controller requires that the Ethernet NIU be located in slots 2 and 3.

Ethernet Transmitter Module (ETM)

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 is recommended that it be installed in slot 5, but it can go in any available slot in the Ethernet NIU backplane.

SNTP Time Synchronization

The SNTP time synchronization feature, if used, expects the Ethernet Transmitter Module to be in slot 4 and/or 5. If the Ethernet Transmitter Modules are in different slots, the initial values of the symbolic variables *lana_slot* and *lanb_slot* must be changed to match the actual slot numbers of the ETMs. Advanced User Parameter (AUP) files for SNTP time synchronization must also define the slots in which the Ethernet Transmitter Modules are located. AUP files for slot 3, 4, and 5 are provided with the ENIU and in the version 1.40 Ethernet NIU template sets. If the ETMs are moved to different slot(s) and time synchronization is used, new AUP files must be created.

Application Program

For Service Request #15 (Read Last-Logged Fault Table Entry) and Service Request #20 (Read Fault Tables), the location of Ethernet NIU faults is the slot the Ethernet NIU is located in (see above).

Series 90 PLCs

Remote Series 90 PLCs that use SRTP Channels COMMREQs expect the Ethernet NIU to be in slot 1 or slot 2. Attempts to establish SRTP channels with Ethernet NIUs 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 Ethernet NIU 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 an Ethernet NIU in any slot.

Serial Ports

The Ethernet 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. Port 1 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. It has a 15-pin, female D-sub connector. Port 2 supports the RS-485 to RS-232 adapter (IC690ACC901). Port 2 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 an Ethernet 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 the 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

Programmer Connection

The programmer can communicate with the Ethernet NIU through 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 Ethernet NIU 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 Ethernet NIU 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 Ethernet NIU RS-232 or RS-485 serial port. When connecting directly to the Ethernet NIU, there is no need to specify the backplane and 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.

Ethernet Connections to the Ethernet Transmitter Module

In an RX3i system, an Ethernet Transmitter Module IC695ETM001 provides the I/O Station's connection to the Ethernet network. In an RX7i system, the Ethernet network connection can be provided by an embedded Ethernet interface board or by an IC698ETM001.

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.

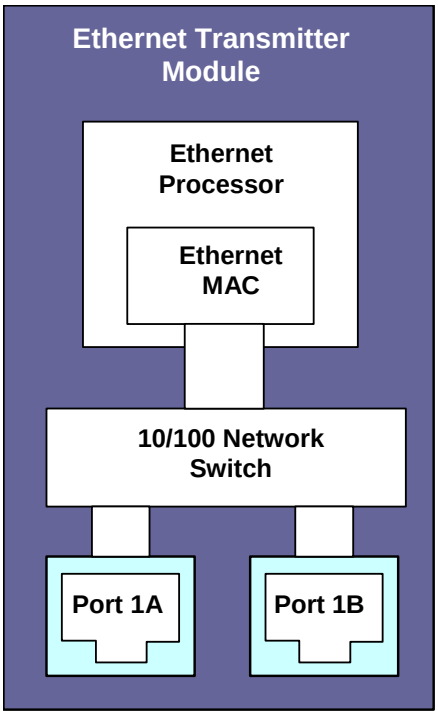
Ethernet Cable

Category 5 cable is recommended for all installations, and required for 100Base-TX operation. 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.

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

Embedded Switch

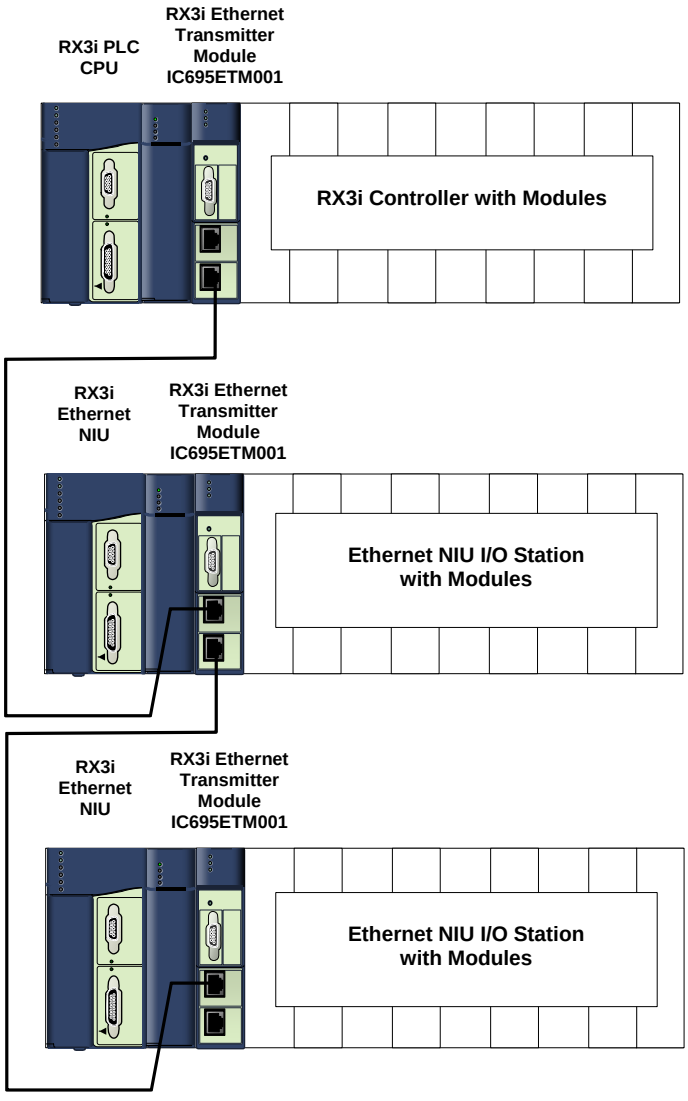
The two Ethernet port connectors on the Ethernet Transmitter Module are controlled by an embedded network switch. Although there are two port connectors, the module has only one interface to the network (one Ethernet address and one IP address).



I/O Station Connections with a Single Controller

The two-port embedded switch on the RX3i or RX7i Ethernet Transmitter Module, or RX7i embedded Ethernet interface 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 in the example below.

The second port connector on the Ethernet Transmitter Module in the I/O Station can then be used to daisy-chain to a third I/O Station and so on. It is important to remember that 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.

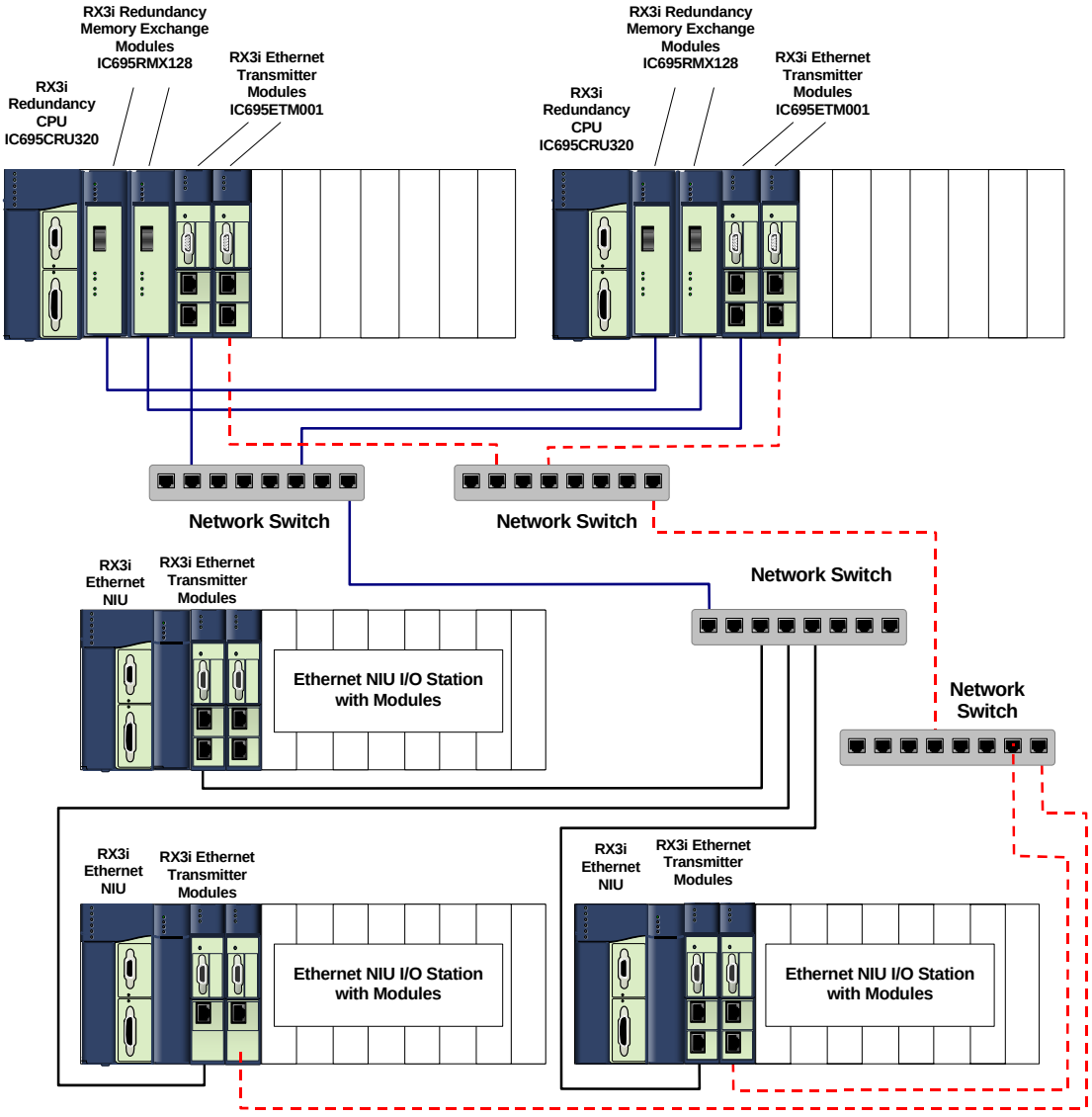


I/O Station Connections for Redundant Controllers with Dual LANs

Ethernet switches can be used to connect 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).

This archictecture requires RX3i IC695CRUxxx or RX7i IC698CRExxx redundancy CPUs. The following example uses RX3i CRU320 CPUs. For additional redundant architectures using an RX3i CRU or RX7i CRE redundancy CPUs with redundancy memory exchange (RMX) modules, refer to the *PACSystems Hot Standby Redundancy User's Manual,* GFK-2308.

Note that this architecture cannot be used with a Max-ON redundancy controlller.



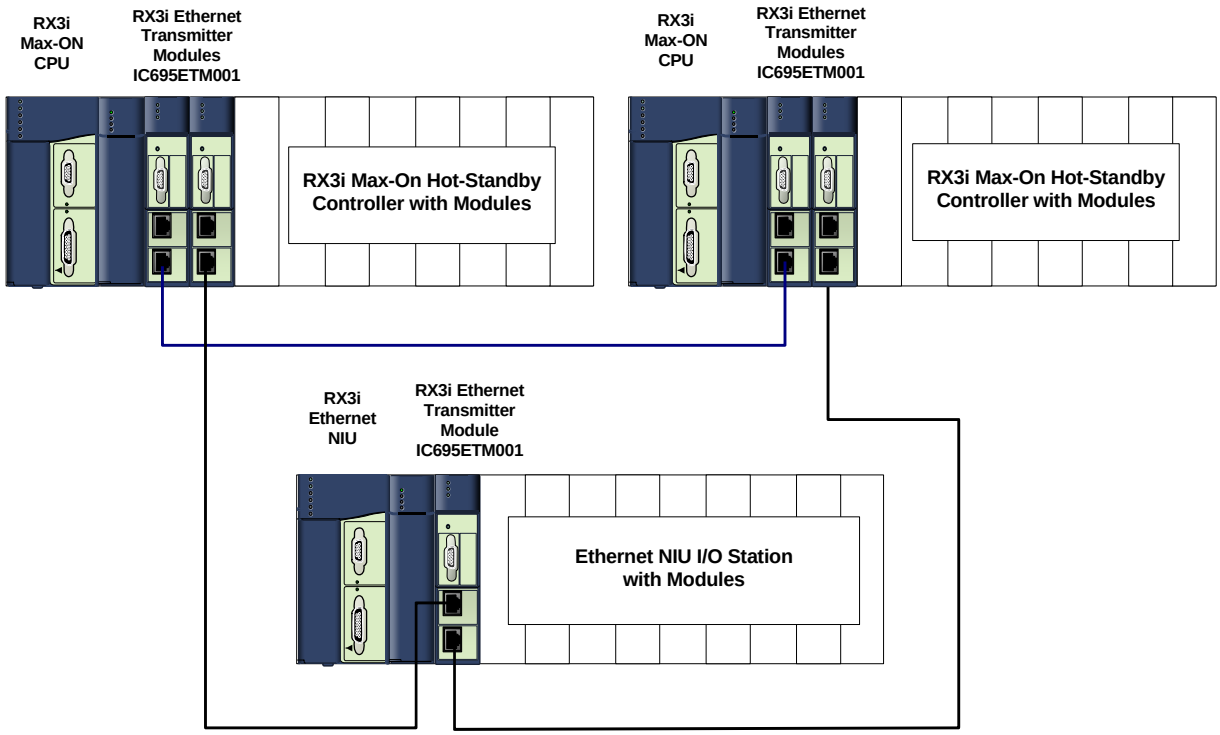
I/O Station Connections with Redundant Max-On Controllers

If only one Ethernet NIU I/O Station is used in a system that includes two 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

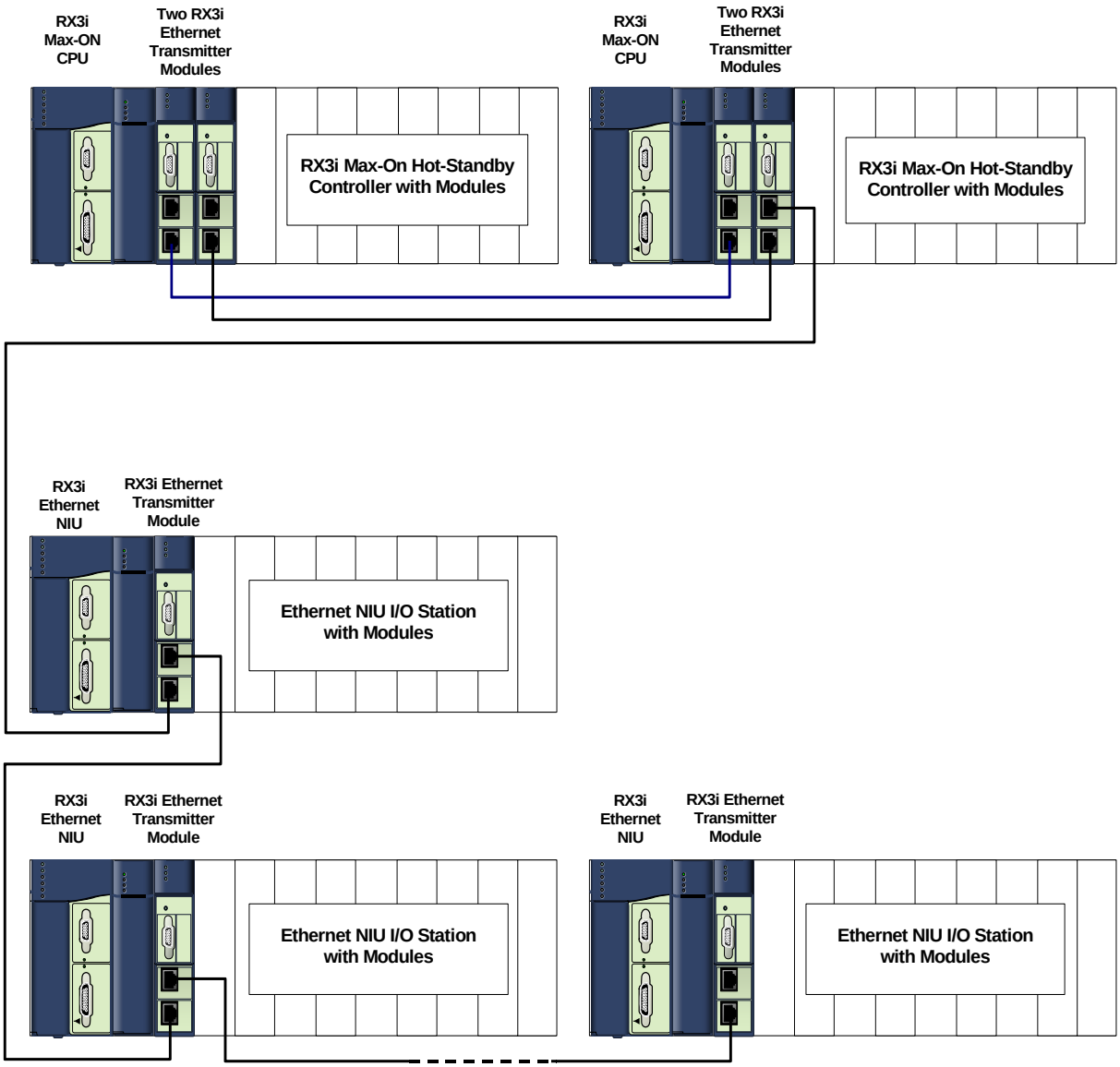
For RX3i Max-ON Hot Standby redundant controllers, there must be a pair of Ethernet Transmitter Modules in each Max-ON controller. The first pair of Ethernet Transmitter Modules in the Max-ON controllers is dedicated to synchronizing application data. To maintain the higher synchronization performance, 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

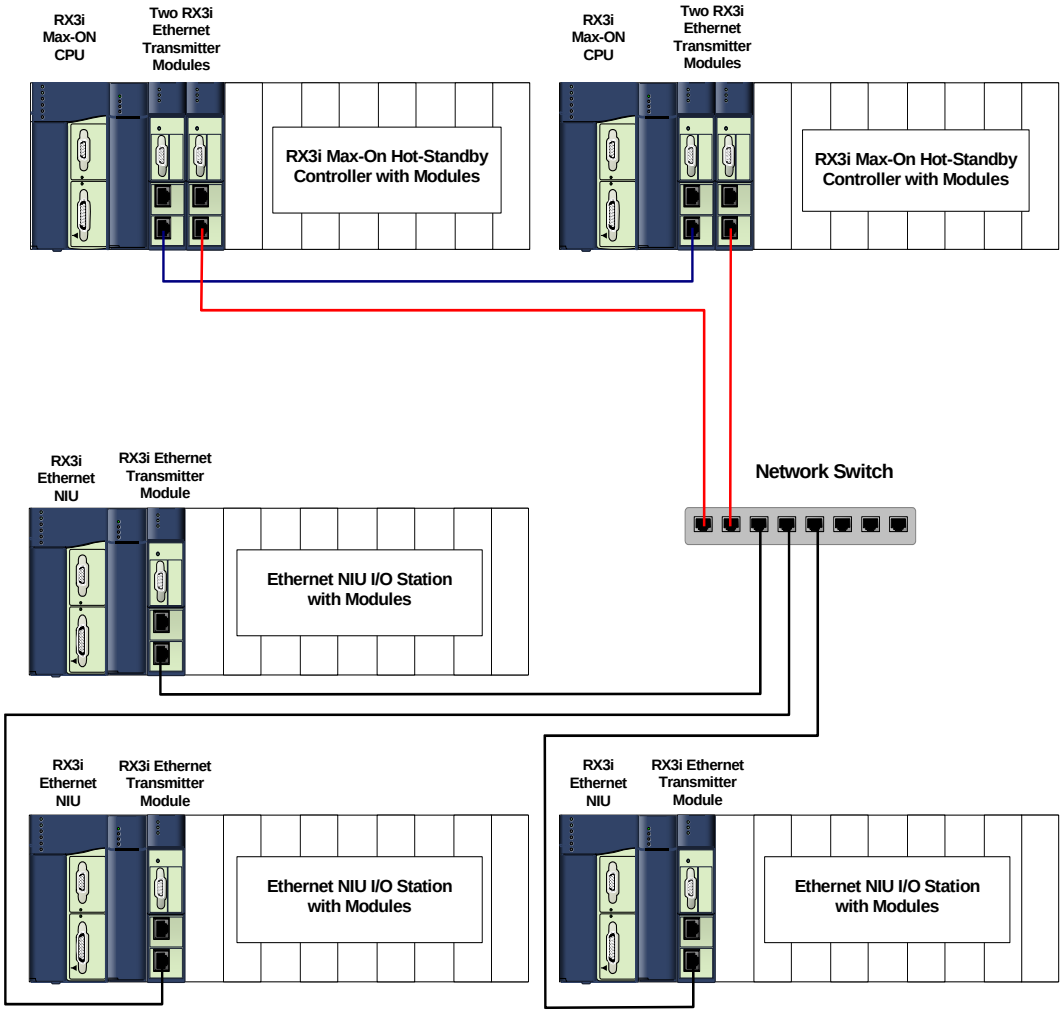
If more than one Ethernet NIU I/O Station will be connected to the Max-ON redundant controllers, the second connector on one of the controllers can be used to extend the daisy-chain to a second Ethernet NIU I/O Station and so on. It is important to remember that if any I/O Station in the chain is powered 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

Most applications require the ability to remove power from an I/O Station or CPU without disturbing the communications on the rest of the I/O network. For this reason, good-quality, Ethernet network switches must be used with multiple I/O Stations. For performance reasons, Ethernet hubs are NOT recommended for use with RX3i Ethernet NIUs. If ten or fewer I/O Stations are used *and* there is no requirement for the I/O network to function under the condition of a powered-down I/O Station, the two-port switch built into 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.

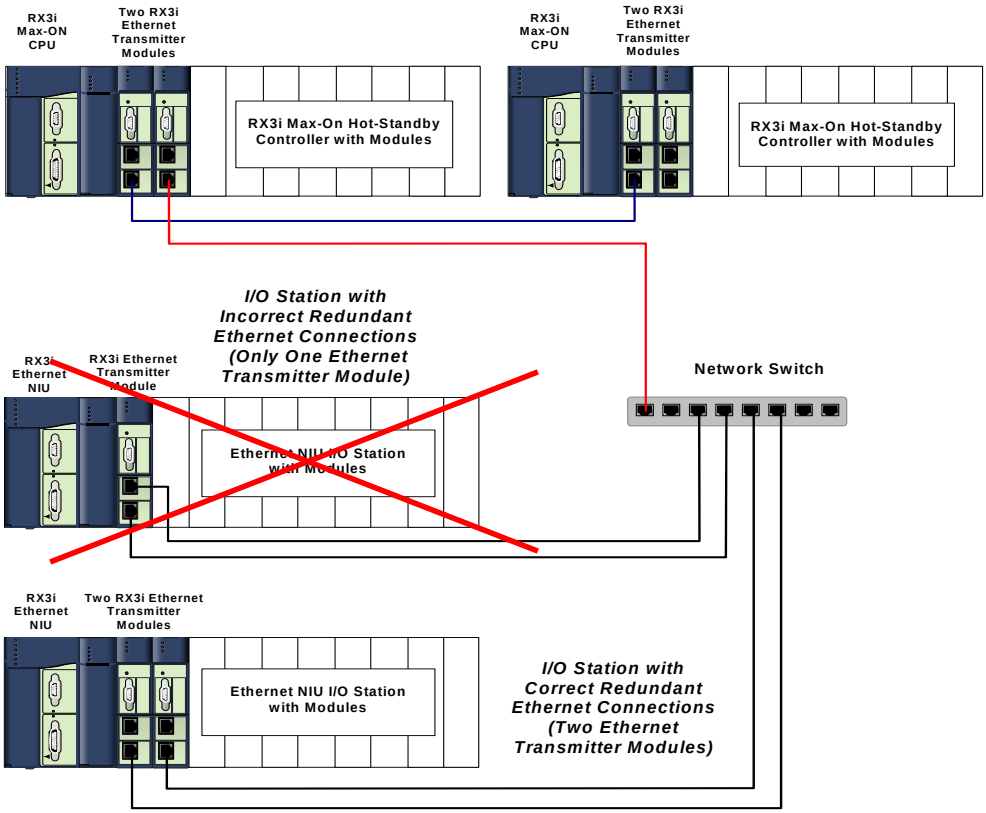
Please contact your local GE Fanuc application engineer for more information regarding recommended network switches.



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. Using two Ethernet Transmitter Modules 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 communication 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	The Battery Low indication is not supported and should be ignored.
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.

Chapter 3 Template Sets for RX3i and RX7i Applications

This chapter describes and explains how to use the preconfigured application templates for RX3i and RX7i controller systems.

- Available Template Sets and C Blocks
  - Template Sets for Single LAN Systems
  - Template Sets for Dual LAN Systems
  - C Blocks for the Application Program in the Controller
- Downloading a Template Set
- Configuring the Controller(s)
  - Configuring Inputs to the Controller
  - Deleting EGD Exchanges for Extra Ethernet NIUs
  - Completing the Controller Configuration
  - Checking Input Operation in a Dual Controller Application
- Configuring the Ethernet NIUs
  - Input References for Ethernet NIUs
- Adjusting I/O Addresses in Dual LAN Systems
  - Configuring the Controller Exchange for NO Discrete Inputs
  - Configuring the Controller Exchange for NO Analog Inputs
- Configuration Examples for a Single LAN System
- Configuration Examples for a Dual LAN System

Available Template Sets and C Blocks

A template set is a zip file containing pre-configured Ethernet NIU and controller projects for Proficy Machine Edition or Proficy Process Control. The template set simplifies the configuration of the controller(s) and ENIUs because the Ethernet Global Data exchanges and a default number of inputs and outputs for the system are already set up. If the default values are used, the only steps needed to implement I/O communication are entering Ethernet IP addresses, configuring I/O modules and storing to the controller(s) and ENIU(s).

Choices in Template Sets

The template sets provide choices for:

PME or PPS

RX7i or RX3i controllers

Single or redundant controller

Single or Dual Ethernet LAN to the ENIUs

Number of ENIUs in the system

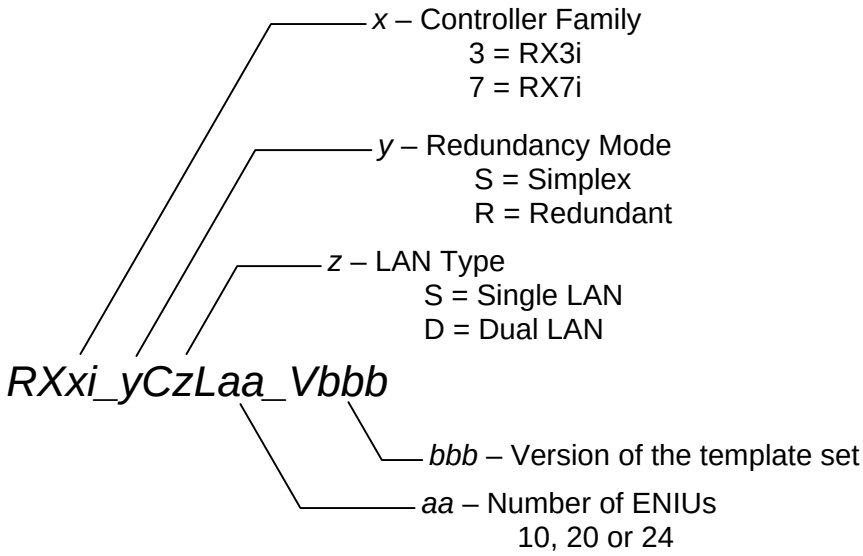
Choosing a Template Set

1. Determine the programmer: either PME or PPS.
2. Determine the controller family: either RX7i or RX3i.
3. Determine whether the controller is simplex or redundant.
4. Determine if a single or dual Ethernet LAN connects to the ENIUs.
5. Determine the number of ENIUs in the system. Pick a template set with the number of ENIUs needed or greater. Do not pick a template set with fewer than the number of ENIUs that is needed.

The Programmer type (PME or PPS) is chosen by going to the correct section of the GE Fanuc support web site either Controller & I/O (PME) or Proficy Process Control (PPS)

Template Set Naming Convention

The template set name identifies the choices that the particular template set provides. The template set name has the following format:



Template Set Contents

Each Template Set is a Zip file that contains:

- A controller folder named CTLRXxi_yCzL_aa_Vbbb
- One or two folders with ENIUs named ENIUa1_a2_yCzL_aa_Vbbb

Sample Template Set

RX3i_SCSL_20_v140.ZIP - RX3i Simplex Controller Single LAN with 20 ENIUs

_ENIU01_12_SCSL_20_v140 – folder with ENIUs 1-12

_ENIU13_20_SCSL_20_v140 – folder with ENIUs 13-20

_CTL_RX3i_SCSL_20_v140 - Controller folder

C Blocks for the Application Program in the Controller

The following C blocks are included in the template sets. Only “C” version 1.33 can be downloaded separately.

Input_Processing	Input handling and Point Fault/Data Quality functionality.
[Input_Arbitration]	Replaced by *Input_Processing*, above. Version 1.33 of the application templates used this block for RX7i CRE Controller, dual LAN applications for NON-PPS applications.
RCCD	For PACSystems RX7i and RX3i controllers with a version 1.3x or higher Ethernet NIU target. This block adds Remote COMMREQ Call and Generic COMMREQ functionality to the application program.
RCCM(not in templates)	For PACSystems RX7i and RX3i controllers with a version 1.2x Ethernet NIU target. This block adds Remote COMMREQ Call functionality to the application program.
ENIU_Faults	For PACSystems RX7i and RX3i controllers with a version 1.3x or higher Ethernet NIU target. This block adds ENIU Fault logging (into Controller PLC Fault Table) functionality to the application program.

Downloading a Template Set

The template sets are available online on the GE Fanuc Support website:

http://support.gefanuc.com

Note: Use of this site requires an account and password.

At the website, follow the steps below to download a template set:

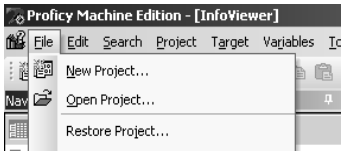
- Under Quick Picks, go to Downloads and select Developer Downloads.
- Search for “templates” to filter the list of downloads.
- Select the appropriate PME or PPS template for your application. Choose the set with the smallest number of Ethernet NIUs (including any expected additions) that fits the application.
- Download the chosen template set to your computer hard drive. After the project is completed, delete any extra Ethernet NIUs. Using a template set with more Ethernet NIUs than are needed will reduce performance.

The Ethernet Global Data exchanges and symbolic variables in a template set are coordinated, which reduces the number of variables in one Machine Edition project folder.

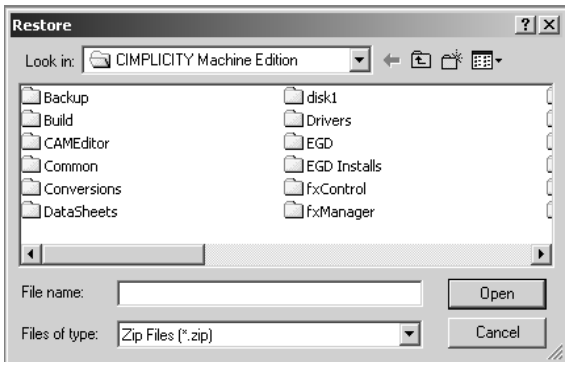
Bringing a Template Set into the Programmer

Unzip the downloaded template set, then use the programmer Restore project feature to bring the folders in the template set into the programmer.

- With no projects currently open, select File>Restore Project. Screens for Proficy Process Developer are similar to the example Machine Edition screens shown in this section.



- With the “Files of Type” field set to Zip Files, navigate to the downloaded template set on the computer’s hard drive.



- Select a template from the downloaded template set. Click Open to add the project to the Navigator window.

Configuring the Controller(s)

In the Navigator, select the appropriate template. Right-click and duplicate the project. In this discussion, the duplicated project has been renamed to *My_Controller_Project.*

This project sets up the CPU hardware configuration, and maps the discrete and analog inputs the controller will receive from each Ethernet NIU.

In an RX7i (NOT CRE) template, the Ethernet Transmitter Module that will be used to communicate with the Ethernet NIUs is located in slot 3. In addition, the RX7i template has a built-in Ethernet port that optionally can be set up to communicate with the Ethernet NIUs. If an RX7i controller's CPU Ethernet port will be connected to the Ethernet NIUs, all Ethernet Global Data exchanges for that LAN (either LAN A or LAN B) must have their adapter name changed to 0.1. Select the adapter name in the drop-down list on each Ethernet Global Data exchange properties page.

In an RX3i (NOT CRU) controller template, the Ethernet Transmitter Module that will be used to communicate with the Ethernet NIUs is located in slot 5. The RX3i template provides for an additional Ethernet Transmitter Module located in slot 4 that can be used for other communications.

- For each controller, set the IP addresses, Subnet Mask, Gateway Address (if required), Status Address, and Redundant IP Address (if used) in the controller's hardware configuration. The LAN(s) to the Ethernet NIUs do not use Redundant IP. It is recommended that the Ethernet NIUs be on a separate Ethernet LANs without HMIs or connection to a plant-wide network. A programmer can be connected to the I/O LAN without impacting the performance of the Ethernet I/O, if the EGD Exchange production and timeout guidelines in chapter 8 are followed.
- If the Ethernet Transmitter Module is moved to a different slot, either drag and drop, or cut and paste the Ethernet Transmitter to the new slot in the configuration. This will preserve all the Ethernet Global Data setup information, and automatically adjust the adapter name for the new location of the Ethernet Transmitter Module. Do not delete and add an Ethernet Transmitter; that would delete the module's EGD Configuration information.

For RX7i CRE controllers only, the templates use two Ethernet Transmitter Modules in slots 5 and 6 of rack 0 for communications with the Ethernet Transmitter Modules in the I/O Stations. In the controller, the Ethernet Transmitter Module in slot 5 controls LAN A and the Ethernet Transmitter Module in slot 6 controls LAN B. Ordinarily, the CPU's Ethernet interface is used for communication with HMIs and plant-wide networks. If the CPU Ethernet port will instead be connected to the Ethernet Transmitter Modules in the I/O Stations, all EGD exchanges for that LAN (either LAN A or LAN B) must have their adapter name changed to 0.1. Select the adapter name in the drop-down list on each EGD exchange properties page.

For RX3i CRU controllers only, the templates use two Ethernet Transmitter Modules in slots 7 and 8 of rack 0 for communications with the Ethernet Transmitter Modules in the I/O Stations. In the CRU controller, the Ethernet Transmitter Module in slot 7 controls LAN A and the Ethernet Transmitter Module in slot 8 controls LAN B. The Ethernet Transmitter Module in slot 6 is used for communication with HMIs and plant-wide networks.

- In ALL templates: in the program block ENIUs_Interface, the call to the block ENIU_Faults has one input, NUM_ENIUs. The template defaults this to the number of Ethernet NIUs in the template. If the application will have a different number of Ethernet NIUs, change the constant on the block's input to match the number of Ethernet NIUs.
- In ALL version 1.40 templates: in the program block ENIUs_Interface, the Call to the block Input_Processing has eight inputs, which are constants. Set the END_ENIU to the number of Ethernet NIUs in the application.

START_ENIU	Must be 1
END_ENIU	Number of the last Ethernet NIU
START_IS	Offset of the memory location where the first discrete inputs from the Ethernet NIUs will be placed in the controllers. This constant must be on a byte boundary (1, 9, 65, …).
START_AIS	Offset of the memory location where the first analog input from the Ethernet NIUs will be placed in the controllers.
MORE_IS	Used to add more discrete inputs to the application without having to re-address all the previously used discrete inputs. Initially this should be set to 0. If used, it is the offset of the memory location where addition discrete inputs from the Ethernet NIUs will be placed in the controllers.
MORE_AIS	Used to add more analog inputs to the application without having to re-address all the previously-used analog inputs. Initially this should be set to 0. If used, it is the offset of the memory location where addition analog inputs from the Ethernet NIUs will be placed in the controllers.
NUM_LANs	Specifies whether the system uses a Single LAN or Dual LANs to the Ethernet NIUs. 1 = Single LAN; 2 = DO NOT USE; 3 = Dual LAN
FAIL_MODE	Selects how inputs in the Controller will be handled if communication to an Ethernet NIU is lost. 0 = Inputs are set to Zero; 1 = Inputs Hold Last State

In Release 1.2 and earlier Proficy Machine Edition templates: in the program block ENIUs_Interface, there is a Call to the block Input_Arbitration. The Call has six inputs, which are constants. Set the END_ENIU to the number of Ethernet NIUs.

START_ENIU	Must be 1
END_ENIU	Number of the last Ethernet NIU
START_IS	Offset of the memory location where the first discrete inputs from the Ethernet NIUs will be placed in the controllers. This constant must be on a byte boundary (1, 9, 65, …).
START_AIS	Offset of the memory location where the first analog input from the Ethernet NIUs will be placed in the controllers.
MORE_IS	Used to add more discrete inputs to the application without having to re-address all the previously used discrete inputs. Initially this should be set to 0. If used, it is the offset of the memory location where addition discrete inputs from the Ethernet NIUs will be placed in the controllers.
MORE_AIS	Used to add more analog inputs to the application without having to re-address all the previously-used analog inputs. Initially this should be set to 0. If used, it is the offset of the memory location where addition analog inputs from the Ethernet NIUs will be placed in the controllers.

Configuring Inputs for the Controller

The templates are set up to receive 200 discrete inputs and 128 (80 in version 1.33) analog inputs from each Ethernet NIU. If the application (including spares) needs this amount or fewer inputs, the default setup should be used. The examples at the end of this chapter describe the default input setup, and show how to adjust the configuration if additional inputs are needed.

In a dual LAN system, if additional discrete or analog inputs are needed for one or more Ethernet NIUs, the size of the inputs variable on both LAN A and LAN B for the affected Ethernet NIUs must be changed. As an example, ENIU01 can use the default values of 200 discrete and 128 analog, so no changes are needed to the variables *ENIU01_LAN_A_InputsDiscrete, ENIU01_LAN_A_InputsAnalog, ENIU01_LAN_B_InputsDiscrete, ENIU01_Xfer_Pri_InputsDiscrete, ENIU01_Xfer_Pri_InputsAnalog, ENIU01_Xfer_Sec_InputsDiscrete, ENIU01_Xfer_Sec_InputsAnalog,* or *ENIU01_LAN_B_InputsAnalog*, but ENIU02 needs 250 discrete inputs and 100 analog inputs. The ArrayDimension1 parameters of the ENIU02 variables need to be changed. ENIU03 can use the defaults.

Note: The Xfer Variables are in version 1.40 and later. They are not in version 1.33.

The table below shows the needed *ArrayDimension1* values for ENIU01, ENIU02, and ENIU03.

Variables	***Number of inputs***	***Value of Array Dimension 1 parameter***	***Starting address calculated by C block***
ENIU01_LAN_A_InputsDiscrete *ENIU01_LAN_B_InputsDiscrete* *ENIU01_Xfer_Pri_InputsDiscrete* *ENIU01_Xfer_Sec_InputsDiscrete*	200	200	1 na na
ENIU01_LAN_A_InputsAnalog *ENIU01_LAN_B_InputsAnalog* *ENIU01_Xfer_Pri_InputsAnalog* *ENIU01_Xfer_Sec_InputsAnalog*	128	128	1
ENIU02_LAN_A_InputsDiscrete *ENIU02_LAN_B_InputsDiscrete* *ENIU02_Xfer_Pri_InputsDiscrete* *ENIU02_Xfer_Sec_InputsDiscrete*	250	250	201
ENIU02_LAN_A_InputsAnalog *ENIU02_LAN_B_InputsAnalog* *ENIU02_Xfer_Pri_InputsAnalog* *ENIU02_Xfer_Sec_InputsAnalog*	100	100	129
ENIU03_LAN_A_InputsDiscrete *ENIU03_LAN_B_InputsDiscrete* *ENIU03_Xfer_Pri_InputsDiscrete* *ENIU03_Xfer_Sec_InputsDiscrete*	200	200	451
ENIU03_LAN_A_InputsAnalog *ENIU03_LAN_B_InputsAnalog* *ENIU03_Xfer_Pri_InputsAnalog* *ENIU03_Xfer_Sec_InputsAnalog*	128	128	229

To change the variables *ArrayDimension1*, in the Navigator window, click on the variable tab to bring up the variable display. Right-click on the variable to bring up the property window. Enter the new value for Array Dimension1. Note: if you change a LAN A variable in a dual LAN application, make sure you also change the corresponding LAN B variable. **Do not delete any *ENIUxx_LAN_x_Inputsxxxx* variables.**

Variable [CRE_HSB_System]	***Property Inspector: Before***	***Property Inspector: After***
Name	*ENIU02_LAN_A_InputsDiscrete*	*ENIU02_LAN_A_InputsDiscrete*
Description	*ENIU02_LAN_A_InputsDiscrete*	*ENIU02_LAN_A_InputsDiscrete*
Publish	External	External
Array Dimension 1	200	250
Array Dimension 2	0	0
Data Source	GE FANUC PLC	GE FANUC PLC
Ref Address		
Input Transfer List	False	False
Output Transfer List	False	False
Data Type	BOOL	BOOL
Current Value	(Array)	(Array)
Initial Value	Off,Off,Off,Off,Off,Off,Off,Off,Off	Off,Off,Off,Off,Off,Off,Off,Off,Off
Default Data Display	On/Off	On/Off
Retentive	True	True

Deleting EGD Exchanges for Extra Ethernet NIUs

If not all the Ethernet NIUs targets in the template will be used, delete the extra exchanges in the controller(s) for each Ethernet NIU that is not present.

There are four consumed EGD exchanges from each ENIU:

Inputs_from_ENIUxx

Inputs_from_ENIUxx_LANB (dual LAN template only)

SVC_Xchg_from_ENIUxx

SVC_Xchg_from_ENIUxx_LANB (dual LAN template only)

There are two produced EGD exchanges to each ENIU:

SVC_Xchg_to_ENIUxx

SVC_Xchg_to_ENIUxx_LANB (dual LAN template only)

Completing the Controller Configuration

After completing the controller configuration:

- add the user application to *_Main* after the Call to *ENIUs_Interface* in rung 1. The added application MUST NOT go before the Call to *ENIUs_Interface*.
- Store to the application to the controller(s).

Checking Input Operation in a Dual Controller Application

In a dual controller application, connect to the primary controller. Make sure the controller is in Run mode, and open the block *ENIUs_Interface*. Check the value on the output status of the Call to *Input_Processing* (*Input_Arbitration* in older versions). The value should be 1. See chapter 4 for more information about error codes.

Open the data watch *InputAddressing* and check the addressing for inputs from the Ethernet NIUs. The data watch is a two index array. The first index is the Ethernet NIU number (0 is not used). The second index is the start address of the inputs. 0 is the start of discrete inputs %I, 1 is the start of analog input %AI, 2 is the start of more discrete inputs, and 3 is the start of more analog inputs. Check that the starting discrete input number and analog input number is correct for each Ethernet NIU.

Configuring the Ethernet NIUs

In the Navigator, select the appropriate template. Right-click and duplicate the project. In this discussion, the duplicated project has been renamed to *My_ENIUs_Project*. This project sets up the communications parameters of the Ethernet NIUs in the system. It also maps input data from modules in the I/O Station to CPU reference addresses.

- For each Ethernet NIU, set the IP Address and Subnet Mask in the hardware configuration for the Ethernet Transmitter Module, which is in slot 4. In Ethernet NIU templates, slot 4 is for LAN A and slot 5 is for LAN B.
- If you need to move the Ethernet Transmitter Module to a different slot, either cut and paste or drag and drop the module in the configuration. This way the Ethernet Global Data Exchanges that have the Ethernet interface information in their configuration will automatically be updated.
- The template sets up fault handling in which the Ethernet NIUs will send non-fatal faults to the controller(s) and the C block *ENIU_FAULTS* in the controller will put the faults into the PLC Fault Table. To instead log faults to the local Ethernet NU fault tables where the programmer will be used to view and clear faults, see chapter 10.
- Input and output modules in an I/O Station must be configured using address ranges assigned to the Ethernet NIU in that I/O Station. See the table on the next page for default input addressing. If the application needs more than 2048 discrete outputs or more than 512 analog outputs see chapter 8.
- If any Ethernet NIUs need more than 200 discrete inputs or 128 (80 in version 1.33 non-PPS) analog inputs, refer to the example in this chapter that shows how to change the Ethernet NIU input size configuration and starting reference addresses.
- If there is no expected expansion of the number of Ethernet NIUs, any unused Ethernet NIU targets should be deleted.
- After completing programming and configuration, store the programs and configurations to the Ethernet NIUs and make sure the Ethernet NIUs are in Run mode.
- If the Remote COMMREQ Calls (RCC) feature will be used, see chapter 12 for RCC configuration steps.

Input References for Ethernet NIUs

The templates set up discrete and analog input references in the controller(s) to match the Ethernet NIU I/O references. The Proficy Machine Edition templates use a standard I/O map with 200 %I discrete inputs, and 128 %Q AI discrete outputsanalog inputs. Proficy Process Systems templates use an I/O map with 200 %I discrete inputs and 128 %Q discrete outputs. Note that Machine Edition Version 1.33 uses 80 %Q discreteAI analog inputs as a default. Each Ethernet NIU in the templates uses the input references shown below.

	Discrete Inputs, All Templates	***Analog Inputs, PME Templates (Version 1.40) and PPS Templates***	***Analog Inputs, PME Templates (Version 1.33)***
ENIU_01	%I001 to %I200	%AI001 to %AI128	%AI001 to %AI080
ENIU_02	%I201 to %I400	%AI129 to %AI256	%AI081 to %AI160
ENIU_03	%I401 to %I600	%AI257 to %AI384	%AI161 to %AI240
ENIU_04	%I601 to %I800	%AI385 to %AI512	%AI241 to %AI320
ENIU_05	%I801 to %I1000	%AI513 to %AI640	%AI321 to %AI400
ENIU_06	%I1001 to %I1200	%AI641 to %AI768	%AI401 to %AI480
ENIU_07	%I1201 to %I1400	%AI769 to %AI896	%AI481 to %AI560
ENIU_08	%I1401 to %I1600	%AI897 to %AI1024	%AI561 to %AI640
ENIU_09	%I1601 to %I1800	%AI1025 to %AI1152	%AI641 to %AI720
ENIU_10	%I1801 to %I2000	%AI1153 to %AI1280	%AI721 to %AI800
ENIU_11	%I2001 to %I2200	%AI1281 to %AI1408	%AI801 to %AI880
ENIU_12	%I2201 to %I2400	%AI1409 to %AI1536	%AI881 to %AI960
ENIU_13	%I2401 to %I2600	%AI1537 to %AI1664	%AI961 to %AI1040
ENIU_14	%I2601 to %I2800	%AI1665 to %AI1792	%AI1041 to %AI1120
ENIU_15	%I2801 to %I3000	%AI1793 to %AI1920	%AI1121 to %AI1200
ENIU_16	%I3001 to %I3200	%AI1921 to %AI2048	%AI1201 to %AI1280
ENIU_17	%I3201 to %I3400	%AI2049 to %AI2176	%AI1281 to %AI1360
ENIU_18	%I3401 to %I3600	%AI2177 to %AI2304	%AI1361 to %AI1440
ENIU_19	%I3601 to %I3800	%AI2305 to %AI2432	%AI1441 to %AI1520
ENIU_20	%I3801 to %I4000	%AI2433 to %AI2560	%AI1521 to %AI1600
ENIU_21 *	%I4001 to %I4200 *	%AI2561 to %AI2688 *	%AI1601 to %AI1680
ENIU_22 *	%I4201 to %I4400 *	%AI2689 to %AI2816 *	%AI1681 to %AI1760
ENIU_23 *	%I4401 to %I4600 *	%AI2817 to %AI2944 *	%AI1761 to %AI1840
ENIU_24 *	%I4601 to %I4800 *	%AI2945 to %AI3072 *	%AI1841 to %AI1920

- PPS templates and RX3i templates include up to 20 Ethernet NIUs. NIUs 21-24 shown here are for RX7i Machine Edition systems only.
- For all template types, %Q0001 to %Q2048 and %AQ001 to %AQ512 are sent from the controller and received by all Ethernet NIUs. For outputs, the references do not have to match; any reference addressing can be used for discrete and analog outputs in the I/O Station.

Input and output modules added to Ethernet NIU I/O Stations must be configured using address ranges for the Ethernet NIU in which they reside. If the application needs more than 2048 discrete outputs or more than 512 analog outputs see chapter 8. Examples of customizing the input configuration are shown later in this chapter.

Adjusting I/O in Dual LAN (and PPS Systems)

In dual LAN systems using the RX7i CRE or RX3i CRU controller templates, the input addresses for %I and %AI are packed in the reference tables in the controller by the *Input_Processing* C block. Using additional %I or %AI inputs would ordinarily mean changing all the %I and %AI references in the system. Instead of readdressing all the references, for each Ethernet NIU needing more %I and %AI, additional data ranges can be added to the EGD *"Input…:"* exchanges. One range is used for %I and another for %AI. In the controllers, the EGD exchanges have corresponding ranges added.

The %I input range will use the variable *ENIUxx_LAN_y_Xtra_InputsDiscrete* and the %AI input range will use the variable *ENIUxx_LAN_y_Xtra_InputsAnalog*. There is also a variable *ENIUxx_LAN_y_Xtra_InputsRegister* if additional word data is needed in an added data range. To synchronize redundant controllers, the variables *ENIUxx_zzz_InputsDiscrete*, *ENIUxx_zzz_InputsAnalog*, *ENIUxx_zzz_InputsRegister*, and *ENIUxx_zzz__Xtra…* are needed. In the variable names, zzz represents either Pri for the primary controller or Sec for the secondary controller. Both the Pri and Sec variables are required.

For the *Input_Processing* block to use this added data, the two inputs to the C block (*MORE_IS* and *MORE_AIS*) must be changed from 0 to the offset in the table where the additional Input data is to be placed.

Example Configuration of Inputs from ENIU_01_LANA

An example of the exchange to add 64 %I (%I4001-%I4064) and 32 %AI (%AI3001-%AI3032) to ENIU01 is shown below.

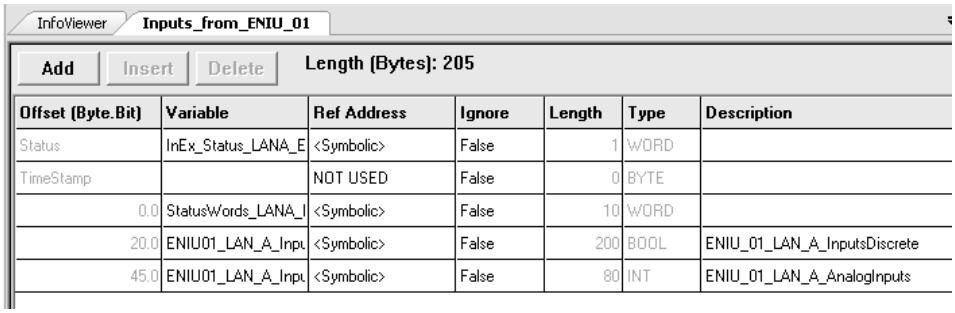
ENIU Setup						Controller EGD Setup					
ENIU 01 Produced Exchange *Inputs_from_ENIU_01*						CRE HSB System Consumed Exchange *Inputs_from_ENIU_01*					
Vari-able	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_01	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from ENIU LAN A	StatusWords_LANA_ENIU_01	<Sym>	False	10	Word	
	%I0001	N/A	200	Bit	Discrete Inputs from ENIU	ENIU01_LAN_A_InputsDiscrete	<Sym>	False	200	Bool	
	%AI001	N/A	128	Word	Analog Inputs from ENIU	ENIU01_LAN_A_InputsAnalog	<Sym>	False	128	Int	
	%I4001	N/A	64	Bit	Extra Discrete Inputs from ENIU	ENIU01_LAN_A_Xtra_Inputs Discrete	<Sym>	False	64	Bool	
	%A3001	N/A	32	Word	Extra Analog Inputs from ENIU	ENIU01_LAN_A_Xtra_Inputs Analog	<Sym>	False	32	Int	

The Call to the Input_Processing block would have the following inputs:

START_ENIU	*1*
END_ENIU	10 (assumes there are 10 Ethernet NIUs)
START_IS	1 regular %I start at %I00001
START_AIS	1 regular %AI start at %AI00001
MORE_IS	4001 Xtra %I start at %I04001
MORE_AIS	3001 Xtra %AI start at %AI03001
Num_LANs	1 or 3: 1 for single LAN or 3 for dual LAN
Fail_Mode	0 or 1 0 for input to Zero, or 1 for inputs to Hold Last State

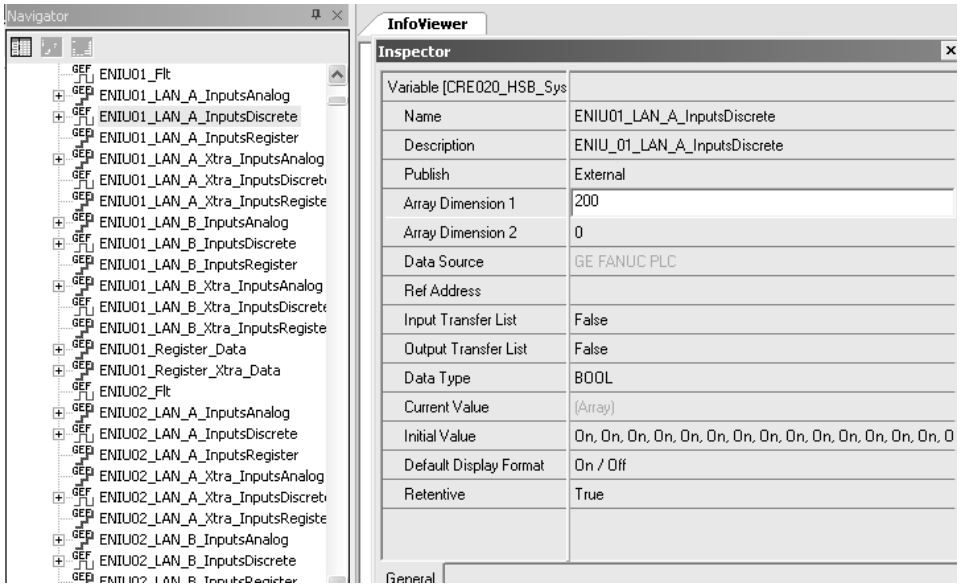
Configuring the Controller Exchange for NO Discrete Inputs

PACSystems RX7i CRE and RX3i CRU controllers in dual/redundant LAN applications use symbolic variables for Input Arbitration. If there are no discrete and/or no analog inputs from one or more Ethernet NIUs, the symbolic variables must be adjusted. For example, this EGD Exchange has 200 discrete inputs and 80 analog inputs in a dual/redundant LAN application:



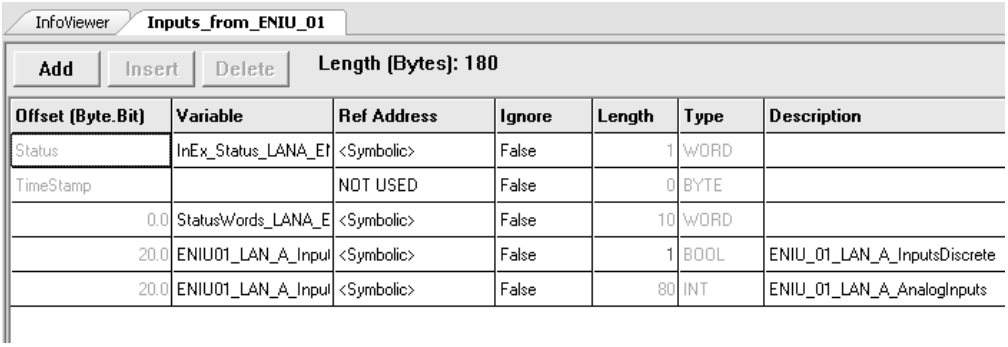
Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type	Description
Status	InEx_Status_LANA_E	<Symbolic>	False	1	WORD	
TimeStamp		NOT USED	False	0	BYTE	
0.0	StatusWords_LANA_I	<Symbolic>	False	10	WORD	
20.0	ENIU01_LAN_A_Inpu	<Symbolic>	False	200	BOOL	ENIU_01_LAN_A_InputsDiscrete
45.0	ENIU01_LAN_A_Inpu	<Symbolic>	False	80	INT	ENIU_01_LAN_A_AnalogInputs

In dual/redundant Ethernet LAN applications with no discrete inputs, the symbolic variable *ENIUxx_LAN_y_InputsDiscrete* (xx represents the ENIU number and y represents the LAN letter) must be used for the Ethernet Global Data exchange. The length of the data area is specified by setting the value in Array Dimension 1in the properties of the variable. These variables contain the discrete input data before the input arbitration logic determines which LAN (variable) is the currently-active one.



For Ethernet NIUs that do not use discrete inputs, the default value of 200 can be left as-is, reserving 200 discrete inputs for future use. Alternately, the value can be changed to 0 to indicate no discrete inputs are required for the Ethernet NIU. Be sure to make the same change for both the LAN A and LAN B variables and the Xfer variables

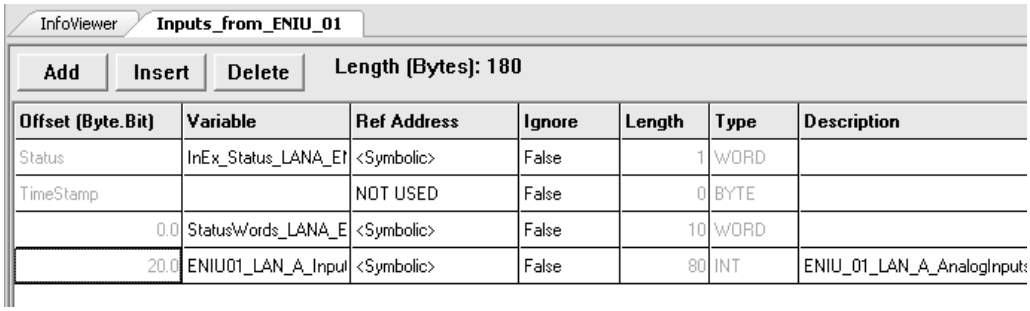
Example EGD Exchange for NO discrete inputs and 80 analog inputs after array dimension value changed to 0:



Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type	Description
Status	InEx_Status_LANA_EI	<Symbolic>	False	1	WORD	
TimeStamp		NOT USED	False	0	BYTE	
0.0	StatusWords_LANA_E	<Symbolic>	False	10	WORD	
20.0	ENIU01_LAN_A_Inpul	<Symbolic>	False	1	BOOL	ENIU_01_LAN_A_InputsDiscrete
20.0	ENIU01_LAN_A_Inpul	<Symbolic>	False	80	INT	ENIU_01_LAN_A_AnalogInputs

Changing the value to 0 means that one boolean of %I is used, because this is a change to the array dimension (making the array variable a non-array, single boolean variable). The Input Arbitration logic understands this to mean that no %I space is to be reserved for the Ethernet NIU. However, the programmer will flag this as an error during validation, so the discrete inputs data area must be deleted from the exchange.

Corrected example of EGD Exchange with no discrete inputs and 80 analog inputs

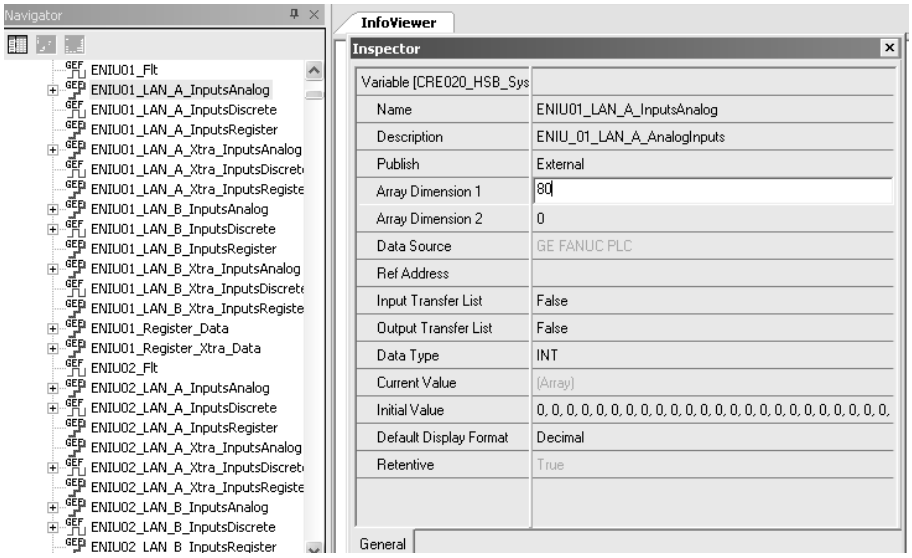


Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type	Description
Status	InEx_Status_LANA_EI	<Symbolic>	False	1	WORD	
TimeStamp		NOT USED	False	0	BYTE	
0.0	StatusWords_LANA_E	<Symbolic>	False	10	WORD	
20.0	ENIU01_LAN_A_Inpul	<Symbolic>	False	80	INT	ENIU_01_LAN_A_AnalogInputs

The EGD exchanges for both LAN A and LAN B need to be modified to remove the discrete input data areas.

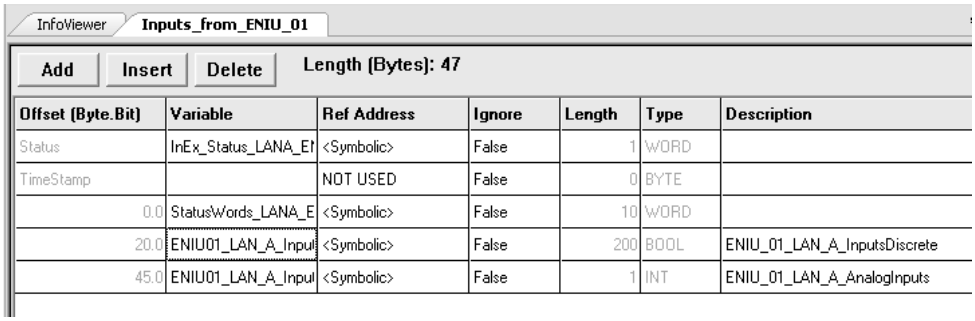
Configuring the Controller Exchange for NO Analog Inputs

In dual/redundant Ethernet LAN applications, the symbolic variable *ENIUxx_LAN_y_I*nputsAnalog (xx represents the Ethernet NIU number and y represents the LAN letter) must be used for the Ethernet Global Data exchange. The length of the data area is specified by setting the value in Array Dimension 1 in the properties of the variable. These variables contain the analog input data before the input arbitration logic determines which LAN (variable) is currently active.



Changing the value to 0 will indicate that one INT of analog input is used. It changes the array dimension, making the array variable into a non-array single variable of type INT. The Input Arbitration logic understands that this means no %AI space should be reserved for the Ethernet NIU. However, the Ethernet Global Data exchange will have two more bytes than expected, so the analog inputs data area must be deleted from the exchange.

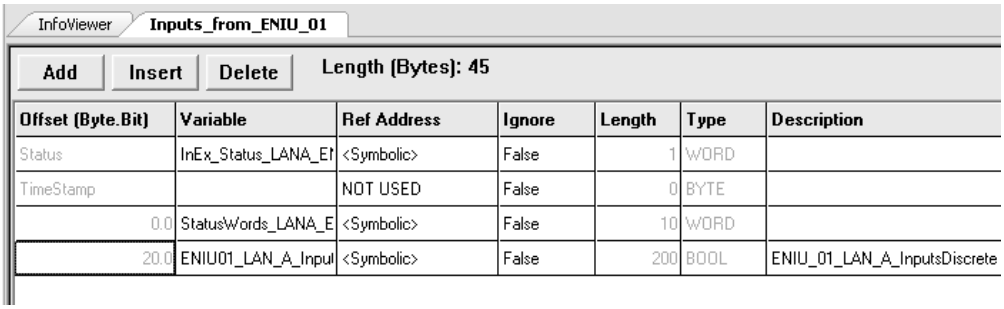
Example of EGD Exchange with 200 discrete inputs and 0 analog inputs after array dimension value set to 0.



Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type	Description
Status	InEx_Status_LANA_EI	<Symbolic>	False	1	WORD	
TimeStamp		NOT USED	False	0	BYTE	
0.0	StatusWords_LANA_E	<Symbolic>	False	10	WORD	
20.0	ENIU01_LAN_A_Inpul	<Symbolic>	False	200	BOOL	ENIU_01_LAN_A_InputsDiscrete
45.0	ENIU01_LAN_A_Inpul	<Symbolic>	False	1	INT	ENIU_01_LAN_A_AnalogInputs

This results in 47 bytes configured, where 45 bytes is expected (for 10 words of status, 200 booleans of discrete inputs, and no analog inputs).

Corrected Ethernet Global Data exchange with 200 discrete inputs and NO analog inputs.



InfoViewer | Inputs_from_ENIU_01

Add | Insert | Delete | Length (Bytes): 45

Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type	Description
Status	InEx_Status_LANA_EI	<Symbolic>	False	1	WORD	
TimeStamp		NOT USED	False	0	BYTE	
0.0	StatusWords_LANA_E	<Symbolic>	False	10	WORD	
20.0	ENIU01_LAN_A_Input	<Symbolic>	False	200	BOOL	ENIU_01_LAN_A_InputsDiscrete

The Ethernet Global Data exchanges for both LAN A and LAN B need to be modified to remove the analog input data areas.

The templates use a range of inputs in each Ethernet NIU. The input reference addresses in the Ethernet NIU are the same as in the controllers. This is done by using the same reference address on the data ranges on the Ethernet Global Data exchange produced in the Ethernet NIU and consumed in the controllers. The Input Arbitration logic in the RX7i CRE and RX3i CRU controller templates sets the addressing in the controllers. For example:

- ENIU #1 sends %I00001 to %I00200 in an EGD exchange to the controllers and the EGD exchange in the controllers puts the data in %I00001 to %I00200.
- ENIU #2 sends %I00201 to %I00400 in an EGD exchange to the controllers and the EGD exchange in the controllers puts the data in %I00201 to %I00400.

If there is no expected expansion of the number of Ethernet NIUs, any unused Ethernet NIU targets should be deleted.

Store the programs and configurations to the Ethernet NIUs and make sure the Ethernet NIUs are in Run mode.

If the Remote COMMREQ Calls (RCC) feature will be used, see chapter 12 for RCC configuration steps.

Configuration Examples for a Single LAN System

Example 1: Default Input Addressing, Single LAN

In this example, input sizes are the same. Reference addresses start at 1 and are sequential. The number of discrete inputs MUST be an even number of bytes, and the starting address MUST be on a byte boundary. The inputs map to the following locations in the controller(s):

		ENIU_01	*ENIU_02*	*ENIU_03*
Discrete Inputs	*Start Address*	%I0001	%I0201	%I0401
	Length (bits)	200	200	200
Analog Inputs	*Start Address*	%AI0001	%AI0129	%AI0257
	Length (words)	128	128	128

Configuration of Inputs from ENIU_01 for Example 1

Ethernet NIU setup						*Controller Setup*					
ENIU 01 Produced Exchange *Inputs_from_ENIU_01*						Controller Consumed Exchange *Inputs_from_ENIU_01*					
Vari-able	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	*InEx_Status_LANA_ENIU_01*	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	*StatusWords_LANA_ENIU_01*	<Sym>	False	10	Word	
	%I0001	N/A	200	Bit	Discrete Inputs from ENIU	*ENIU01_LAN_A_Inputs Discrete*	<Sym>	False	200	Bit	
	%AI001	N/A	128	Word	Analog Inputs from ENIU	*ENIU01_LAN_A_Inputs Analog*	<Sym>	False	128	Word	

Configuration of Inputs from ENIU_02 for Example 1

Ethernet NIU setup						*Controller Setup*					
ENIU 02 Produced Exchange *Inputs_from_ENIU_02*						Controller Consumed Exchange *Inputs_from_ENIU_02*					
Vari-able	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	*InEx_Status_LANA_ENIU_02*	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	*StatusWords_LANA_ENIU_02*	<Sym>	False	10	Word	
	%I0201	N/A	200	Bit	Discrete Inputs from ENIU	*ENIU02_LAN_A_Inputs Discrete*	<Sym>	False	200	Bit	
	%AI129	N/A	128	Word	Analog Inputs from ENIU	*ENIU02_LAN_A_Inputs Analog*	<Sym>	False	128	Word	

Configuration of Inputs from ENIU_03 for Example 1

Ethernet NIU setup						*Controller Setup*					
ENIU 03 Produced Exchange *Inputs_from_ENIU_03*						Controller Consumed Exchange *Inputs_from_ENIU_03*					
Vari-able	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	*InEx_Status_LANA_ENIU_03*	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	*StatusWords_LANA_ENIU_03*	<Sym>	False	10	Word	
	%I0401	N/A	200	Bit	Discrete Inputs from ENIU	*ENIU03_LAN_A_Inputs Discrete*	<Sym>	False	200	Bit	
	%AI257	N/A	128	Word	Analog Inputs from ENIU	*ENIU03_LAN_A_Inputs Analog*	<Sym>	False	128	Word	

Example 2: Adding Inputs to the Configuration, Single LAN

If an Ethernet NIU needs more than 200 discrete inputs or 128 analog inputs, the Ethernet NIU configuration must be changed. In this example, ENIU02 has 250 discrete inputs and 100 analog inputs. As shown in the example, because ENIU02 changes the length of data, this results in the starting addresses in ENIU03 and any higher ENIUs to change. Reference addresses start at 1 and are sequential. Input usage in the Ethernet NIUs for example 2 is:

		ENIU_01	*ENIU_02*	*ENIU_03*
Discrete Inputs	*Start Address*	%I0001	%I0201	%I0451
	Length (bits)	200	250	200
Analog Inputs	*Start Address*	%AI0001	%AI0129	%AI0229
	Length (words)	128	100	128

Configuration of Inputs from ENIU_01 for Example 2

ENIU_01 has no changes to its setup from Example 1.

Configuration of Inputs from ENIU_02 for Example 2

The Items in *italics* below are the items to be changed for ENIU_02 for Example 2.

Ethernet NIU setup						*Controller Setup*					
ENIU 02 Produced Exchange Inputs_from_ENIU_02						Controller Consumed Exchange Inputs_from_ENIU_02					
Vari-able	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_02	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_02	<Sym>	False	10	Word	
	%I0201	N/A	*250*	Bit	Discrete Inputs from ENIU	ENIU02_LAN_A_Inputs Discrete (array dimension 250)	<Sym>	False	*250*	Bit	
	%AI129	N/A	*100*	Word	Analog Inputs from ENIU	ENIU02_LAN_A_Inputs Analog (array dimension 100)	<Sym>	False	*100*	Word	

Configuration of Inputs from ENIU_03 for Example 2

The Items in italics below are the items to be changed for ENIU_03 for Example 2.

Ethernet NIU setup						*Controller Setup*					
ENIU 03 Produced Exchange Inputs_from_ENIU_03						Controller Consumed Exchange Inputs_from_ENIU_03					
Vari-able	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_03	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_03	<Sym>	False	10	Word	
	%I0451	N/A	200	Bit	Discrete Inputs from ENIU	ENIU03_LAN_A_InputsDiscrete	<Sym>	False	200	Bit	
	%AI229	N/A	128	Word	Analog Inputs from ENIU	ENIU03_LAN_A_InputsAnalog (array dimension 128)	<Sym>	False	128	Word	

Configuration Examples for a Dual LAN System

Example 3: Default Input Addressing, Dual LAN

In this example, input sizes are the same. Reference addresses start at 1 and are sequential. The number of discrete inputs MUST be an even number of bytes, and the starting address MUST be on a byte boundary. The inputs map to the following locations in the controller(s):

		ENIU_01	*ENIU_02*	*ENIU_03*
Discrete Inputs	*Start Address*	%I0001	%I0201	%I0401
	Length (bits)	200	200	200
Analog Inputs	*Start Address*	%AI0001	%AI0129	%AI0257
	Length (words)	128	128	128

Configuration of Inputs from ENIU_01_LANA for Example 3

Ethernet NIU setup						*Controller Setup*					
ENIU 01 Produced Exchange Inputs_from_ENIU_01						CRE HSB System Consumed Exchange Inputs_from_ENIU_01					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_01	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_01	<Sym>	False	10	Word	
	%I0001	N/A	200	Bit	Discrete Inputs from ENIU	ENIU01_LAN_A_Inputs Discrete	<Sym>	False	200	Bool	
	%AI001	N/A	128	Word	Analog Inputs from ENIU	ENIU01_LAN_A_Inputs Analog	<Sym>	False	128	Int	

Configuration of Inputs from ENIU_01_LANB for Example 3

Ethernet NIU setup						*Controller Setup*					
ENIU 01 Produced Exchange Inputs_from_ENIU_01_LANB						CRE HSB System Consumed Exchange Inputs_from_ENIU_01_LANB					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0081	False	16	Bit	Status LAN B	InEx_Status_LANB_ENIU_01	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1151	N/A	10	Word	Status Words from Ethernet NIU LAN B	StatusWords_LANB_ENIU_01	<Sym>	False	10	Word	
	%I0001	N/A	200	Bit	Discrete Inputs from ENIU	ENIU01_LAN_B_Inputs Discrete	<Sym>	False	200	Bool	
	%AI001	N/A	128	Word	Analog Inputs from ENIU	ENIU01_LAN_B_Inputs Analog	<Sym>	False	128	Int	

Inputs produced on LAN B are the same as in LAN A, only Status is different. Variables used in the consumed exchange are different.

Configuration of Transfer Variables in the Controller for ENIU01 for Example 3

Ethernet NIU setup	*Controller Setup*						
	Variable	Array Dimension					
[N/A]	ENIU01_Xfer_Pri_InputsDiscrete	200					
	ENIU01_Xfer_Pri_InputsAnalog	128					
	ENIU01_Xfer_Sec_InputsDiscrete	200					
	ENIU01_Xfer_Sec_InputsAnalog	128					

Configuration of Inputs from ENIU_02_LANA for Example 3

Ethernet NIU setup						*Controller Setup*					
ENIU 02 Produced Exchange Inputs_from_ENIU_02						CRE HSB System Consumed Exchange Inputs_from_ENIU_02					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_02	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_02	<Sym>	False	10	Word	
	%I0201	N/A	200	Bit	Discrete Inputs from ENIU	ENIU02_LAN_A_Inputs Discrete	<Sym>	False	200	Bool	
	%AI129	N/A	128	Word	Analog Inputs from ENIU	ENIU02_LAN_A_Inputs Analog	<Sym>	False	128	Int	

Configuration of Inputs from ENIU_02_LANB for Example 3

Ethernet NIU setup						*Controller Setup*					
ENIU 02 Produced Exchange Inputs_from_ENIU_02_LANB						CRE HSB System Consumed Exchange Inputs_from_ENIU_02_LANB					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0081	False	16	Bit	Status LAN B	InEx_Status_LANB_ENIU_02	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1151	N/A	10	Word	Status Words from Ethernet NIU LAN B	StatusWords_LANB_ENIU_02	<Sym>	False	10	Word	
	%I0201	N/A	200	Bit	Discrete Inputs from ENIU	ENIU02_LAN_B_Inputs Discrete	<Sym>	False	200	Bool	
	%AI129	N/A	128	Word	Analog Inputs from ENIU	ENIU02_LAN_B_InputsAnalog	<Sym>	False	128	Int	

Inputs produced on LAN B are the same as in LAN A, only Status is different. Variables used in the consumed exchange are different.

Configuration of Transfer Variables in the Controller for ENIU02 for Example 3

Ethernet NIU setup	*Controller Setup*						
	CRE Internal Transfer Variables for ENIU02						
	Variable	Array Dimension					
[N/A]	ENIU02_Xfer_Pri_InputsDiscrete	200					
	ENIU02_Xfer_Pri_InputsAnalog	128					
	ENIU02_Xfer_Sec_InputsDiscrete	200					
	ENIU02_Xfer_Sec_InputsAnalog	128					

Configuration of Inputs from ENIU_03_LANA for Example 3

Ethernet NIU setup						*Controller Setup*					
ENIU 03 Produced Exchange Inputs_from_ENIU_03						CRE HSB System Consumed Exchange Inputs_from_ENIU_03					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_03	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_03	<Sym>	False	10	Word	
	%I0401	N/A	200	Bit	Discrete Inputs from ENIU	ENIU03_LAN_A_Inputs Discrete	<Sym>	False	200	Bool	
	%AI257	N/A	128	Word	Analog Inputs from ENIU	ENIU03_LAN_A_Inputs Analog	<Sym>	False	128	Int	

Configuration of Inputs from ENIU_03_LANB for Example 3

Ethernet NIU setup						*Controller Setup*					
ENIU 03 Produced Exchange Inputs_from_ENIU_03						CRE HSB System Consumed Exchange Inputs_from_ENIU_03					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0081	False	16	Bit	Status LAN B	InEx_Status_LANB_ENIU_03	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1151	N/A	10	Word	Status Words from Ethernet NIU LAN B	StatusWords_LANB_ENIU_03	<Sym>	False	10	Word	
	%I0401	N/A	200	Bit	Discrete Inputs from ENIU	ENIU03_LAN_B_Inputs Discrete	<Sym>	False	200	Bool	
	%AI257	N/A	128	Word	Analog Inputs from ENIU	ENIU03_LAN_B_Inputs Analog	<Sym>	False	128	Int	

Inputs produced on LAN B are the same as in LAN A, only Status is different. Variables used in the consumed exchange are different.

Configuration of Transfer Variables in the Controller for ENIU03 for Example 3

Ethernet NIU setup	*Controller Setup*					
	CRE Internal Transfer Variables for ENIU03					
	Variable	Array Dimension				
[N/A]	ENIU03_Xfer_Pri_InputsDiscrete	200				
	ENIU03_Xfer_Pri_InputsAnalog	128				
	ENIU03_Xfer_Sec_InputsDiscrete	200				
	ENIU03_Xfer_Sec_InputsAnalog	128				

Example 4: Adding Inputs to the Configuration, Dual LAN

If an Ethernet NIU needs more than 200 discrete inputs or 128 analog inputs, the Ethernet NIU configuration must be changed. In this example, ENIU02 has 250 discrete inputs and 150 analog inputs. As shown in the example, because ENIU02 changes the length of data, this results in the starting addresses in ENIU03 and any higher Ethernet NIUs to change. Reference addresses start at 1 and are sequential. Input usage in the Ethernet NIUs for example 4 is shown below. Note that the starting addresses in ENIU_02 are directly after the last address in ENIU_01.

		ENIU_01	*ENIU_02*	*ENIU_03*
Discrete Inputs	*Start Address*	%I0001	%I0201	%I0451
	Length (bits)	200	250	200
Analog Inputs	*Start Address*	%AI0001	%AI0129	%AI0279
	Length (words)	128	150	128

Configuration of Inputs from ENIU_01 for Example 4

ENIU_01 has no changes to its setup from Example 4.

Configuration of Inputs from ENIU_02 for Example 4

The Items in italics below are the items to be changed for ENIU_02 for Example 4.

Ethernet NIU setup						*Controller Setup*					
ENIU 02 Produced Exchange Inputs_from_ENIU_02						CRE HSB System Consumed Exchange Inputs_from_ENIU_02					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_02	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_02	<Sym>	False	10	Word	
	%I0201	N/A	*250*	Bit	Discrete Inputs from ENIU	ENIU02_LAN_A_Inputs Discrete	<Sym>	False	*250*	Bit	
	%AI129	N/A	*150*	Word	Analog Inputs from ENIU	ENIU02_LAN_A_Inputs Analog	<Sym>	False	*150*	Word	

Ethernet NIU setup						*Controller Setup*					
ENIU 02 Produced Exchange Inputs_from_ENIU_02_LANB						CRE HSB System Consumed Exchange Inputs_from_ENIU_02_LANB					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN B	InEx_Status_LANB_ENIU_02	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN B	StatusWords_LANB_ENIU_02	<Sym>	False	10	Word	
	%I0201	N/A	*250*	Bit	Discrete Inputs from ENIU	ENIU02_LAN_B_Inputs Discrete	<Sym>	False	*250*	Bit	
	%AI129	N/A	*100*	Word	Analog Inputs from ENIU	ENIU02_LAN_B_Inputs Analog	<Sym>	False	*100*	Word	

Configuration of Transfer Variables in the Controller for ENIU02 for Example 4

Ethernet NIU setup	*Controller Setup*					
	CRE Internal Transfer Variables for ENIU02					
	Variable	Array Dimension				
[N/A]	ENIU02_Xfer_Pri_InputsDiscrete	250				
	ENIU02_Xfer_Pri_InputsAnalog	150				
	ENIU02_Xfer_Sec_InputsDiscrete	250				
	ENIU02_Xfer_Sec_InputsAnalog	150				

Configuration of Inputs from ENIU_03 for Example 4

The Items in italics below are the items to be changed for ENIU_03 for Example 4.

Ethernet NIU setup						*Controller Setup*					
ENIU 03 Produced Exchange Inputs_from_ENIU_03						CRE HSB System Consumed Exchange Inputs_from_ENIU_03					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0033	False	16	Bit	Status LAN A	InEx_Status_LANA_ENIU_03	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1101	N/A	10	Word	Status Words from Ethernet NIU LAN A	StatusWords_LANA_ENIU_03	<Sym>	False	10	Word	
	%I0451	N/A	200	Bit	Discrete Inputs from ENIU	ENIU03_LAN_A_Inputs Discrete	<Sym>	False	200	Bit	
	%AI279	N/A	128	Word	Analog Inputs from ENIU	ENIU03_LAN_A_Inputs Analog	<Sym>	False	128	Word	

Ethernet NIU setup						*Controller Setup*					
ENIU 03 Produced Exchange Inputs_from_ENIU_03_LANB						CRE HSB System Consumed Exchange Inputs_from_ENIU_03_LANB					
Var.	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
	%T0081	False	16	Bit	Status LAN B	InEx_Status_LANB_ENIU_03	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
	%R1151	N/A	10	Word	Status Words from Ethernet NIU LAN B	StatusWords_LANB_ENIU_03	<Sym>	False	10	Word	
	%I0451	N/A	200	Bit	Discrete Inputs from ENIU	ENIU02_LAN_B_Inputs Discrete	<Sym>	False	200	Bit	
	%AI279	N/A	128	Word	Analog Inputs from ENIU	ENIU02_LAN_B_Inputs Analog	<Sym>	False	128	Word	

Configuration of Transfer Variables in the Controller for ENIU03 for Example 4

Ethernet NIU setup	*Controller Setup*					
	CRE Internal Transfer Variables for ENIU03					
	Variable	Array Dimension				
[N/A]	ENIU03_Xfer_Pri_InputsDiscrete	200				
	ENIU03_Xfer_Pri_InputsAnalog	128				
	ENIU03_Xfer_Sec_InputsDiscrete	200				
	ENIU03_Xfer_Sec_InputsAnalog	128				

Powerup Operation

At powerup, the Ethernet NIU searches for a healthy communications link from which to receive its outputs. In a dual LAN system, the Ethernet NIU searches for a healthy communications link in a specific order and accept outputs from the first healthy link it encounters.

The search order is:

Primary Controller LAN A

Primary Controller LAN B

Secondary Controller LAN A

Secondary Controller LAN B

Chapter 4 Input Processing by a CPU

Controller(s) that are driving Ethernet NIUs must handle the inputs returned by the Ethernet NIUs. The input-handling function is provided by a C block in the template set used for the application. The name of the C block that handles ENIU inputs depends on the version of the application templates that are being used.

- Starting with version 1.40 of the Ethernet NIU application templates, the C block *Input_Processing* is used in all the controller application programs in the GE Fanuc-provided template sets.
- With earlier versions of the application template sets, the C block that handled inputs was named *Input_Arbitration* for PME projects and *Input_Processing* for PPS projects Operation of the *Input_Arbitration* C block is basically the same as operation of the *Input_Processing* C block, which is described in this chapter.

This chapter covers:

- *Input Processing* Block
  - Configuring the Input Processing C Block
- Symbolic Variables for the Input Processing Function
- Input Processing Error Codes
- Redundant Controller, Dual LAN Considerations
  - Switching Logic
  - Predefined Signals for Custom Switching Logic
  - Dedicated Signals
- Point Fault / Data Quality Feature
  - Enabling Point Fault References
  - Operation of Point Fault References

For more information about the application templates, see chapter 3.

The Input_Processing Block

The *Input_Processing* block provides the following functions:

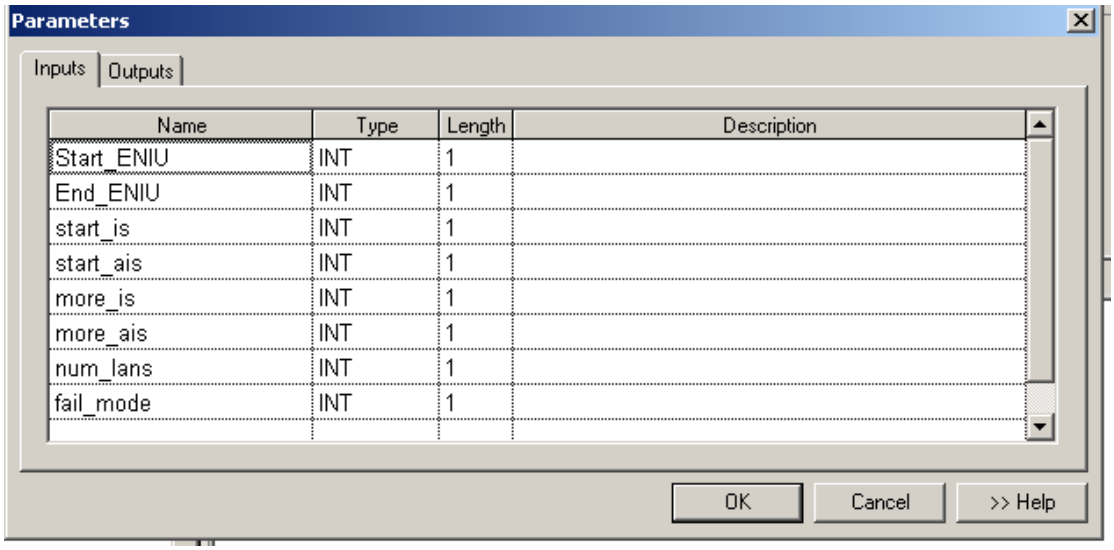
- Checks if inputs are being received from the Ethernet NIU(s).
- Zeroes or Holds Last State for inputs if communication is lost to the Ethernet NIU(s).
- Generates fault table entries if communication with the Ethernet NIU is lost.
- If Point Faults are enabled, sets Point Faults if communication with the Ethernet NIU is lost.
- Provides status bits for health of communication to the Ethernet NIUs.
- In Redundant Controller and Dual LAN systems, provides status bits that show the controller and LAN currently controlling the Ethernet NIU.
- In Redundant Controller and Dual LAN systems, provides status bits that show the health of each communication path to the Ethernet NIU(s).
- In Dual LAN system, checks communication with the Ethernet NIU on both LANs.
- In Dual LAN systems, decides which LAN to use for communication with the Ethernet NIU.

If the templates described in Chapter 3 have been used, the *Input_Processing* C block is automatically included and configured.

Configuring the Input Processing C Block

The inputs and outputs of the C block are described below.

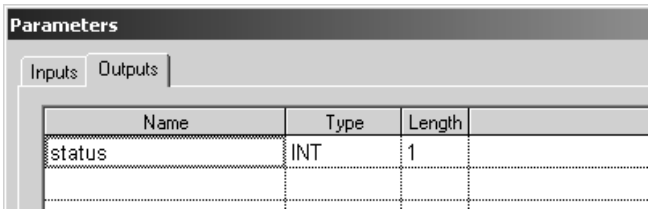
Inputs for the C Block



Start_ENIU	Constant that is the Ethernet NIU number of the first Ethernet NIU, typically: 1
End_ENIU	Constant that is the number of the last Ethernet NIU. Ethernet NIUs should be numbered consecutively.
start_is	Constant that is the offset in %I memory where the first discrete input from the Start ENIU will be written. Inputs are packed consecutively in the controller. It does not need to start at 1, but must be on a byte boundary (33, 129 etc…).
start_ais	Constant that is the offset in %AI memory where the first analog input from the Start ENIU will be written. Inputs are packed consecutively in the controller.
more_is	Constant that is the offset in %I memory where additional discrete inputs from the Start ENIU will be written. Inputs are packed consecutively in the controller. It does not need to start at 1, but must be on a byte boundary (33, 129 etc…). Must not overlap with start_is range.
more_ais	Constant that is the offset in %AI memory where additional analog input from the Start ENIU will be written. Must not overlap with start_ais range.
*num_lans	1= single LAN, 2= (not applicable), 3 = dual LAN with dual interfaces (2 Ethernet Transmitter Modules)
*fail_mode	specifies mode when there is communication fault with the Ethernet NIU, 0 = zero inputs, any other value (non-zero) = hold last state.

* not present on *Input_Arbitration* C block

Output of the C Block



status — Status of the Input_Processing C block call. Enter a Variable of type Integer. This location will be monitored to determine completion and success of the configuration of the Input_Processing block. Input_Processing block will check all inputs to the block the first time it is executed, and will output an error of the form xxyy (in hex) if an error is found. xx is the Ethernet NIU number and yy is the Error Code. If the xx is 00, the error code is a common parameter. If no error is found the block returns a value of 1 and the inputs are provided to the controller. If an error is returned, the error must be corrected and the controller must be restarted to reset the Input_Processing block. If the Input_Processing block indicates an error in the status output, the inputs are not updated in the controller. See page 4-6 for the list of error codes.

Note: The Input Processing block does not support Run Mode Store (RMS) of EGD. RMS should be set to False for EGD exchanges going to the ENIUs

Symbolic Variables for the Input Processing Function

The input processing function of the C block requires that specific symbolic variables be used in the controller for all Ethernet Global Data exchanges. If the specified symbolic variables are not used, input processing will not work properly.

The symbolic variables must be declared as variables in the controller and published either internally or externally. Otherwise, either the controller program will not store to the PLC, or the PLC will have a fault when it attempts to go into Run mode.

The project templates automatically declare the variables in the controller and set up all the exchanges with the correct variables.

The variables that must be used on the *Input_from_ENIU_xx* exchanges are:

- *ENIUxx_LAN_y_InputsDiscrete* – Discrete Inputs from Ethernet NIU xx on LAN y.
- *ENIUxx_LAN_y_InputsAnalog* – Analog Inputs from Ethernet NIU xx on LAN y.
- *ENIUxx_LAN_y_InputsRegister* – Optional word inputs from Ethernet NIU xx on LAN y. These input are placed in the symbolic variable *ENIUxx_Register_Data* after input processing.
- *ENIUxx_LAN_y_Xtra_InputsDiscrete* – Optional additional discrete inputs from Ethernet NIU xx on LAN y, only used if all discrete inputs are all used and more inputs are needed.
- *ENIUxx_LAN_y_Xtra_InputsAnalog* – Optional additional analog inputs from Ethernet NIU xx on LAN y, only used if all analog inputs are all used and more are needed.

In Redundant controller configurations, the Input Processing block also uses a number of symbolic variables per ENIU to transfer the information from the active controller to the backup controller. These symbolic variables will have be of the form:

ENIUxx_yy_Xfer_zzzzzz.

Input Processing – Error Codes

The input processing function requires configuration of the I/O parameters. The status variable on the output of the block contains a value of 0001 when the configuration is correct. Most error codes identify the number of the Ethernet NIU in the upper byte and the error code in the lower byte. Error codes that do not include an Ethernet NIU number have 00 in the upper byte and report the error code in the lower byte. It is best to display the error code in Hex.

Note: Inputs are not passed to the %I and %AI reference tables of the controller until the Input_Processing block is configured correctly and returning a status of 0001.

Error	*Error Code*	*Error*	*Error Code*
BAD_I_DATATYPE_LANA	xx10	I_MORE_DATASIZE_A_B_DIFFERENT	xx5c
BAD_AI_DATATYPE_LANA	xx11	AI_MORE_DATASIZE_A_B_DIFFERENT	xx5d
BAD_EX_DATATYPE_LANA	xx12	EX_MORE_DATASIZE_A_B_DIFFERENT	xx5e
BAD_I_DATATYPE_LANB	xx13		
BAD_AI_DATATYPE_LANB	xx14	I_MORE_DATA_SIZE_NOT_BYTE_MULTIPLE	xx60
BAD_EX_DATATYPE_LANB	xx15	HIGHEST_MORE_I_EXCEEDS_32000	xx61
BAD_I_DATASIZE_LANA	xx16	HIGHEST_MORE_AI_EXCEEDS_TABLE	xx62
BAD_AI_DATASIZE_LANA	xx17		
BAD_EX_DATASIZE_LANA	xx18	I_MORE_DATA_START_NOT_BYTE_BOUNDARY	0064
BAD_I_DATASIZE_LANB	xx19	START_ADDR_ZERO_WITH_I_MORE_DATA_LEN	xx66
BAD_AI_DATASIZE_LANB	xx1a	START_ADDR_ZERO_WITH_AI_MORE_DATA_LEN	xx67
BAD_EX_DATASIZE_LANB	xx1b	I_DATASIZE_PRI_XFER_DIFFERENT	xx71
I_DATASIZE_A_B_DIFFERENT	xx1c	I_DATASIZE_SEC_XFER_DIFFERENT	xx72
AI_DATASIZE_A_B_DIFFERENT	xx1d	AI_DATASIZE_PRI_XFER_DIFFERENT	xx73
EX_DATASIZE_A_B_DIFFERENT	xx1e	AI_DATASIZE_SEC_XFER_DIFFERENT	xx74
		EX_DATASIZE_PRI_XFER_DIFFERENT	xx75
		EX_DATASIZE_SEC_XFER_DIFFERENT	xx76
HIGHEST_I_EXCEEDS_32000	xx21	I_MORE_DATASIZE_PRI_XFER_DIFFERENT	xx77
HIGHEST_AI_EXCEEDS_TABLE	xx22	I_MORE_DATASIZE_SEC_XFER_DIFFERENT	xx78
		AI_MORE_DATASIZE_PRI_XFER_DIFFERENT	xx79
I_DATA_START_NOT_BYTE_BOUNDARY	0024	AI_MORE_DATASIZE_SEC_XFER_DIFFERENT	xx7a
		I_DATA_OVERWRITES_XTRA_I_DATA	0xa0
BAD_I_DATATYPE_PRI_XFER	xx41	AI_DATA_OVERWRITES_XTRA_AI_DATA	0xa1
BAD_I_DATATYPE_SEC_XFER	xx42		
BAD_AI_DATATYPE_PRI_XFER	xx43	I_XTRA_DATA_OVERWRITES_I_DATA	0xa3
BAD_AI_DATATYPE_SEC_XFER	xx44	AI_XTRA_DATA_OVERWRITES_AI_DATA	0xa4
BAD_EX_DATATYPE_PRI_XFER	xx45	START_ENIU_INPUT_BAD	0xb0
BAD_EX_DATATYPE_SEC_XFER	xx46	END_ENIU_LESS_THAN_START_ENIU	0xb1
BAD_I_MORE_DATATYPE_PRI_XFER	xx47	ENIU_GREATER_THAN_63	0xb2
BAD_I_MORE_DATATYPE_SEC_XFER	xx48	I_DATA_START_ADDRESS_BAD	00c1
BAD_AI_MORE_DATATYPE_PRI_XFER	xx49	AI_DATA_START_ADDRESS_BAD	00c2
BAD_AI_MORE_DATATYPE_SEC_XFER	xx4a	I_XTRA_DATA_START_ADDRESS_BAD	00c4

Error	Error Code	Error	Error Code
BAD_I_MORE_DATATYPE_LANA	xx50	AI_XTRA_DATA_START_ADDRESS_BAD	00c5
BAD_AI_MORE_DATATYPE_LANA	xx51		
BAD_EX_MORE_DATATYPE_LANA	xx52		
BAD_I_MORE_DATATYPE_LANB	xx53		
BAD_AI_MORE_DATATYPE_LANB	xx54	IO_LAN_STATUS_BAD	0xf2
BAD_EX_MORE_DATATYPE_LANB	xx55	ETM_LANA_BAD	0xf3
BAD_I_MORE_DATASIZE_LANA	xx56	ETM_LANB_BAD	0xf4
BAD_AI_MORE_DATASIZE_LANA	xx57	REGISTER_DATA_TOO_SMALL	0x31
BAD_EX_MORE_DATASIZE_LANA	xx58		
BAD_I_MORE_DATASIZE_LANB	xx59		
BAD_AI_MORE_DATASIZE_LANB	xx5a		
BAD_EX_MORE_DATASIZE_LANB	xx5b		

Redundant Controller, Dual LAN Considerations

Switching Logic

Switching logic is provided in CRE and CRU Controller templates. Switching conditions are generated in the ladder block *ENIUs_Ctrl_Check.* The switching conditions are solved in the block *ENIUs_Interface*. The basic operation of the switching logic is:

- When an Ethernet NIU is powered up, the Ethernet NIU defaults to LAN A. The application can use the switching logic to request that the Ethernet NIUs use LAN B.
- Reestablishment or loss of the inactive LAN does not cause any switching action to the inactive LAN.
- Transfer of controller causes the Ethernet NIU to follow the other controller using the same LAN as used before.

The switching logic can be customized for the application by revising the logic in the block *ENIUs_Interface*.

There are up to four rungs with similar structure, which are used for switching to primary controller LAN A, primary controller LAN B, secondary controller LAN A or secondary controller LAN B. The following control signals control switching between the LANs:

- *ControlWords_LANA_B[0].X[3]*,
- *ControlWords_LANA_B[0].X[13]*,
- *ControlWords_LANA_B[0].X[4],* and
- *ControlWords_LANA_B[0].X[14]*

The logic controlling these coils can be modified as needed.

Predefined Signals for Custom Switching Logic

The following predefined signals are available for use in any custom switching logic or HMI application. These signals are in addition to the standard system variables (# variables) that are available in controller applications. See GFK-2222 *PACSystems CPU Reference Manual* and GFK-2308 *PACSystems Hot Standby CPU Redundancy Manual* for more details on system variables.

Signal Name	Description
ENIUxx_on_PriA	ENIUxx is being controlled by primary controller on LAN A
ENIUxx_on_SecA	ENIUxx is being controlled by secondary controller on LAN A
ENIUxx_on_PriB	Dual LAN only ENIUxx is being controlled by primary controller on LAN B
ENIUxx_on_SecB	Dual LAN only ENIUxx is being controlled by secondary controller on LAN B
ENIUxx_CommOK_PrimaryA	ENIUxx communication good to primary controller on LAN A
ENIUxx_CommOK_SecondaryA	ENIUxx communication good to secondary controller on LAN A
ENIUxx_CommOK_PrimaryB	Dual LAN only ENIUxx comm good to primary controller on LAN B
ENIUxx_CommOK_SecondaryB	Dual LAN only ENIUxx comm good to secondary controller on LAN B
ENIUxx_Flt	No communication to ENIUxx
Ctl_by_Standby	Standby controller only – Standby controller in controlling one or more Ethernet NIUs
Pri_Ctl_On_A_B	Primary controller is controlling some Ethernet NIUs on LAN A and some on LAN B
Sec_Ctl_On_A_B	Secondary controller is controlling some Ethernet NIUs on LAN A and some on LAN B
NoENIUonPriLANA	No Ethernet NIUs are being controlled by primary controller on LAN A
NoENIUonPriLANB	No Ethernet NIUs are being controlled by primary controller on LAN B
NoENIUonSecLANA	No Ethernet NIUs are being controlled by secondary controller on LAN A
NoENIUonSecLANB	No Ethernet NIUs are being controlled by secondary controller on LAN B
ENIUonPriLANA	At least one Ethernet NIU is being controlled using primary controller on LAN A
ENIUonPriLANB	At least one Ethernet NIU is being controlled using primary controller on LAN B
ENIUonSecLANA	At least one Ethernet NIU is being controlled using secondary controller on LAN A

Signal Name	*Description*
ENIUonSecLANB	At least one Ethernet NIU is being controlled using secondary controller on LAN B
Pri_New_Mstr	Operation has switched to the primary controller
Sec_New_Mstr	Operation has switched to the secondary controller
StatusWords_LANy_ENIU_xx	10 words of status data sent by Ethernet NIU to controller(s)**

** See chapter 9 for information on the ten words of status data sent by the Ethernet NIU to the controller(s).

Dedicated Signals

Certain signals should not be either deleted or renamed. These signals must be marked as Publish in their properties. Renaming, deleting or marking Publish as false will cause a failure when the program is downloaded to the controller. These signals are available for use in any custom switching logic or HMI annunciation.

Signal Name	Signal Name
Signals for each Ethernet NIU (indicated by xx) and both LANs (indicated by y)	
ENIUxx_LAN_y_InputsDiscrete	*InEx_Status_LANy_ENIU_xx*
ENIUxx_LAN_y_InputsAnalog	*StatusWords_LANy_ENIU_xx*
ENIUxx_LAN_y_InputsRegister	*SVC_In_Status_LANy_ENIU_xx*
ENIUxx_LAN_y_Xtra_InputsAnalog	*SVC_Out_Status_LANy_ENIU_xx*
ENIUxx_LAN_y_Xtra_InputsDiscrete	*Fltdata_LANy_ENIUxx*
ENIUxx_LAN_y_Xtra_InputsRegister	*RCC_Response_LANy_ENIUxx*
ENIUxx_Xfer_Pri_InputsDiscrete	*ENIUxx_Xfer_Sec_InputsDiscrete*
ENIUxx_ Xfer_Pri _InputsAnalog	*ENIUxx_ Xfer_Sec _InputsAnalog*
ENIUxx_ Xfer_Pri _InputsRegister	*ENIUxx_ Xfer_Sec_InputsRegister*
ENIUxx_ Xfer_Pri _Xtra_InsAnalog	*ENIUxx_ Xfer_Sec_Xtra_InsAnalog*
ENIUxx_ Xfer_Pri _Xtra_InsDiscrete	*ENIUxx_ Xfer_Sec_Xtra_InsDiscrete*
ENIUxx_ Xfer_Pri _Xtra_InsRegister	*ENIUxx_ Xfer_Sec_Xtra_InsRegister*
ENIUxx_ Xfer_Pri _StWords	*ENIUxx_ Xfer_Sec_StWords*
Signals for each Ethernet NIU (indicated by xx)	
RCC_Request_ENIUxx	*fltbuf_eniu_xx*
Fltack_ENIUxx	*fltptr_eniu_xx*
ENIUxx_Register_Data	*ENIUxx_Register_Xtra_Data*
Signals for both LANs (indicated by y)	
OutEx_Status_LANy	
Signals common to all Ethernet NIUs and both LANs	
ControlWords_LANA_B	*Sw_Pri_A*
Eniuoffsetarray	*Sw_Pri_B*
LANA_LSW	*Sw_Sec_A*
LANB_LSW	*Sw_Sec_B*

Point Fault/Data Quality Feature

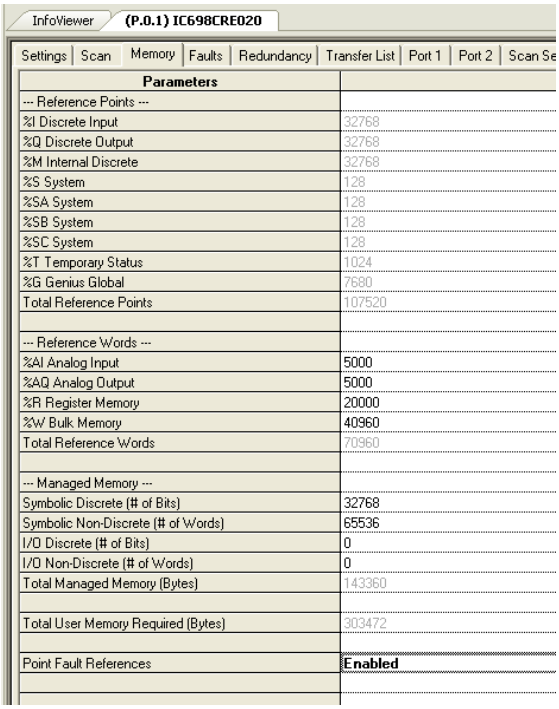
If enabled, Discrete and Analog inputs have a property known as Point Faults References.

PPS Systems data quality uses the Point Fault feature of the PACSystems controller. Refer to Proficy Process System documentation for use of Data Quality feature with in PPS Controller Function Block.

See GFK-2222 *PACSystems CPU Reference Manual* for more details on Point Faults.

Enabling Point Fault References

Point Faults are always enabled in a PPS System. In a non-PPS System, the Point Faults feature is disabled by default and if needed must be enabled as shown below.



Parameters	
--- Reference Points ---	
%I Discrete Input	32768
%Q Discrete Output	32768
%M Internal Discrete	32768
%S System	128
%SA System	128
%SB System	128
%SC System	128
%T Temporary Status	1024
%G Genius Global	7680
Total Reference Points	107520
--- Reference Words ---	
%AI Analog Input	5000
%AQ Analog Output	5000
%R Register Memory	20000
%W Bulk Memory	40960
Total Reference Words	70960
--- Managed Memory ---	
Symbolic Discrete (# of Bits)	32768
Symbolic Non-Discrete (# of Words)	65536
I/O Discrete (# of Bits)	0
I/O Non-Discrete (# of Words)	0
Total Managed Memory (Bytes)	143360
Total User Memory Required (Bytes)	303472
Point Fault References	Enabled

Operation of Point Fault References

When communication to an Ethernet NIU is lost, the CPU sets Point Faults for all %I and %AI inputs from that ENIU. When communication to an Ethernet NIU is restored and inputs are received the CPU clears Point Faults for all %I and %AI inputs from that ENIU.

For dual LAN systems, the CPU sets Point Faults if communication to an Ethernet NIU is lost on both LANs. When there are dual LANs, two sets of inputs from the Ethernet NIUs are available to each of the redundant controllers.

The setting/clearing of Point Fault data is performed by the Input_Processing block, which must be included in the controller application programs to determine which inputs should be used.

Chapter 5 Configuring PC-Based Controllers

Any GE Fanuc Ethernet interface master capable of exchanging Ethernet Global Data messages, such as the Series 90-30 CPU, Series 90-70 CPU, or PC Control can function as a controller for the Ethernet NIU. However, these other controllers do NOT support Redundant I/O LANs, Fault Reporting to the controller or Remote COMMREQ Calls.

In a system that has a primary and secondary controller, it is not necessary for the controllers to be the same type.

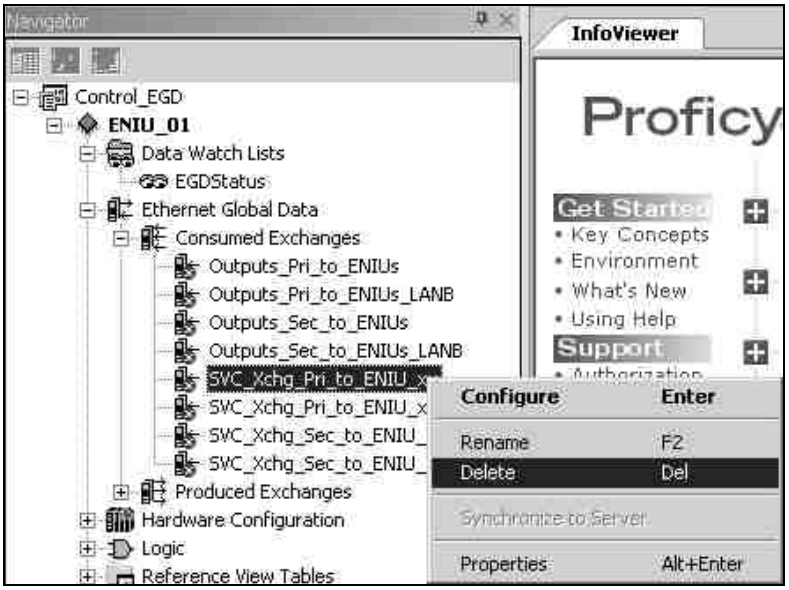
This chapter describes configuration steps when a PC-based controller (either Quickpanel Control or PC Control) will be used as a controller for an Ethernet NIU:

- Configuring the ENIU
- Configuring the Controller

Configuring the Ethernet NIU

Follow the steps in chapter 6 to complete the hardware configuration of an Ethernet NIU. Follow the steps in chapter 7 to configure Ethernet Global Data exchanges.

Because other controllers DO NOT support fault reporting to the controller or Remote COMMREQ Calls, if you are using any of the provided templates, delete the SVC exchanges from the Ethernet Global Data configuration in the Ethernet NIU.



Configuring the PC-Based Controller

For each Ethernet NIU, control variables are needed to handle both produced and consumed data.

The produced data structure consists of three elements.

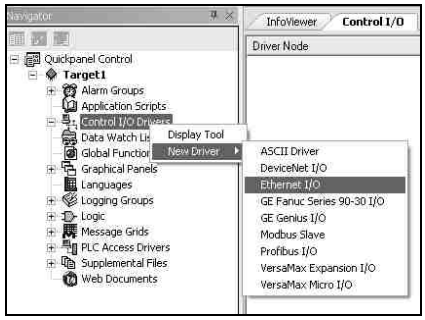
- Command, 10 words
- Discrete out, 2048 bits
- Analog out, 512 words

The controller receives a consumed data structure from each Ethernet NIU. This structure consists of:

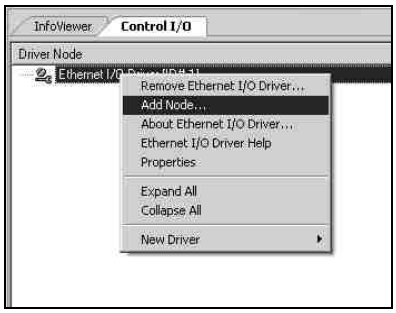
- Status, 10 words
- Discrete input data, as required for each Ethernet NIU
- Analog input data, as required for each Ethernet NIU

Logic is required to drive the command words and respond to the status feedback. Refer to chapter 9 for more information on the data in the command and status words.

Add an Ethernet Global Data driver for each LAN, if it is not already present .

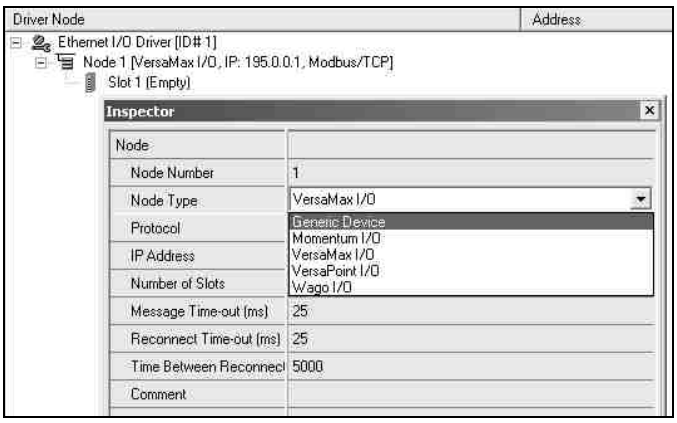


1. Select Ethernet I/O from the New Driver list from Control I/O Drivers.
2. Right-click on Ethernet I/O Driver and select Add Node.

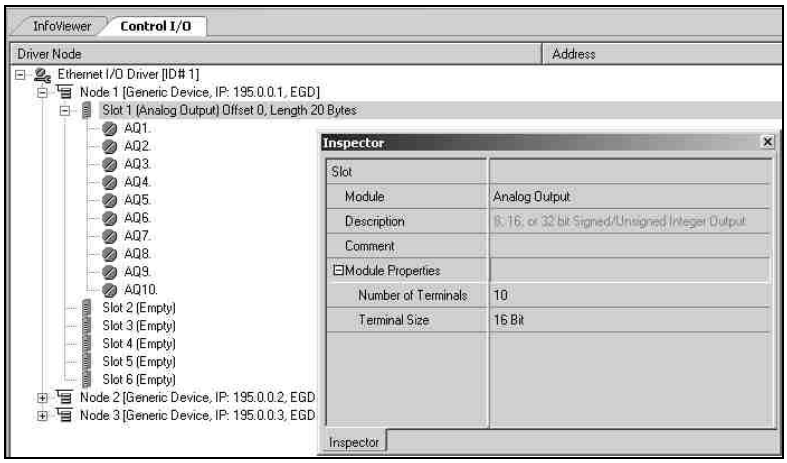


3. Create a node for each Ethernet NIU in the system.

   - Right-click on a Node and select ‘Properties’ to open the Inspector window.
   - Select Generic Device as the Node Type:



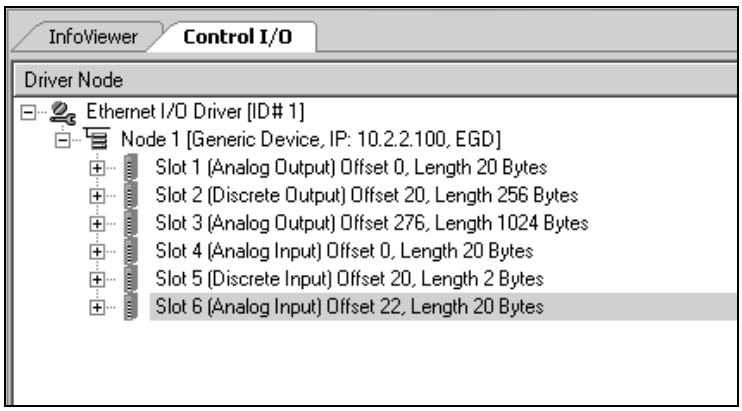
   - Enter the IP Address of the Ethernet NIU.
   - Enter 6 for Number of Slots.
   - Select Slot 1 and open the properties inspector, expand the Module Properties field.
   - Select Analog Output as the module type.
   - Enter 10 for Number of Terminals (this is the 10 words of command data to the Ethernet NIU).



4. Configure the remaining slots, as follows;

   - Slot 2 - Discrete Outputs, 2048 Terminals
   - Slot 3 – Analog Outputs, 512 Terminals
   - Slot 4 – Analog Inputs, 10 Terminals (this is the ten status words from the Ethernet NIU)
   - Slot 5 - Discrete Inputs, enter number of digital inputs from Ethernet NIU
   - Slot 6 – Analog Inputs, enter number of analog inputs from Ethernet NIU

When completed, the configuration screen should be similar to the example below:



As shown above, slots 2, 3, 5, and 6 are for discrete/analog outputs and inputs, slot 1 is the ten control words to the Ethernet NIU, and slot 4 is the ten status words from the Ethernet NIU.

Set Up I/O and Control/Feedback Data

Create the variables to contain the data for the Ethernet Global Data exchanges. Each variable will be an array of the appropriate size. For each Ethernet NIU create the variables described below.

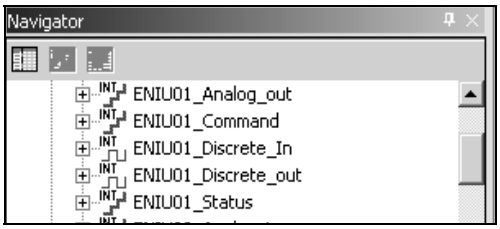
The produced data consists of:

- Command, 10 words
- Discrete out, 2048 bits
- Analog out, 512 words

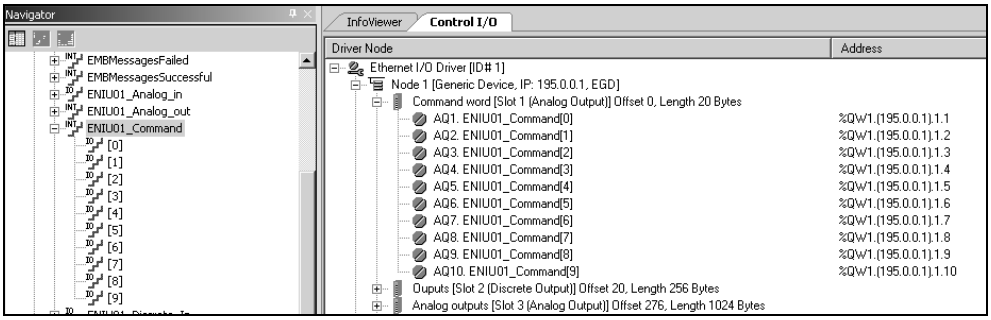
This data is common to all Ethernet NIUs. However, each Ethernet NIU must have its own copy of this data. Provision must be made to aggregate the outputs and then copy all elements to the individual Ethernet NIU variables.

The consumed data consists of:

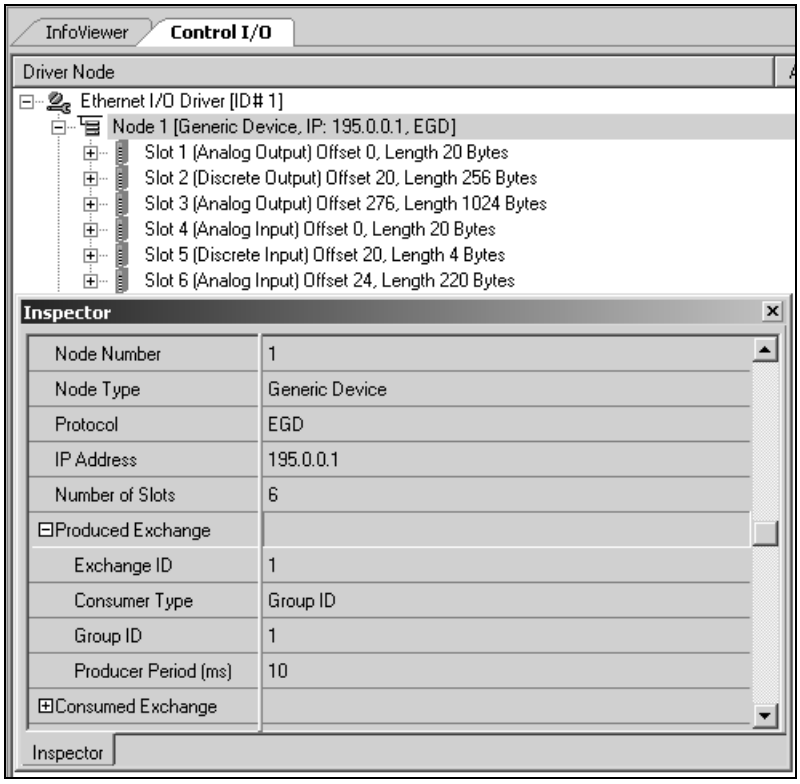
- Status data, 10 words.
- Discrete inputs, as required for each Ethernet NIU.
- Analog inputs, as required for each Ethernet NIU.



1. Open each slot and drag and drop the appropriate variable to each terminal:

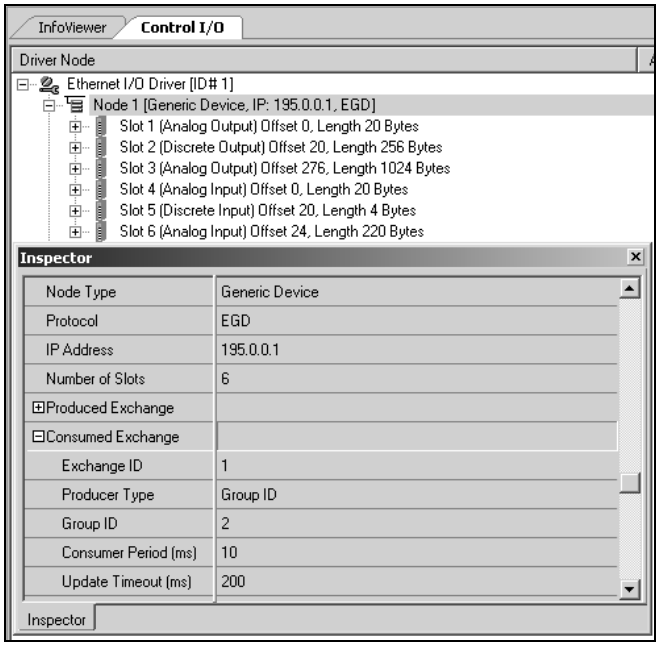


2. Select the node and open the Produced Exchange field in the property inspector window:



- Enter the Exchange ID. This must match the Exchange ID used in the Ethernet NIU's consumed exchange.
- Set the Consumer Type field to Group ID and the Group ID to 1. This must match the value used in the Ethernet NIU's consumed exchange
- Set the Producer Period (see chapter 8 for recommendations on Producer Period).

3. Open up the Consumed Exchange field.



- Enter Exchange ID. This must match the Exchange ID used in the Ethernet NIU produced exchange.
- Set the Producer Type to Group ID and set the Group ID to 2. The Group ID value must match the Destination value in the ENIU produced data exchange.
- Consumer Period (leave at 10ms).
- Update Timeout (see chapter 7 for recommendations on Update Timeout value).
- Repeat for each Ethernet NIU node.

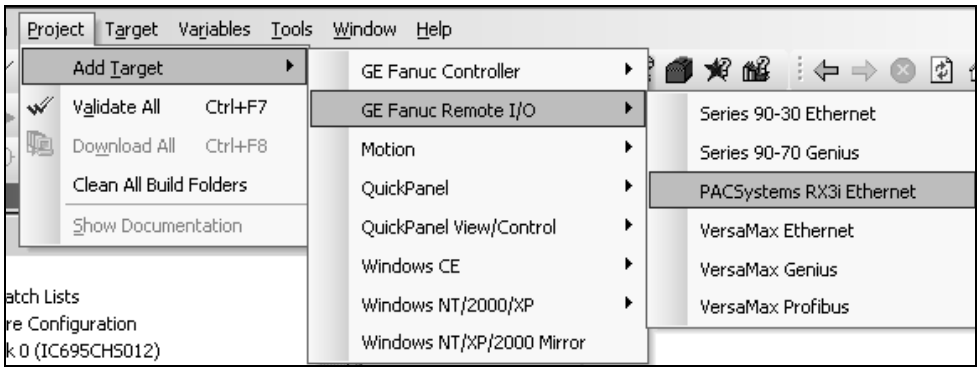
Chapter 6 Hardware Configuration

This chapter explains how to add an Ethernet NIU target to the project, and how to set up the basic hardware configuration of an Ethernet NIU and Ethernet Transmitter Module.

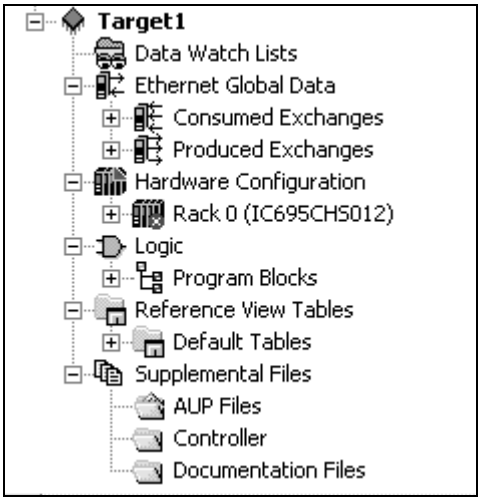
- Adding an Ethernet NIU Target to the Project
  - Completing the Hardware Configuration in the ENIU Target
- Configuring the Ethernet NIU
  - Memory Tab for the Ethernet NIU
  - Faults Tab for the Ethernet NIU
  - Port Tabs for the Ethernet NIU
    - RTU Slave Port Configuration
    - Message Mode Port Configuration
    - SNP Port Configuration
    - Serial I/O Port Configuration
- Configuring the Ethernet Transmitter Module in the I/O Station

Adding an Ethernet NIU Target to the Project

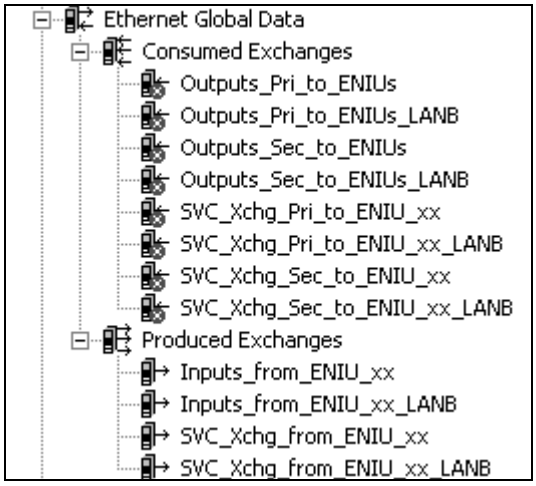
Configuration starts by creating an RX3i Ethernet NIU target in the project. In the programmer, select Project> Add Target> GE Fanuc Remote I/O> PACSystems RX3i Ethernet to add an Ethernet NIU target to the project:



The Ethernet NIU target for Proficy Machine Edition version 5.7 or later and for Proficy Process Systems version 1.0 or later is shown below.

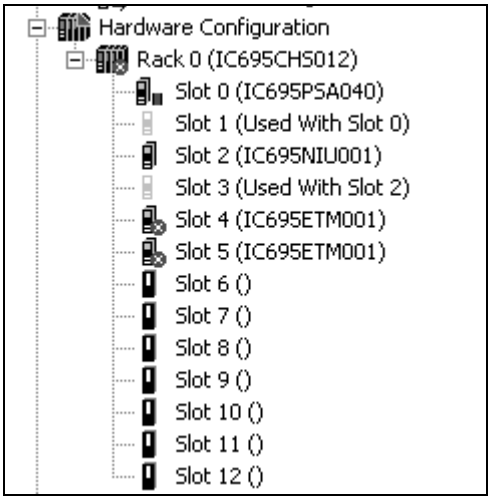


Expanding the Ethernet Global Data component of the target shows the produced and consumed EGD exchanges that are already present:



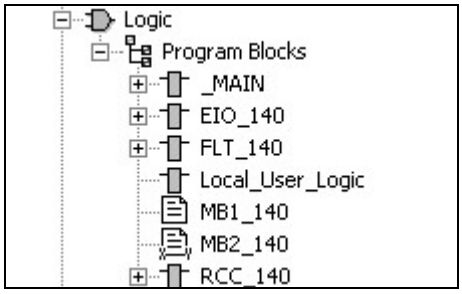
See chapters 3 and 7 for information about these EGD exchanges.

Expanding the Hardware Configuration component of the target shows the modules that are already present:



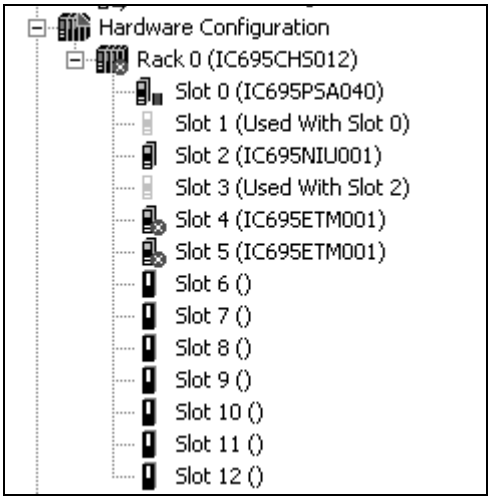
Expanding the Logic component of the target shows the logic blocks that are already present:

Note: Newer versions of the ENIU will show a 140 version or later in the block name.



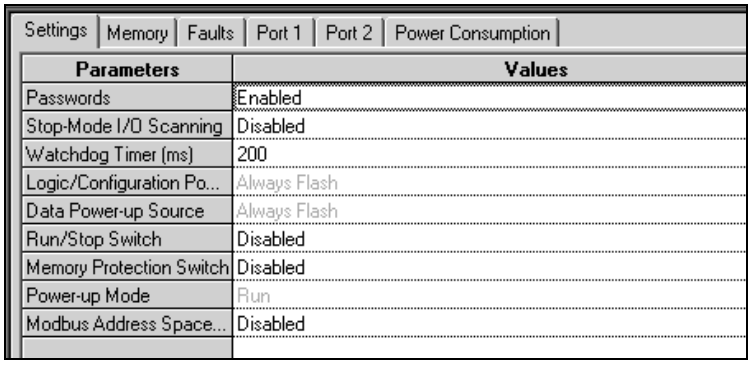
Completing the Hardware Configuration in the ENIU Target

The hardware configuration for the target includes an Ethernet NIU in slot 2, and Ethernet Transmitter Modules in slots 4 and 5. If you are only using one LAN, the Ethernet Transmitter Module in slot 5 should be deleted. Use the additional slots to add the hardware configuration of the individual modules in the Ethernet NIU I/O Station.

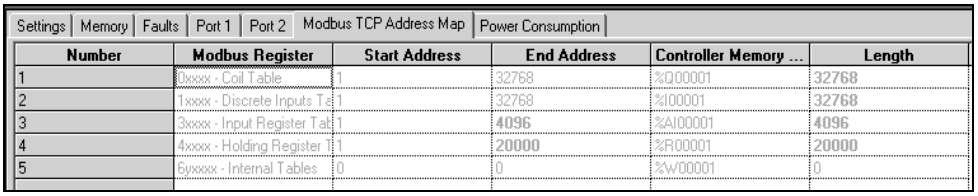


Configuring the Ethernet NIU

Double-click on the Ethernet NIU in the Navigator to configure or view its properties. The Settings tab for the Ethernet NIU is shown below. The default settings should be used as-is.

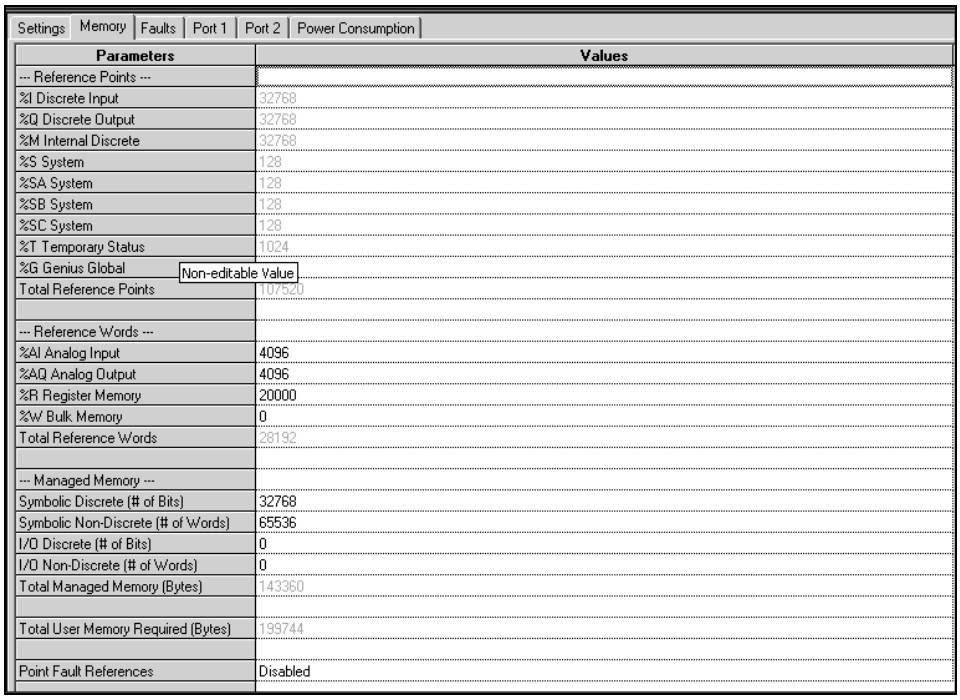


Feature	*Default*	*Choices*	*Description*
Passwords	Enabled	Enabled, Disabled	If passwords are disabled, they cannot be re-enabled without clearing controller memory.
Stop-Mode I/O Scanning	Disabled	Disabled, Enabled	Enables or disables I/O scanning while the NIU is in Stop mode.
Watchdog Timer (ms)	200	10 to 2550ms in 10ms increments	Select a value that is greater than the program sweep time. The NIU restarts the watchdog timer at the beginning of each sweep. The watchdog timer increments during the sweep.
Logic / Configuration Power-up Source	Always Flash	Fixed	The source of the logic and configuration data that will be used at powerup.
Data Power-up Source	Always Flash	Fixed	The source of the reference data that will be used at powerup.
Run/Stop Switch	Disabled	Disabled, Enabled	Enables or disables use of the switch on the NIU to place the NIU in Stop mode, or to go from Stop to Run mode and clear non-fatal faults.
Memory Protection Switch	Disabled	Disabled, Enabled	Enables or disables the memory protect feature of the Run/Stop Switch. If enabled, it prevents writing to program memory and configuration and forcing or overriding discrete data.
Power-up Mode	Run	Fixed	The ENIU always powers up in Run mode.
MODBUS Address Space	Disabled	Disabled, Enabled	If this feature is enabled, a new tab appears that shows standard MODBUS address assignments for the Ethernet NIU's assigned data references. Read-only. See below.



Memory Tab for the Ethernet NIU

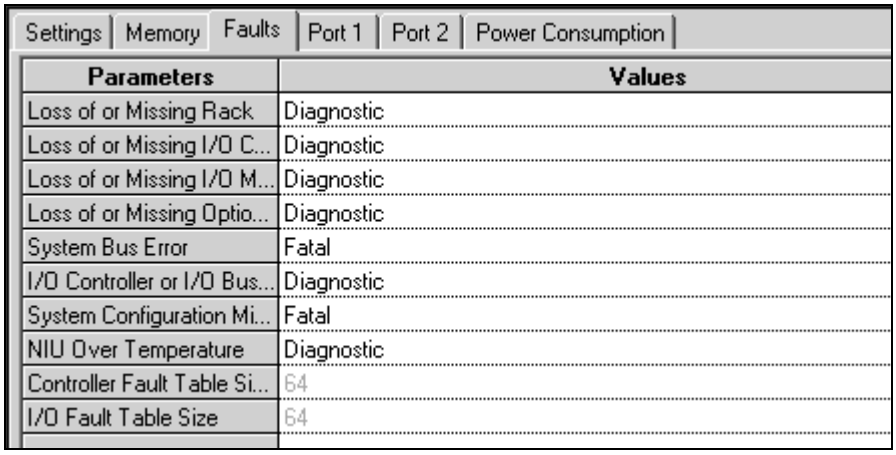
The Memory tab is used to specify the upper limits of each configurable memory area. The upper limits of non-configurable memory areas are also displayed. The default Memory Allocations for the Ethernet NIU are shown below



Parameters	Values
--- Reference Points ---	
%I Discrete Input	32768
%Q Discrete Output	32768
%M Internal Discrete	32768
%S System	128
%SA System	128
%SB System	128
%SC System	128
%T Temporary Status	1024
%G Genius Global	
Total Reference Points	107520
--- Reference Words ---	
%AI Analog Input	4096
%AQ Analog Output	4096
%R Register Memory	20000
%W Bulk Memory	0
Total Reference Words	28192
--- Managed Memory ---	
Symbolic Discrete (# of Bits)	32768
Symbolic Non-Discrete (# of Words)	65536
I/O Discrete (# of Bits)	0
I/O Non-Discrete (# of Words)	0
Total Managed Memory (Bytes)	143360
Total User Memory Required (Bytes)	199744
Point Fault References	Disabled

Faults Tab for the Ethernet NIU

The Faults tab for the Ethernet NIU is shown below. The defaults shown can be edited as appropriate for the application.

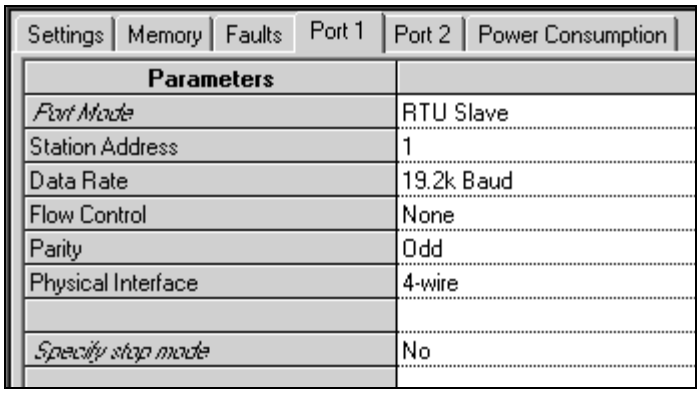


Parameters	Values
Loss of or Missing Rack	Diagnostic
Loss of or Missing I/O C...	Diagnostic
Loss of or Missing I/O M...	Diagnostic
Loss of or Missing Optio...	Diagnostic
System Bus Error	Fatal
I/O Controller or I/O Bus...	Diagnostic
System Configuration Mi...	Fatal
NIU Over Temperature	Diagnostic
Controller Fault Table Si...	64
I/O Fault Table Size	64

All faults set a diagnostic variable and are logged in a fault table. A fatal fault stops the Ethernet NIU.

Port Tabs for the Ethernet NIU

Ports 1 and 2 can be configured for RTU Slave, Message Mode, SNP Slave, or Serial I/O communications. Set up the communications parameters to match those of a master device connected to the port.



When changing the Port Mode, set the Port Mode parameter first; the communications parameters are different for each protocol type.

RTU Slave: The Port Mode defaults to RTU Slave. Note that an SNP Master such as Logic Developer or Winloader can communicate on a port even though it is configured as an RTU Slave.

Message Mode: sets up the port so that C blocks can perform serial I/O operations using the C Runtime Library functions.

SNP Slave: reserves the port for use by an SNP Master such as Logic Developer or Winloader.

Serial I/O: sets up the port to communicate using COMMREQs in the local logic.

RTU Slave Port Configuration

RTU Slave (the default) sets up the port for MODBUS RTU Slave protocol. The parameters of RTU slave are shown above.

Station Address: specify a Station Address from 1 to 247. Do not use the station address of 1 for any other device in the control system. The NIU will use the station address of 1 when powered up with no configuration.

Data Rate: 1200 Baud, 2400 Baud, 4800 Baud, 9600 Baud, 19.2k Baud, 38.4k Baud, 57.6k Baud, 115.2k Baud.

Flow Control: The type of flow control to be used on the port: none or hardware. The hardware flow-control is RTS/CTS crossed.

Parity: The parity for serial communications (odd or even).

Physical Interface: The choices are:

- 2-wire: single path for both receiving and transmitting communications. The receiver is disabled while transmitting.
- 4-wire: separate paths for receiving and transmitting communications. The transmit line is driven only while transmitting.
- 4-wire transmitter on (port 2 only): separate paths for receiving and transmitting communications. The transmit line is driven continuously.

Specify Stop Mode: To specify the protocol to be used in Stop mode, set Specify Stop Mode to yes. For the Stop mode, select either RTU Slave or SNP Slave, and configure the communications parameters.

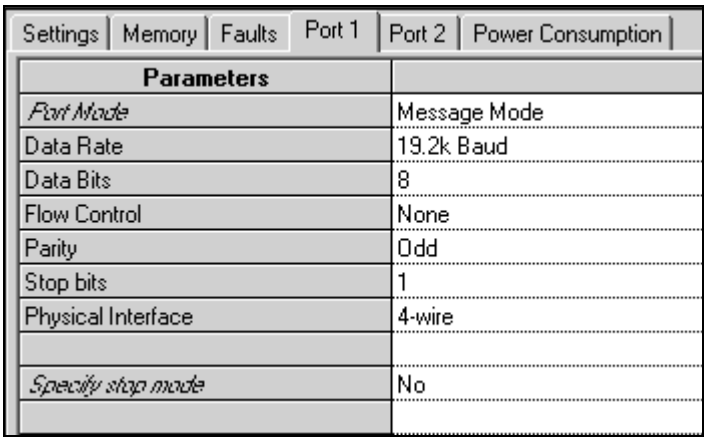
Specify stop mode	Yes
Stop Mode	SNP Slave
Turn Around Delay Time (ms)	0
Timeout (s)	10
SNP ID	

Because the port is already configured for RTU slave, it will use the same communications parameters if the Stop mode communications protocol is also RTU slave.

If the Stop mode communications protocol is SNP slave, configure the communications parameters:

- Turnaround Delay: the minimum time interval required between reception of a message and the next transmission. In 2-wire mode, this interval is required for switching the direction of data transmission on the communication line. The range is 0 to 2550ms in 10ms increments.
- Timeout: The maximum time the port will wait to receive a message from the master. If a message is not received within this interval, the port assumes that communications have been disrupted. It then waits for a new Attach message from the master.
- SNP ID: The port ID to be used for SNP communications. In SNP multi-drop communications, this ID is used to identify the intended receiver of a message. This parameter can be left blank if communication is point to point. The SNP ID can be up to seven alphanumeric characters (A through Z, 0 through 9) or the underline (_) character.

Message Mode Port Configuration



Configure the parameters of Message Mode protocol.

Data Rate: 1200 Baud, 2400 Baud, 4800 Baud, 9600 Baud, 19.2k Baud, 38.4k Baud, 57.6k Baud, 115.2k Baud.

Data Bits: The number of bits in a word for serial communication (7or 8).

Flow Control: The type of flow control to be used on the port: none or hardware. The hardware flow-control is RTS/CTS crossed.

Parity: The parity for serial communications (odd or even).

Stop Bits: The number of stop bits for serial communications (1 or 2).

Physical Interface: The choices are:

- 2-wire: single path for both receiving and transmitting communications. The receiver is disabled while transmitting.
- 4-wire: separate paths for receiving and transmitting communications. The transmit line is driven only while transmitting.
- 4-wire transmitter on (port 2 only): separate paths for receiving and transmitting communications. The transmit line is driven continuously.

Specify Stop Mode: To specify the protocol to be used in Stop mode, set Specify Stop Mode to yes. For the Stop mode, select either RTU Slave or SNP Slave, and configure the communications parameters.

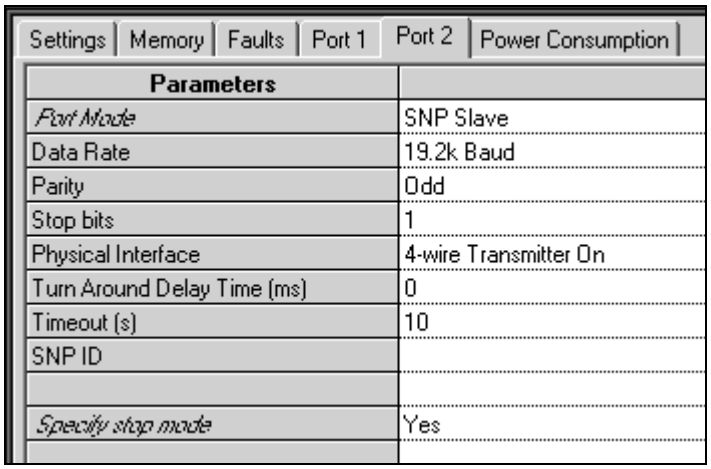
Specify stop mode	Yes
Stop Mode	SNP Slave
Turn Around Delay Time (ms)	0
Timeout (s)	10
SNP ID	

For RTU Slave: specify a Station Address from 1 to 247. Do not use the station address of 1 for any other device in the control system. The Ethernet NIU uses the station address of 1 when powered up with no configuration, and when the Port Mode parameter is set to Message Mode with MODBUS as the stop mode protocol.

For SNP Slave: configure the Turnaround Delay Time in milliseconds, Timeout, and the SNP ID.

- Turnaround Delay: the minimum time interval required between reception of a message and the next transmission. In 2-wire mode, this interval is required for switching the direction of data transmission on the communication line. The range is 0 to 2550ms in 10ms increments,
- Timeout: The maximum time the port will wait to receive a message from the master. If a message is not received within this interval, the port assumes that communications have been disrupted. It then waits for a new Attach message from the master,
- SNP ID: The port ID to be used for SNP communications. In SNP multi-drop communications, this ID is used to identify the intended receiver of a message. This parameter can be left blank if communication is point to point. The SNP ID can be up to seven alphanumeric characters (A through Z, 0 through 9) or the underline (_).

SNP Slave Port Configuration



Configure the parameters of SNP Slave protocol.

Data Rate: 1200 Baud, 2400 Baud, 4800 Baud, 9600 Baud, 19.2k Baud, 38.4k Baud, 57.6k Baud, 115.2k Baud.

Parity: The parity for serial communications (odd or even).

Stop Bits: The number of stop bits for serial communications (1 or 2). SNP uses one stop bit.

Physical Interface: The choices are:

- 2-wire: single path for both receiving and transmitting communications. The receiver is disabled while transmitting.
- 4-wire: separate paths for receiving and transmitting communications. The transmit line is driven only while transmitting.
- 4-wire transmitter on (port 2 only): separate paths for receiving and transmitting communications. The transmit line is driven continuously. This choice is inappropriate for SNP multi-drop communications, because only one device on the multi-drop line can be transmitting at a given time.

Turnaround Delay: the minimum time interval required between reception of a message and the next transmission. In 2-wire mode, this interval is required for switching the direction of data transmission on the communication line. The range is 0 to 2550ms in 10ms increments.

Timeout: The maximum time (0 to 60 seconds) the port will wait to receive a message from the master. If a message is not received within this interval, the port assumes communications have been disrupted, and waits for a new Attach message from the master.

SNP ID: The port ID to be used for SNP communications. In SNP multi-drop communications, this ID is used to identify the intended receiver of a message. This parameter can be left blank if communication is point-to-point. The SNP ID can be up to seven alphanumeric characters (A through Z, 0 through 9) or the underline (_).

Specify Stop Mode: To specify a protocol to be used in Stop mode, set Specify Stop Mode to yes

Specify stop mode	Yes
Stop Mode	SNP Slave
Turn Around Delay Time (ms)	0
Timeout (s)	10
SNP ID	

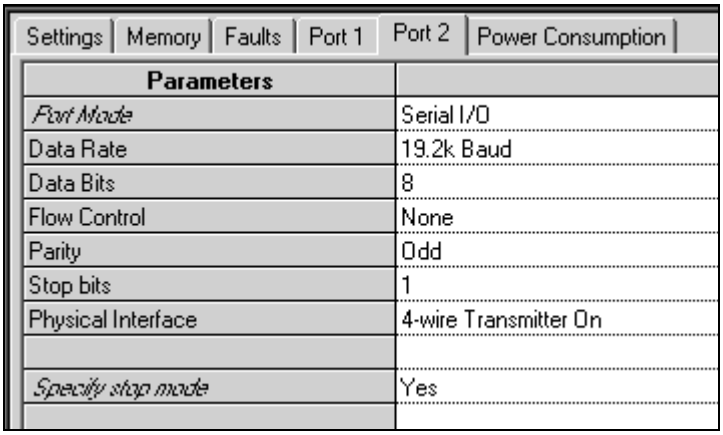
Because the port is already configured for SNP slave, it will use the same communications parameters in Stop mode.

Serial I/O Port Configuration

Serial I/O protocol is active only in Run mode. If the NIU can be set to Stop mode, it will switch to the configured Stop Mode protocol (see below). As long as the NIU can be stopped, the protocol can be auto-switched to one that enables serial protocol connection.

Caution

If neither port is configured for either RTU Slave or SNP Slave, DO NOT DISABLE the Run/Stop switch unless an alternate way is provided to take control of or stop the module.



Configure the parameters for Serial I/O protocol.

Data Rate: 1200 Baud, 2400 Baud, 4800 Baud, 9600 Baud, 19.2k Baud, 38.4k Baud, 57.6k Baud, 115.2k Baud.

Data Bits: The number of bits in a word for serial communication (7or 8).

Flow Control: The type of flow control to be used on the port: none, hardware, or software (XON/XOFF). The hardware flow-control is RTS/CTS crossed.

Parity: The parity for serial communications (odd or even).

Stop Bits: The number of stop bits for serial communications (1 or 2).

Physical Interface: The choices are:

- 2-wire: single path for both receiving and transmitting communications. The receiver is disabled while transmitting.
- 4-wire: separate paths for receiving and transmitting communications. The transmit line is driven only while transmitting.
- 4-wire transmitter on (port 2 only): separate paths for receiving and transmitting communications. The transmit line is driven continuously.

Specify Stop Mode: To specify the protocol to be used in Stop mode, set Specify Stop Mode to yes. For the Stop mode, select either RTU Slave or SNP Slave, and configure the communications parameters.

Specify stop mode	Yes
Stop Mode	SNP Slave
Turn Around Delay Time (ms)	0
Timeout (s)	10
SNP ID	

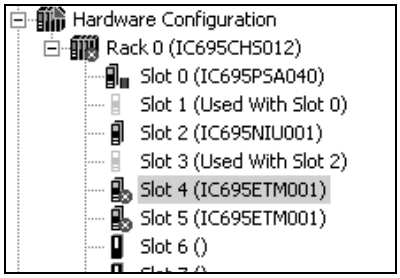
For RTU Slave: specify a Station Address from 1 to 247. Do not use the station address of 1 for any other device in the control system. The NIU uses the station address of 1 when powered up with no configuration.

For SNP Slave: configure the Turnaround Delay Time in milliseconds, Timeout, and the SNP ID.

- Turnaround Delay: the minimum time interval required between reception of a message and the next transmission. In 2-wire mode, this interval is required for switching the direction of data transmission on the communication line. The range is 0 to 2550ms in 10ms increments,
- Timeout: The maximum time the port will wait to receive a message from the master. If a message is not received within this interval, the port assumes that communications have been disrupted. The port then waits for a new Attach message from the master,
- SNP ID: The port ID to be used for SNP communications. In SNP multi-drop communications, this ID is used to identify the intended receiver of a message. This parameter can be left blank if communication is point to point. The SNP ID can be up to seven alphanumeric characters (A through Z, 0 through 9) or the underline (_).

Configuring the Ethernet Transmitter Module in the I/O Station

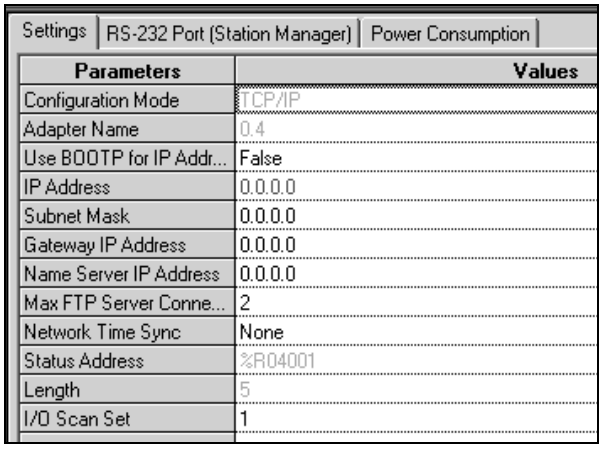
The hardware configuration for the Ethernet NIU target includes Ethernet Transmitter Modules in slots 4 and 5.



Double-click on an Ethernet Transmitter Module in the Navigator to configure or view its properties.

Settings Tab for the Ethernet Transmitter Module

Default settings for an Ethernet Transmitter Module are shown below. These can be edited as appropriate for the application.



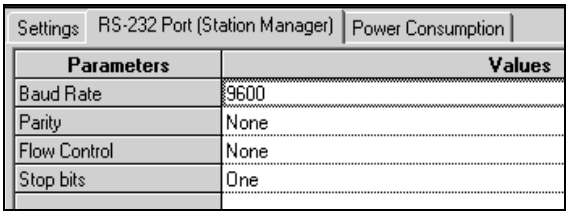
In the default configuration the Ethernet Transmitter Module is in slot 4 of the I/O Station.

Feature	*Description*	*Configuration 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 Ethernet NIU.

The Ethernet Transmitter Module is linked to the EGD exchanges for communications. If the module needs to be moved to another slot in the hardware configuration, either copy and paste or drag and drop it to the new slot. This will automatically adjust the adapter name property (which is the rack.slot location of the Ethernet module, for example: 0.4) in all the EGD exchanges.

RS-232 Port Tab for the Ethernet Transmitter Module

Default communications parameters for an Ethernet Transmitter Module Station Manager port are shown below. These can be edited as appropriate for the application.



Chapter 7 Creating an Application without the Templates

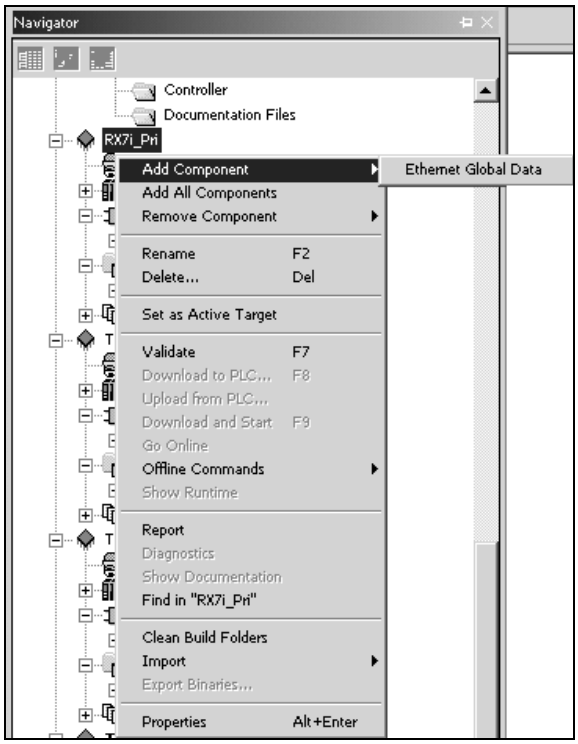
This chapter explains how to configure Ethernet Global Data exchanges **when the application templates are not used** (see chapter 3). The information in this chapter can also be used to add Ethernet NIUs to existing system configurations.

- Controller Configuration for Ethernet NIUs
  - Ethernet Interface Configuration in the Controller
  - Configuring the EGD Exchanges in the Controller Target
  - Configuring the Input Data Areas of the Input from ENIU Exchanges
  - Configuring the Number of Ethernet NIUs in the Controller Application
- Configure Ethernet Global Date in the Ethernet NIU Target
- Setting Up EGD Exchanges
  - Viewing and Editing EGD Exchange Properties
  - Viewing and Editing the EGD Exchange Table
  - Configuring EGD Outputs from the Controller to Ethernet NIUs
  - Single Controller Single LAN Exchanges
    - Properties and Data Ranges for *Inputs_from_ENIU_xx*
    - Properties and Data Ranges for *Outputs_Pri_to_ENIUs*
    - Properties and Data Ranges for *SVC_Xchg_Pri_to_ENIU_xx*
    - Properties and Data Ranges for *SVC_Xchg_from_ENIU_xx*
  - Added exchanges for Two Controllers
    - Properties *and Data Ranges for Outputs_Sec_to_ENIUs*
    - *Properties* and Data Ranges for *SVC_Xchg_Sec_to_ENIU_xx*
  - Added exchanges for Two Controller Dual LAN
    - Properties and Data Ranges for *Inputs_from_ENIU_xx_LANB*
    - *Properties and Data Ranges for Outputs_Pri_to_ENIUs_LANB*
    - *Properties and Data Ranges for Outputs_Sec_to_ENIUs_LANB*
    - *Properties and Data Ranges for SVC_Xchg_Pri_to_ENIU_xx_LANB*
    - *Properties and Data Ranges for SVC_Xchg_Sec_to_ENIU_xx_LANB*
    - Properties and Data Ranges for *SVC_Xchg_from_ENIU_xx_LANB*

Controller Configuration for Ethernet NIUs

If the system is not created using one of the templates, 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 controller 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 Ethernet Global Data exchanges must be created and defined. Configuring the exchanges assigns the data in the exchange to controller memory references. 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 Ethernet Global Data exchanges must also be created and configured. Exchanges for the secondary controller 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.

Ethernet Interface Configuration in the Controller

Enter the Ethernet IP Address, the Subnet Mask and (if needed) the Gateway IP Address for the Ethernet module in the controller

Feature	Description	Configuration 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.

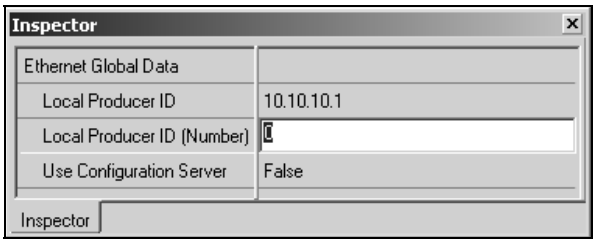
The Ethernet module is linked to the Ethernet Global Data exchanges for communications. If the module needs to be moved to another slot in the hardware configuration, either copy and paste or drag and drop it to the new slot. This will automatically adjust the adapter name property (which is the rack.slot location of the Ethernet Transmitter Module, for example: 0.4) in all the EGD exchanges.

Redundant IP should not be used for the Ethernet interfaces that connect to Ethernet NIUs.

For CRE or CRU systems, the produced EGD exchanges in the controller should be set for Produce in Backup Mode.

Controller Ethernet Global Data Local Producer ID

The controller needs to have an Ethernet Global Data Local Producer ID. In the Navigator window, right click on the Ethernet Global Data item to bring up the Property Inspector.



If the Local Producer ID is 0.0.0.0 or if it is an incorrect value, enter a value in the Local Producer ID.

For all controllers except PACSystems Redundancy controllers, the primary and secondary controllers must have unique Local Producer IDs. PACSystems Redundancy controllers in redundant mode are considered a single system and share a common Local Producer ID.

Configuring the EGD Exchanges in the Controller Target

Ethernet Global Data Exchanges need to be added to the controller target for each Ethernet NIU in the project. Follow the instructions above for adding the EGD Exchanges *Inputs_from_ENIU_xx, SVC_Xchg_from_ENIU_xx, SVC_Xchg_to_ENIU_xx* (and the _LANB Exchanges if this is a Dual Lan System). No changes are required for the exchanges *Outputs_to_ENIUs* and *Outputs_to_ENIUs_LANB*.

Configuring the Input Data Areas of the Input from ENIU Exchanges

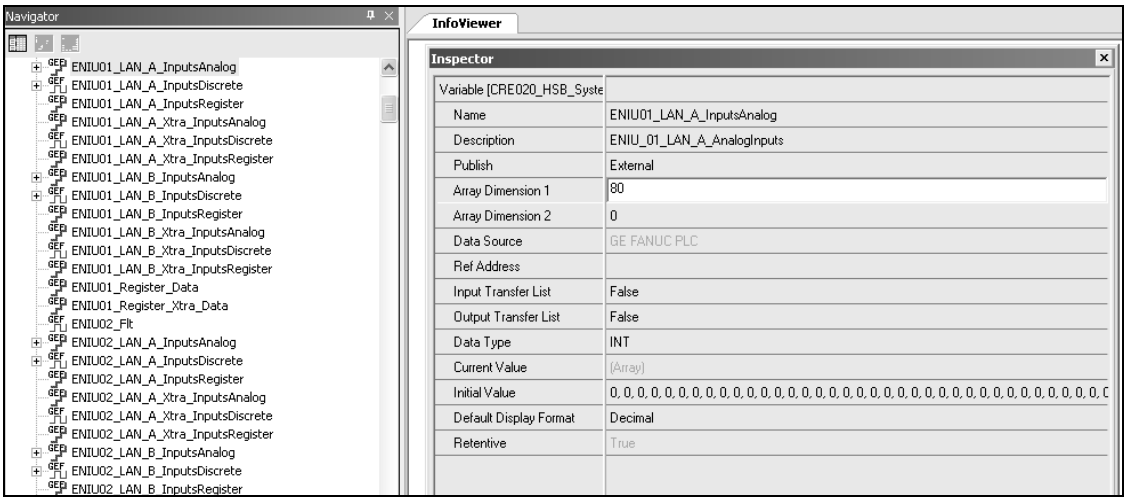
In dual LAN systems, the input data areas for the discrete, analog and word data from the additional Ethernet NIU(s) must be configured to match the data areas used in the Ethernet NIU. After the sizes of the inputs in %I, %AI and words have been configured for the Ethernet NIUs, these values need to be entered as the array dimension 1 values of these variables:

ENIUxx_LAN_A_InputsAnalog, ENIUxx_LAN_B_InputsAnalog,

ENIUxx_LAN_A_InputsDiscrete, ENIUxx_LAN_B_InputsDiscrete,

ENIUxx_LAN_A_InputsRegister, and *ENIUxx_LAN_B_InputsRegister*

If any of the data types are NOT used, leave the Array Dimension 1 value(s) of the appropriate variable set at 0. The example below shows the size of *ENIU01_LAN_A_InputsAnalog* set at 80 INTs (words). For Release 1.40 projects, this variable would be set to 128 INTs.



Configuring the Number of Ethernet NIUs in the Controller Application

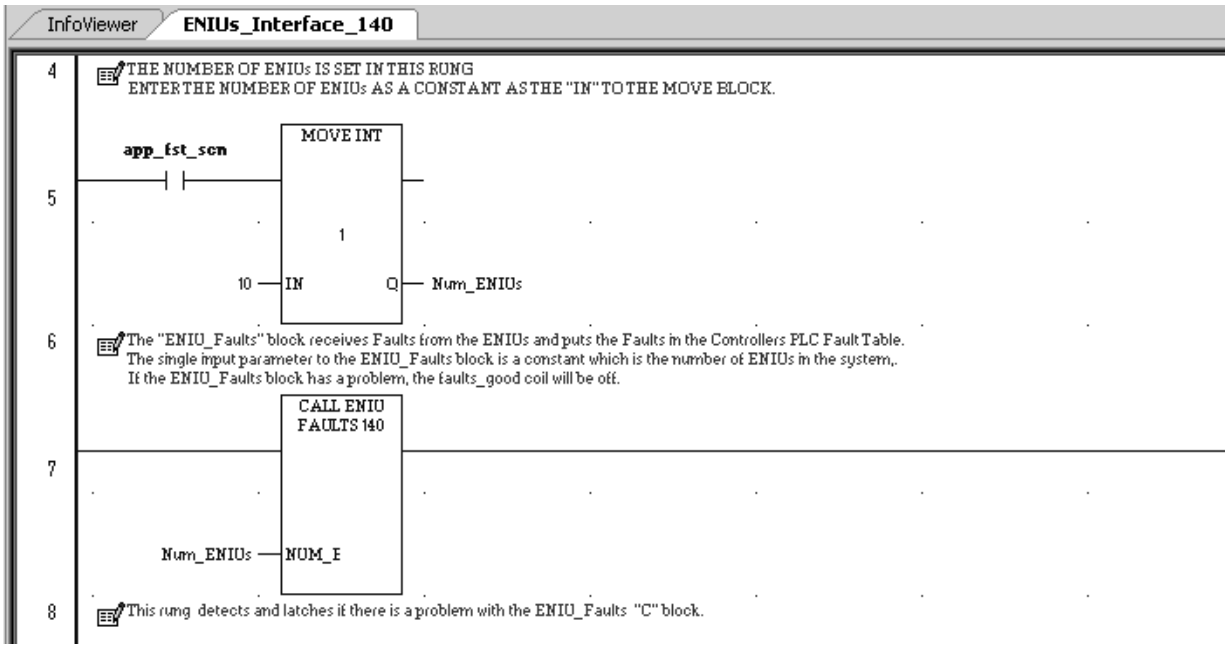
In the controller ladder application, the number of Ethernet NIUs must be changed to match the number of Ethernet NIUs in the system.

In version 140 and later the Number of ENIUs is set in Rung 5, which moves a constant into the variable *Num_ENIUs*. The variable *Num_ENIUs* is used in the calls to *ENIU_Faults* and *Input_Processing*.

In Version 133 and earlier, the number of Ethernet NIUs appears as the Input parameter *NUM_ENIUs* for the Call of ladder block *ENIUxx_Fault_yyy* in rung 2 of ladder block *ENIUs_Interface_yyy*.

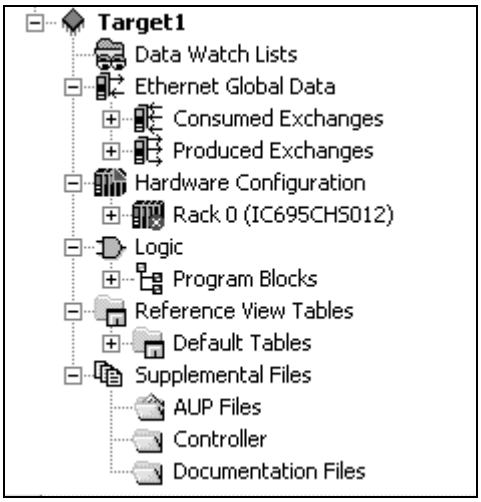
For dual LAN applications, the number of Ethernet NIUs must also be entered as the Input Parameter *END_ENIU* for the Call of ladder block *Input_Arbitration_yyy* (ladder block Input_Processing_yyy in PPS Applications) in rung 6 of ladder block *ENIUs_Interface_yyy*.

The example logic below shows the number of Ethernet NIUs set to 10 for a dual LAN application:

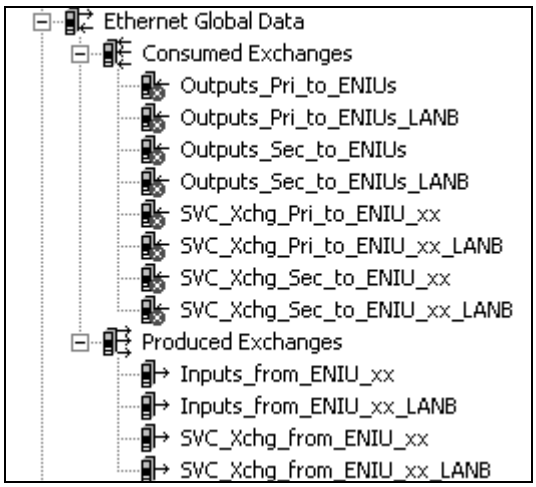


Configure Ethernet Global Data in the Ethernet NIU Target

The Ethernet NIU target for Proficy Machine Edition version 5.7 or later and for Proficy Process Systems version 1.0 or later is shown below.



Expanding the Ethernet Global Data component of the Ethernet NIU target shows the produced and consumed EGD exchanges that are already present:



The Ethernet NIU target includes the Ethernet Global Data exchanges described below, for single controller, dual controller, single LAN, and dual LAN systems.

Ethernet Global Data Exchange Descriptions

The exchanges in the Ethernet NIU target are described below. Exchanges that will be used for the application are set up as described later in this chapter. Exchanges that are not needed for the application can be deleted.

Inputs_from_ENIU_xx transfers inputs from the Ethernet NIU to the controllers.

- Produced by: the Ethernet NIU
- Consumed by: both controllers

Outputs_Pri_to_ENIUs transfers outputs from the primary controller to the Ethernet NIUs.

- Produced (multicast) by: the primary controller
- Consumed by: all Ethernet NIUs

Outputs_Sec_to_ENIU transfers outputs from the secondary controller to the Ethernet NIUs.

- Produced (multicast) by: the secondary controller
- Consumed by: all Ethernet NIUs

SVC_Xchg_from_ENIU_xx transfers faults from an Ethernet NIU to the controllers. In addition, if the Ethernet NIU receives a Remote COMMREQ Call request from the controller, it responds to the Remote COMMREQ Call request in this exchange.

- Produced by: the Ethernet NIU
- Consumed by: both controllers

SVC_Xchg_Pri_to_ENIU_xx has three functions; it acknowledges that the controller has received fault data from an Ethernet NIU, it allows clearing of faults in an Ethernet NIU, and it transfers Remote COMMREQ Call requests from the controller to the Ethernet NIU.

- Produced by: the primary controller
- Consumed by: the Ethernet NIU

SVC_Xchg_Sec_to_ENIU_xx has three functions; it acknowledges that the controller has received fault data from the Ethernet NIU, it allows clearing of faults in the Ethernet NIU, and it transfers Remote COMMREQ Call requests from the controller to the Ethernet NIU.

- Produced by: the secondary controller
- Consumed by: the Ethernet NIU

Inputs_from_ENIU_xx_LANB transfers inputs from the Ethernet NIU to the controllers. In this exchange, xx is the Ethernet NIU number on LAN B.

- Produced by: the Ethernet NIU
- Consumed by: both controllers

Outputs_Pri_to_ENIUs_LANB transfers outputs from the primary controller to the Ethernet NIUs on LAN B.

- Produced (multicast) by: the primary controller
- Consumed by: all Ethernet NIUs

Outputs_Sec_to_ENIUs_LANB transfers outputs from the secondary controller to the Ethernet NIUs on LAN B.

- Produced (multicast) by: the secondary controller
- Consumed by: all Ethernet NIUs

SVC_Xchg_from_ENIU_xx_LANB has two functions; it transfers faults from the Ethernet NIU to the controllers and when the Ethernet NIU receives Remote COMMREQ Calls from the controller, the Ethernet NIU returns the response in this exchange. xx is the ENIU number on LAN B.

- Produced by: the Ethernet NIU
- Consumed by: both controllers

SVC_Xchg_Pri_to_ENIU_xx_LANB has three functions; it acknowledges that the controller has received fault data from the Ethernet NIU, it allows clearing of faults in an Ethernet NIU, and it transfers a Remote COMMREQ Call request from the controller to the Ethernet NIU. xx is the ENIU number on LAN B.

- Produced by: the primary controller
- Consumed by: the Ethernet NIU

SVC_Xchg_Sec_to_ENIU_xx_LANB has three functions; it acknowledges that the controller has received fault data from the Ethernet NIU, it allows clearing of faults in the Ethernet NIU, and it transfers a Remote COMMREQ Call request from the controller to the Ethernet NIU. xx is the ENIU number on LAN B.

- Produced by: the secondary controller
- Consumed by: the Ethernet NIU

Setting Up Ethernet Global Data Exchanges

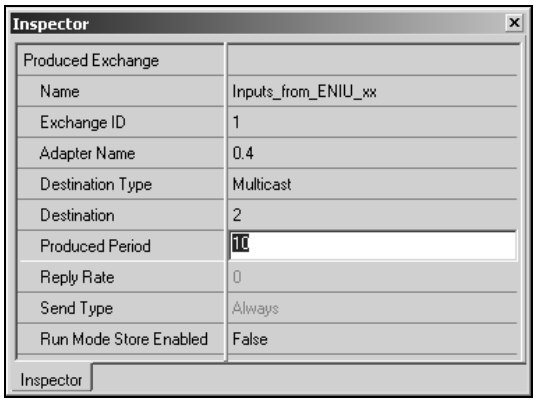
This section explains how to view and edit the properties of Ethernet Global Data exchanges, and how to view and edit the data tables of EGD memory assignments.

Viewing and Editing EGD Exchange Properties

In the Navigator, right-click on an exchange to view or change its properties. Click on a field to edit. For all exchanges:

- Group ID in the consumed exchange must match the Destination in the corresponding produced exchange.
- The Exchange ID must match in the corresponding produced and consumed exchanges.

The Update Timeout in the consumed exchange must be approximately 2ms more than 3 times the Produced Period in the corresponding produced exchange.



Sets of tables in this chapter show the EGD exchange properties in the controller and Ethernet NIUs side-by-side. The Ethernet NIU configuration is on the left and the controller configuration is on the right.

The name of each exchange can be different in the producer and consumer, but using the same name at both ends makes understanding and troubleshooting much easier.

ENIU Produced Exchange ***Example_Exchange_1***	
Name	**Example_Exchange_1**
Exchange ID	Number from 1 to 16000
Adapter Name	0.x x = slot of ETM
Destination Type	Always Multicast
Destination	Number from 1 to 32
Produced Period	Number in milliseconds
Reply Rate	Fixed not editable
Send Type	Fixed not editable
Run ModeStore	False

Controller Consumed Exchange ***Example_Exchange_1***	
Name	**Example_Exchange_1**
Producer ID	a.b.c.d
Group ID	Number from 1 to 32
Exchange ID	Number from 1 to 16000
Adapter Name	Rack.slot of ETM (0.x)
Consumed Period	Fixed not editable
Reply Rate	Fixed not editable
Update Timeout	Number in milliseconds
Run Mode Store	False

Run Mode Store of Ethernet Global Data to the Ethernet NIU

Logic Developer version 5.8 and later includes a Run Mode Store property for Ethernet Global Data exchanges, as shown in the illustration of the Inspector window on the previous page. By default, the Run Mode Store property is set to False. It should NOT be changed for most Ethernet NIU applications. Run Mode Store for Ethernet Global Data is not supported for redundancy systems.

It is always possible to add new Ethernet Global Data exchanges in a Run Mode Store without setting the Run Mode Store property to True. Also, programmer-generated Ethernet Global Data exchanges in PPS applications can always be modified in a Run Mode Store, whether or not Run Mode Store is enabled in the exchange configuration.

Caution

The Ethernet NIU uses Ethernet Global Data to exchange input and output data for the I/O modules in the station with one or more CPUs. Modifying an Ethernet Global Data exchange in the Ethernet NIU using Run Mode Store can cause a bump in data transfer, or possibly take the exchange offline. It is important to understand the risk of data loss when making a Run Mode Store change to Ethernet Global Data.

After careful consideration of the consequences, individual exchanges can be configured to allow modification or deletion in a Run Mode Store. Use of this feature requires an Ethernet NIU with revision 5.50 or later NIU firmware and revision 5.50 or later Ethernet firmware. The Ethernet Transmitter Module in the I/O Station must also support Run Mode Store of Ethernet Global Data (firmware revision 5.50 or later is required).

Effect of a Run Mode Store on EGD Operation

Changes to the Ethernet Global Data in the Ethernet NIU must correspond to changes made in the CPU(s). It is important to be aware that the Ethernet NIU and CPU(s) will NOT receive and apply new EGD configurations simultaneously.

Performing a Run Mode Store does not stop the production or consumption of existing exchanges that have not been modified.

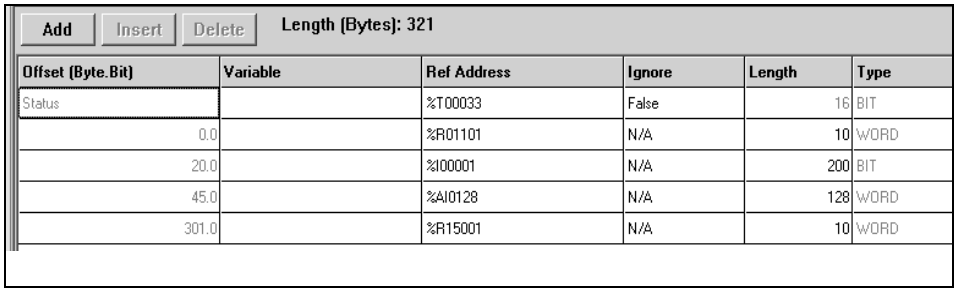
Changes implemented with a Run Mode Store of Ethernet Global Data take effect while the Ethernet NIU is running - no stop is required.

Added exchanges will start consumption or production shortly after the activation of any logic that is part of the Run Mode Store sequence.

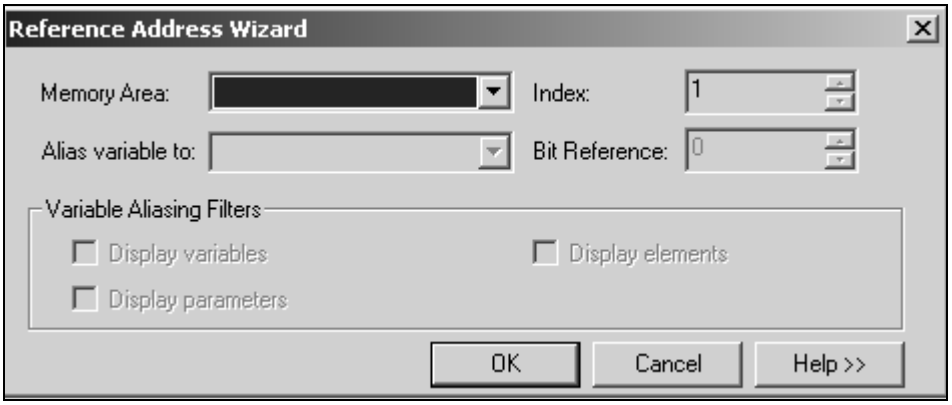
The Ethernet NIU does not use EGD signatures (if EGD signatures are enabled in the controller, they are ignored by the Ethernet NIU). After a Run Mode Store, the Ethernet NIU can only determine the compatibility of a modified Ethernet Global Data exchange that it consumes by the checking the length of the exchange. If the length does not match, the Ethernet NIU stops consuming the exchange. A store to update the corresponding producer/consumer is needed to resume consumption of the exchange.

Viewing and Editing the EGD Exchange Data Range Table

Double-click on an EGD exchange in the Navigator to view its reference assignments:



Double-click on a field to edit:



For all exchanges, each row in the Data Range Table must have the same data length in the Ethernet NIU and controller.

ENIU Produced Exchange Example_Exchange_1						***Controller Consumed Exchange Example_Exchange_1***					
Offset	Ref Address	Ignore	Len	Type	Desc	Variable	Ref-Addr	Ignore	Len	Type	Desc
Status	Must be a specific %T	False	16	Bit	EGD Status	* See below		False	16 or 1	Bit Word	EGD Status
						Time Stamp	Not Used	False	0	Byte	
	Must be a specific %R	N/A	10	Word	ENIU Status	* See below	<Sym>	False	10	Word	ENIU Status
	** %I0001	N/A	200	Bit	1st data range	* See below	<Sym>	False	200	Bool	1st data range
	** %AI001	N/A	80	Word	2nd data range	* See below	<Sym>	False	128	Int	2nd data range

*If using GE Fanuc C blocks in the controller, must be a specific symbolic variable for consumed exchanges. Otherwise it will be a reference address in Ref-Addr column.
** Input (%I and %AI) addresses used for the ENIU will be per the application.

Configuring EGD Outputs from the Controller to Ethernet NIUs

The exchange *Outputs_Pri_to_ENIUs* will be used to send outputs from the primary controller to the Ethernet NIUs. Corresponding exchanges must be set up in the controller and in the Ethernet NIU. The two tables below show the parameter and data range configuration for this exchange.

Consumed Exchange (ENIU)	
Name	**Outputs_Pri_to_ENIUs**
Producer ID	0.0.0.0 *
Group ID	1 **
Exchange ID	1 ***
Adapter Name	0.4
Consumed Period	200 (read only, not used)
Update Timeout	32 †

Produced Exchange (Controller)	
Name	**Outputs_to_ENIUs**
Exchange ID	1 ***
Adapter Name	0.5
Destination Type	Multicast
Destination	1 **
Produced Period	10 †
Reply Rate	0 (read only, not used)
Send Type	Always (read only)

* The Producer ID must match the Local Producer ID in the controller. If using the individual RX3i Ethernet NIU template this field will be 0.0.0.0 and must be changed.

** Must match in ENIU Consumed Exchange and Controller Produced Exchange.

*** Must match in ENIU Consumed Exchange and Controller Produced Exchange.

† Produced Period and Update Timeout need to be set for the system, see chapter 8.

All other EGD exchange parameters are pre-populated if using the templates. These defaults should be used for most applications. If a template is not used, the controller exchange must be created and the parameters must be entered.

The data ranges for the exchange (the outputs coming from the primary controller) are:

ENIU 01 Consumed Exchange Outputs_Pri_to_ENIUs						***Controller Produced Exchange Outputs_to_ENIUs***					
Offset	*Ref Addr*	*Ignore*	*Length*	*Type*	*Description*	*Variable*	*Ref Addr*	*Ignore*	*Length*	*Type*	*Desc*
Status	%T0001	False	16	Bit	EGD Status	OutEx_Status_LANA	<Sym>	False	1	Word	Status
Time Stamp	NOT USED	False	0	Byte							
0.0	%R01111	False	10	Word	Control Words from Primary LAN A	ControlWords_LANA_B	<Sym>	N/A	10	Word	
20.0	%M00001	False	2048	Bit	Discrete Outputs from Primary LAN A		%Q00001	N/A	2048	Bit	
276.0	%R00001	False	512	Word	Analog Outputs from Primary LAN A		%AQ0001	N/A	512	Word	

The first data range, offset 0.0, is for the 10 words of Control data from the primary controller.

The second data range, offset 20.0, is for the discrete outputs sent from the primary controller. The default output range shown above will receive 2048 discrete outputs and put them in %M0001 through %M2048. The discrete outputs MUST use these reference addresses for the Ethernet NIU to operate properly. If the primary controller will send more than 2048 discrete outputs to the Ethernet NIUs, discrete outputs above 2048 will not operate properly. If the system needs less than 2048 discrete outputs for all the Ethernet NIUs, this data range can be edited to a smaller number, but the same change must be made in all of the Ethernet NIUs and in the controller.

The third data range for this EGD exchange, offset 276.0, is for the analog outputs sent from the controller. The default output range shown above will receive 512 analog outputs and put them in %R001 through %R512. The analog outputs MUST use these reference addresses for the ENIU to operate properly. If more than 512 analog outputs are sent to the Ethernet NIUs, analog outputs above 512 will not operate properly. If the system needs fewer than 512 analog outputs total for all the ENIUs, this data range can be edited to a smaller number, but the same change must be made in all of the ENIUs and in the controller.

If the control system includes multiple Ethernet NIUs and a total of more than 2048 discrete or 512 analog outputs, the controller must use separate (and separately-configured) exchanges to send the outputs. See chapter 8 for more information.

Configuring the EGD Exchange Inputs_from_ENIU_xx

In this EGD exchange, xx represents the ENIU number, such as ENIU_01 as shown here. When creating new ENIUs, replace the xx with the actual ENIU number.

The two tables below show the parameter and data range configuration for this exchange.

Produced Exchange (ENIU)	
Name	**Inputs_from_ENIU_01**
Exchange ID	1 **
Adapter Name	0.4
Destination Type	Multicast
Destination	2 ***
Produced Period	10 †
Reply Rate	0 (read only, not used)
Send Type	Always (read only)

Consumed Exchange (Controller)	
Name	**Inputs_from_ENIU_01**
Producer ID	*
Group ID	2 ***
Exchange ID	1 **
Adapter Name	0.5
Consumed Period	200 (read only, not used)
Update Timeout	32 †

* The Producer ID for the consumed exchange must match the Local Producer ID of the Ethernet NIU. The consumed exchange at the controller must match the Local Producer ID of the Ethernet NIU.

** Must match in Ethernet NIU Produced Exchange and Controller Consumed Exchange.

*** Must match in Ethernet NIU Produced Exchange and Controller Consumed Exchange.

† Produced Period and Update Timeout need to be set for the system, see chapter 8.

All other Ethernet NIU parameters should be left at their default values unless special circumstances exist.

Configuration of Inputs from ENIU_01

ENIU 01 Produced Exchange *Inputs_from_ENIU_01*						**Controller Consumed Exchange*Inputs_from_ENIU_01***					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
Status	%T0033	False	16	Bit	EGD Status		* note	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
0.0	%R1101	N/A	10	Word	Status Words from ENIU LAN A		* note	False	10	Word	
20.0	**%I0001**	**N/A**	**200**	**Bit**	Discrete Inputs from ENIU		**%I0001**	**False**	**200**	**Bit**	
45.0	**%AI001**	**N/A**	**128**	**Word**	Analog Inputs from ENIU		**%AI001**	**False**	**128**	**Word**	

* Note: Assign free memory addresses or create variables for the EGD Status and Ethernet NIU Status variables in the controller.

Add data ranges as needed to configure the I/O Station's discrete and analog inputs. These inputs must not duplicate or overlap any inputs in other NIUs or in the local CPU. The ranges entered will correspond to inputs %I and %AI 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 unless that gap is intended to be spare space for the Ethernet NIU.

As an example, if a system had three Ethernet NIUs and each Ethernet NIU had 200 discrete inputs and 128 analog inputs the addressing of the inputs would be:

ENIU01	Discrete Inputs %I1-200	Analog Inputs %AI1-128
ENIU02	Discrete inputs %I201-400	Analog Inputs %AI129- 256
ENIU03	Discrete Inputs %I401-600	Analog Inputs %AI257 - 385

In the controller

Discrete Inputs %I1-200 and Analog Input %AI1-128 are from ENIU01

Discrete Inputs %I201-400 and Analog Input %AI129 - 256 are from ENIU02

Discrete Inputs %I401-600 and Analog Input %AI257 - 385 are from ENIU03

Properties and Data Ranges for SVC_Xchg_Pri_to_ENIU_xx

In this Ethernet Global Data exchange, xx represents the Ethernet NIU number, such as ENIU_01. When creating new Ethernet NIUs, replace the xx with the actual Ethernet NIU number.

The two tables below show the parameter and data range configuration for this exchange.

Consumed Exchange (ENIU)	
Name	***SVC_Xchg_Pri_to_ENIU_01***
Producer ID	0.0.0.0 *
Group ID	31 ***
Exchange ID	101 **
Adapter Name	0.4
Consumed Period	200 (read only, not used)
Update Timeout	230

Produced Exchange (Controller)	
Name	***SVC_Xchg_to_ENIU_01***
Exchange ID	101 **
Adapter Name	0.5
Destination Type	Multicast
Destination	31 ***
Produced Period	75
Reply Rate	0 (read only, not used)
Send Type	Always (read only)

* The Producer ID must match the EGD Local Producer ID in the controller. If using the individual RX3i Ethernet NIU template this field will be 0.0.0.0 and must be changed. 10.10.10.101 is the EGD Local Producer ID used in the GE Fanuc templates for the controller.

** The Exchange ID must match in Ethernet NIU Consumed Exchange and Controller Produced Exchange. Each Ethernet NIU must have a unique exchange number for this exchange.

*** Must match in Ethernet NIU Consumed Exchange and Controller Produced Exchange.

All of the other Ethernet NIU parameters should be left at the default values unless special circumstances exist.

The data ranges for the exchange are:

ENIU 01 Consumed Exchange *SVC_Xchg_Pri_to_ENIU_01*						**Controller Exchange** *SVC_Xchg_to_ENIU_01*					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
Status	%R01042	False	1	Word	EGD Status	SVC_Out_Status_LANA_ENIU01	<Sym>	False	1	Word	
Time Stamp	Not used	False	0	Byte							
0.0	%R04931	False	2	Word	Fault Ack and Fault Clear	FItack_ENIU01	<Sym>	N/A	1	Word	
						ClearFaults_ENIU01	<Sym>	N/A	1	Word	
4.0	**%R02401**	False	**200**	**Word**	RCC Request	RCC_Request_ENIU01	<Sym>	N/A	**200**	**Word**	

All the data ranges must be as shown, the Ethernet NIU expects to have the status and data ranges at these addresses. All Ethernet NIUs use the same data ranges for the exchange *SVC_Xchg_Pri_to_ENIU_xx*.

In the controller, Symbolic variables are shown for the variables used by the exchange.

Properties and Data Ranges for SVC_Xchg_from_ENIU_xx

In this EGD exchange, xx represents the ENIU number, such as ENIU_01. When creating new Ethernet NIUs, replace the xx with the actual Ethernet NIU number.

The two tables below show the parameter and data range configuration for the exchange.

Produced Exchange (ENIU)	
Name	***SVC_Xchg_from_ENIU_01***
Exchange ID	101 **
Adapter Name	0.4
Destination Type	Multicast
Destination	32 ***
Produced Period	75
Reply Rate	0 (read only, not used)
Send Type	Always (read only)

Consumed Exchange (Controller)	
Name	***SVC_Xchg_from_ENIU_01***
Producer ID	*
Group ID	32 ***
Exchange ID	101 **
Adapter Name	0.5
Consumed Period	200 (read only, not used)
Update Timeout	230

* The Producer ID for the consumed exchange must match the Local Produced ID of the Ethernet NIU. The consumed exchange at the controller must match the Local Producer ID of the Ethernet NIU.

** The Exchange ID must match in Ethernet NIU Produced Exchange and Controller Consumed Exchange.

*** Must match in Ethernet NIU Produced Exchange and Controller Consumed Exchange.

All other Ethernet Global Data exchange parameters are pre-populated and should be left at the default values unless special circumstances exist. If a template was not used, the controller exchange needs to be created and the parameters need to be entered

The data ranges for the exchange are:

ENIU 01 Produced Exchange *SVC_Xchg_from_ENIU_01*						**Controller Consumed Exchange** *SVC_Xchg_from_ENIU_01*					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
Status	%R1041	False	1	Word	EGD Status	*SVC_In_Status_LANA_ENIU_01*	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
0.0	%R04951	N/A	24	Word	Fault Data from ENIU	*Fltdata_LANA_ENIU01*	<Sym>	False	24	Word	
48.0	**%R01201**	**N/A**	**200**	**Word**	RCC Response from ENIU	***RCC_Response_LANA_ENIU01***	<Sym>	**False**	**200**	**Word**	

All the data ranges must be as shown, the Ethernet NIU expects to have the status and data ranges at these addresses. All ENIUs use the same data ranges for the exchange *SVC_Xchg_from_ENIU_xx*.

In the controller, Symbolic variables are shown for the variables used by the exchange.

Properties and Data Ranges for Outputs_Pri_to_ENIUs_LANB

The two tables below show the parameter and data range configuration for this exchange.

Consumed Exchange (ENIU)	
Name	**Outputs_Pri_to_ENIUs_LANB**
Producer ID	0.0.0.0 *
Group ID	1 **
Exchange ID	3 ***
Adapter Name	0.5
Consumed Period	200 (read only, not used)
Update Timeout	32 †

Produced Exchange (Primary Controller)	
Name	**Outputs_to_ENIUs_LANB**
Exchange ID	3 ***
Adapter Name	P.0.6\S.0.6
Destination Type	Multicast
Destination	1 **
Produced Period	10 †
Reply Rate	0 (read only, not used)
Send Type	Always (read only)
Produce in Backup Mode	See note ††
Effective Exchange ID	See note ††

* The Producer ID must match the Local Producer ID in the controller. If using the individual RX3i Ethernet NIU template this field will be 0.0.0.0 and must be changed. 10.10.10.101 is the Local Producer ID used in the GE Fanuc templates for the controller.

The same Producer ID is used on both LAN A and LAN B.

** Must match in ENIU Consumed Exchange and Controller Produced exchange.

*** Must match in ENIU Consumed Exchange and Controller Produced exchange.

† Produced Period and Update Timeout need to be set for the system, see chapter 8.

†† If an RX7i CRE or RX3i CRU controller is used, there is not an *Exchange Outputs_Sec_to_ENIUs* in the controllers. The exchange *Outputs_to_ENIUs* is used by both controllers, Produce in Backup mode is true and the Exchange ID in the Secondary controller has an offset added to the Exchange ID. The default offset added is 1000.

For an RX7i CRE or RX3i CRU controller, the same EGD Local Producer ID is used by both the primary and secondary controllers.

All of the other EGD exchange parameters are pre-populated. These defaults should be used unless special circumstances exist. If a template was not used, the controller exchange must be created and the parameters must be entered.

The data ranges for the exchange (the outputs coming from the controller) are:

ENIU 01 Consumed Exchange Outputs_Pri_to_ENIUs_LANB						**CRE/CRU HSB System Produced Exchange Outputs_to_ENIUs_LANB**					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
Status	%T0049	False	16	Bit	EGD Status	OutEx_Status_LANB	<Sym>	False	1	Word	
Time Stamp	Not used	False	0	Byte							
0,0	%R1161	False	10	Word	Control Words from Primary LAN B	ControlWords_LANA_B	<Sym>	N/A	10	Word	
20.0	**%M10001**	False	**2048**	**Bit**	Discrete Outputs from Primary LAN B		%Q00001	**N/A**	**2048**	**Bit**	
276.0	**%R10001**	False	**512**	**Word**	Analog Outputs from Primary LAN B		%AQ0001	**N/A**	**512**	**Word**	

The first data range, offset 0.0, is 10 words of Control data.

The second data range, offset 20.0, is the discrete outputs sent from the controller. The default output range shown above will receive 2048 discrete outputs and put them in %M0001 through %M2048. The discrete outputs MUST use these reference addresses for the ENIU to operate properly. If more than 2048 discrete outputs are sent to the ENIUs, discrete outputs above 2048 will not operate properly. If the system needs less than 2048 discrete outputs for all the ENIUs, this data range can be edited to a smaller number, but the same change must be made in all of the ENIUs and in the controller.

The third data range, offset 276.0, is the analog outputs sent from the controller. The default output range shown above will receive 512 analog outputs and put them in %R001 through %R512. The analog outputs MUST use these reference addresses for the ENIU to operate properly. If more than 512 analog outputs are sent to the Ethernet NIUs, analog outputs above 512 will not operate properly. If the system needs fewer than 512 analog outputs total for all the ENIUs, this data range can be edited to a smaller number but the same change must be made in all of the ENIUs and in the controller.

If the control system includes multiple Ethernet NIUs and a total of more than 2048 discrete outputs or 512 analog outputs, the controller must use separate (and separately-configured) exchanges to send the outputs. See chapter 8 for more details.

Properties and Data Ranges for Outputs_Sec_to_ENIUs_LANB

This exchange is not needed for single controllers systems and can be deleted.

For RX7i CRE and RX3i CRU controllers, this exchange is only configured in the Ethernet NIUs, it is not present in the CRE/CRU controllers. See notes for *Outputs_Pri_to_ENIUs*.

The two tables below show the parameter and data range configuration for this exchange.

Consumed Exchange (ENIU)	
Name	**Outputs_Sec_to_ENIUs_LANB**
Producer ID	See Notes Below *
Group ID	1 **
Exchange ID	1003
Adapter Name	0.5
Consumed Period	200 (read only, not used)
Update Timeout	32

Produced Exchange (Secondary Controller)	
Name	See Notes below
Exchange ID	1003 *
Adapter Name	0.6
Destination Type	Multicast
Destination	1 **
Produced Period	10
Reply Rate	0 (read only, not used)
Send Type	Always (read only)
Produce in Backup Mode	*
Effective Exchange ID	*

* When using an RX7i CRE or RX3i CRU controller in redundant mode, the secondary controller uses the same exchange as the primary. Produced Exchange (Name) and EGD Producer ID for the secondary controller are the same as the primary, the Exchange ID in the secondary has an offset added automatically (default offset is 1000) and the values used show in the row Effective Exchange ID. The ENIU is defaulted to the Exchange name *Outputs_Sec_to_ENIUs_LANB* and it should not be changed. If a different controller is used or a CRE controller is used in non-redundant mode, the Exchange Name and Exchange ID in the controller should be *Outputs_Sec_to_ENIUs_LANB* and 1003 respectively as shown in the table above.

** Must match in ENIU Consumed Exchange and Controller Produced exchange.

All of the other EGD exchange parameter defaults should be used unless special circumstances exist.

The data ranges for the exchange (the outputs coming from the controller) are:

ENIU 01 Consumed Exchange Outputs_Sec_to_ENIUs_LANB						**CRE/CRU HSB System Produced Exchange Outputs_Sec_to_ENIUs_LANB**					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Description
Status	%T0065	False	16	Bit	EGD Status	OutEx_Status_LANB	<Sym>	False	1	Word	
Time Stamp	Not used	False	0	Byte							
0.0	%R1171	False	10	Word	Control Words from Pri LAN **B**	ControlWords_LANA_B	<Sym>	N/A	10	Word	
20.0	**%M12049**	False	**2048**	**Bit**	Discrete Outputs from Pri LAN **B**		%Q00001	**N/A**	**2048**	**Bit**	
276.0	**%R10513**	False	**512**	**Word**	Analog Outputs from Pri LAN **A**		%AQ0001	**N/A**	**512**	**Word**	

The first data range, at offset 0.0, is 10 words of Control data,

The second data range, at offset 20.0, is the discrete outputs sent from the controller. The default output range shown above will receive 2048 discrete outputs and puts them in %M12049 through %M14096. The discrete outputs MUST use these reference addresses for the Ethernet NIU to operate properly. If more than 2048 discrete outputs are sent to the Ethernet NIUs, the discrete outputs above 2048 will not operate properly. If the system needs less than 2048 discrete outputs for all the Ethernet NIUs, this data range can be edited to a smaller number, but the same change must be done in all of the Ethernet NIUs and in the controller.

The third data range, offset 276.0, is the analog outputs sent from the controller. The default output range shown above will receive 512 analog outputs and puts them in %R10513 through %R11024. The analog outputs MUST uses these reference addresses for the Ethernet NIU to operate properly. If more than 512 discrete outputs are sent to the Ethernet NIUs, the discrete outputs above 512 will not operate properly. If the system needs less than 512 analog outputs for all the Ethernet NIUs, this data range can be edited to a smaller number, but it must be done in all of the Ethernet NIUs and in the controller.

If the control system includes multiple Ethernet NIUs and a total of more than 2048 discrete or 512 analog outputs, the controller must use separate (and separately-configured) exchanges to send the outputs.

Properties and Data Ranges for Inputs_from_ENIU_xx_LANB

In this Ethernet Global Data exchange, xx represents the Ethernet NIU number, such as ENIU_01. When creating new Ethernet NIUs, replace the xx with the actual Ethernet NIU number.

The two tables below show the parameter and data range configuration for this exchange.

Produced Exchange (ENIU)	
Name	***Inputs_from_ENIU_01_LANB***
Exchange ID	3 **
Adapter Name	0.5
Destination Type	Multicast
Destination	2 ***
Produced Period	10 †
Reply Rate	0 (read only, not used)
Send Type	Always (read only)

Consumed Exchange (Controller)	
Name	***Inputs_from_ENIU_01_LANB***
Producer ID	*
Group ID	2 ***
Exchange ID	3 **
Adapter Name	P.0.6\S.0.6
Consumed Period	200 (read only, not used)
Update Timeout	32 †

* The Producer ID for the consumed exchange must match the Local Producer ID of the ENIU. The consumed exchange at the controller must match the Local Producer ID of the ENIU.

** Must match in ENIU Produced Exchange and Controller Consumed Exchange.

*** Must match in ENIU Produced Exchange and Controller Consumed Exchange.

† Produced Period and Update Timeout need to be set for the system, see chapter 8.

All other EGD exchange parameters should be left at the default values unless special circumstances exist. If a template was not used, the controller exchange needs to be created and the parameters need to be entered.

The same Producer ID is used on both LAN A and LAN B.

Configuration of Inputs from ENIU_01

ENIU 01 Produced Exchange Inputs_from_ENIU_01_LANB						CRE/CRU HSB System Consumed Exchange Inputs_from_ENIU_01_LANB					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
Status	%T0081	False	16	Bit	EGD Status	InEx_Status_LANB_ENIU_01	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
0.0	%R1151	N/A	10	Word	Status Words from ENIU LAN B	StatusWords_LANB_ENIU_01	<Sym>	False	10	Word	
20.0	**%I0001**	**N/A**	**200**	**Bit**	**Discrete Inputs from ENIU**	**ENIU01_LANB_InputsDiscrete**	**<Sym>**	**False**	**200**	**Bool**	
45.0	**%AI001**	**N/A**	**128**	**Word**	**Analog Inputs from ENIU**	**ENIU01_LANB_InputsAnalog**	**<Sym>**	**False**	**128**	**Int**	

Add data ranges as needed to configure the I/O Station's discrete and analog inputs. These inputs must not duplicate or overlap any inputs in other NIUs or in the local CPU. The ranges entered will correspond to inputs %I and %AI 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 unless that gap is intended to be spare space reserved for the Ethernet NIU.

As an example, if a system had three Ethernet NIUs and each one had 200 discrete inputs and 128 analog inputs the addressing of the inputs would be:

ENIU01	Discrete Inputs %I1-200	Analog Inputs %AI1-128
ENIU02	Discrete inputs %I201-400	Analog Inputs %AI129 - 256
ENIU03	Discrete Inputs %I401-600	Analog Inputs %AI257 - 385

In the controller:

Discrete Inputs %I1-200 and Analog Input %AI1-128 are from ENIU01

Discrete Inputs %I201-400 and Analog Input %AI129 - 256 are from ENIU02

Discrete Inputs %I401-600 and Analog Input %AI257 - 385 are from ENIU03

Properties and Data Ranges for SVC_Xchg_Pri_to_ENIU_xx_LANB

In this EGD exchange, xx represents the Ethernet NIU number, such as ENIU_01. When creating new ENIUs, replace the xx with the actual Ethernet NIU number.

The two tables below show the parameter and data range configuration for this exchange.

Consumed Exchange (ENIU)	
Name	***SVC_Xchg_Pri_to_ENIU_01_LANB***
Producer ID	0.0.0.0 *
Group ID	31 ***
Exchange ID	201 **
Adapter Name	0.5
Consumed Period	200 (read only, not used)
Update Timeout	230

Produced Exchange (Primary or Only Controller)	
Name	***SVC_Xchg_to_ENIU_01_LANB***
Exchange ID	201 **
Adapter Name	P.0.6\S.0.6
Destination Type	Multicast
Destination	31 ***
Produced Period	75
Reply Rate	0 (read only, not used)
Send Type	Always (read only)
Produce in Backup Mode	See note ††
Effective Exchange ID	See note ††

* The Producer ID must match the EGD Local Producer ID in the controller. If using the individual RX3i Ethernet NIU template this field will be 0.0.0.0 and must be changed. 10.10.10.101is the Producer ID used in the GE Fanuc templates for the primary controller.

** The Exchange ID must match in Ethernet NIU Consumed Exchange and Controller Produced Exchange, The GE Fanuc Templates use [200 + Ethernet NIU number] for this exchange number. Each Ethernet NIU must have a unique exchange number for this exchange

*** Must match in Ethernet NIU Consumed Exchange and Controller Produced Exchange.

†† If an RX7i CRE or RX3i CRU controller is used, there is no exchange *SVC_Xchg_Pri_to_ENIUs_LANB* in the controllers. The Exchange *SVC_Xchg_to_ENIUs_LANB* is used by both controllers, Produce in Backup mode is true and the Exchange ID in the Secondary controller has an offset added to the Exchange ID. The default offset added is 1000.

All other EGD exchange parameters should be left at the default value unless special circumstances exist. If a template was not used, the controller exchange needs to be created and the parameters need to be entered.

The data ranges for the exchange are:

ENIU 01 Consumed Exchange SVC_Xchg_Pri_to_ENIU_01_LANB						**CRE/CRU HSB System Produced Exchange** SVC_Xchg_to_ENIU_01_LANB					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Description
Status	%R01025	False	1	Word	EGD Status	SVC_Out_Status_LANB_ENIU_01	<Sym>	False	1	Word	
Time Stamp	Not used	False	0	Byte							
0.0	%R04935	False	2	Word	Fault Ack and Clear from PRI LAN B	FItack_ENIU01	<Sym>	N/A	1	Word	
						ClearFaults_ENIU01	<Sym>	N/A	1	Word	
4.0	**%R02801**	False	**200**	**Word**	RCC Request from Pri LAN B	RCC_Request_ENIU01	<Sym>	**N/A**	**200**	**Word**	

All the data ranges must be as shown; the Ethernet NIU expects to have the status and data ranges at these addresses. All Ethernet NIUs will have the same data ranges for the exchange *SVC_Xchg_Pri_to_ENIU_xx_LANB***.**

In the controller, Symbolic variables are shown for the variables used by the exchange.

Properties and Data Ranges for SVC_Xchg_Sec_to_ENIU_xx_LANB

In this EGD exchange, xx represents the Ethernet NIU number, such as ENIU_01. When creating new Ethernet NIUs, replace the xx with the actual Ethernet NIU number.

The two tables below show the parameter and data range configuration for this exchange.

Consumed Exchange (ENIU)	
Name	**SVC_Xchg_Sec_to_ENIU_01_LANB**
Producer ID	0.0.0.0 *
Group ID	31 **
Exchange ID	1201 *
Adapter Name	0.5
Consumed Period	200 (read only, not used)
Update Timeout	230

Produced Exchange (Secondary Controller)	
Name	**SVC_Xchg_Sec_to_ENIU_01_LANB**
Exchange ID	1201 *
Adapter Name	0.6
Destination Type	Multicast
Destination	31 **
Produced Period	75
Reply Rate	0 (read only, not used)
Send Type	Always (read only)
Produce in Backup Mode	*
Effective Exchange ID	*

* When using an RX7i CRE or RX3i CRU controller in redundant mode, the secondary controller uses the same exchange as the primary. Produced Exchange (Name) and EGD Producer ID are the same as the primary. The Exchange ID in the secondary has an offset added automatically, and the values used show in the row Effective Exchange ID. The Ethernet NIU is defaulted to the Exchange name *SVC_Xchg_Sec_to_ENIU_xx_LANB,* which should not be changed. If a different controller is used or a CRE controller is used in non-redundant mode, the Exchange Name and Exchange ID in the controller should be *SVC_Xchg_Sec_to_ENIU_xx_LAN*B and 1201 respectively as shown in the table above. The Exchange ID is (1200 + ENIU# +) . The example shows it for ENIU_01 (1201).

** Must match in ENIU Consumed Exchange and Controller Produced Exchange.

All other Ethernet Global Data exchange parameters should be left at the default value unless special circumstances exist. If a template was not used, the controller exchange needs to be created and the parameters need to be entered.

The data ranges for the exchange are:

ENIU 01 Consumed Exchange SVC_Xchg_Sec_to_ENIU_01_LANB						**CRE/CRU HSB System Produced Exchange** SVC_Xchg_Sec_to_ENIU_01_LANB					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Description
Status	%R01026	False	1	Word	EGD Status	SVC_Out_Status_LANB_ENIU_01	<Sym>	False	1	Word	
Time Stamp	Not used	False	0	Byte							
0.0	%R04937	False	2	Word	Fault Ack and Clear from Sec LAN **B**	FItack_ENIU01	<Sym>	N/A	1	Word	
						ClearFaults_ENIU01	<Sym>	N/A	1	Word	
4.0	**%R03801**	False	**200**	**Word**	RCC Request from Sec LAN B	RCC_Request_ENIU01	<Sym>	**N/A**	**200**	**Word**	

All the data ranges must be as shown; the Ethernet NIU expects to have the status and data ranges at these addresses. All ENIUs will have the same data ranges for the exchange *SVC_Xchg_Sec_to_ENIU_xx_LANB***.**

In the controller, Symbolic variables are shown for the variables used by the exchange.

Properties and Data Ranges for SVC_Xchg_from_ENIU_xx_LANB

In this Ethernet Global Data exchange, xx represents the Ethernet NIU number, such as ENIU_01. When creating new Ethernet NIUs, replace the xx with the actual Ethernet NIU number. The two tables below show the parameter and data range configuration for this exchange.

Produced Exchange	
Name	***SVC_Xchg_from_ENIU_01_LANB***
Exchange ID	201 **
Adapter Name	0.5
Destination Type	Multicast
Destination	32 ***
Produced Period	75
Reply Rate	0 (read only, not used)
Send Type	Always (read only)

Consumed Exchange	
Name	***SVC_Xchg_from_ENIU_01_LANB***
Producer ID	*
Group ID	32 ***
Exchange ID	201 **
Adapter Name	P.0.6\S.0.6
Consumed Period	200 (read only, not used)
Update Timeout	230

* The Producer ID for the consumed exchange must match the Local Produced ID of the Ethernet NIU. The consumed exchange at the controller must match the Local Producer ID of the Ethernet NIU.

** The Exchange ID must match in Ethernet NIU Produced Exchange and Controller Consumed Exchange.

*** Must match in Ethernet NIU Produced Exchange and Controller Consumed Exchange.

All of the other EGD exchange parameters should be left at the default value unless special circumstances exist. If a template was not used, the controller exchange needs to be created and the parameters need to be entered. The producer ID in the controller configuration is the EGD Producer ID of the Ethernet NIU. It should be 10.10.10.<ENIU#>. For example, for ENIU_01, it is: 10.10.10.1, as shown. The Data ranges for the exchange are:

ENIU 01 Produced Exchange *SVC_Xchg_from_ENIU_01_LANB*						**CRE/CRU HSB System Consumed Exchange** *SVC_Xchg_from_ENIU_01_LANB*					
Offset	Ref Addr	Ignore	Length	Type	Description	Variable	Ref Addr	Ignore	Length	Type	Desc
Status	%R1027	False	1	Word	EGD Status	SVC_In_Status_LANB_ENIU_01	<Sym>	False	1	Word	
						TimeStamp	Not used	False	0	Byte	
0.0	%R04951	N/A	24	Word	Fault Data from ENIU	Fltdata_LANB_ENIU01	<Sym>	False	24	Word	
48.0	**%R01201**	**N/A**	**200**	**Word**	RCC Response from ENIU	**RCC_Response_LANB_ENIU01**	<Sym>	**False**	**200**	**Word**	

All the data ranges must be as shown; the Ethernet NIU expects to have the status and data ranges at these addresses. All ENIUs will have the same data ranges for the exchange *SVC_Xchg_from_ENIU_xx_LANB*.

In the controller, Symbolic variables are shown for the variables used by the exchange.

Chapter 8 Ethernet Global Data

This chapter describes the operation of Ethernet Global Data, and explains how to optionally customize EGD exchanges.

- EGD Exchanges in an Ethernet NIU Target
- EGD Exchanges for I/O, Status, and Control Data
  - Update Time for I/O, Status, and Control Data
  - *Inputs_from_ENIU_xx*
    - Adding Data to Input Exchanges
    - Configuring Input Registers in the Controller
    - Using Input Register Data in the Controller
    - Configuring ENIU Exchanges for NO Discrete/Analog Inputs
  - *Outputs_xxx_to_ENIUs*
    - Adding Data to Output Exchanges
    - Adding Extra Produced Exchanges *Outputs_to_ENIUs*
- EGD Exchanges for Faults and Remote COMMREQ Calls
  - SVC Operation
- EGD Timing for the Project Templates
- Setting Up SNTP Time Synchronization

EGD Exchanges in an Ethernet NIU Target

Each version 1.33 and higher Ethernet NIU target in the configuration includes three I/O-related and three *SVC_Xchg*-related Ethernet Global Data exchanges:

- *Outputs_Pri_to_ENIU*: Consumed Output data exchange from the primary controller.
- *Outputs_Sec_to_ENIU*: Consumed Output data exchange from the secondary controller.
- *Inputs_from_ENIU*: Produced Input data exchange back to the controller(s).
- *SVC_Xchg_Pri_to_ENIU*: Consumed data exchange for faults and Remote COMMREQ Call commands from the primary controller.
- *SVC_Xchg_Sec_to_ENIU*: Consumed data exchange for faults and Remote COMMREQ Call commands from the secondary controller.
- *SVC_Xchg_from_ENIU*: Produced data exchange for faults and Remote COMMREQ Call status back to controller(s).

The target also includes equivalent exchanges for LAN B in a dual LAN system.

EGD Addresses for Multiple Controllers on One LAN

If multiple controllers, each with a set of Ethernet NIUs, are used on the same LAN, each system **MUST** use different group address for the Ethernet Global Data exchanges. Each system must have a unique set of group addresses. The table below lists the exchanges and group addresses used in the Ethernet NIU template and suggests group numbers to be used for a second and third system.

Exchange Name	***Default Group #s***	***Suggested Group #s for 2nd System***	***Suggested Group #s for 3rd System***
Inputs_from…	2	4	6
Outputs_to…	1	3	5
SVC_Xchg_from…	32	30	28
SVC_Xchg_to…	31	29	27

EGD Exchanges for I/O, Status, and Control Data

An Ethernet NIU uses one Ethernet Global Data consumed data exchange to receive output and control data from a controller, and one Ethernet Global Data produced data exchange to send input and status data to a controller.

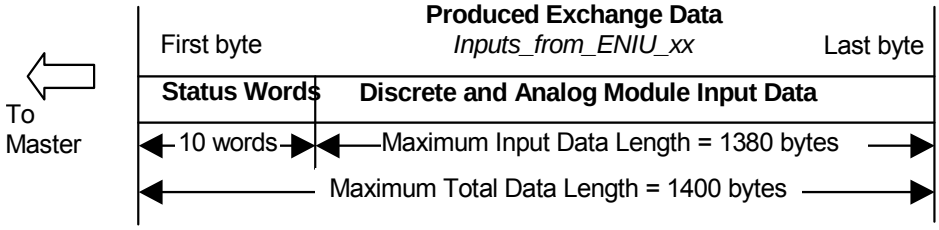
For redundant CPU systems, there is a pair of Ethernet Global Data produced and consumed exchanges for each path between the Ethernet NIU and a controller.

Update Time for the I/O, Status, and Control EGD Exchanges

A typical system might consist of a controller with five Ethernet NIU I/O Stations. In this example, the controller sends 1300 bytes of outputs, and each Ethernet NIU sends 301 bytes of inputs to the controller. This typical system would have its I/O updates occur in less than 32 milliseconds. If the controller scan time is greater than 32 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.

Inputs_from_ENIU_xx

The Ethernet NIU's produced Ethernet Global Data exchange *Inputs_from_ENIU_xx* begins with 10 words of status data followed by up to 1380 bytes of input data.

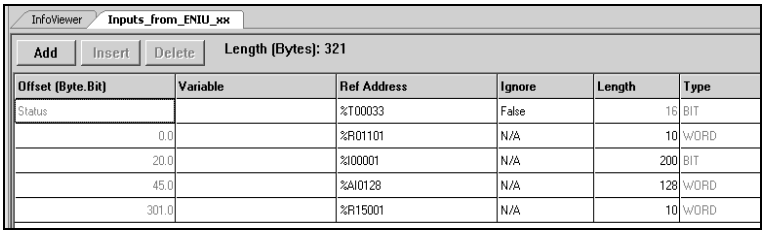


Adding Data to Input Exchanges

In addition to input data, the Ethernet NIU can send back other data to the controller(s). For example, intelligent modules or local logic in the I/O Station can produce data that is needed in the controller. There are two ways to send this additional data to the controller:

- Another data range can be added to the *Inputs_from_ENIU_xx* exchange.
- Local_User_Logic can move the data into unused portions on the %I or %AI data ranges.

For example, the following Ethernet Global Data Exchange for ENIU_01 has ten input register words (%R15001-10) assigned for extra I/O station data.



Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type
Status		%T00033	False	16	BIT
0.0		%R01101	N/A	10	WORD
20.0		%I00001	N/A	200	BIT
45.0		%AI0128	N/A	128	WORD
301.0		%R15001	N/A	10	WORD

Multiple data areas can be added as long as the total length of the EGD exchange is less than 1400 bytes in length.

Configuring Input Registers in the Controller

Input register data must be added to the Ethernet Global Data input exchange from the I/O Station in the controller application. The input register data is added in one or more separate data areas after the %AI memory in the exchange. In controller applications using a single Ethernet I/O LAN, this data can be placed in any %memory or symbolic memory in the controller.

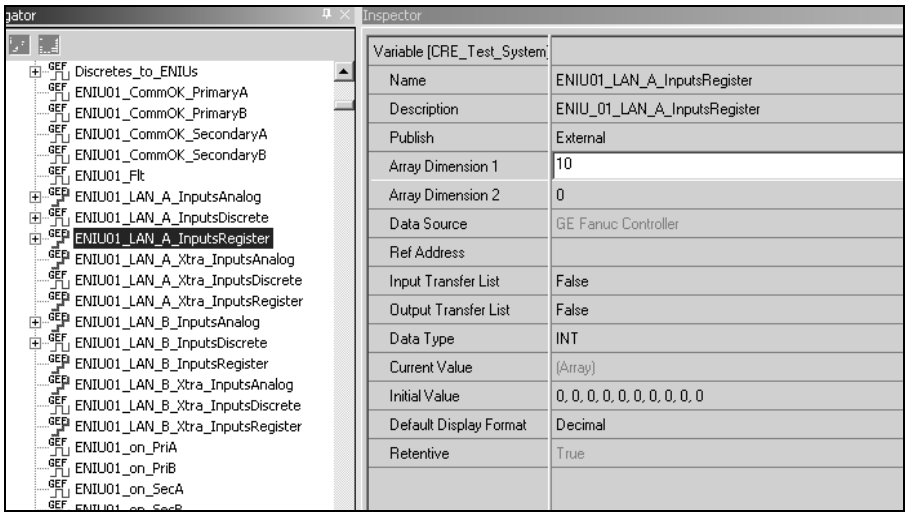
In dual/redundant Ethernet LAN applications, specific symbolic variables must be used. An example EGD Exchange for a dual/redundant LAN application is shown below.

InfoViewer | Inputs_from_ENIU_01

Add | Insert | Delete | Length (Bytes): 321

Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type
Status	InEx_Status_LANA_ENIU_01	<Symbolic>	False	1	WORD
TimeStamp		NOT USED	False	0	BYTE
0.0	StatusWords_LANA_ENIU_01	<Symbolic>	False	10	WORD
20.0	ENIU01_LAN_A_InputsDiscrete	<Symbolic>	False	200	BOOL
45.0	ENIU01_LAN_A_InputsAnalog	<Symbolic>	False	128	WORD
301.0	ENIU01_LAN_A_InputsRegister	<Symbolic>	False	10	INT

In dual/redundant Ethernet LAN applications, the symbolic variable *ENIUxx_LAN_y_InputsRegister* (xx represents the Ethernet NIU number and y represents the LAN) must be used for the Ethernet Global Data exchange. The length of the data area is specified by setting the value in Array Dimension 1 in the properties of the variable. These variables contain the input register data before the input processing logic determines which LAN (variable) is currently active.



In addition, the length of variable *ENIUxx_Register_Data* must be set to match the length of variable *ENIUxx_LAN_y_InputsRegister*.

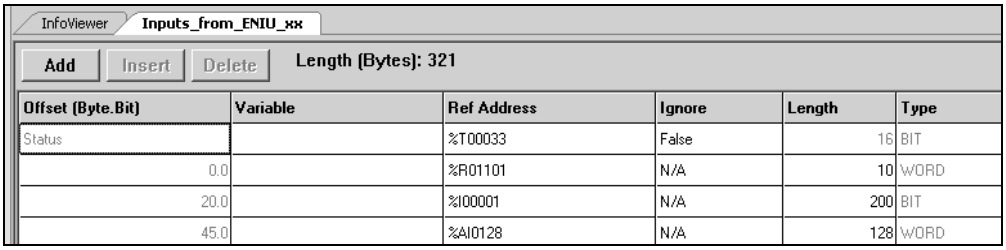
The Ethernet Global Data exchanges for both LAN A and LAN B need to be modified to include the input register data.

Using Input Register Data in the Controller

ENIUxx_Register_Data is the symbolic variable provided for use in the controller application. It contains the selected data (based on which LAN is being used) from variables *ENIUxx_LAN_A_InputsRegister* and *ENIUxx_LAN_B_InputsRegister*.

Configuring ENIU Exchanges for NO Discrete and/or Analog Inputs

Example EGD exchange for ENIU_01 with 200 discrete inputs and 128 analog inputs assigned:



InfoViewer | Inputs_from_ENIU_xx

Add | Insert | Delete | Length (Bytes): 321

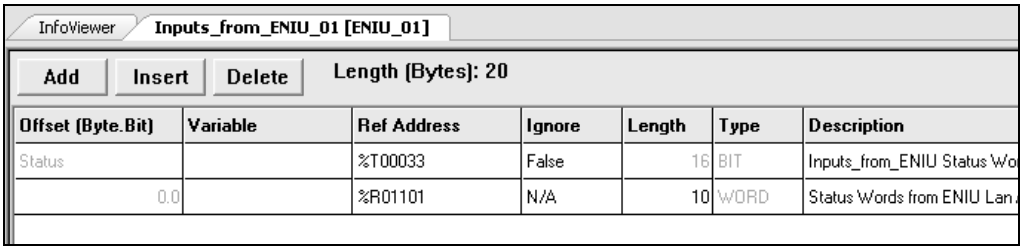
Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type
Status		%T00033	False	16	BIT
0.0		%R01101	N/A	10	WORD
20.0		%I00001	N/A	200	BIT
45.0		%AI0128	N/A	128	WORD

The data areas for either discrete inputs or analog inputs, or both, can be deleted if not required for the Ethernet NIU.

In dual LAN applications with no discrete and/or analog inputs, the Ethernet Global Data exchanges for both LAN A and LAN B must be modified to remove the discrete input and/or analog input data areas.

Caution – Do ***not*** delete the entire exchange. The ten Status Words are needed by the controller for proper operation.

Example EGD Exchange for ENIU_01 with NO discrete inputs and NO analog inputs assigned:



InfoViewer | Inputs_from_ENIU_01 [ENIU_01]

Add | Insert | Delete | Length (Bytes): 20

Offset (Byte.Bit)	Variable	Ref Address	Ignore	Length	Type	Description
Status		%T00033	False	16	BIT	Inputs_from_ENIU Status Wo
0.0		%R01101	N/A	10	WORD	Status Words from ENIU Lan

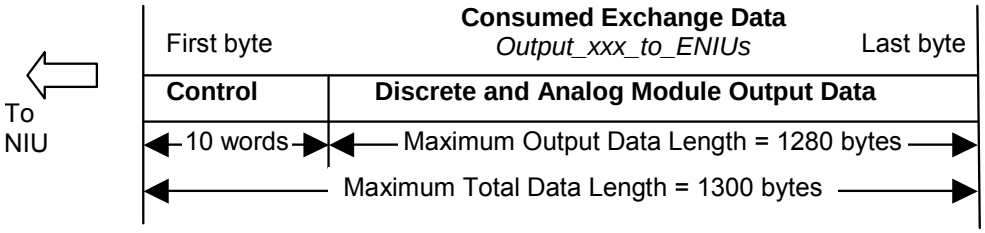
The EGD Status and ENIU Status data areas MUST still exist for proper communication with the controller.

Configuring the Controller Exchange for NO Discrete Inputs

For applications that use a single I/O LAN with either one or two controllers, data areas for the discrete and/or analog inputs can be deleted in the Ethernet Global Data exchange in the controller, to make them match the changes configured in the exchange in the Ethernet NIU (see above).

Outputs_xxx_to_ENIUs

The Ethernet NIU's consumed data exchange *Output_xxx_to_ENIUs* consists of ten words of control data followed by up to 1280 bytes of output data from the controller. This Ethernet Global Data Exchange is sent from the controller and received by all Ethernet NIUs. (Although the maximum length of an Ethernet Global Data exchange for other devices is 1400 bytes, the maximum data length of the consumed EGD output exchange for the Ethernet NIU is 1300 bytes).



In a system with redundant CPUs, both CPUs send output data to the Ethernet NIU. The Ethernet NIU uses data from only one output exchange based on health of communication and on command action.

Adding Data to Output Exchanges

In some circumstances, word data (%R, %W, symbolic) needs to be sent to the Ethernet NIUs in the *Outputs_xxx_to_ENIUs* exchange. Because the Ethernet NIU can only use %Q and %AQ for receiving data, the application logic must move the word data into an unused area of the %AQ data to be sent to the Ethernet NIUs. In the Ethernet NIUs, word data can be used directly from the %AQ references. Alternatively, the Local_User_Logic can be used to move the word data in the %AQ reference into %R, %W, or symbolic variables as required.

Remember that the *Outputs_xxx_to_ENIUs* exchange goes to all the Ethernet NIUs.

Adding Extra Produced Exchanges Outputs_xxx_to_ENIUs

There are two cases when the controller(s) may need to have more than one EGD Exchange of the type *Outputs_xxx_to_ENIUs*: (xxx = Pri or Sec)

- If there are more than 2048 discrete outputs and/or 512 analog outputs, the exchange *Outputs_xxx_to_ENIUs* must be made into multiple exchanges.
- If the Ethernet NIUs are separated onto multiple LANs, and the controller has two or more Ethernet Transmitter modules with different Ethernet NIUs handled by each Ethernet Transmitter Module.

If there are two controllers, the exchange *Outputs_Sec_to_ENIUs* must be made into multiple exchanges.

Example of Increasing the Number of Outputs in an ENIU System

A single-controller system with ten Ethernet NIUs has a total of 3500 discrete outputs and 750 analog outputs. That is more than the maximum number of 2048 discrete outputs and/or 512 analog outputs that can be handled by an Ethernet NIU.

In this example, the EGD Exchange *Outputs_Pri_to_ENIU_1_5* is set up to go to Ethernet NIUs 1 through 5. The controller is configured to send outputs %Q0001 to %Q2048 and %AQ0001 to %AQ0512 using this exchange. The Ethernet NIUs will put these outputs in their standard internal references: %M0001 to %M2048 and %R0001 to %R0512.

Another EGD Exchange, *Outputs_Pri_to_ENIU_6_10*, is set up to go to Ethernet NIUs 6 through 10. The controller is configured to send %Q2049 to %Q4096 and %AQ0513 to %AQ1024 using this exchange. The Ethernet NIUs will put these outputs in their standard internal references: %M0001 to %M2048 and %R0001 to %R0512. In this case, the controller outputs %Q2049 to %Q4096 and %AQ0513 to %AQ1024 get mapped to %Q0001 to %Q2048 and %AQ0001 to %AQ0512 in ENIUs 6 through 10.

If the Ethernet NIUs are on different LANs, the two exchanges *Outputs_Pri_to_ENIU_1_5* and *Outputs_Pri_to_ENIU_6_10* would need to be sent to different Ethernet interfaces (adapter name in the EGD exchange properties). If the number of discrete and analog outputs is less than one exchange requires, the same ranges can be configured for exchanges on both Ethernet interfaces. Different LANs are usually used to increase the performance of the I/O.

EGD Exchanges for Faults and Remote COMMREQ Calls

SVC (Service Exchange) Operation

The SVC Ethernet Global Data Exchanges only work with PACSystems RX7i or RX3i controllers. If Rx3i Ethernet NIUs are used with any other controllers, the SVC Exchanges should be deleted from the ENIU template in the programmer.

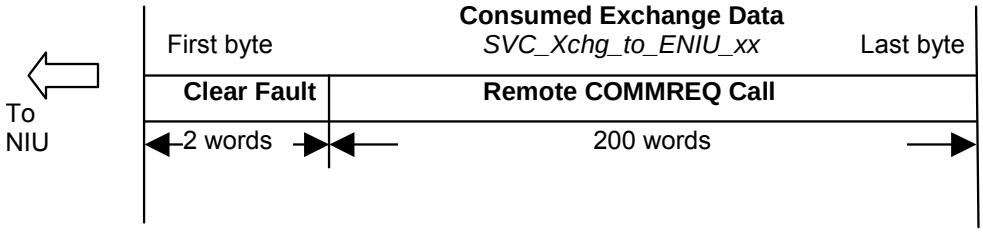
The number of SVC Ethernet Global Data Exchanges that will be used for each Ethernet NIU in an RX7i or RX3i system depends on whether the Ethernet NIU has dual controllers and /or dual LANs.

- For each Ethernet NIU with a single controller, there will be two SVC Xchgs, one from the controller to the Ethernet NIU and the other from the Ethernet NIU to the controller.
- For each Ethernet NIU with two controllers, there will be four SVC Xchgs.
- For each Ethernet NIU with two controllers and dual LANs, there will be eight SVC Xchgs.

SVC Xchg from the Controller to the Ethernet NIU

The controller uses the SVC Xchg **to** the Ethernet NIU to send commands to execute COMMREQs in the Ethernet NIU. A C block in the RX7i or RX3i controller formats the COMMREQ so it can be sent in the SVC Xchg to the Ethernet NIU.

The Ethernet NIU's consumed exchange *SVC_Xchg_to_ENIU_xx* begins with 2 words of data that can be used to clear faults, followed by 200 words of Remote COMMREQ Call data.



The second word in the *SVC_Xchg_to_ENIU_xx* can be used to clear faults in an individual ENIU. See chapter 10 for details. All other fields in the SVC_Xchg Ethernet Global Data exchanges are used internally by the Ethernet NIU and C blocks in the controller(s); they do not have fields that need to be accessed.

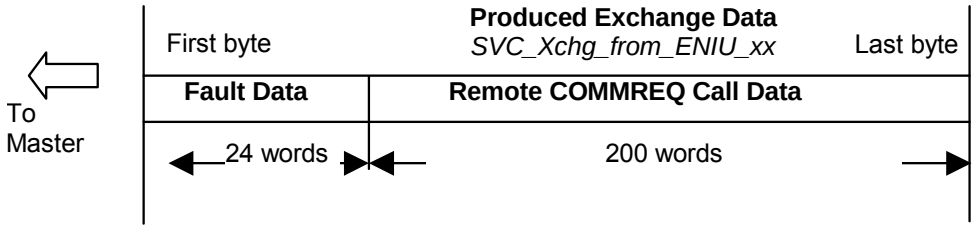
SVC Xchg from the Ethernet NIU to the Controller

The Ethernet NIU uses its SVC Xchg to the controller to return the results of the COMMREQ execution. In addition, when non-fatal faults occur in the Ethernet NIU, they are also sent in the SVC Xchg **from** the Ethernet NIU to the controller.

A C block in the RX7i or RX3i controller puts the faults in the controller PLC Fault Table. If multiple faults occur in an Ethernet NIU, they are queued up and spooled off to the controller.

In addition to the Ethernet Global Data exchanges for I/O, status, and control data, version 13x or later of the Ethernet NIU templates provides another pair of Ethernet Global Data exchanges that can be used for Fault Reporting and Remote COMMREQ Call features. Both functions are combined in these exchanges. One pair of exchanges is required for each path from the Ethernet NIU to the controller(s).

The Ethernet NIU's produced exchange *SVC_Xchg_from_ENIU_xx* begins with 24 words of Fault Reporting data followed by 200 words of Remote COMMREQ Call data.



All other fields in the *SVC_Xchg Ethernet Global Data* exchanges are used internally by the Ethernet NIU and C blocks in the controller(s); they do not have fields that need to be accessed.

EGD Timing for the Project Templates

In the preconfigured RX3i and RX3i project templates, the Ethernet Global Data I/O exchanges have been set up with appropriate default produced periods and default consumed update timeouts. These default values accommodate almost any I/O mix and corresponding exchange sizes for a system with the number of Ethernet NIUs in the template, and accommodate a single programmer on the same Ethernet LAN that is used for the Ethernet I/O. The produced periods and consumer update timeouts can be modified, as long as the maximum number of I/O Stations and the exchange sizes listed below is not exceeded.

Ethernet NIU Ethernet Global Data Settings for Version 1.3x or Greater

Tested produced period and consumed update timeout settings for Ethernet NIU EGD Exchanges* with PACSystems RX7i or RX3i controllers for PPS/PME:

			IO Exchanges Suggested Values		*SVC Exchanges Suggested Values*	
ENIU I/O Stations	***Input Exchanges per I/O Station***	***Output Exchanges from Controller(s)***	***Produced Period(ms)***	***Consumed Timeout (ms)***	***Produced Period(ms)***	***Consumed Timeout(ms)***
RX7i Settings						
Up to 5	1 (205 bytes)	1 (1300 bytes)	10	32	75	230
Up to 10	1 (205 bytes)	1 (1300 bytes)	14	44	75	230
Up to 20	1 (205 bytes)	2 (1300 bytes)	26	80	75	230
RX3i Settings						
Up to 10	1 (205 bytes)	1 (1300 bytes)	16	50	75	230
Up to 20	1 (205 bytes)	2 (1300 bytes)	26	80	75	230

If SVC exchanges (used for fault-clearing and Remote COMMREQ Call commands) are not needed for the application, they should all be deleted from the controller(s) and the Ethernet NIUs in Machine Edition. If all SVC exchanges in all Ethernet NIUs and controllers(s) are deleted, the I/O exchanges will execute more quickly and the settings below can be used instead.

Ethernet NIU Ethernet Global Data Settings for Version 1.2x

Tested produced period and consumed update timeout settings for Ethernet NIU EGD Exchanges* with PACSystems RX7i or RX3i controllers.

			IO Exchanges Suggested Values		*SVC Exchanges Suggested Values*
ENIU I/O Stations	***Input Exchanges per I/O Station***	***Output Exchanges from Controller(s)***	***Produced Period(ms)***	***Consumed Timeout (ms)***	Not used. Therefore, the I/O exchanges can run at faster rates.
Up to 5	1 (205 bytes)	1 (1300 bytes)	6	20	
Up to 10	1 (205 bytes)	1 (1300 bytes)	8	26	
Up to 20	1 (205 bytes)	1 (1300 bytes)	14	44	
Up to 42	1 (205 bytes)	4 (1300 bytes)	25	77	

* The tables above are based on Ethernet NIU I/O Stations that consume a 1300-byte output data Ethernet Global Data exchange, and produce a 205-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 produced period and consumed update timeout. For any Ethernet NIU I/O Stations 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 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 of the Ethernet Global Data exchanges and their frequency of production have 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 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 determining the consumed update timeout parameter is to multiply the produced period by 3 and add 2 milliseconds. If the update timeout period of a consumer 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 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 consumed update times listed in the previous table. For example, an output exchange in a system that is configured as described above with five I/O Stations will be received in 18 milliseconds or less. This can be considered the one-way update rate for that configuration. As usual, for the overall system I/O response time, controller scan time also must be taken into consideration.

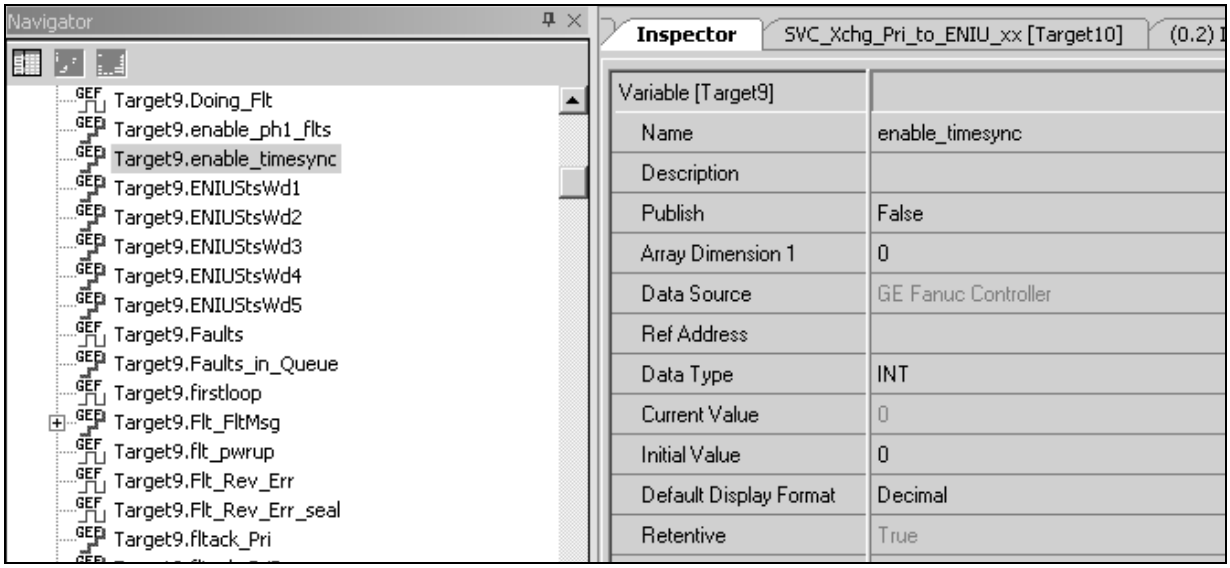
For applications that require performance or update rates faster than the times listed in the preceding tables, consider breaking your single I/O network into two separate I/O networks. This will require an additional Ethernet module in each controller.

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 network with 18 I/O Stations into two networks with 9 I/O Station each. Each 9 I/O Station network then can be configured for produce periods of 8 milliseconds and EGD exchange consumed timeout values of 24 milliseconds.

Setting Up SNTP Time Synchronization

SNTP Time Synchronization sets the Ethernet NIU Time of Day clock to UTC (Greenwich Mean Time), not to local time. An SNTP Time Server must be present on the LAN.

The Time of Day clock can be set up for synchronization to a SNTP Time Server using the template sets. The setup can be completed by changing the initial value of the variable *enable_timesync* to 1, and storing to the Ethernet NIU.



This needs to be done for each Ethernet NIU that will use SNTP Time Synchronization.

Advantage of Using SNTP Time Synchronization

In addition to the benefit of setting the clocks in all the Ethernet NIUs to the same value, each Ethernet Global Data exchange produced has a timestamp. Normally, the timestamp comes from the Ethernet Interface, either time-synchronized or not. When Time Synchronization is enabled, the EGD exchange timestamp comes from the Ethernet NIU's Time of Day clock instead.

Time Synchronization and the AUP File in the Ethernet Transmitter

The Ethernet NIU is told to synchronize to a SNTP time server by storing the Advanced User Parameters file to the Ethernet NIU.

Ethernet Global Data produced exchanges have a flag that indicates whether their time is synchronized or not. If the time is not synchronized, the controller receiving the EGD exchange will receive a status of 3 (EGD exchange updated correctly but time not synchronized). If this causes a problem with the controller, delete the Advanced User Parameters file for the Ethernet Transmitter Module and download to the Ethernet NIU. To delete the Advanced User Parameters file from the Ethernet Transmitter Module, select the Ethernet interface in the navigator, right click, select Properties, and delete the AUP File Name shown in the property field.

If the CPU receives a fault message that the SNTP Time Synchronization COMMREQ has failed, the Advanced User Parameters file may be missing or incorrect. If the Ethernet Transmitter Module has been moved to a different slot, both the Advanced User Parameters filename and the contents of the AUP file that specifies the location of the Ethernet Interface must be corrected.

AUP files are provided for slot 3, slot 4, and slot 5. They can be found in the supplemental files of the targets in the templates.

Disabling Fault Table SNTP Alarms when SNTP is Not Used

If SNTP is not enabled (*enable_timesync* is 0), alarms will be placed in the ENIU Fault table and forwarded to the controller.

The Fault Table entries can be shut off by changing the AUP file(s) to disable SNTP. If there are two Ethernet Transmitter Modules in the Ethernet NIU's I/O Station, the AUP file for each Ethernet Transmitter Module must be changed.

To disable SNTP, in the AUP file find the line *ncpu_sync* = 1. Change it to *ncpu_sync* = 0

Chapter 9 I/O Data - Control, Status, and I/O Data Formats

This chapter describes the data that an Ethernet NIU regularly exchanges with its controller(s).

- Data in the ENIU System
  - Addressing I/O in an ENIU System
  - Data Memory in the Ethernet NIU

    References Used in the Ethernet NIU

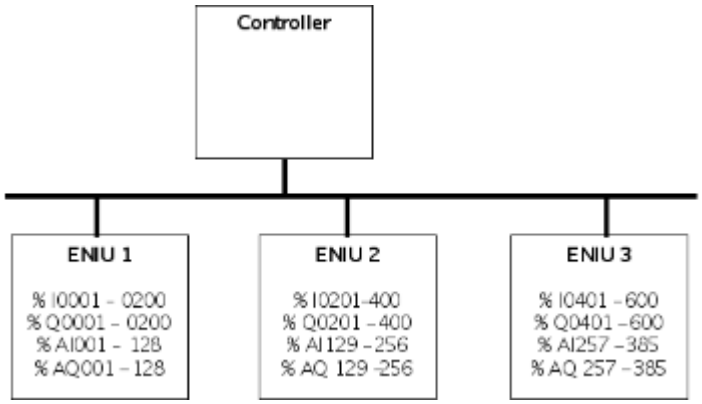
    Discrete and Analog Outputs in the Ethernet NIU
- Exchanging Data with One or Two Controllers
  - ENIU Operation with Two Controllers
  - ENIU Operation with Two Controllers and Dual Ethernet LANs
  - ENIU Operation if No Data is Received
  - ENIU Operation if Data is Received from the Backup Controller
- Exchanges for I/O, Status, and Control Data
  - Format of Output Exchanges
  - Format of Input Exchanges
  - Control Data Format
  - Status Data Format
- EGD Exchanges for Faults and Remote COMMREQ Calls

System I/O Data References

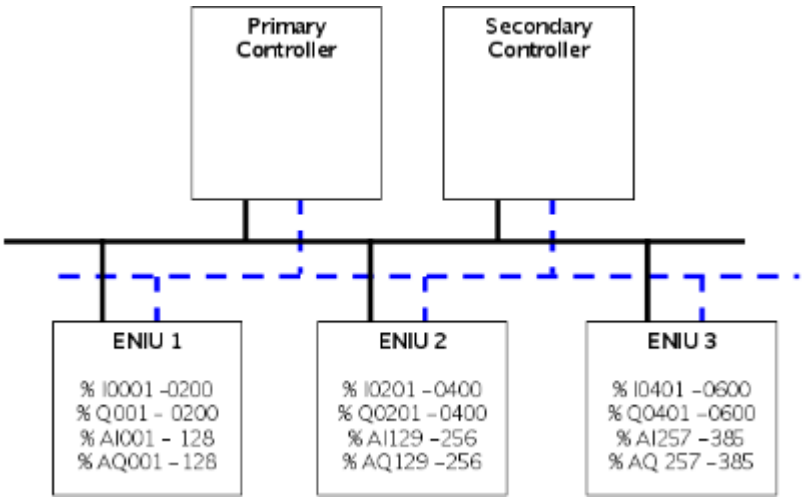
I/O modules are added to the Ethernet NIU configuration and their parameters are configured the same way they are configured in a PLC system.

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. Note that the example uses 200 discrete and 80 analog for each ENIU, which are the defaults when the 1.2 and 1.33 Ethernet NIU templates are used. Proficy Process Systems 1.33 templates and all 1.40 and later templates use 128 analog as the default.

Duplicated input references from 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 20000)
Analog Inputs - %AI	32640 (Default is 4000)
Analog Outputs - %AQ	32640 (Default is 4000)
Bulk Memory - %W	(Default is 20480)

Dedicated Data Memory 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 in this section. These reference addresses are automatically configured in Ethernet Global Data exchanges when the Ethernet NIU target is created. The reference assignments that are made are different for the 1.3x and later versions of the Ethernet NIU target than for 1.2x versions of the Ethernet NIU target. If the Ethernet NIU that uses local logic is upgraded from version 1.2x to a later version of the templates, the references used in the local logic may need to be adjusted.

References Assigned in a 1.2x Ethernet NIU Target

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.*

References Assigned in a 1.3x and Higher Ethernet NIU Target

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)
Discrete Outputs from primary controller optional LAN B	Must be %M10001 - %M12048 (bits)
Discrete Outputs from secondary controller optional LAN B	Must be %M12049 - %M14096 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)
Ethernet Global Data Exchange status (consumed from primary controller) LAN B	Must be %T0049 - %T0064 (bits)
Ethernet Global Data Exchange status (consumed from secondary controller) LAN B	Must be %T0065 - %T0080 (bits)
Ethernet Global Data Exchange status (produced by ENIU) LAN B	Must be %T0081 - %T0096 (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)
Analog Outputs from primary controller optional LAN B	Must be %R10001 - %R10512 (words)
Analog Outputs from secondary controller optional LAN B	Must be %R10513 - %R11024 (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)
ENIU Status data to be sent to controller(s) LAN B	Must be %R1151 - %R1160 (words)
Control Data (from primary / only controller) LAN B	Must be %R1161 - %R1170 (words)
Control Data (from secondary controller) LAN B	Must be %R1171 - %R1180 (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 can receive discrete and analog output data from a primary controller, and optionally from a secondary controller. For applications created using 1.3x or later Ethernet NIU targets (this includes Ethernet NIUs in the application templates), the Ethernet NIU can also receive output data from both controllers on a second LAN (LAN B). In order to allow the Ethernet NIU to use default values for outputs when communication to the controller(s) is lost, the Ethernet NIU places the discrete output data in the Ethernet Global Data consumed exchange into its %M memory, and it places the analog output data into its %R memory.

If the controllers are synchronized, the same data is sent from both controllers. For systems with dual LANs, the same data is sent on both LAN A and LAN B.

Number of Outputs

The controller can send 2048 discrete outputs and 512 analog outputs, which is the maximum number of outputs the Ethernet NIU is designed to use. If there are multiple Ethernet NIUs present on the LAN, all Ethernet NIUs receive the same set of outputs. Each Ethernet NIU uses only the output references that have been assigned to the modules in the I/O Station during hardware configuration. When the data is received, the Ethernet NIU places it in memory beginning at the first reference in each table (for example, %Q0001). The exchange definitions for both the controller and the Ethernet NIU can be adjusted for improved performance by transferring only the data actually used in the system.

Systems Requiring Over 2048 Discrete Outputs or 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. See chapter 8 for details.

Exchanging Data with 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. Optionally, dual LANs can also be used in applications created with 1.3x and later Ethernet NIU targets.

ENIU Operation with Two Controllers and One Ethernet LAN

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 Ethernet NIU stops receiving data from the primary controller, the Ethernet NIU begins using output and control data from the secondary controller instead.

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

Either 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 with Two Controllers and Dual Ethernet LANs

In a system with a primary controller, secondary controller, and dual LANs, both controllers regularly send output and control data for the I/O Station on both LANs. Both controllers receive the latest input and status data from the Ethernet NIU on both LANs.

During normal operation, the Ethernet NIU uses the output and control data it receives from its primary controller on LAN A. However, if the Ethernet NIU stops receiving data from the primary controller on LAN A, the Ethernet NIU begins using output and control data from the primary controller on LAN B. If the primary controller goes away entirely, the Ethernet NIU starts using output and control data from the secondary controller. It will use the LAN it was last using before the primary controller went away.

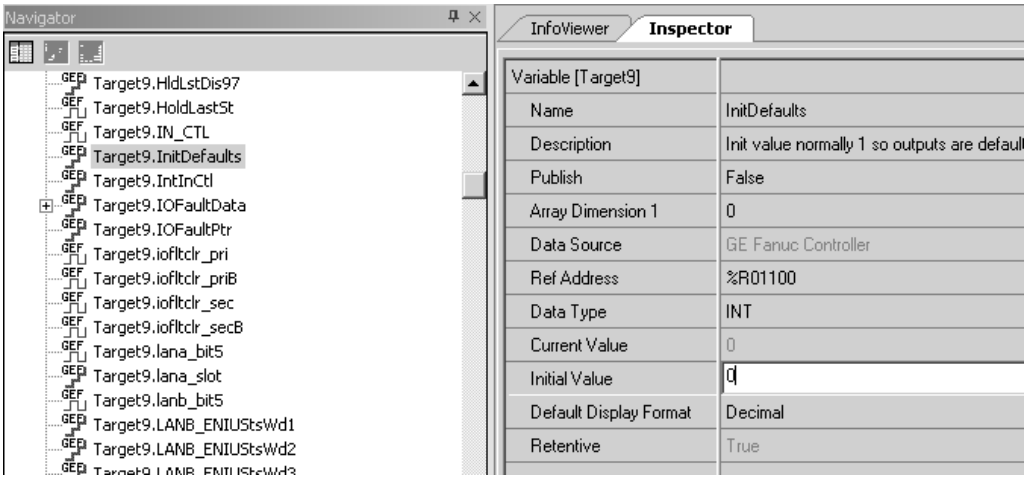
After the Ethernet NIU switches to a different LAN and / or controller, it will keep using the new LAN and controller until it receives a command from the A controller (in the control data portion of the output message) telling it to switch. If it does not receive a switch command from controller A, it continues using the current LAN and controller until communication is lost.

Either controller can also command the Ethernet NIU to switch to any combination of LAN and controllers. If the commanded combination is not available, the Ethernet NIU will NOT switch.

Ethernet NIU Operation if No Data is Received

If the Ethernet NIU does not receive output and control data from either controller within the configured timeout period, the Ethernet NIU 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 chapter).

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



ENIU Operation if Data is Received from the Backup Controller

In systems with redundant controllers, the Ethernet NIUs can receive outputs from either the active or backup controller. The best choice is to have all Ethernet NIUs receive their outputs from the same controller and on a dual LAN system, from the same LAN. Command bits can be sent to the Ethernet NIUs to direct them to use a specific controller and LAN if that communication path is functioning.

If not all the communication paths from the controllers to the Ethernet NIUs are functioning, it is possible for some Ethernet NIUs to be receiving outputs from LAN A and some from LAN B. It is also possible for some Ethernet NIUs to be receiving outputs from the active controller and other Ethernet NIUs to be receiving outputs from the backup controller.

Outputs in a redundant controller system are solved in the active controller and then transferred to the backup controller using the RMX link. If an Ethernet NIU can only communicate with the backup controller and cannot communicate with the active controller,

the outputs the Ethernet NIU receives from the backup controller are from a solution where the controller does not get input updates from the Ethernet NIU. Note: if the active controller can communicate with the Ethernet NIU, the outputs are solved with inputs from the Ethernet NIU.

Selecting Outputs from the Active Redundant Controller Only Mode

Release 1.40 adds the ability to configure a high-availability system with redundant controllers so that the Ethernet NIU will update output points with values received from the active controller only. If communications with the active controller are lost, the Ethernet NIU will set outputs to a commanded state (Hold Last State, zero, or defaults).

By default, the Ethernet NIU will continue to update output points from the back-up controller if communications with the active controller are lost. To enable this option set the initial value of the variable *enableoutputsfromactiveonly* to 1 in the Ethernet NU. It is also possible to modify the value of the variable *ApplyOutputsFromBackupUnit* at run-time using logic in the Ethernet NIU or from a programmer. In that case, the behavior in the Ethernet NIU is dependent on the current value of the *ApplyOutputsFromBackupUnit*.

These variables are only meaningful in applications with redundant controllers. In other applications, this variable has no effect on operation.

Caution

Placing the active controller in run disabled mode with this feature enabled on an Ethernet NIU will cause that NIU to set its outputs to the commanded state. In this case the active controller is no longer sending outputs to the NIU and it is configured not to accept outputs from the backup controller.

By default, a loss of communication between the active controller and Ethernet NIUs does not cause a switchover to the backup controller. Therefore in a single LAN configuration with the new outputs from active controller only option enabled, a loss of communication between the active controller and any Ethernet NIU causes that Ethernet NIU to default its outputs unless a role switch is forced by the application logic. Similarly, a loss of communication between the active controller and a switch/ring causes all Ethernet NIUs to default their outputs unless a role switch is forced by the application logic. In a dual LAN system, both controllers must be lost for outputs to default.

In templates all outputs are transferred from the active controller to the backup controller so in this case the active controller is still controlling outputs. In this case the active controller would not be receiving inputs from the ENIU so the inputs would have went to the commanded state (Hold Last, zero, or defaults). If Point Fault references are enabled the point faults would be set for inputs from the ENIU as well.

Delay in Switching to Active Controller

If a role switch occurs with the outputs from active only option enabled, there is a delay before the Ethernet NIU sees outputs from the newly active controller. During this delay, the Ethernet NIU does not know that a switch in control has occurred. As long an Ethernet NIU still has communication to the backup controller, it does not default its outputs until a timeout period has passed. During the timeout period, the Ethernet NIU's outputs hold their last states. By default, the Ethernet NIU uses a timeout delay of 1 second when waiting for an update from the newly activated controller. This timeout value can be changed for applications that have a very long production period.

Loss of Synchronization

If the RMX link(s) are lost between two redundant controllers, the controllers become unsynchronized and both become active. An Ethernet NIU that was receiving outputs from both controllers continues to accept outputs from the controller that was active before synchronization was lost.

If the active controller fails after the controllers become unsynchronized, the Ethernet NIU starts accepting outputs from the former backup controller (which is now active).

In systems where the Ethernet NIU is configured to accept outputs from the active controller only, an Ethernet NIU that is not communicating with the active controller starts accepting outputs from the former backup controller as soon as synchronization is lost.

Compatibility

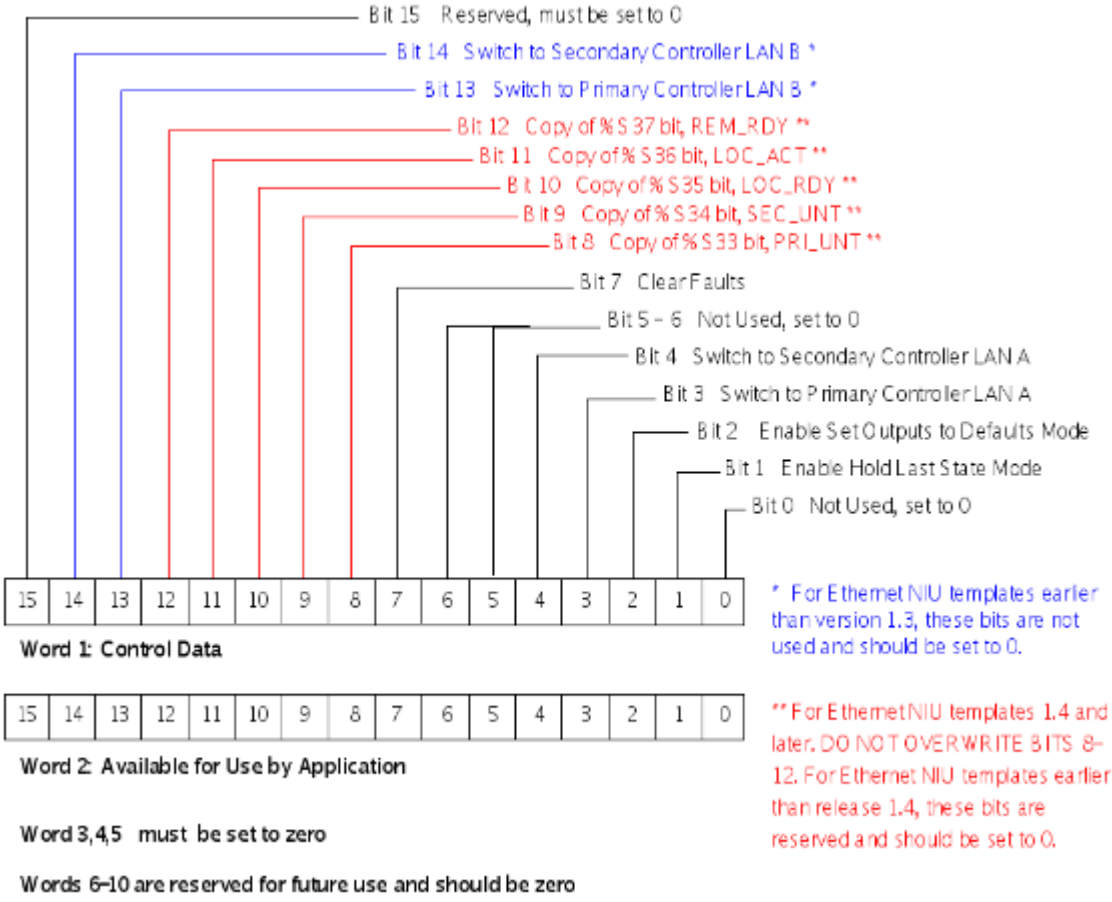
Operation of the outputs from active only feature requires specific application code in the Ethernet NIU and in the redundant controllers. In a system that uses this feature, it is important to be sure that the Ethernet NIUs and controllers can support it.

If the Ethernet NIU supports this feature but the controller application does not, there will not be any effect on the application. However, if the new feature is erroneously enabled in the Ethernet NIU (by setting the global variable *ApplyOutputsfromBackupUnit* to 1), the Ethernet NIU will not receive notification of which controller is active, and it will default its outputs.

Control Data Format

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

In addition, if there are two controllers, the control data can be used to determine which of them will supply the I/O Station outputs.



Control Data Definitions

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.

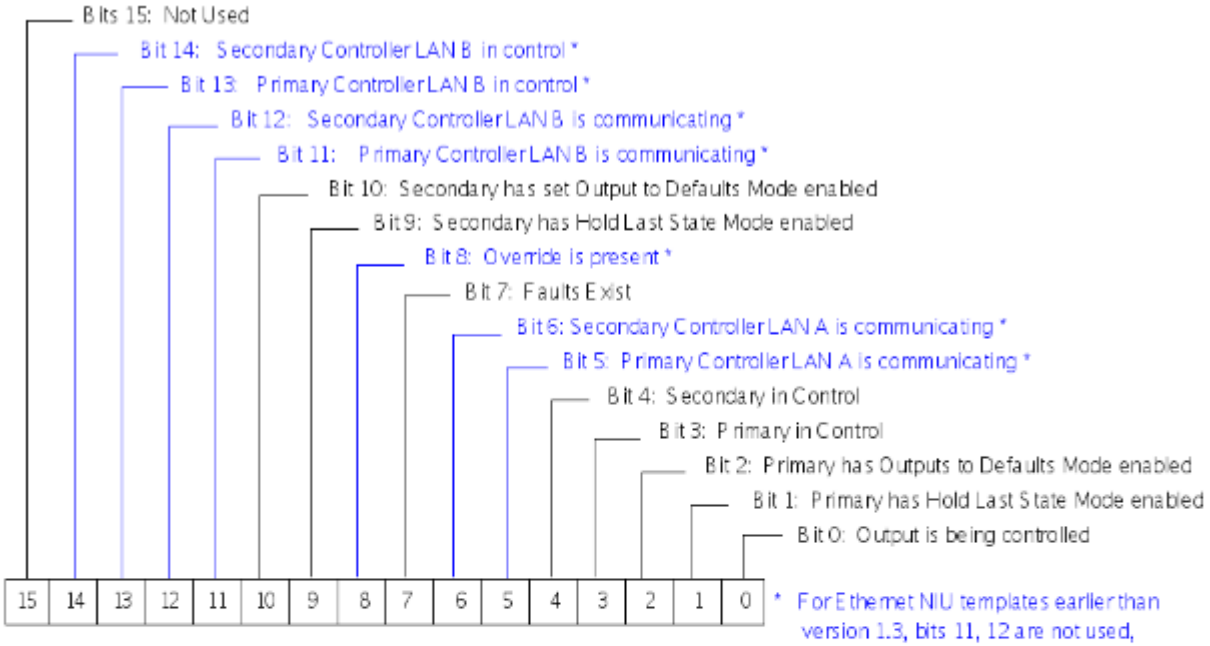
Bit	*Assignment*	*Using this Bit in the Application*
Word 1 Bit 1	Enable Hold Last State Mode:*	Set this bit if outputs in the I/O Station should hold their last commanded state when communications are lost. For systems with two controllers, this bit should be the same in both the primary and secondary controller exchanges.
Word 1 Bit 2	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.
Word 1 Bit 3	Switch to Primary Controller LAN A:	Setting this bit to 1 tells all ENIUs to switch to Primary Controller LAN A For Systems without Dual LANs this bit is used to switch to Primary Controller
Word 1 Bit 4	Switch to Secondary Controller LAN A:	Setting this bit to 1 tells all ENIUs to switch to Secondary Controller LAN A. For Systems without Dual LANs this bit is used to switch to Secondary Controller.
Word 1 Bit 7	Clear Faults:	This bit clears all faults in ALL Ethernet NIUs that receive the exchange. **The ENIUs clear faults when they see a rising edge on this bit.** In a system with two controllers, only the exchange from the currently-active controller is used to clear faults. In Version 132 and higher, faults can be cleared in individual ENIUs by using the ClearFault Word in the SVC_Exch to the ENIU.
Word 1 bits 8-12	Copies of %S Status bits	(For use with ENIU templates version 1.4x and later only; for earlier versions, these bits should be set to 0). The ENIU_Check block in the controller sets these bits to match the states of the CPU's %S bits that track the state of the redundant controllers. The application logic should NOT overwrite these bits. The Ethernet NIU can monitor these bits to identify the active controller, and to determine when control has moved from one controller to the other. When a role switch in the controllers occurs, the controllers see that the new controller is in control the next scan. The "Control by Standby Controller" bit uses feedback from the Ethernet NIUs, which does not occur until the next EGD exchange is received. Because of this, the "Control by Standby Controller" indication can momentarily flash On when a role switch occurs.
Word 1 Bit 13	Switch to Primary Controller LAN B:	For Ethernet NIU templates 1.3x and later, setting this bit to 1 tells all ENIUs to switch to Primary Controller LAN B. For Ethernet NIU templates earlier than version 1.3x, this bit is not used and should be set to 0.

Bit	Assignment	Using this Bit in the Application
Word 1 Bit 14	Switch to Secondary Controller LAN B:	For Ethernet NIU templates 1.3x and later, setting this bit to 1 tells all Ethernet NIUs to switch to Secondary Controller LAN B. For Ethernet NIU templates earlier than version 1.3x, this bit is not used and should be set to 0.
Word 2	Word 2, Available to Application:	The application program in the controller(s) can optionally use word 2 as described later in this section.
Word 8	Exchange Sequence Number	This word contains a sequence number. The Ethernet NIU increments the sequence number every scan. The range of the sequence number is 1 to 32000.
Word 10	ENIU Version	This word is reserved for an Ethernet NIU version number.

* See the section on setting up the output defaults.

Status Data Format

The 10 words 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 every Ethernet NIU.



Word 1: Status and Fault Data

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

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

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

Word 4: Copy of Control Data Word 2 (Application Based) from Primary Controller LAN B

Word 5: Copy of Control Data Word 2 (Application Based) from Secondary Controller LAN B

Words 6, 7: Reserved for Future use

Word 8: Controller Sequence Number

Word 9: Reserved for Future use

Word 10: Used for Ethernet NIU Revision Control

Note If a CPU is in Run-I/O Disabled mode, it no longer sends valid outputs to the Ethernet NIU, so it is no longer communicating with the Ethernet NIU. However, the Ethernet NIU continues to provide valid input data to the CPU.

Status Data Definitions

Bits	Assignment	Using the Bits in the Application
Word 1 Bit 0	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.
Word 1 Bit 1 & 9	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
Word 1 Bit 2 & 10	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.
Word 1 Bit 3	Primary LANA in Control	Set when the Primary controller LAN A is presently controlling the ENIU and providing output data for the I/O Station. If this bit is set, bit 4 (Secondary in control) should NOT be set.
Word 1 Bit 4	Secondary LANA in Control	Set when the Secondary controller LAN A is presently controlling the ENIU and providing output data for in the I/O Station. If this bit is set, bit 3 (Primary in Control) should NOT be set.
Word 1 Bit 5	Primary LAN A is Comm	LAN is Communicating (EGD Input Exchange Status variable (in ENIU) is a value of 1, 3, or 5)
Word 1 Bit 6	Secondary LAN A is Comm	LAN is Communicating: (EGD Input Exchange Status variable (in ENIU) is a value of 1, 3, or 5)
Word 1 Bit 7	Faults Exist	Set when any fault exists in the ENIU.
Word 1 Bit 8	Override is Present	Set when an override is present.
Word 1 Bit 9	Secondary 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
Word 1 Bit 10	Secondary 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.
Word 1 Bit 11	Primary LAN B is Comm	LAN is Communicating: (EGD Input Exchange Status variable (in ENIU) is a value of 1,3 or 5)
Word 1 Bit 12	Secondary LAN B is Comm	LAN is Communicating: (EGD Input Exchange Status variable (in ENIU) is a value of 1,3 or 5)
Word 1 Bit 13	Primary LAN B in Control	Set when the Primary controller LAN B is presently controlling the ENIU and providing output data for the I/O Station.
Word 1 Bit 14	Secondary LAN B in Control	Set when the Secondary controller LAN B is presently controlling the ENIU and providing output data for in the I/O Station.

Bits	*Assignment*	*Using the Bits in the Application*
Words 2-5	Words 2, 3,4 & 5 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. For Dual LAN systems, words 4 and 5 will have content
Word 8	Controller Sequence Number	
Word 10	Controller Version Number	

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.

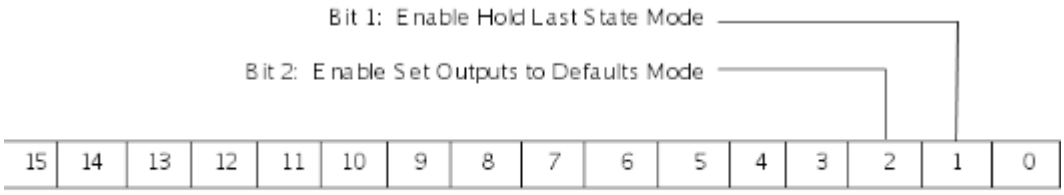
Switching Control Back to the Primary Controller

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 or bit 13 (if LAN B is used) 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 and bit 13 reset.
2. Synchronize the program state with data from the secondary controller.
3. Set output bit 3 (Switch to Primary Controller) or bit 13 (if LAN B is used) of the data going to the Ethernet NIU.

Commanding Output Operation if Communication is Lost

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 Ethernet NIU will hold outputs at their last commanded states. If control bit 2 is set to 1, the Ethernet NIU will set outputs to their individual default states (see below). Bit 2 takes precedence; if both bits 1 and 2 are inadvertently set, the Ethernet NIU 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 and dual LANs, 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. The values sent by all controllers can be monitored using Status Words 2-5 which echo back the Command Word 1 from the four communication channels.

NOTE: If an output module loses communication with the Ethernet NIU, the outputs for that module ONLY are set to the module's default state, regardless of the commanded output state from the controller. See GFK-2314 *PACSystems RX3i System Manual* for more information regarding output modules that support default states when the output module loses communication.

Specifying Individual Output Defaults

If the control outputs are set to default instead of hold their last state, all outputs usually default to zero. If that is suitable for the application, no further action is needed.

For some applications, taking outputs to a safe state requires setting discrete outputs to On, or forcing analog outputs to individually-specified values. To establish output defaults for applications where defaults are needed, follow these steps:

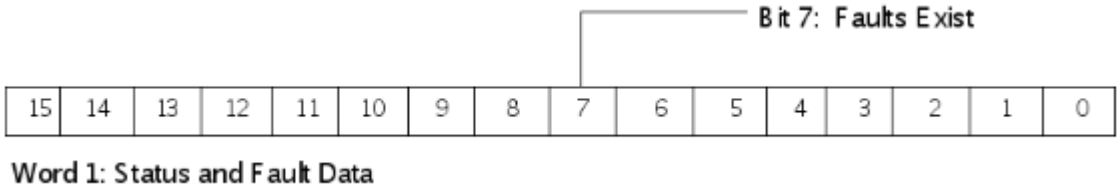
1. In the programmer, select the Variable Table and locate the Ethernet NIU variables. The variables are in a table of the form <Devicename><variable name>
2. In the Ethernet NIU section, locate the output (Qxxxx or AQxxxx) that is to be given a default value. If there is no variable with the current 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-clicking.
3. When creating output variables (Qxxxx), set the Retentive property to True. Otherwise, the default value will not be stored properly.
4. Do not execute the command to delete unused variable as this will delete the 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 memory. Default values are loaded into a holding buffer from flash when the Ethernet NIU starts up.

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. Fault handling operation is different for version 1.3x 2x or later of the Ethernet NIU target in the programmer. Both versions should not be used at the same time.

Checking Faults in the Input Status Data

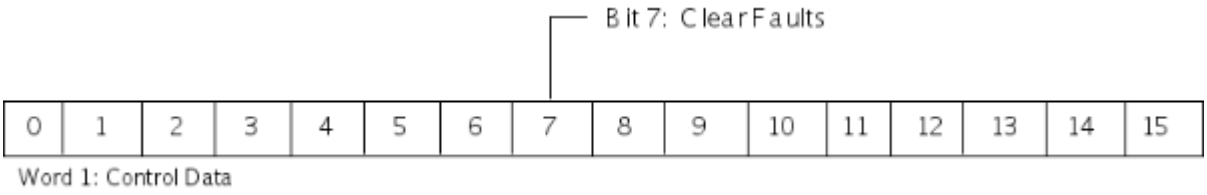
To use the Input Status Data for fault handling, the application program in the controller(s) should monitor Ethernet NIU status word 1 bit 7 to check for faults:



Word 1: Status and Fault Data

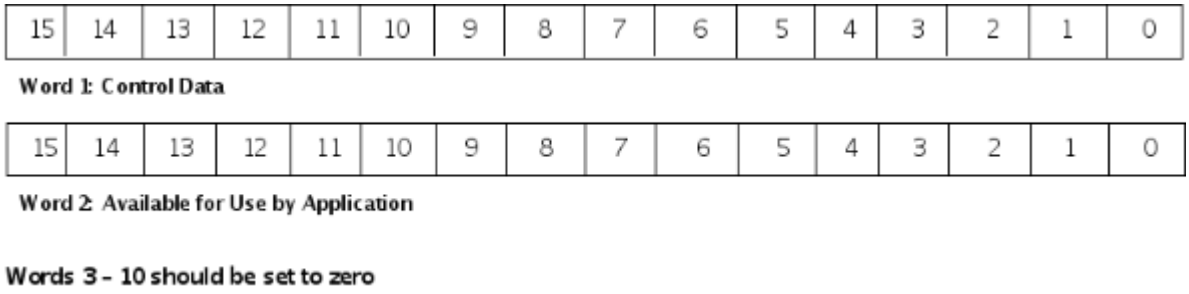
Clearing Faults in the Output Control Data

The controller can set bit 7 in the Ethernet NIU Output Control Data to clear faults in ALL Ethernet NIUs that receive the *Outputs_to_ENIUs* exchange.



Using the Optional Application-Specific Command Word

Word 2 of the command data can be used by the controller for several purposes, as described below.



Setting Up a Heartbeat

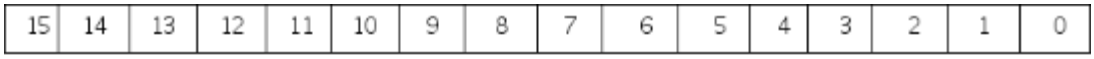
For example, the controller could use a free-running counter as a heartbeat for the value of this word, then check the incoming Ethernet NIU status block to make sure the ENIU is still running. In redundancy applications, each controller could check the other controller's heartbeat to determine whether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

Sequencing Outputs

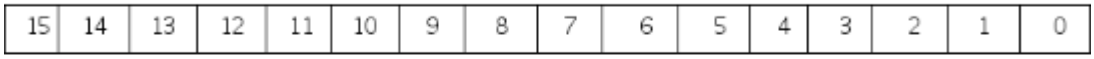
This word could also be used to sequence outputs. The controller would set the outputs to a particular state and set the sequence number in the command data. When the Ethernet NIU returns the same sequence number in its status data, the controller knows that the Ethernet NIU has received the outputs.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

Checking the Status of the Heartbeat / Sequence

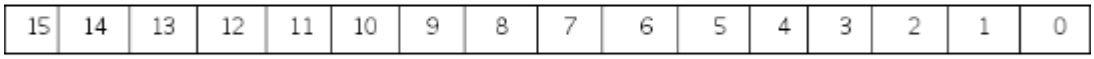
The primary controller's heartbeat/sequence ID word is returned in the second word of the Ethernet NIU status data. The secondary controller's heartbeat/sequence ID word is returned in the third word of the Ethernet NIU status data.



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 2 (Application-based) from Secondary Controller

Word 4: Copy of Control Data Word 2 (Application based) from Primary Controller LAN B

Word 5: Copy of Control Data Word 2 (Application based) from Secondary Controller LAN B

Words 6 – 10: Reserved

Chapter 10 Diagnostics

This chapter describes:

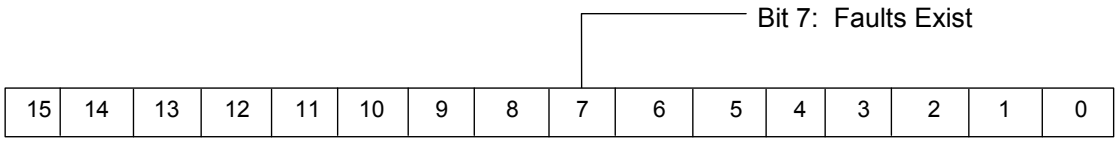
- Using the Status and Control Data for Fault Monitoring
  - Checking Faults in the Input Status Data
  - Clearing Faults in the Output Control Data
- Viewing the Fault Tables
- Viewing Faults in the Controller PLC Fault Table
- Viewing Ethernet NIU Faults in the Ethernet NIU Fault Table
- Enhanced Fault Handling
  - Disabling Enhanced Fault Handling
  - The *ENIU_Faults* C Block
  - Clearing Faults using *SVC_Xchg_to_ENIU_xx*
  - Symbolic Variables Used in Fault Handling
- Using the Station Manager
  - Checking the IP Address of the Ethernet NIU
  - Checking communications on the network
  - Viewing the Exception Log
  - Checking for Stale Ethernet Global Data Status
  - Checking Exchanges with the Stat Command
- Testing Communications after Setup
  - Verifying that ECG Exchanges are Working
  - Checking the Cable Connections
- Troubleshooting Ethernet I/O
- Checking Communications in the Programmer Watch Windows
- What to do if you can’t solve the problem

Using the Status and Control Data for Fault Monitoring

The regular exchange of status and control data described in chapter 9 provides the controller with the ability to check for fault conditions and clear faults. For applications created using version 1.3x or later of the Ethernet NIU target in the programmer, if the enhanced diagnostics feature is used to monitor and clear faults, the status and control data should *not* also be used for fault handling in the controller(s) at the same time.

Checking Faults in the Input Status Data

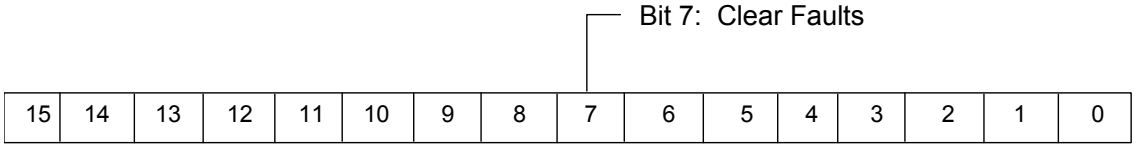
To use the Input Status Data for fault handling, the application program in the controller(s) should monitor Ethernet NIU status word 1 bit 7 to check for faults:



Word 1: Status and Fault Data

Clearing Faults in the Output Control Data

The controller can set bit 7 in the Ethernet NIU Output Control Data to clear faults in ALL Ethernet NIUs that receive the *Outputs_to_ENIUs* exchange.



Word 1: Control Data

The Ethernet NIU Fault Tables

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

The *PLC Fault Table* lists faults associated with the Ethernet NIU itself. Fault descriptions are listed on the next page of this chapter. For each fault in the table, see the *User Action* column for information about correcting the fault. At powerup and at Stop to Run transitions of an Ethernet NIU, the NIU logs a message into its internal PLC Fault Table indicating that the event has occurred. In addition, the NIU retransmits the contents of its fault tables to the controller. If the Ethernet NIU's internal PLC Fault Table has more than 16 faults stored, only the FIRST 8 and LAST 8 faults from the PLC Fault Table are retransmitted to the controller.

The *I/O Fault Table* lists faults associated with modules in the I/O Station, such as the loss or addition of a module. If the I/O Fault Table has more than 32 faults stored, only the FIRST 16 and LAST 16 faults from the I/O Fault Table are transmitted to the controller.

Viewing Faults in the Controller PLC Fault Table

Like other faults from the Ethernet NIU, PLC and I/O Fault Table faults are logged into the controller's PLC Fault Table as application messages. A fault in the controller PLC fault table from an Ethernet NIU looks like this:

Application Msg (ENIU #) - <fault text, 24 characters>

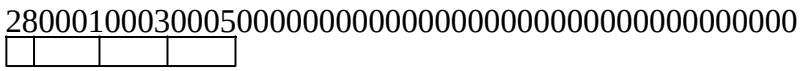
For example, a Loss of an IO Module in rack 0 slot 7 from Ethernet NIU 02 would produce the following message:

Application Msg (2) - 0.7: Loss of Module

Any fatal faults stop the Ethernet NIU. Because the Ethernet NIU has stopped, the fatal faults are not sent to the controller. Fatal faults can be viewed by connecting the programmer to the Ethernet NIU and viewing the faults before the Ethernet NIU is restarted.

Viewing Extra Fault Data in the Ethernet NIU's PLC Fault Table

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



The Station Manager utility can be used to view more detailed information about specific faults, as explained later in this chapter.

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 Log Event 28, Entry 2=1, Entry 3=3, and Entry 4=5.

Ethernet NIU 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.
Configured gateway address 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.

Enhanced Fault Handling

Version 1.3x and later of the Ethernet NIU target provide the enhanced fault handling features described below:

- ENIU non-fatal faults are sent to a PACSystems controller in the *SVC_Xchg_from_ENIU_xx* exchange.
- The *ENIU_Faults* C block in the controller puts the faults in the controller's PLC Fault Table. The *ENIU_Faults* C block also writes the complete fault information including Fault Extra Data to a buffer in the controller.
- The controller can clear faults in an individual Ethernet NIU by sending commands in the ClearFaults data range of the *SVC_Xchg_to_ENIU_xx* exchange.

If enhanced fault handling is used in the application, fault handling using the fault tables should not be used.

Disabling Enhanced Fault Handling

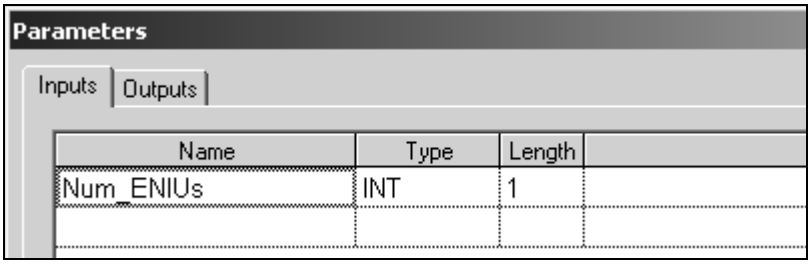
Enhanced fault handling can be disabled if it is not needed for the application. Disabling enhanced fault handling and deleting its SVC exchanges can improve the overall Ethernet Global Data exchange time. See chapter 8 for details. To disable enhanced fault handling, set the initial value of the Ethernet NIU variable *enable_ph1_flts to* 1 and store the program to each Ethernet NIU that should have enhanced fault handling disabled.

If the SVC exchanges have not been deleted from version 1.3x or later EGD templates, enhanced fault handling can be re-enabled. To re-enable enhanced fault handling, change the initial value of *enable_ph1_flts* to 0 and store the program to each Ethernet NIU that should have enhanced fault handling re-enabled.

The ENIU_Faults C Block

The C block *ENIU_Faults* is required in PACSystems controller to enhanced fault handling. This block is present and configured if the project templates are used.

If the C block is added to the controller the parameter for the block must be configured as described here. The input parameter specifies the number of Ethernet NIUs (number 1 to number of Ethernet NIUs) for fault processing.



When the C block is called the input should be a constant that is the number of Ethernet NIUs in the system. If the input is left blank, a fault is generated in the controller's PLC Fault Table.

The block will only process faults from the number of Ethernet NIUs specified by this input. If the number specified here is greater than the number of Ethernet NIUs that are present, a fault is generated in the PLC Fault Table for each missing Ethernet NIU.

Fault Data Circular Buffer

The *ENIU_Faults* C block places fault data from the Ethernet NIUs into circular buffers. Each Ethernet NIU has a separate circular buffer. The buffers are in symbolic variable arrays named *fltbuf_eniu_xx*, where xx is the Ethernet NIU number. The symbolic variable *fltptr_eniu_xx* points to the first word of the latest 24-word entry in the fault buffer. By subtracting 24 from the pointer, the next-oldest entry can be read. The array for each Ethernet NIU is 1152 words in length.

Clearing Faults Using SVC_Xchg_to_ENIU_xx

Faults in an individual ENIU can be cleared by placing a value of 1 or 2 in the variable *ClearFaults_ENIUxx* in the controller. xx is the Ethernet NIU number. Placing a value of 1 in this variable will clear the Ethernet NIU PLC Fault Table. Placing a value of 2 in this variable will clear the Ethernet NIU IO Fault Table. Clearing is a one-shot operation. After the clear is done, set the *ClearFaults_ENIUxx* variable to 0 for long enough for the Ethernet NIU to receive it before another clear is done.

Clearing the fault table in the controller does NOT clear the fault tables in the Ethernet NIU(s).

Symbolic Variables for Fault Handling

Specific symbolic variables must be used in the controller for the *SVC_Xchg* Ethernet Global Data exchanges, or the enhanced fault-handling feature will not log faults.

These symbolic variables must be declared as variables in the controller and published either internally or externally. Otherwise, either the controller program will not store to the PLC or the PLC will log a fault when it attempts to go to Run mode.

The project templates described in this manual will automatically declare the variables in the controller and set up the *SVC_Xchgs* with the correct variables.

If the controller is not set up using a template, the variable file *ENIU_Faults_1xx_variables.csv* should be imported into the controller to create the symbolic variables and the *SVC_Xchgs* will need to be set up as described in chapter 5. Note: In the filename, 1xx represents release 1.20, 1.21, 1.33 or 1.40 of the Ethernet NIU application project.

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.

For Ethernet NIU systems with dual LANs, each Ethernet interface has a separate Station Manager and both Station Managers must be accessed separately.

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/support.

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.

Checking Communications on the Network

During system setup, use the Station Manager utility to test each installed Ethernet device to be sure it is operating 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 for 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 Ethernet NIU 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.

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
- 03H indicates that the Ethernet Interface in the producer is configured for network time synchronization, but is not synchronized to an SNTP server. The data was refreshed on schedule.
- 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 the Station Manager.

Type Xchange <producer ID> <exchange ID> or Xchange <Ndx>

The data ranges in each exchange can be viewed using the *EGDREAD* command in the Station Manager.

Type EGDREAD <producer ID> <exchange ID> or EGDRead <Ndx>

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.

Testing Communications after Setup

After completing the configuration and other steps necessary to set up the system, it is important to make sure that communications are working well. Some guidelines are given below.

Verifying that All Ethernet Global Data Exchanges are Working

1. One at a time, connect a Station Manager to each Ethernet port in the controller(s) and each Ethernet NIU.
2. For each connection, execute the *stat g* command in Station Manager to be sure all EGD Exchanges are working properly.

Stat g should return a display like the one illustrated below, which shows *stat g* response data from ENIU_01 in a system with two controllers.

Each line of the response represents one exchange.

The State column shows whether an exchange is working. The state should be Active and the value should be 00H, 01H, 03H, or 05H if the exchange is working. If the value is 06H or 07H, the exchange is not being received or is taking too long to be received. If the state is 0eH, the size of the exchange in bytes at the producing and receiving ends is not the same and must be fixed.

```
> stat g
<<< EGD Status >>>   22-MAY-2000  17:23:58.0

                                                                  Transfers
Ndx  Producer ID      Exchange ID  Mode      State               Completed
---  ---------------  -----------  --------  ---------------     ---------
 0H     10.10.10.101          101  CONSUMER      ACTIVE(00H)      1712567
 1H     10.10.10.101         1101  CONSUMER      ACTIVE(00H)      1712579
 2H     10.10.10.101            1  CONSUMER      ACTIVE(00H)     12015851
 3H     10.10.10.101         1001  CONSUMER      ACTIVE(00H)     12052804
 4H       10.10.10.1          101  PRODUCER      ACTIVE(01H)      1712087
 5H       10.10.10.1            1  PRODUCER      ACTIVE(01H)      8560434
```

EGD Exchanges in a Single Controller System

In a single controllers system, each Ethernet NIU should have the following exchanges (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.xx; ExchangeID 1 – Inputs from ENIUxx

Produced Exchange – ProducerID 10.10.10.xx; ExchangeID 1xx – SVC from ENIUxx

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1 – Outputs Pri to ENIUs

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1001 – Outputs Sec to ENIUs

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1xx – SVC Pri to ENIUs

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 11xx – SVC Sec to ENIUs

EGD Exchanges in a Dual Controller, Single LAN System

In a dual controller system single LAN, the primary controller should have the following exchanges (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1 – Outputs to ENIUxx

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1xx – SVC to ENIUxx (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1 – Inputs from ENIUxx (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1xx – SVC from ENIUxx (one per ENIU)

The secondary controller should have the following exchanges (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1001 – Outputs to ENIUxx

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 11xx – SVC to ENIUxx (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1 – Inputs from ENIUxx (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1xx – SVC from ENIUxx (one per ENIU)

EGD Exchanges in a Dual Controller, Dual LAN System

Each Ethernet NIU should have the following exchanges on LAN A (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.xx; ExchangeID 1 – Inputs_from_ENIU_xx

Produced Exchange – ProducerID 10.10.10.xx; ExchangeID 1xx – SVC_Xchg_from_ENIU_xx

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1 – Outputs_Pri_to_ENIUs

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1001 – Outputs_Sec_to_ENIUs

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1xx – SVC_Xchg_Pri_to_ENIU_xx

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 11xx – SVC_Xchg_Sec_to_ENIU_xx

Each Ethernet NIU should have the following exchanges on LAN B (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.xx; ExchangeID 3 – Inputs_from_ENIU_xx_LANB

Produced Exchange – ProducerID 10.10.10.xx; ExchangeID 2xx – SVC_Xchg_from_ENIU_xx_LANB

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 3 – Outputs_Pri_to_ENIUs_LANB

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 1003 – Outputs_Sec_to_ENIUs_LANB

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 2xx – SVC_Xchg_Pri_to_ENIU_xx_LANB

Consumed Exchange – ProducerID 10.10.10.101; ExchangeID 12xx – SVC_Xchg_Sec_to_ENIU_xx_LANB

The Primary redundant controller should have the following exchanges on LAN A (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1 – Outputs_to_ENIUs

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1xx – SVC_Xchg_to ENIU_xx (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1 – Inputs_from_ENIU_xx (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1xx – SVC_Xchg_from_ENIU_xx (one per ENIU)

The Primary redundant controller should have the following exchanges on LAN B (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 3 – Outputs_to_ENIUs_LANB

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 2xx – SVC_Xchg_to_ENIU_xx_LANB (one per ENIU

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 3 – Inputs_from_ENIU_xx_LANB (one per ENIU

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 2xx – SVC_Xchg_from_ENIU_xx_LANB (one per ENIU)

The Secondary redundant controller should have the following exchanges on LAN A (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1001 – Outputs_to_ENIUs

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 11xx – SVC_Xchg_to_ENIU_xx (one per ENIU

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1 – Inputs_from_ENIU_xx (one per ENIU

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 1xx – SVC_Xchg_from_ENIU_xx (one per ENIU)

The Secondary redundant controller should have the following exchanges on LAN B (xx=ENIU#)

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 1003 – Outputs_to_ENIUs_ LANB

Produced Exchange – ProducerID 10.10.10.101; ExchangeID 12xx – SVC_Xchg_to_ENIU_xx_LANB (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 3 – Inputs_from_ENIU_xx_LANB (one per ENIU)

Consumed Exchange – ProducerID 10.10.10.xx; ExchangeID 2xx – SVC_Xchg_from_ENIU_xx_LANB (one per ENIU)

Outputs %Q001 to %Q2048 are sent from both the controllers to all Ethernet NIUs over both LANs.

Analog outputs %AQ001 to %AQ512 are sent from both the controllers to all the Ethernet NIUs over both LANs.

Checking the Cable Connections

Verification should be done on the communication path between each controller and the Ethernet NIUs. Put controller(s) in Run mode. Disconnect the Ethernet cable at each controller. One at a time, connect the Ethernet cables. For each cable connection:

- In the controller, turn one (or more) discrete outputs on and off. Verify that the outputs come on in all Ethernet NIUs. This can be done by using Reference View tables in the controller to control the outputs, and viewing the outputs in the Ethernet NIU's Reference View table. Do the same with analog outputs.
- Repeat the same steps to check that the configured inputs from the Ethernet NIU are being correctly received by the controller(s). Each Ethernet NIU will be sending a different range of input to the controller(s).

Troubleshooting Ethernet I/O

If discrete outputs and analog outputs are not changing in the Ethernet NIU, check the following when they are changed in the controller:

- Make sure both the controller and Ethernet NIU are in Run mode.
- Make sure the controller and Ethernet NIU are both connected to the Ethernet I/O LAN.
- Verify that both controller and Ethernet NIU are sending and receiving Ethernet Global Data exchanges without error. This can be checked by using Station Manager on the Ethernet ports and issuing a *stat g* command. This should be done at the controller and at each ENIU.
- If discrete inputs and analog inputs are not changing in the controller(s) when they are changed in the Ethernet NIU station, check to see that both controller and the Ethernet NIU are sending and receiving Ethernet Global Data exchanges without error. This can be done using the Station Manager on the Ethernet ports and issuing a *stat g* command. This should be done at each controller and at each Ethernet NIU.

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 a hardware problem. If this occurs, follow the procedure below.

1. Make sure 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. *TALLY L* 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. Make sure 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 Communications in the Programmer Watch Windows

Communication between devices can be checked in the Ethernet Global Data Status word of an EGD Exchange. The status of the input exchanges (to the controller) can be checked via the status word of the Consumed EGD Exchanges in the controller. The status of the output exchanges (to the Ethernet NIU) can be checked via the status of the Consumed EGD Exchanges in the ENIU.

Checking the Ethernet Global Data Status

Watch Windows in the controller and Ethernet NIU projects can be used to view the communications status variables. For an exchange to be properly configured, the Producer ID, the Exchange Number and the Number of Bytes in the exchange must match in the Ethernet Global Data exchange setup in both the controller and the Ethernet NIU.

EGD Status Watch Windows in the Controller Project

InputsExchangeStatus (Inputs from ENIU)

InEx_Status_LANA_ENIU_01

InEx_Status_LANB_ENIU_01

Variables for additional Ethernet NIUs can be added as needed.

SVCInputExchangeStatus (SVC data from ENIU)

SVC_In_Status_LANA_ENIU_01

SVC_In_Status_LANB_ENIU_01

Variables for additional Ethernet NIUs can be added as needed.

EGD Status Watch Windows in the Ethernet NIU Project

EGDStatus (Outputs from Controllers and SVC Requests from Controllers)

OutputsEx_Status_Pri_LANA

OutputsEx_Status_Pri_LANB

OutputsEx_Status_Sec_LANA

OutputsEx_Status_Sec_LANB

SVCEx_Status_Pri_LANA

SVCEx_Status_Pri_LANB

SVCEx_Status_Sec_LANA

SVCEx_Status_Sec_LANB

LAN B variables only exist in projects that use dual LANs.

Exchange Status Values

Typical values and descriptions of an EGD status variable are:

0 - Data has not been consumed, exchange is not setup or devices are not communicating

1 - Exchange working properly (data has been consumed)

3 - Exchange working properly, but Network Time Sync enabled and ENIU is not time synced.

5 - Stale data, data was consumed, but the producer has not updated it since the last consumption

6 - Timeout in exchange

7 – Data received after timeout

e – Mismatch in size of EGD exchange between Controller and ENIU.

The variables for the Output Exchanges from controllers will toggle between the values of 1 and 5, because the Ethernet NIU scans faster than the controller produces data. That means the controller has not always refreshed the data at the time of consumption. This is normal and to be expected.

Please refer to the *TCP/IP Ethernet Communications for PACSystems Manual,* GFK-2224, for a detailed discussion of EGD Status values, and troubleshooting tips

Watching Ethernet NIU Status with the Controller

Core to the system is the status of the Ethernet NIUs as viewed in the controller. Each I/O Input Exchange from an Ethernet NIU has ten status words indicating the status of the Ethernet NIU (see chapter 9 for more information). Status words are transmitted on both LAN A and LAN B. Typically, when both LANs are functional, the values for the status in the LAN A and LAN B status words are the same. There is a Watch Window displaying the first status word for both LAN A and LAN B for Ethernet NIU number 1. Status words for additional Ethernet NIUs can be added as needed. The status words are only current if the Ethernet Global Data exchange is healthy. The health can be determined by the EGD Status word of the Exchange as described above.

Ethernet NIU Status Watch Window in the Controller Project

ENIU_Status_Words (Status words from the Ethernet NIU(s))

StatusWords_LANA_ENIU_01[0] - the first word of the status of Ethernet NIU 1 via LAN A

StatusWords_LANB_ENIU_01[0] - the first word of the status of Ethernet NIU 1 via LAN B

Variables can be added for additional Ethernet NIUs as needed.

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 11 Local Program Logic in the Ethernet NIU

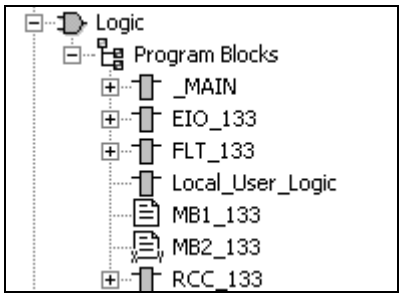
This section describes the Local Logic feature of the Ethernet NIU.

- 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 can execute up to 20K bytes of logic locally in the I/O Station.

When the RX3i Ethernet NIU target is created in the programmer, an empty LD logic block named *Local_User_Logic* is created that is called from *Main*. The logic program blocks in the Ethernet NIU target are named with the target version number as the last three characters of the block names. For example:



This block can be changed to ST or FBD by deleting the LD block *Local_User_Logic* and the creating a new block with the type of ST or FBD.

NOTE: Even if *Local_User_Logic* is not used, a block with that name MUST be present 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 both require the specific reference table addresses in the Ethernet NIU. These addresses MUST NOT be written to by *Local_User_Logic*:

%R00001 to %R09999

%R10000 to %R11024 *

%M00001 to %M04096

%M10001 to %M14096 *

%T0001 to %T0512

* For applications using version 1.3x or later of the PACSystems RX3i Ethernet NIU target application. These references are not restricted for earlier versions of the Ethernet NIU target application.

Addresses Written to by EGD Exchanges

The Ethernet Global Data exchanges *Outputs_Pri _to_ENIU*, *Outputs_Sec_to_ENIU*, *Outputs_Pri_to_ENIU_LANB*, and *Outputs_Sec_to_ENIU_LANB* write to the restricted %M and %R addresses listed above. The Ethernet NIU then writes to the following addresses (listed below) every scan of the Ethernet NIU:

%Q00001 to %Q02048

%AQ0001 to %AQ0512

If Local User Logic writes to these addresses, the Ethernet NIU functionality overwrites the values and it will look like the Local User 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 take into account.

1. A COMMREQ in *Local_User_Logic* 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 Ethernet NIU. The COMMREQ will put the data in the Ethernet NIU memory.

   If the result of a COMMREQ in the local logic must be provided to the controller(s) associated with the Ethernet NIU, it 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 also 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 command.

Chapter 12 Remote COMMREQ Calls

This chapter describes the Remote COMMREQ Call (RCC) feature that allows PACSystems RX7i and RX3i controllers to pass a predefined set of COMMREQs to intelligent 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
- Adding the RCC C Block to the Controller Target
- Adding Logic to Sequence RCC Commands and Check Return Status
- Monitoring Remote COMMREQ Calls for Completion
- Diagnostics for Remote COMMREQ Calls
- Remote COMMREQ Calls in a Redundancy System

The predefined COMMREQs that can be sent using Remote COMMREQ Calls are described in chapter 13. If an RX7i or RX3i controller needs to send the I/O Station a different type of COMMREQ, the Generic Remote COMMREQ Call feature can be used instead. See chapter 14 for details.

Using Remote COMMREQ Calls

The Remote COMMREQ Call (RCC) feature utilizes of a pair of Ethernet Global Data exchanges between the controller and an Ethernet NIU. These exchanges also move faults from the Ethernet NIU to the controller. One of the two exchanges sends standard COMMREQs from the controller to the Ethernet NIU. The other exchange sends the result of the COMMREQ from the RX3i Ethernet NIU back to the controller.

In a system with redundant controllers, there must be a Remote COMMREQ Call exchange from each controller to the Ethernet NIU. The Remote COMMREQ call exchange from the Ethernet NIU is sent to a group destination so that both controllers can receive it.

In Dual LAN systems, there is a pair of exchanges for each LAN.

Remote COMMREQ Call Functionality in the Ethernet NIU

Remote COMMREQ Call functionality is built into the Ethernet NIU target in the programmer. The Ethernet Global Data exchanges must be set up as described in this chapter. For applications using the Ethernet NIU templates, the EGD exchanges have already been set up with the needed information.

Remote COMMREQ Call Functionality in the Controller

Remote COMMREQ Call functionality is only available in PACSystem RX7i or RX3i controllers. The functionality is provided by the two EGD exchanges per ENIU and a parameterized C block.

- In version 1.2x of the controller target, the C block is named *RCCM_12x_yy.*
- In version 1.3x of the controller target, the C block is named *RCCD_13x_yy*. For applications using the Ethernet NIU templates, this C block is automatically included.
- In version 1.4x of the controller target, the C block is named *RCCD_14x_yy*. For applications using the Ethernet NIU templates, this C block is automatically included.

In the 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. It is expected that yy is the Ethernet NIU number. The controller needs a separate C block for each Ethernet NIU that will receive RCC commands. Each C block must have a unique name.

The C block drives the EGD exchange *SVC_Xchg_to_ENIU_xx* (or *RCC_Pri_request_to ENIU_xx* in a version 1.2x template application) that sends the COMMREQ to the Ethernet NIU. The Ethernet NIU sends the result back to the controller using the Ethernet Global Data exchange: *SVC_Xchg_from_ENIU_xx (RCC_response_from_ENIU_xx* for version 1.2x). For Dual LAN systems there are corresponding exchanges for LANB.

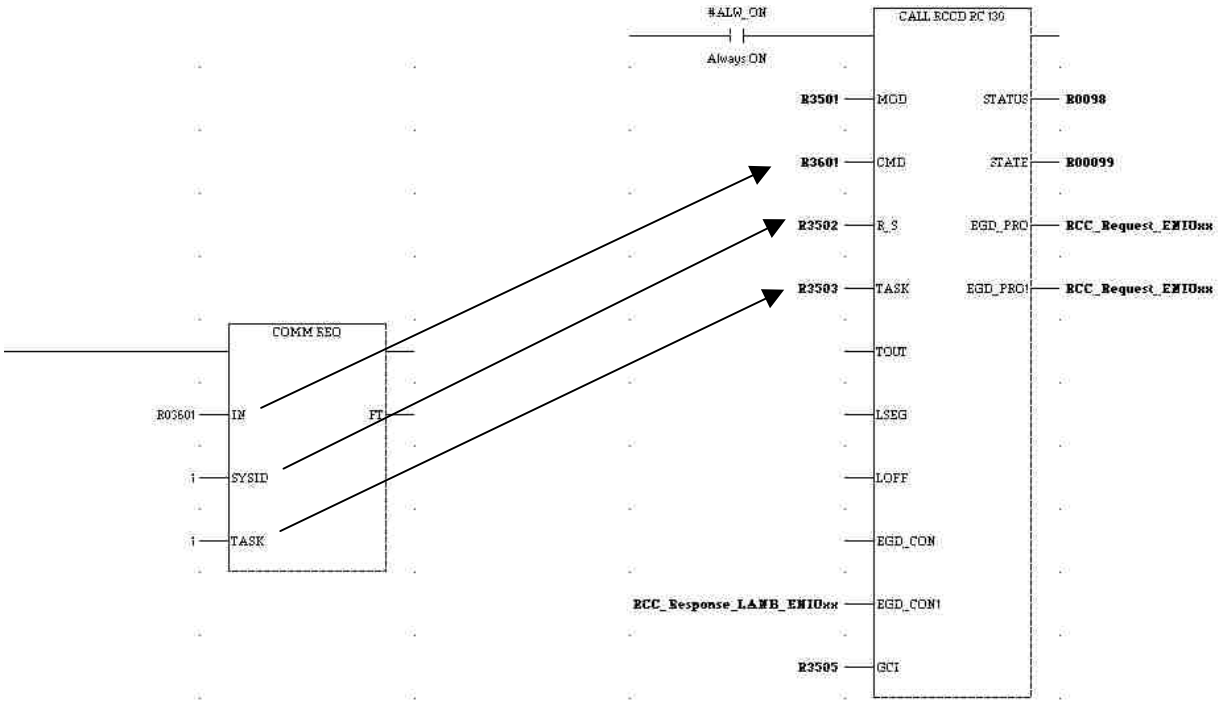
The C block in the controller puts the results into the status and data areas that are specified in the Remote COMMREQ request.

Inputs to the Remote COMMREQ Call in the Controller Logic

Like a COMMREQ instruction, the C block in the controller has input parameters. The application program in the controller must include logic to sequence the commands to the C block and to monitor the status for completion of the Remote COMMREQ Call.

As the picture below shows:

- The IN input of the COMMREQ becomes the CMD input of the Remote COMMREQ Call C block
- The SYSID input of the COMMREQ becomes the R_S input of the Remote COMMREQ Call.
- The TASK input of the COMMREQ is the TASK input 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.



Remote COMMREQ Call Operation

A PACSystems controller with the Remote COMMREQ Call C block sends COMMREQs to the Ethernet NIU in an Ethernet Global Data exchange named *SVC_Xchg_to_ENIU_xx* (for 1.2x templates, the exchange is named *RCC_Pri_request_to_EIU_xx*).

After receiving the Remote COMMREQ Call, the RX3i Ethernet NIU executes the COMMREQ. The Ethernet NIU then sends the result back to the controller in an Ethernet Global Data exchange named *SVC_Xchg_from_ENIU_xx* (for 1.2x templates the exchange is named *RCC_response_from_EIU_xx*). For Dual LAN systems there are corresponding exchanges for LAN B.

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 *SVC_Xchg_to_ENIU_xx* (for 1.2x templates, the exchange is named *RCC_Pri_request_to_ENIU_xx*) exchange and adds a sequence number for checking.
- Monitors for a response in the exchange SVC_Xchg_from_ENIU_xx , returns the COMMREQ Status Word, and returns data if a response is expected if 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.

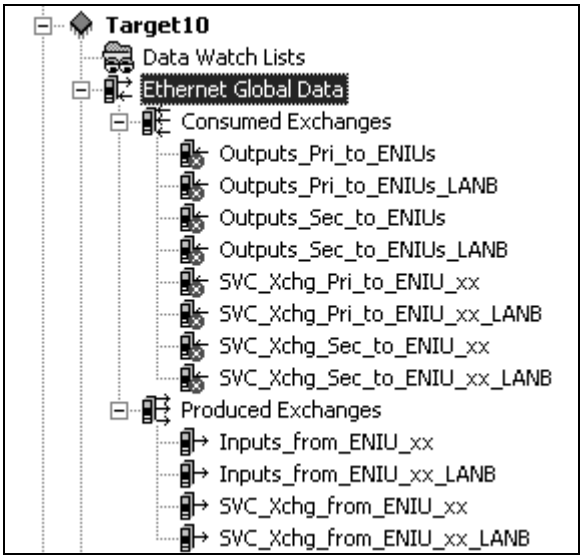
Configuring EGD Exchanges for Remote COMMREQ Calls

The Ethernet Global Data exchanges used for Remote COMMREQ Calls are included an Ethernet NIU target in the programmer. For version 1.3x and later Ethernet NIU targets, these Ethernet Global Data exchanges have names that start with *SVC*. For version 1.2x of the Ethernet NIU targets, these EGD exchanges have names that start with *RCC*.

If the application uses the RX7i or RX3i controller templates, the EGD exchanges for Remote COMMREQ calls are already present and configured in the controller target. EGD exchanges only need to be configured in the controller target, as described in this chapter, if:

- a template set was not used or
- additional Ethernet NIU targets have been added to the project.

Ethernet Global Data exchanges for Remote COMMREQ Calls are included in the EGD exchanges component of the Ethernet NIU target configuration as shown below.



The figure shows the exchanges in the Ethernet NIU for a dual LAN, dual controller system. Note that the Ethernet NIU needs consumed exchanges from each controller, designated by Pri and Sec in the exchange names. When using the templates for a CRE or CRU redundancy CPU, the Pri and Sec designation are not in the names of the exchanges in the controller target.

EGD Exchanges for Remote COMMREQ Calls

The EGD Exchanges for Remote COMMREQ Calls in version 1.3x and later Ethernet NIU targets are:

- *SVC_Xchg_Pri_to_ENIU_xx*: Consume a RCC request from the primary controller
- *SVC_Xchg_Sec_to_ENIU_xx*: Consume a RCC request from the secondary controller
- *SVC_Xchg_from_ENIU_xx*: Produce a RCC response back to the controller(s)
- *SVC_Xchg_Pri_to_ENIU_xx_LANB*: Consume a RCC request from the primary controller
- *SVC_Xchg_Sec_to_ENIU_xx_LANB*: Consume a RCC request from the secondary controller
- *SVC_Xchg_from_ENIU_xx_LANB*: Produce a RCC response back to the controller(s)

The exchanges for LAN B are only used if the ENIU has dual LANs connected to it. If the ENIU only has a single LAN, the *_LANB* exchanges should be deleted if they are present in the target.

RCC Exchange Names for Earlier Releases

The Remote COMMREQ Call EGD exchanges in version 1.2x Ethernet NIU templates 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)

Timing for EGD Exchanges for Remote COMMREQ Calls

Each of the EGD exchanges for Remote COMMREQ Calls has a default Produced Period of 50 milliseconds or 75 milliseconds and a consumed Update Timeout of 150 milliseconds or 230 milliseconds.

Configuring EGD Exchanges for RCC (Version 1.3x and later ENIU Target)

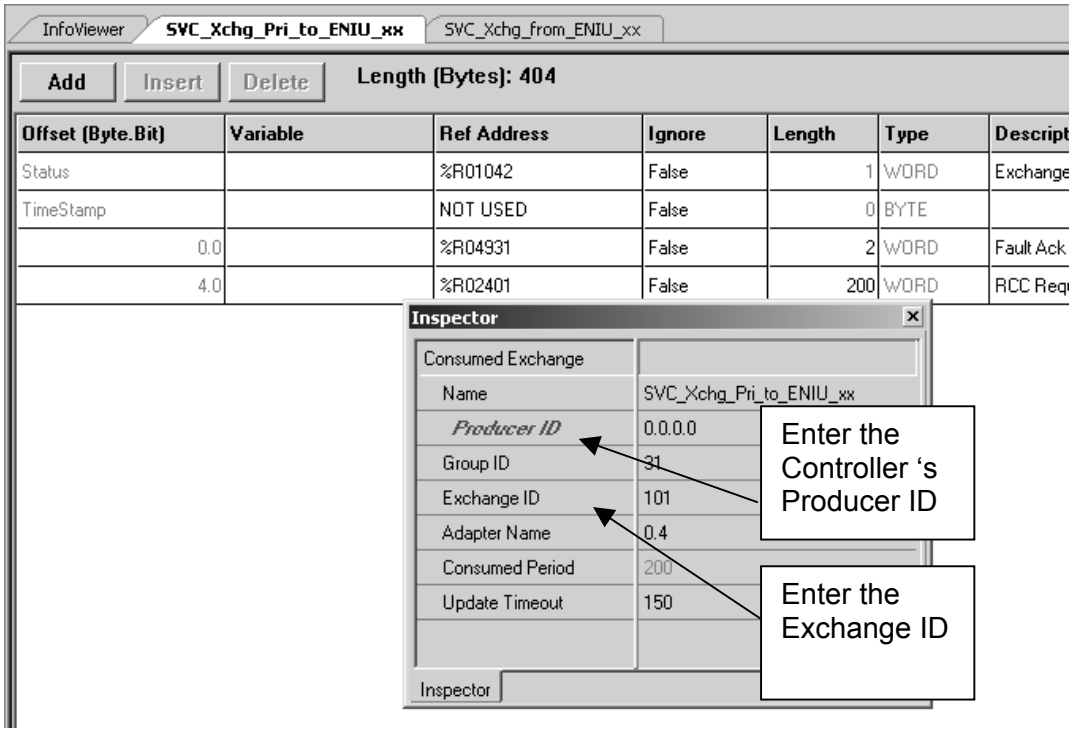
If a new Ethernet NIU target is added to an application, configure the EGD exchanges for Remote COMMREQ Calls as described in this section.

Configuring the ENIU's Consumed Exchange to Receive RCC)

For Release 1.3x or later of the application templates, the Ethernet Global Data exchange named *SVC_Xchg_Pri_to_ENIU_xx* delivers the COMMREQ request from the primary controller to the Ethernet NIU. When using a template set, this exchange is pre-configured and no additional configuration of the SVC exchanges is required.

If the RX3i Ethernet NIU was created by making a new project or by adding a target to an existing folder, the EGD exchange needs to supply the Producer ID of the controller and the Exchange ID.

The data ranges must not be changed. The Update timeout should be changed to match the value for the other Ethernet NIU or to the value in the timing chart if many Ethernet NIUs are added.



Each exchange that is sent to each Ethernet NIU must have a unique Exchange ID. The templates use the numbering convention [100+ the Ethernet NIU number]. For example, for Ethernet NIU 1 the Exchange ID would be 101. If a secondary controller is used, the same Producer ID used for the exchange from the primary controller, and the Exchange ID need to

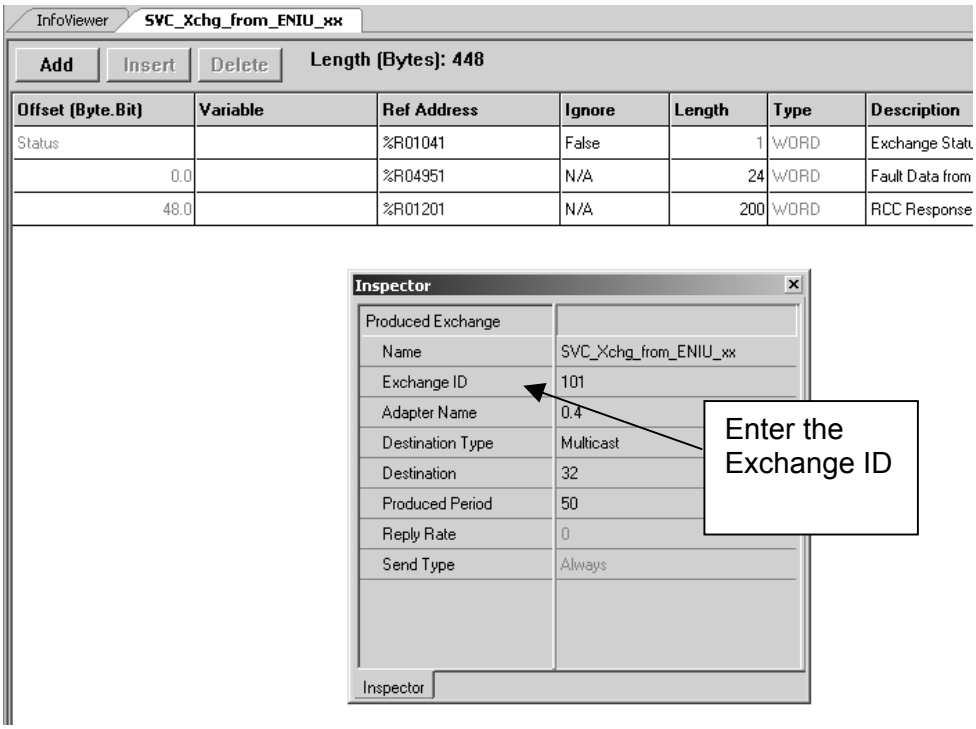
be configured. For the exchange from the secondary controller, the Exchange ID is the number used for the exchange from the primary, plus an offset which is configured in the controller application. The default is 1000. For this example the Exchange ID for the EGD exchange from the secondary controller to ENIU 1 would be 1101 (101 + 1000).

Configuring the ENIU's Produced Exchange for Response to RCC

For release 1.3x or later of the application templates, the EGD exchange *SVC_Xchg_from_ENIU_xx* delivers the response to the COMMREQ request back to the controller.

In the Ethernet NIU, only the Exchange ID needs to be entered, as shown below.

The data ranges must not be changed. The other exchange properties' parameters should not be changed.

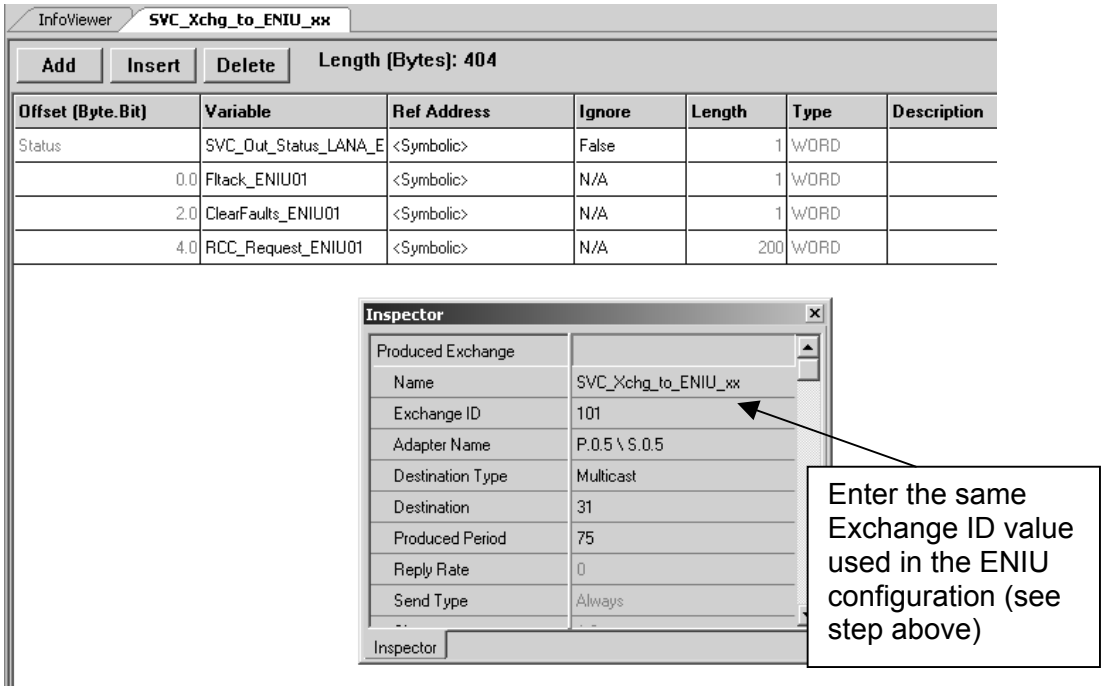


Use a unique Exchange ID for the exchange for each Ethernet NIU. A numbering convention like 100+ the Ethernet NIU number can be used to generate a number, so for Ethernet NIU 1, the Exchange ID would be 101 and for ENIU2, it would be 102.

Configuring the Controller's Produced Exchange to Send RCC

For Release 1.3x and later of the application templates, the request exchange *SVC_Xchg_to_ENIU_xx* must be configured in the controller as shown below. A separate exchange is required for each Ethernet NIU. Except for the Exchange ID, the parameters should be the same for the exchanges.

- If multiple Ethernet NIUs will receive Remote COMMREQ Call commands, each Ethernet NIU must have a separate Ethernet Global Data exchange. Each exchange must have a unique Exchange ID.
- The Adapter Name identifies the Ethernet module that is sending the EGD Exchange.
- The Destination Type is Multicast.
- Destination is 31 (this is the same value used for Group ID in the ENIU configuration, see above)
- The Destination is the IP Address of the Ethernet NIU.
- The Produced period should be 50 milliseconds or 75 milliseconds to match the ENIU default.



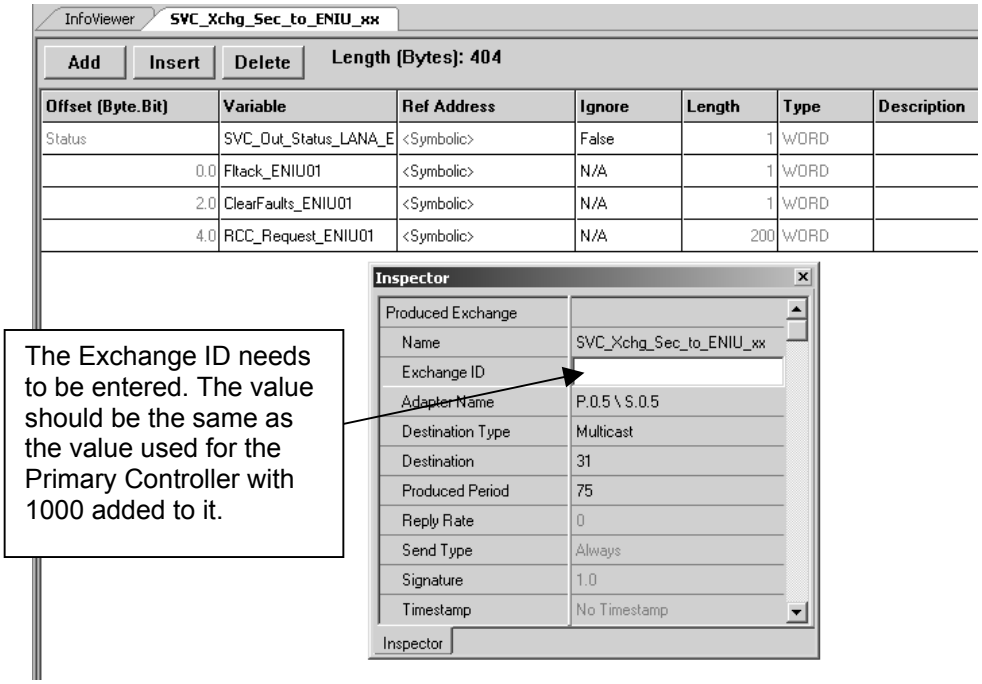
- The Data Ranges must be configured as shown.

- Symbolic variables are used for the Data Ranges.
  - The EGD Exchange Status variable should be *SVC_Out_Status_LANy_ENIU_xx*
  - The Fault Ack symbolic variable should be *Fltack_ENIUxx*
  - The Clear Faults symbolic variable should be *ClearFaults_ENIUxx*
  - The RCC Request symbolic variable should be *RCC_Request_ENIUxx*

    where xx = ENIU #, and y = A or B (for LANA or LANB)

If the controller target was created from the templates or if RCC functionality already exists in the controller target (if an additional Ethernet NIU is being added to an application), the required symbolic variables already exist in the controller target. If the controller target was NOT created from the templates, and if RCC functionality does NOT already exist in the controller target, the user must create the symbolic variables.

If the secondary controller is not a PACSystems RX7i CRE-type CPU or an RX3i CRU-type CPU, the request exchange needs to be configured in the secondary controller as shown below.

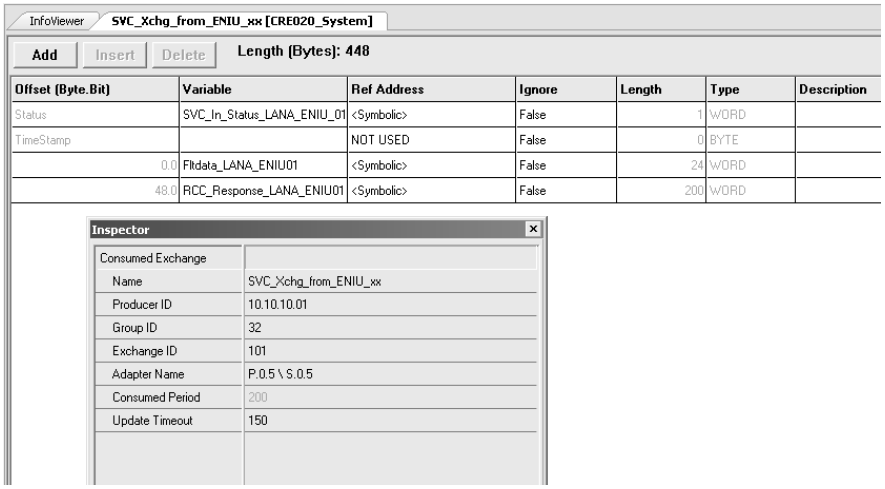


All the data ranges and parameters are the same as in the exchange from the primary controller except the Exchange ID.

Configuring the Controller's Consumed EGD Exchange for RCC Response

For release 1.3x and later of the application templates, the *SVC_Xchg_from_ENIU_xx* response exchange needs to be configured in the controller as shown below. A separate exchange is required for each Ethernet NIU. Except for the Producer ID and the Exchange ID, the parameters should be the same for the exchanges.

- If multiple Ethernet NIUs will receive Remote COMMREQ Call commands, each Ethernet NIU must have a separate Ethernet Global Data exchange. Each exchange must have a unique Exchange ID.
- The Producer ID is the Local Producer ID Address of the Ethernet NIU.
- The Group ID is 32 (this is the same value used for Destination ID in the Ethernet NIU configuration, see above)
- The Exchange ID is the format of 1xx, where xx is the Ethernet NIU number
- The Adapter Name identifies the Ethernet module that is receiving the EGD Exchange.
- The Update Timeout period should be 150 milliseconds or 225 milliseconds to match the ENIU default.



The Data Ranges must be configured as shown. Symbolic variables are used for the Data Ranges:

- The EGD Exchange Status variable should be *SVC_In_Status_LANy_ENIU_xx*
- The Fault Data symbolic variable should be *Fltdata_LANy_ENIUxx*
- The RCC Response symbolic variable should be *RCC_Response_LANy_ENIUxx*

  where xx = ENIU #, and y = A or B (for LANA or LANB)

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

For Release 1.3x and later of the application templates, if multiple Ethernet NIUs will receive Remote COMMREQ Call commands, each exchange must each have a different Exchange ID. That makes each Ethernet Global Data Exchange unique on the Ethernet LAN. 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 SVC Ethernet Global Data for the first Ethernet NIU uses Exchange ID 101. The secondary controller uses Exchange ID 1101. For each additional ENIU the Exchange ID should be incremented by 1.

The Exchange ID needs to be incremented on both ends: at the controllers, and at the Ethernet NIU.

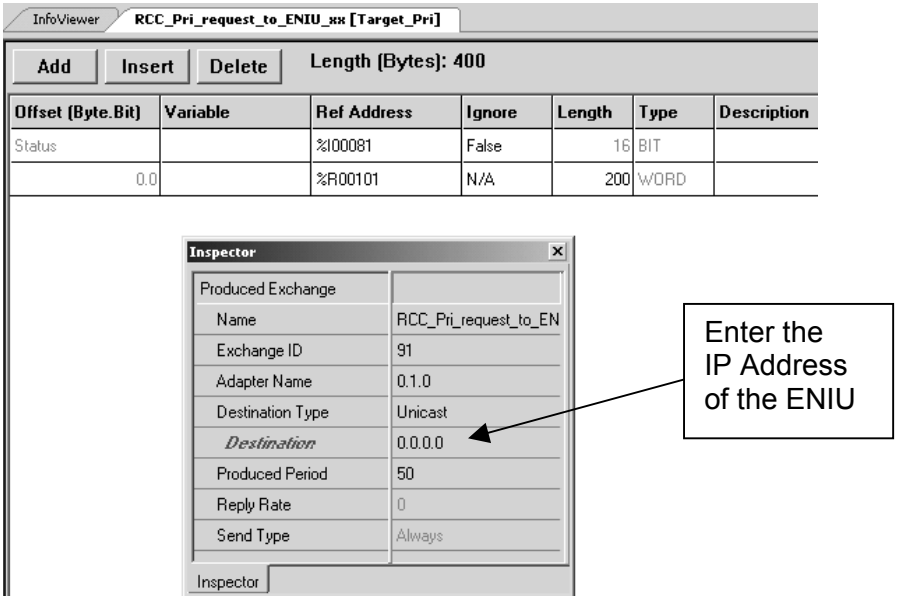
Configuring EGD Exchanges for RCC (Version 1.2x ENIU Target)

For applications created using Proficy Machine Edition prior to release 5.7, an Ethernet NIU target can be configured for Remote COMMREQ Calls as described below.

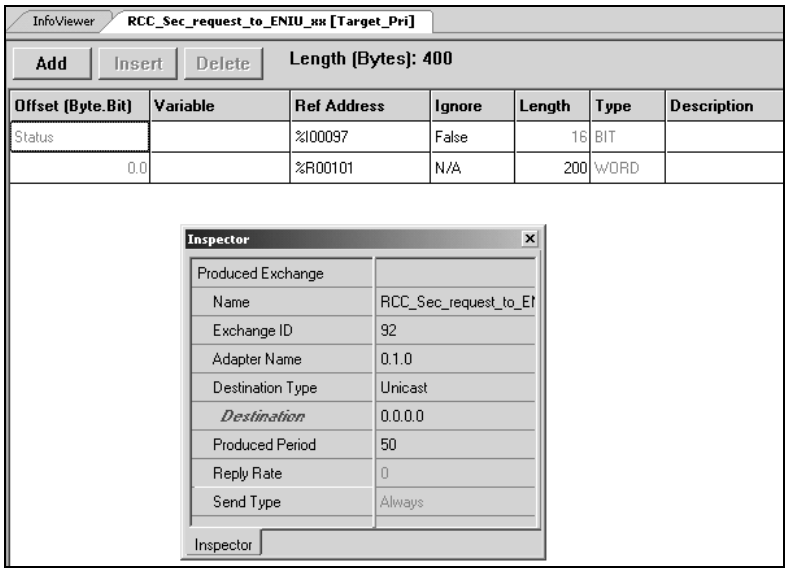
Configuring the Controller's Produced Exchange to Send RCC

For release 1.2x of the application target, 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 Ethernet NIUs will receive RCC commands, each ENIU needs a separate exchange and each exchange must have 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.



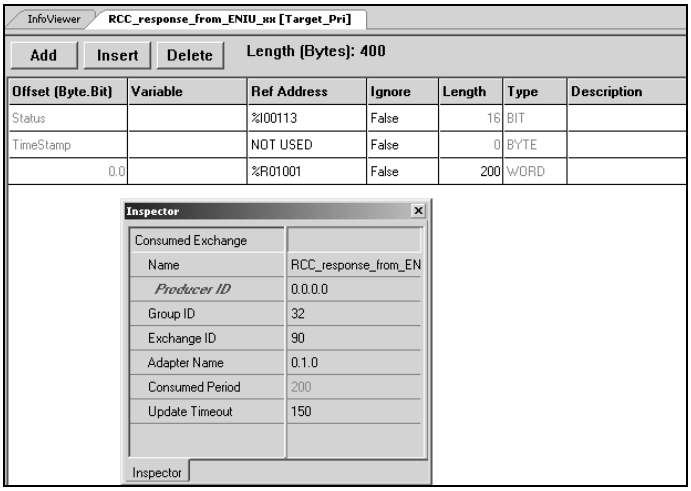
If there is a secondary controller, the request exchange needs to be configured in the secondary controller as shown below.



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

For release 1.2x of the application target, 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 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

For release 1.2x of the application target, 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.

The Exchange 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 Target

This step is only necessary if the controller target was not built from the template set or if the RCC block has not already been added to the controller target.

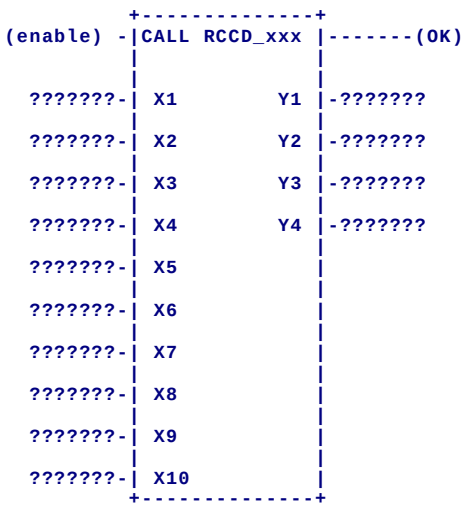
In the programmer navigator tree view, right-click Program Blocks in the logic area of the controller.

Click on add C block. A dialog box to add the C block will come up.

Browse to the file *RCCD_1xx.gefElf* (for version 1.3x and later templates) or *RCCM_12x.gefElf* (for version 1.2x templates).

To add it to the controller target, double-click or select and open it.

The C block for version 1.3x and later templates is shown below. For version 1.2x templates, the C block is similar, but with fewer inputs and fewer outputs as noted.

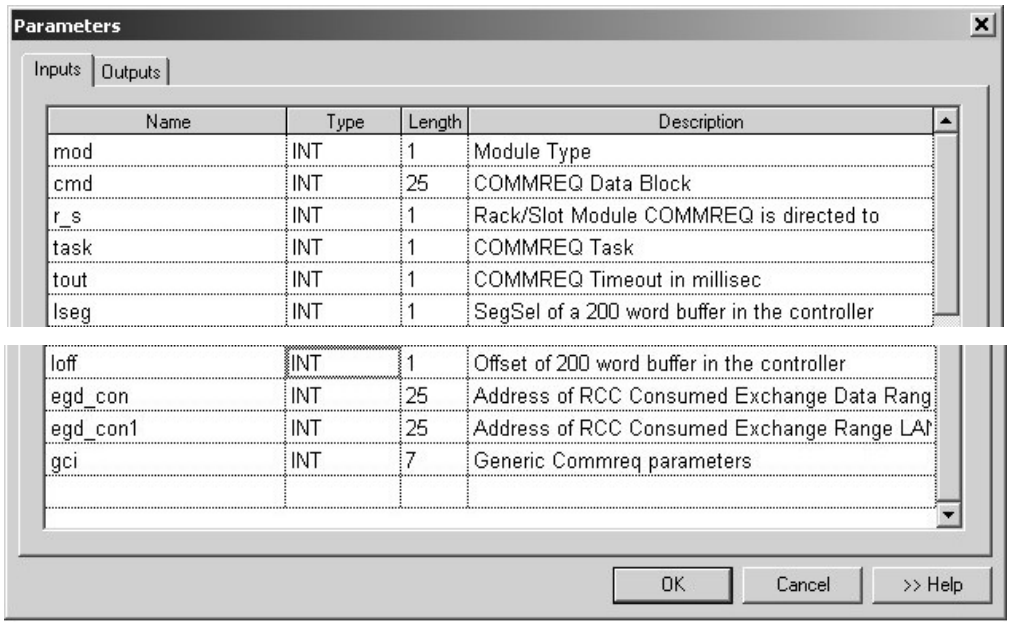


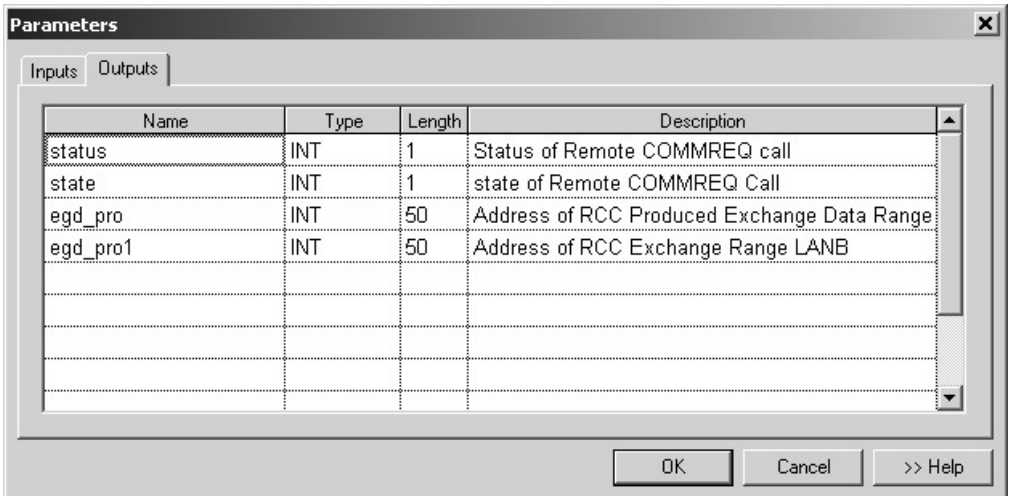
The inputs and outputs parameters for the C block have the following labels:

	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.
X9	egd_con1	int	25	(Version 1.3x) Address of RCC Consumed Exchange data range LAN B.
X10	gci	int	7	(Version 1.3x) Generic commreqs parameters. See chapter 14.
Y1	stat	int	1	Status of Remote COMMREQ Call.
Y2	stat	int	1	State of Remote COMMREQ Call.
Y3	egd_pro	int	50	Address of RCC Produced Exchange data range.
Y4	Egd_pro1	int	50	(Version 1.3x) Address of RCC Produced Exchange data range LAN B.

Adding the C Block Parameters

Right-click on the C block in the Navigator tree view. Click on Properties. Click on the parameters line in the Property Inspector to open the parameters dialog. Enter the parameters as shown below. (Parameters shown are for version 1.3x and later templates. Version 1.2x have fewer parameters as described on the previous page).





Adding the C Block Call to Controller Logic

Create a LD block called *RCC* in *_Main*, and add a Call to RCC that is called every scan.

- For version 1.4x templates, in the *RCC* block, add a call to *RCCD_14x_yy* that is called every scan.
- For version 1.3x templates, in the *RCC* block, add a call to *RCCD_13x_yy* that is called every scan.
- For version 1.2x templates, in the *RCC* block, add a call to *RCC_12x_yy* that is called every scan.

  yy indicates the Ethernet NIU number that is to receive the RCC command.

Enter the inputs and outputs of the C block as described below.

Inputs for the C Block

mod Module type the COMMREQ is being sent to. Enter a Register reference, then place the module code (see below) into the register.

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

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 the 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 Data Area in *SVC_Xchg_from_ENIU_xx* Exchange. Data area is a symbolic variable *RCC_Response_from_ENIUxx*. It must have an array dimension of 200.

Note: for Version 1.2x the EGD Pointers are Reference addresses such as %R1201

egd_c1 (Version 1.3x templates only) Pointer to the RCC Data Area in *SVC_Xchg_from_ENIU_xx_LANB* Exchange. Data Area is a symbolic variable *RCC_Response_from_ENIUxx_LANB*. It must have an array dimension of 200.

gci (Version 1.3x templates only) Pointer to the starting address of the generic COMMREQ parameter data. See chapter 14.

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. This tells the intermediate step that the C block is on.

egd_p (Version 1.3x and later templates) Pointer to the data area in *SVC_Xchg_to_ENIU_xx* exchange. Data Range is a symbolic variable *RCC_Request_ENIUxx*. It must have an array dimension of 200.

(Version 1.2x templates) Pointer to the *RCC_request _to_ENIU_xx* exchange. Enter the starting register reference of the exchange data.

egd_p1 (Version 1.3x and later only) Pointer to the data area in *SVC_Xchg_to_ENIU_xx_LANB* exchange. Data Range is a symbolic variable *RCC_Request_ENIUxx*. It must have an array dimension of 200.

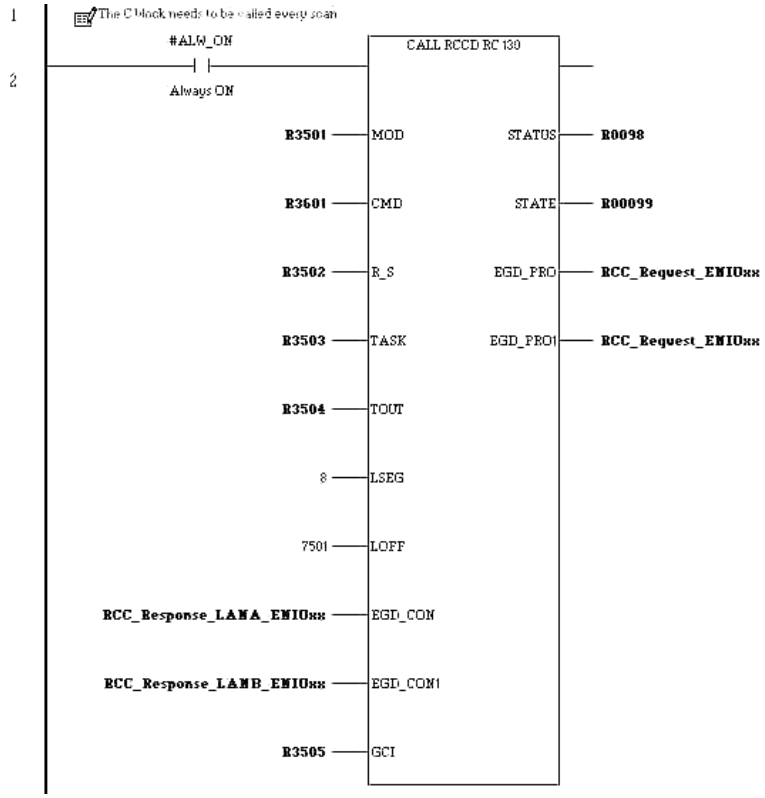
Adding Logic to Sequence RCC Commands and Check Return Status

A Remote COMMREQ Command works like a standard COMMREQ.

- 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

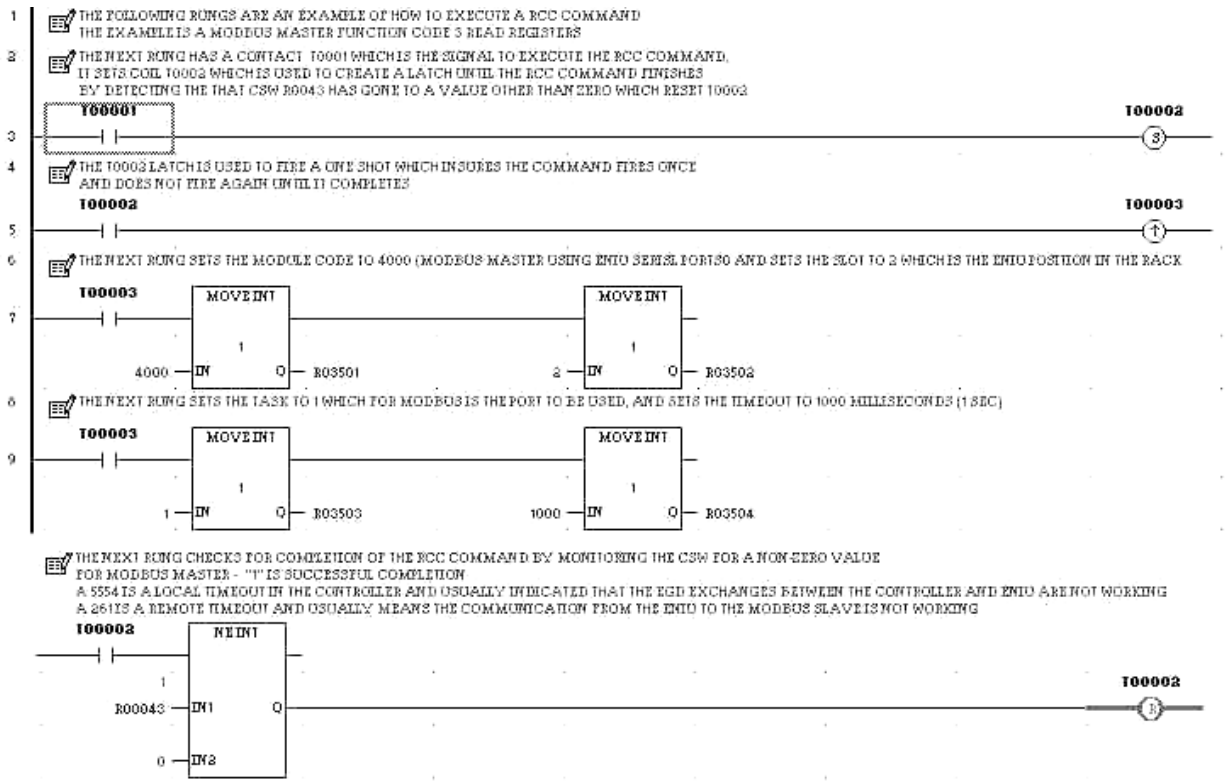
Example logic for two commands is shown on the following pages. Both commands use the same C block. The C block for version 1.3x templates is shown below. Differences in the C block for version 1.3x or later, and 1.2x templates were described on the previous pages.



Sample Logic for Modbus Master

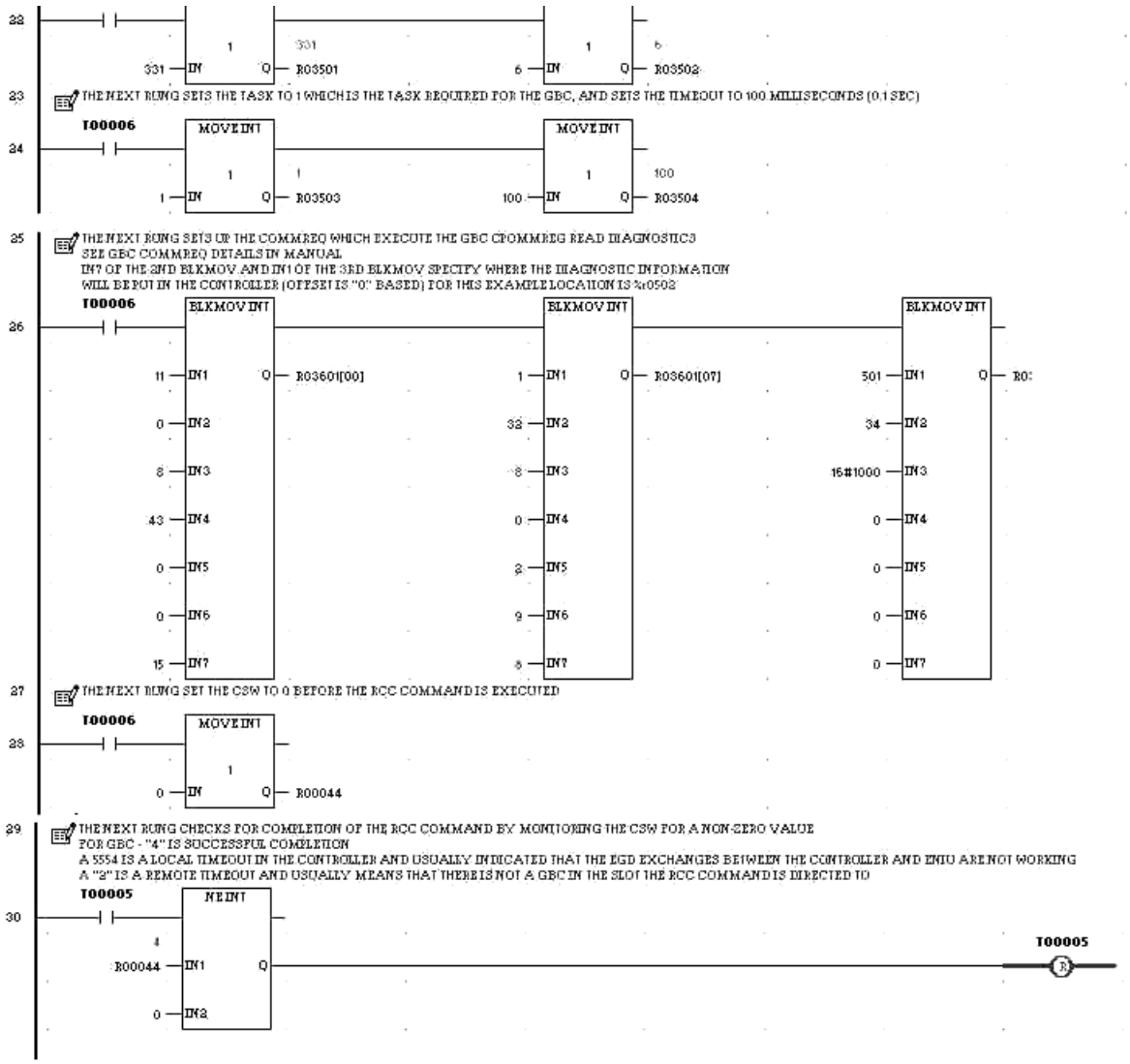
For port 1 of the Ethernet NIU, the Port Mode MUST be changed to Serial I/O and the Baud rate and parity need to be set to match the Modbus Slave settings.

After the port setup is changed, it must be downloaded to the Ethernet NIU. If the port settings 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 C block will supply a result for the requested Remote COMMREQ Call command. The application in the controller needs to monitor it for completion.

The C block performs the following sequence of checks:

1. 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:

   A. After power is cycled to the Ethernet NIU, when a Status output value of 9999 indicates that the ENIU has powered up.

   B. (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.

2. The requested Remote COMMREQ Call command is checked to be sure that:

   A. The requested COMMREQ is one that is supported by the Remote COMMREQ Call feature (see chapter 13).

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

3. The Ethernet NIU responds with a completion code, indicating success or an error condition. The Remote COMMREQ Call 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 13 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	RCC Command sent to Ethernet NIU, waiting for response
333	Local Timeout in Controller waiting for ENIU response
222	Waiting for Response from ENIU and Sequence Number does not match
47	Initial value

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.

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 10.

   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 if the primary is in control. Bit 4 indicates if 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 an 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 *SVC_Xchg_from_ENIU_xx* (*RCC_response_from_ENIU_xx* in version 1.2x templates).

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 that 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 *SVC_Xchg_from_ENIU_xx* (*RCC_response_from_ENIU_xx* in version 1.2x templates). 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 the Remote COMMREQ Call C block:

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 Sequence 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 subsequent registers

COMMREQ Status Word Error Codes in the Controller

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 Words from the Ethernet NIU

The Ethernet NIU responds to the controller with a COMMREQ Status Word. The COMMREQ Status Word can come from the Ethernet NIU or from the module to which the COMMREEQ was sent. The COMMREQ Status Word Error Codes generated by the Ethernet NIU are:

Decimal	*Hex*	
-2	FFFE	The Ethernet NIU timed out before receiving a response to the RCC command.
1234	4D2	The RCC command caused a non-fatal COMMREQ fault in the Ethernet NIU.

The 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 designated COMMREQ Status Word location.

COMMREQ Status Word Error Codes generated by the modules are forwarded directly to the controller.

Chapter 13 COMMREQs for Remote COMMREQ Calls

Chapter 14 described the Remote COMMREQ Calls feature of Ethernet NIUs. This chapter explains how PACSystems RX7i or RX3i controllers can use Remote COMMREQ Calls to send COMMREQs to intelligent modules in the I/O Station.

- COMMREQs Supported by Remote COMMREQ Calls
- COMMREQs for DeviceNet Master Modules
- COMMREQs for Genius Bus Controller Modules
- COMMREQs for RX3i Profibus Master Module
- COMMREQ for RX3i and Series 90-30 Motion Controller Modules
- COMMREQ for High-Speed Counter Modules
- COMMREQs for MODBUS RTU Master on the RX3i Ethernet NIU Serial Ports
- COMMREQ Error Codes by Module Type
- Status Values for MODBS Master Communications

The COMMREQ Command Data blocks listed on the next page are supported by all RX3i Ethernet NIU template applications. Ethernet NIU templates that are version 1.3x or later provide additional COMMREQ capability using the Generic COMMREQ feature, which is described in chapter 14. Generic COMMREQs can be used to send other types of COMMREQs that are not listed here.

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

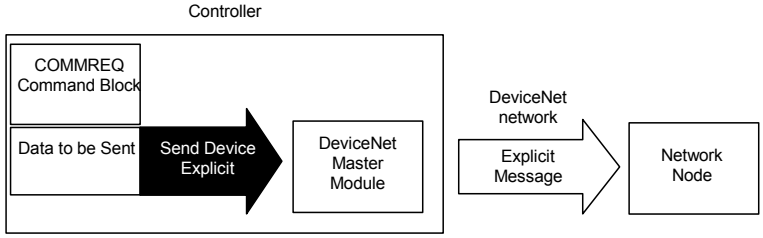
The controller can use a Remote COMMREQ Call to send the following COMMREQs to a PACSystems RX3i DeviceNet Master Module IC694DNM200 or Series 90-30 DeviceNet Master Module IC693DNM200 in the I/O Station:

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

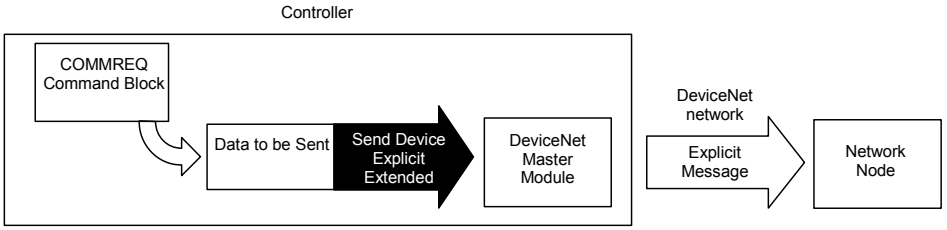
DeviceNet Master Modules, COMMREQ 1: Send Device Explicit

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 is 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.

Word	Dec	(Hex)	Description
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 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 byte 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 1 (this indicates success). 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 (Specific error codes are in word 6)			
	Values above 0xFF are internal 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	*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
7 - 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 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)
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.

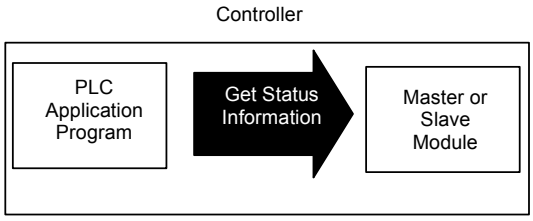
<table>
<tr><th>Word</th><th colspan="4">Description</th></tr>
<tr><td>1</td><td colspan="4">Command number that this data block is replying to. (4)</td></tr>
<tr><td rowspan="15">2
low
byte</td><td colspan="4">Status Code: Number indicating the status of the client connection to the device.</td></tr>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

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.

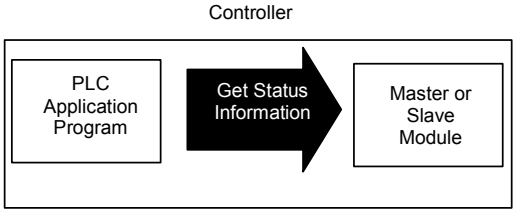
Word	*Dec/Bin*	*Hex*	*Description*	
1	00005	(0005)	Command number that this data block is replying to. (5)	
2 low byte	00000001	(01)	Indicates whether the module is set up for slave operation (Network Settings configured). For this example, the module is being scanned.	
			0x00	Idle (Group 2 master/slave 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 Input 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 Input Status Information 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

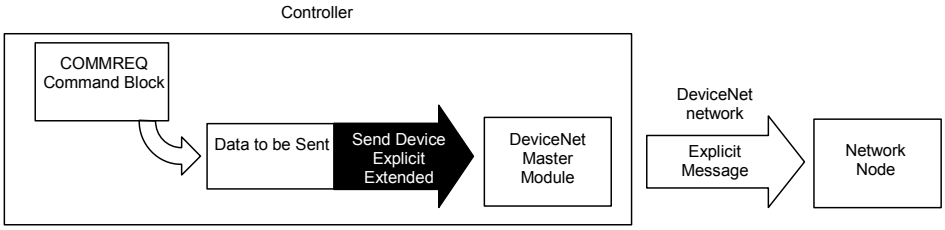
<table>
<tr><th>Word</th><th>Description</th></tr>
<tr><td>1</td><td>Command code that this data block is replying to. (6)</td></tr>
<tr><td>2 - 5</td><td>Device Status. Each bit corresponds to an individual device MAC ID. The state of that bit indicates the device's status: 0 = Device is not active (not configured, faulted, etc…), 1 = Device is active, being scanned. For the master's own MAC ID, the status bit is always 0.
<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>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>byte 1</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td></tr>
<tr><td>byte 2</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td>byte 3</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td></tr>
<tr><td>byte 4</td><td>39</td><td>38</td><td>37</td><td>36</td><td>35</td><td>34</td><td>33</td><td>32</td></tr>
<tr><td>byte 5</td><td>47</td><td>46</td><td>45</td><td>44</td><td>43</td><td>42</td><td>41</td><td>40</td></tr>
<tr><td>byte 6</td><td>55</td><td>54</td><td>53</td><td>52</td><td>51</td><td>50</td><td>49</td><td>48</td></tr>
<tr><td>byte 7</td><td>63</td><td>62</td><td>61</td><td>60</td><td>59</td><td>58</td><td>57</td><td>56</td></tr>
</table></td></tr>
<tr><td>6</td><td>Server Status
<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>res.</td><td>AKS</td><td>CYC</td><td>COS</td><td>res.</td><td>ST</td><td>P</td><td>EX</td></tr>
<tr><td>byte 1</td><td colspan="3">Reserved</td><td>SERA</td><td>IDLE2</td><td>IDLE1</td><td colspan="2">G3</td></tr>
</table>
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</td></tr>
<tr><td>7</td><td>CAN Network Status.
<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>reserved</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 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</td></tr>
<tr><td>8</td><td>Firmware ID, Minor revision:</td><td>In BCD four hex digits. For example, revision 1.10 = 01 10 hex.</td></tr>
<tr><td></td><td>Firmware ID, Major revision:</td><td>See above.</td></tr>
</table>

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><td></td><td>bit 7</td><td>bit 6</td><td>bit 5</td><td>bit 4</td><td>bit 3</td><td>bit 2</td><td>bit 1</td><td>bit 0</td></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>
</table>

Word	Description
65	Number of minor ticks per major tick interval

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.

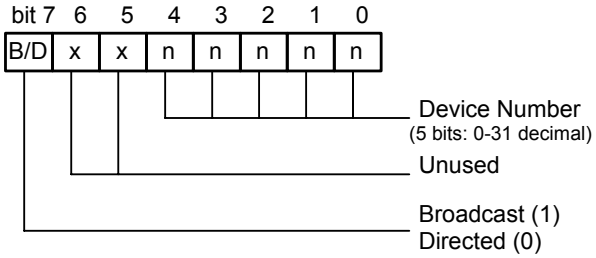
Note that the Dequeue Datagram queue is operated as a first–in–first–out (FIFO) queue. Specific datagrams within the queue cannot be dequeued without first dequeueing datagrams received earlier.

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

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; must be 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

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, 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; must be 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 Master 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 .
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.
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, 196; 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, 196; 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 13-39.

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; must be 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	58625	(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	57857	(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 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 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

<table>
<tr><th>Word</th><th>Dec</th><th>(Hex)</th><th colspan="4">Description</th></tr>
<tr><td rowspan="6">12</td><td rowspan="6"></td><td rowspan="6"></td><td colspan="4">Value depends on Function Code</td></tr>
<tr><td>1</td><td rowspan="2">Reference table for response: %I = 16, %Q = 18, %T = 20, %M = 22, %AI = 10, %AQ = 12, %R = 8, %W =196</td><td>6</td><td>Not used, must be 0</td></tr>
<tr><td>2</td><td>7</td><td>Reference table for response: %I = 16, %Q = 18, %T = 20, %M = 22, %AI = 10, %AQ = 12, %R = 8, %W =196</td></tr>
<tr><td>3</td><td rowspan="2">Reference table for response: %AI = 10, %AQ = 12, %R = 8, %W =196</td><td>15</td><td>Reference table for data source: %Q= 18, %R = 8, %W =196</td></tr>
<tr><td>4</td><td>16</td><td>Reference table for data source: %AI = 10, %AQ = 12, %R = 8, %W =196</td></tr>
<tr><td>5</td><td>Not used, must be 0</td><td>17</td><td>Reference table for response: %AI = 10, %AQ = 12, %R = 8, %W =196</td></tr>
<tr><td rowspan="6">13</td><td rowspan="6"></td><td rowspan="6"></td><td colspan="4">Value depends on Function Code</td></tr>
<tr><td>1</td><td rowspan="3">Offset in Reference table to put response</td><td>6</td><td>Not used, must be 0</td></tr>
<tr><td>2</td><td>7</td><td rowspan="3">Offset in Reference table to get data</td></tr>
<tr><td>3</td><td>15</td></tr>
<tr><td>4</td><td>16</td><td></td></tr>
<tr><td>5</td><td>Not used, must be 0</td><td>17</td><td>Offset in Reference table to put response</td></tr>
<tr><td>14</td><td>0500</td><td>(01F4)</td><td colspan="4">Timeout in Milliseconds: Range 0 to 10,000 (500 ms in this example)</td></tr>
</table>

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 IC693DSM324 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 14 Generic Remote COMMREQ Calls

This chapter describes the Generic Remote COMMREQ Command (RCC) feature of the RX3i Ethernet NIU, which allows PACSystems RX7i and RX3i controllers to pass any COMMREQ to a module in an RX3i ENIU I/O Station. This capability is only available with PACSystems RX7i and RX3i controllers.

All RX3i Ethernet NIUs support a pre-defined set of COMMREQs that can be sent using Remote COMMREQ Calls. Use of that feature was described in chapters 12 and 13.

Ethernet NIU targets that are version 1.3x or later allow any COMMREQ that is supported by a module in an RX3i Ethernet NIU station to be used, as described in this chapter. When it receives a Generic Remote COMMREQ Command, the module in the RX3i Ethernet NIU station error-checks the COMMREQ.

If a COMMREQ is supported by Remote COMMREQ Calls, that method should be used instead of the Generic COMMREQ method. Supported COMMREQs have more error checking and also support more reference tables for source and destination addresses.

Generic COMMREQs should only be used if the COMMREQ is not directly supported by Remote COMMREQ Commands.

Generic Remote COMMREQ C Block

Generic COMMREQs are implemented using an additional input to the RCC C block. That input defines the generic operation of the COMMREQ. It provides a set of pointers identifying where source and destination information can be found in the COMMREQ data words.

RCC C Block Generic COMMREQ Input Parameters

The Generic COMMREQ Input (GCI) to the Remote COMMREQ Call C block is a block of seven words, which are described below. The C Block input parameter needs to be set as an INT of length seven. The application logic must place an address (reference table or symbolic) on this input, which the C block uses as a pointer.

Word	***Description***	***Usage***	***Values***
Word 1	Generic COMMREQ type	All types	0 – not Generic 1 – Type 1 2 - Type 2 3 - Type 3
Word 2	Count to the Segment Selector	Type 2 only	Length in words
Word 3	Count to the Data Length	Type 2 only	Length in words
Word 4	Reserved	Na	
Word 5	Count to the Segment Selector	Type 3 only	Length in words
Word 6	Count to the Data Length	Type 3 only	Length in words
Word 7	Base	Type 2 & 3	0 – base 0 1 – base 1

Using Generic COMMREQs

To use a Generic COMMREQ, follow the setup for a regular COMMREQ with Remote COMMREQ Call as described in chapter 12, in addition, provide the GCI input to the Remote COMMREQ Call C block as described below.

Completing the GCI Inputs to an RCC C Block

1. Determine the type of Generic COMMREQ to be used:

   Type 1 – Executes an operation and returns the status of the operation in the COMMREQ status word.

   Type 2 – Executes an operation, typically a read, and returns the status of the operation in the COMMREQ status word. It also returns data to another location that is specified in the Generic COMMREQ data block.

   Type 3 – Executes an operation, typically a write, and returns status of the operation in the COMMREQ status word. It also fetches data from a location in the controller that is specified in the Generic COMMREQ data block. If the data to be written is contained in the COMMREQ data block, it is a Type 1 Generic COMMREQ (see above), not a Type 3.

   If there are no Segment Selectors and Offsets after the word that contains the COMMREQ Command code, it is a Type 1 COMMREQ which only returns a COMMREQ status word. Put a value of 1 in the first word of GCI and the Generic COMMREQ setup is complete.

   If there are a Segment Selector and Offset after the COMMREQ Command code, it is a Type 2 or Type 3 COMMREQ.

   - If the Segment Selector and Offset specify a destination of data coming back to the controller, it is a Type 2. Enter 2 in word 1 of input parameter GCI, and continue at step 2.
   - If the Segment Selector and Offset specify a source in the controller where the data comes from, it is a Type 3. Enter 3 in word 1 of input parameter GCI, and continue at step 3.

2. For Type 2 Generic COMMREQ only:

   a. In word 2 of the GCI input, enter the value to point to the Segment Selector in the COMMREQ data block. To do this, find the COMMREQ command number in word 7 of the COMMREQ data block. Count from this location (do not count the COMMREQ command code location) to the Segment Selector, and enter the count into word 2 of the GCI input.

b. In word 3 of the GCI input, enter the value to point to the data length in the COMMREQ data block. Count to the Data Length parameter the same way as to the Segment Selector and enter this count in word 3 of the GCI input

c. The offset to the data in the COMMREQ is either at the specified offset (base 1) or at the next highest value (base 0). Enter the base (either 0 or 1) in word 7 of the GCI input.

3. For Type 3 Generic COMMREQ only:

a. In word 5 of the GCI input, enter the value to point to the Segment Selector in the COMMREQ data block. To do this, find the COMMREQ command number in word 7 of the COMMREQ data block. Count from this location (do not count the COMMREQ command code location) to the Segment Selector, enter the count into word 5 of the GCI input.

b. In word 6 of the GCI input, enter the value to point to the data length in the COMMREQ data block. Count to the data length parameter the same way as to the Segment Selector and enter this count in word 6 of the GCI input.

c. The offset to the data in the COMMREQ is either at the specified offset (base 1) or at the next highest value (base 0). Enter the base (either 0 or 1) in word 7 of the GCI input.

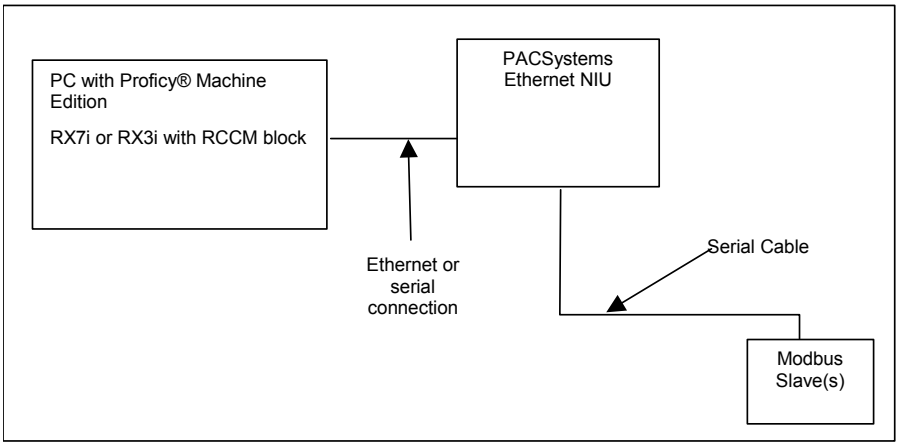
Chapter 15 Modbus Master for the Ethernet NIU

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 implemented in two ways in the PACSystems RX3i Ethernet NIU:

- Remote COMMREQ Call commands can be sent from a PACSystems RX7i or RX3i controller to tell the Ethernet NIU to do MODBUS communications. When the Remote COMMREQ Call method is used, 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 that is built into the Ethernet NIU template to do MODBUS communication. When the Local User Logic method is used, the addresses for the data to be written to the MODBUS Slave and 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 must 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, and the response to the could be returned to either place where the C block was called, resulting in loss of data.

- Remote COMMREQ Call communications always use the C block *MB1_xxx* (for example. *MB1_133*). Remote COMMREQ Call communications can be used for either or both ports.
- Local User Logic must use the C block *MB2_xxx* . It can be used for either or both ports. The *MB2_xxx* block has input and output parameters for setting up the communication requests. Local User Logic cannot use both ports simultaneously.

Hardware Configuration for MODBUS Master

Hardware configuration is done with Proficy Machine Edition. Any RX3i Ethernet NIU serial port that will be used for MODBUS Master protocol must be configured for Serial I/O with the communication parameters shown below.

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

Version 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 the first time the C block is called. 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 MB2_xxx for MODBUS Master

The input parameters of the C function block used for MODBUS Master communications must be correctly set up specifying:

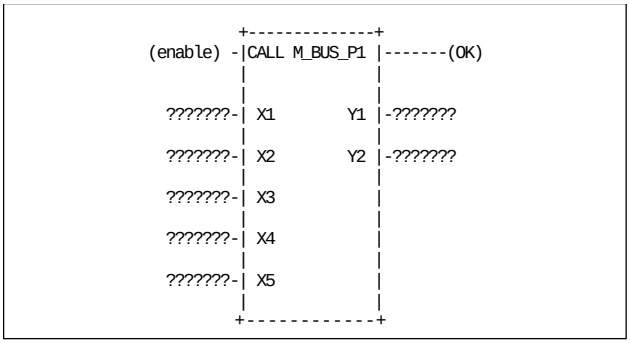
- the slot the ENIU is located in
- the port to be used for MODBUS Master communications
- a required buffer area in %R or %W memory (150 registers in size).
- the particular settings for an individual MODBUS Message

When MODBUS Master is implemented using Remote COMMREQ Call commands, all setup is done in the controller.

When MODBUS Master is implemented using 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.

Name	Type	Length	Description
cmd	INT	8	Modbus command parameters
r_s	INT	1	rack slot location of eniu (default is 2)
port	INT	1	port number 1 or 2
seg	INT	1	Segment selector for internal buffer %R or %W
seg_off	INT	1	segment offset for internal buffer

Parameters

Inputs | Outputs

Name	Type	Length	Description
status	INT	1	status of commreq (0 in process, 1 success, error >1)
state	INT	1	internal state of commreq (idle, in process, broadcast timeout)

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 easiest approach is to call the C block every scan of the NIU. Optionally, the C block could be called only when MODBUS Master requests are being executed, but it must be called until the request completes.

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. %R11000). 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 holds the value for the rack/slot location of the module that is to execute the COMMREQ (for MODBUS commands this is the NIU module).

Format	Int (i.e. %R10211), 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 holds the value for the serial port number on the NIU that is to execute the COMMREQ.

Format	Int (i.e. %R10212) , 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 hold the value for the Local Buffer Segment Selector.

Format	Int (i.e. 8) , 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 holds the value for the Local Buffer Offset.

Format	Int (i.e. 13000) , example uses a constant (13000)
Length	1 Note X5 must be configured as Type: Int; Len:1
Usage	Local Buffer Offset: Valid numbers 1 to 125 less the end of selected 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. %R11098). 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. %R11099). 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 using Block Move or Data Init instructions.

Word	*Contains*	*Description*	
First word	MODBUS query Command: 8002	The C block sets this input parameter to 0 after 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 section focuses on Local User Logic use of MODBUS Master. Remote COMMREQ Calls are described in chapter 12. 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 *MB2_xxx* is the same as the setup for the Remote COMMREQ Call CMD input. The only difference is that the Remote COMMREQ Call CMD input starts with the seventh word. That is the word that corresponds to the COMMREQ command number in all 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 either 8002 or 8006 in the first word of the command block, the C block performs a MODBUS query based on the values in the command block.
3. The C block validates the command block inputs and then:
   - Writes a zero back into the first word of the command block
   - Writes 2 into the output Y1 (Status) of the C block if the command block is good and communication is started. If there is an error in the command block or port setup, an error code is written into output Y1 (Status).
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.

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. When reading real or dword data from some types of MODBUS slaves, the two registers that form the 32-bit data type may be reversed. For slaves that have this 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 Ethernet NIU 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 parameter 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 Exception 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 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 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 Ethernet NIU 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 five 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 sslave device.

For a GE Fanuc PLC MODBUS slave the meaning of the words 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 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 = Timed out after sending a Broadcast message

Programming Examples

This section describes application logic to implement MODBUS Master communications from:

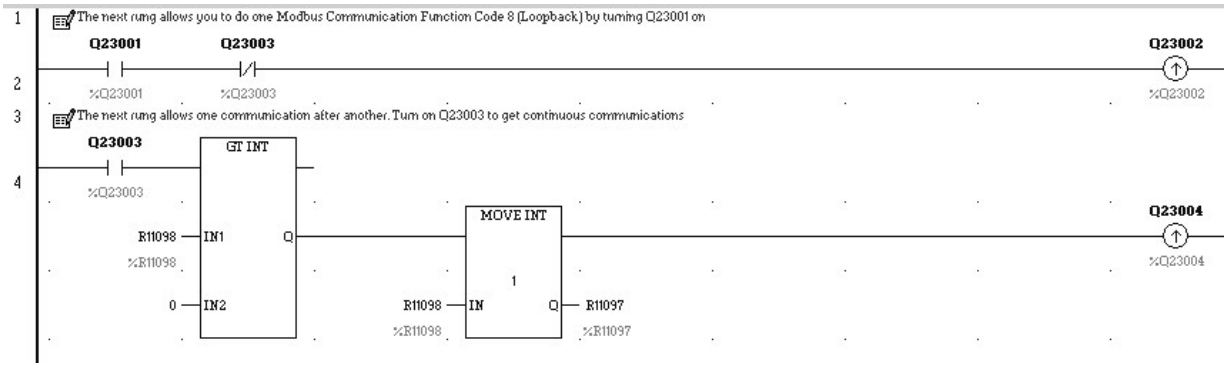
- Local User Logic in the Ethernet NIU
- Remote COMMREQ Call Communications from the master

Example 1: MODBUS Master Using Local User Logic

The MODBUS Master ladder code is in the Ladder block *Local_User_Logic*. This block is called every scan. The example shows how to do a Function Code 8 and how to continuously do a Function Code 3.

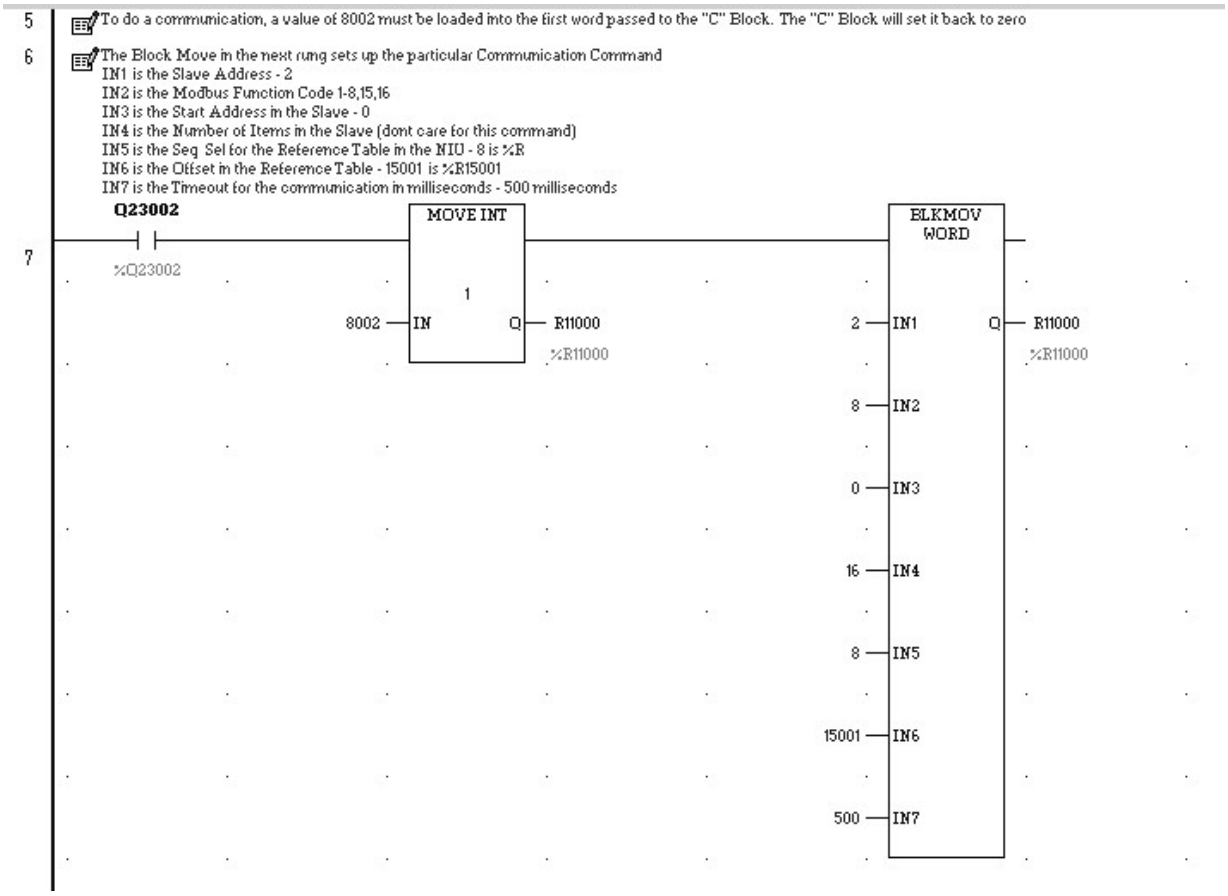
Rung 1: An individual MODBUS message to do Function Code 8 Loopback can be sent by toggling %Q23001. A message to Read Registers can be sent continuously, and as quickly as possible, by turning on %Q23003.

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 %R11097. 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 %Q23004 would need to be used to load different parameters into the BLKMOV in Rung 9.



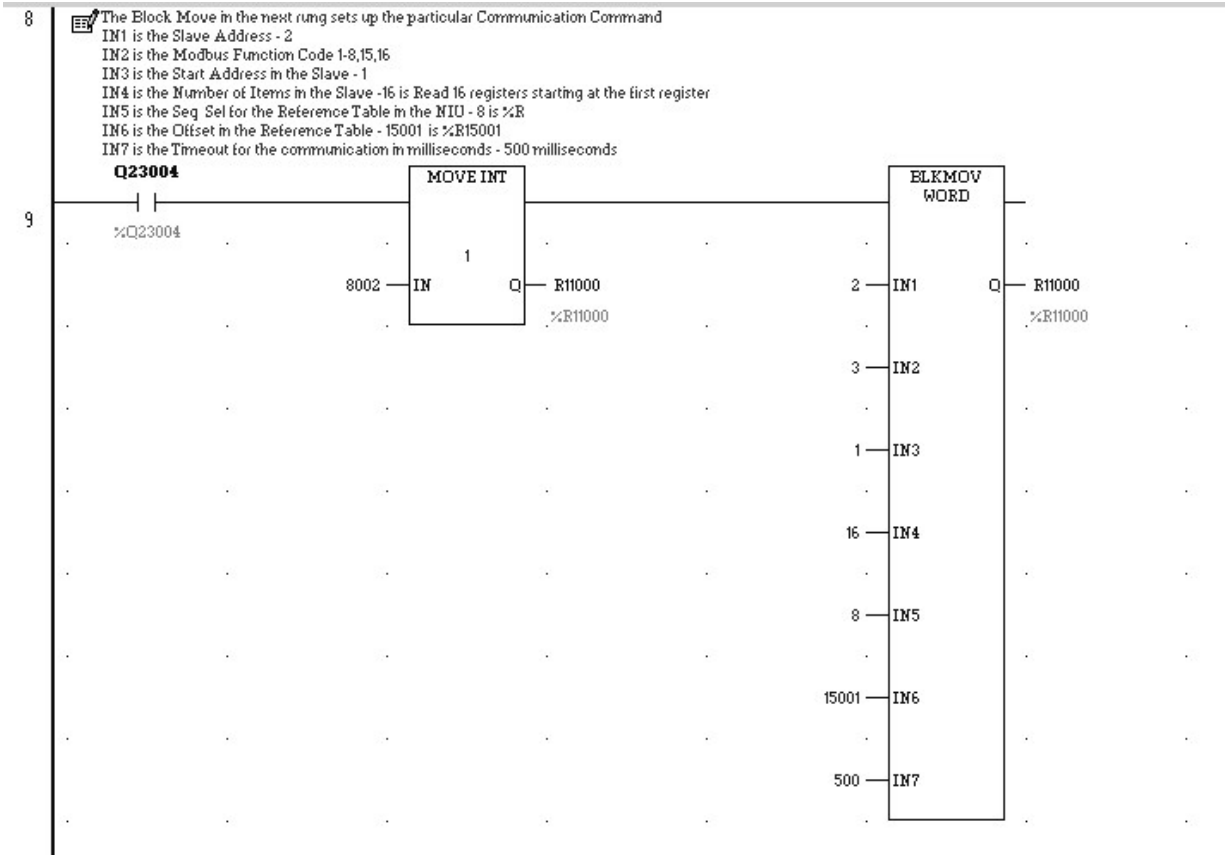
The BLKMOV in rung 7 sets up a MODBUS query to:

- Slave #2,
- Function Code 8 Loopback,
- Timeout is 500 milliseconds.

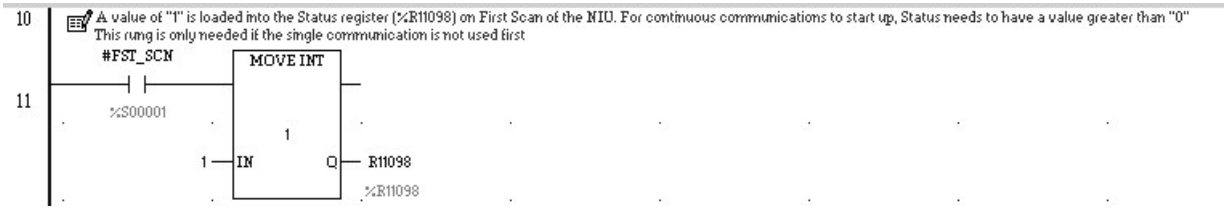


The BLKMOV in rung 9 sets up a MODBUS query to:

- Slave #2,
- Function Code 3 Read Registers,
- Starting at address 400001 (first Register),
- Read 16 Registers,
- Put the data in %R registers
- Starting at %R15501,
- Timeout is 500 milliseconds.

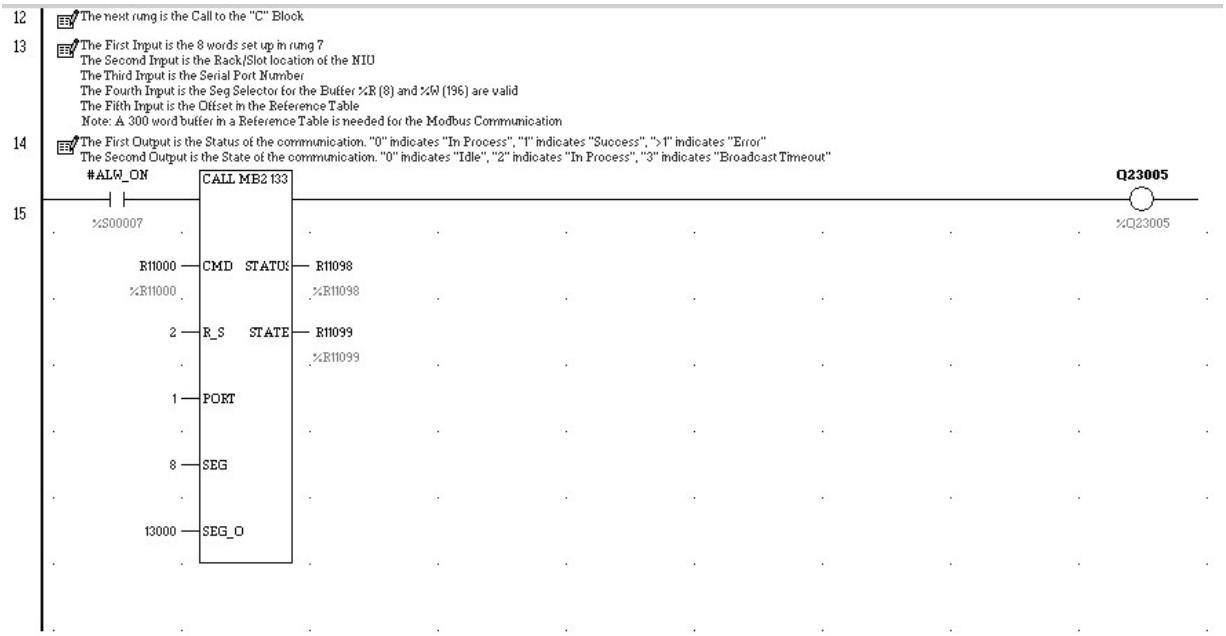


Rung 11 is used to initialize the status value register (%R11098) to 1. This allows the continuous communication to start if the single communication is not done first.



Rung 15 of the example logic calls the C block. Setup for the example C block is:

- Command block of 8 registers starting at %R11000
- Ethernet NIU located in slot 2 (Must be rack 0)
- Communication to use port 1 (RS232 port) of the Ethernet NIU.
- Internal 150-register buffer in %R memory
- Starting at register %R13000



Error-checking for the MODBUS communication is done by monitoring %R11098 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 the Ethernet NIU Hardware Configuration to Ethernet NIU will fail.
- If using %W, make sure you configure %W in the ENIU Hardware Configuration, as it defaults to length of 0.

Example 2: MODBUS Master Using RCC Communications

Note: This example shows the logic in the controller, not logic in the ENIU.

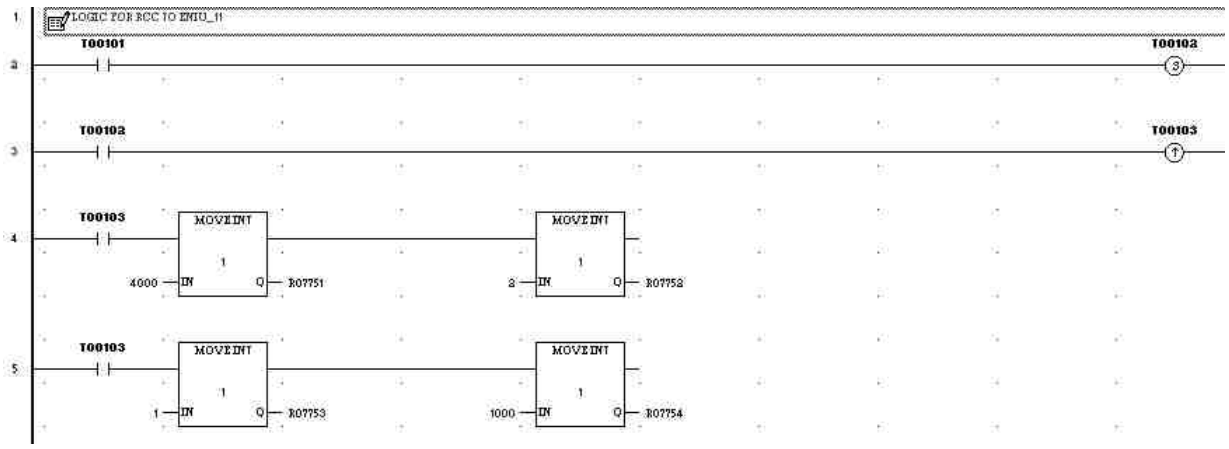
This example uses a MODBUS Master command to Port 1 of the Ethernet NIU (Function Code 3 Read 24 Registers from slave #1 starting at the first register and put the data in %R101).

See the RCC call at the end of the example.

Rung 2 – Turning on %T00101 sets %T00102. That fires the one-shot %T00103 in rung 3. %T00102 will be used to check for completion.

Rung 4 loads 4000 (the module code for MODBUS Master) into %R07751, and loads 2 (the slot number of the Ethernet NIU) into %R07752.

Rung 5 loads 1 (the port number) into %R07753, and loads 1000 (the timeout at the controller in milliseconds) into %R07754.

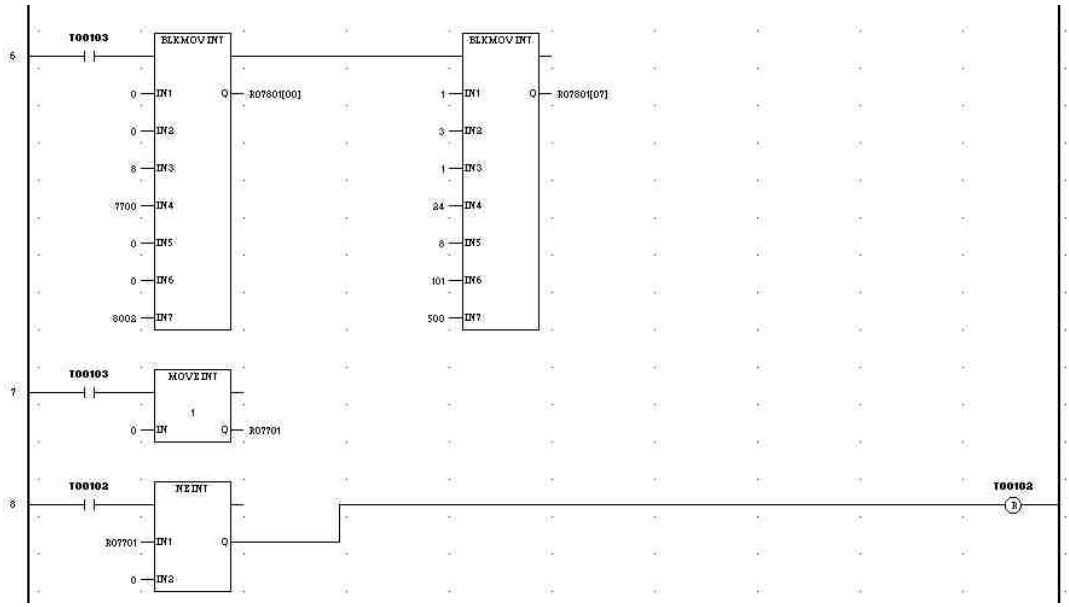


Rung 6 loads the RCC command into %R7801 thru %R7814

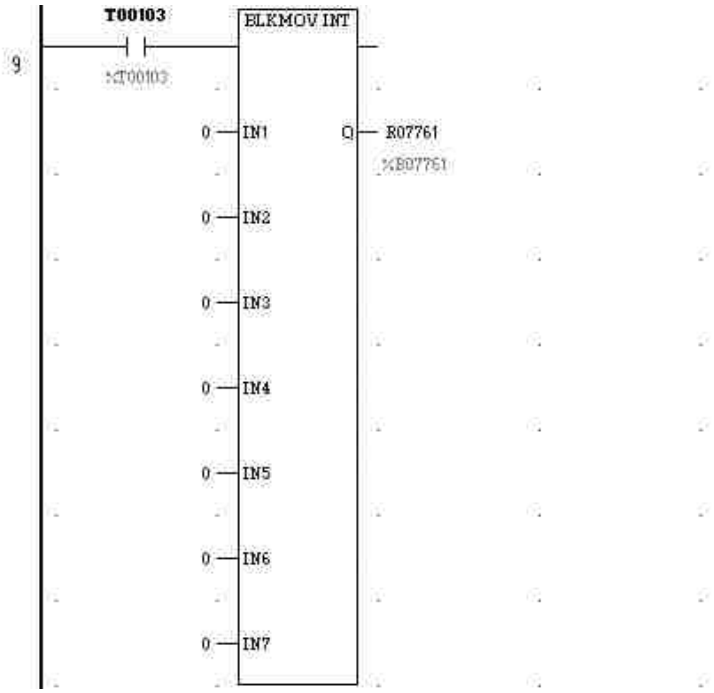
%R7801 – always 0 for MODBUS Master
%R7802 – always 0 for MODBUS Master
%R7803 - CSW segment selector, 8 for %R memory
%R7804 – CSW offset (0 based), %R7701 is CSW in this example
%R7805 – always 0
%R7806 – always 0
%R7807 – COMMREQ Command number 8002 for MODBUS Master
%R7808 – Slave ID (Slave #1)
%R7809 – Function Code (3: Read Registers)
%R7810 – Starting Address (1: First Register) First Register is 40001 in Modicon addressing
%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 completion

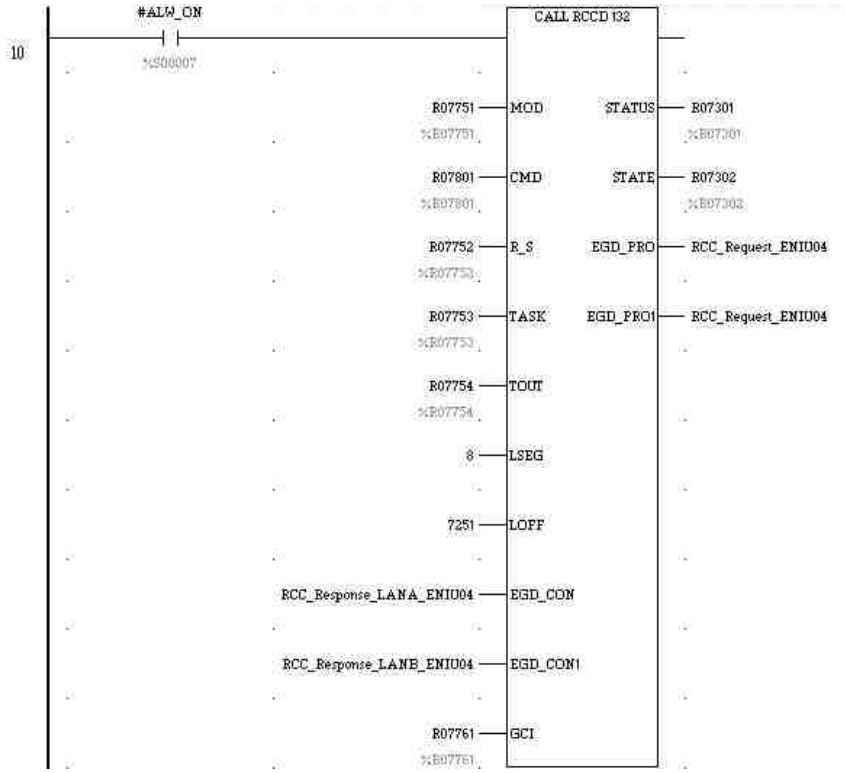
Rung 8 checks for COMMREQ Status Word going non-zero (completion). The end of the rung has a Reset coil %T102 (not shown). The value 1 indicates success.



Rung 9 shows setting up the Generic COMMREQ (GCI) fields for the C block. For MODBUS Master commands, Generic COMMREQ is not used, so the seventh register must be 0. The first six registers are not used but should be set to zero.



Rung 10 executes the Remote COMMREQ communications call:



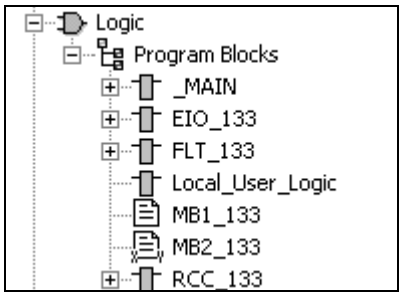
Chapter 16 Upgrading a Release 1.2x Ethernet NIU Application

This chapter describes upgrading a version1.2x Ethernet NIU application to version 1.3x or later. Version 1.2x of the Ethernet NIU application does not provide the enhanced fault reporting or Generic RCC features that are available with Release 1.3x and later versions. Existing version 1.2x applications can be upgraded to include theses features.

- Determining the version Ethernet NIU application that is needed
  - Deciding if Fault Reporting is to be used
  - Deciding if Generic RCC Commands are to be used
- Upgrading the Ethernet NIU Application
  - Save the existing application target data
- Creating a New Target
- Adding Fault Reporting and/or Generic RCC to the Ethernet NIU Application
  - Adding/modifying Ethernet Global Data Exchanges
  - Ethernet NIU Ladder Blocks
  - Adding Controller Logic
- Adding Fault Reporting and/or Generic Remote COMMREQ Calls to the Controller
  - Symbolic Variables
  - Adding/modifying Ethernet Global Data Exchanges
  - Adding Controller Logic

Determining the Ethernet NIU Application Needed

Logic program blocks in an Ethernet NIU target are shown below.



The blocks named with the version number as the last three characters of the block names.

- The first digit specifies the type of Ethernet NIU
  - "0" is a Series90-30 ENIU
  - "1" is a RX3i ENIU
- The second digit is the major revision Note: RX3i Ethernet NIU starts with major version 4.
- The third digit is a minor revision.

The illustration above shows 133 logic blocks. The block version depends on which revision of Proficy Machine Edition or Proficy Process Systems software is being used:

Logic Program Blocks Version	***Proficy Machine Edition Version***	***Proficy Process Systems Version***	***Ethernet Firmware Version of Ethernet NIU and ETM***
1.2x	Prior to 5.7	Not applicable	4.82 or later
1.3x	5.7	1.0	5.01 or later
1.4x	5.8 Sim 11	1.5 SIM 11	5.50 or later

All Rx3i Ethernet NIUs can be updated as described in this chapter. For some applications, updating is not necessary.

Update the Application if Fault Reporting will be Used

Version 1.3x and later Ethernet NIU applications can report non-fatal Ethernet NIU faults in the Controller's Fault Table. See chapter 10 for more information about this feature. If you want to add Fault Reporting to an earlier application, upgrade the application to version 1.3x or later.

Update the Application if Generic RCC will be Used

Version 1.2x of the Ethernet NIU application is able to receive a predefined set of COMMREQs from RX7i and RX3i controllers. These COMMREQs are listed in chapter 13. Version 1.3x and later also supports the Generic Remote COMMREQ Command feature, which can be used to send other types of COMMREQs to the I/O Station. If want to add the Generic Remote COMMREQ Call feature to an earlier application, upgrade to version 1.3x or later.

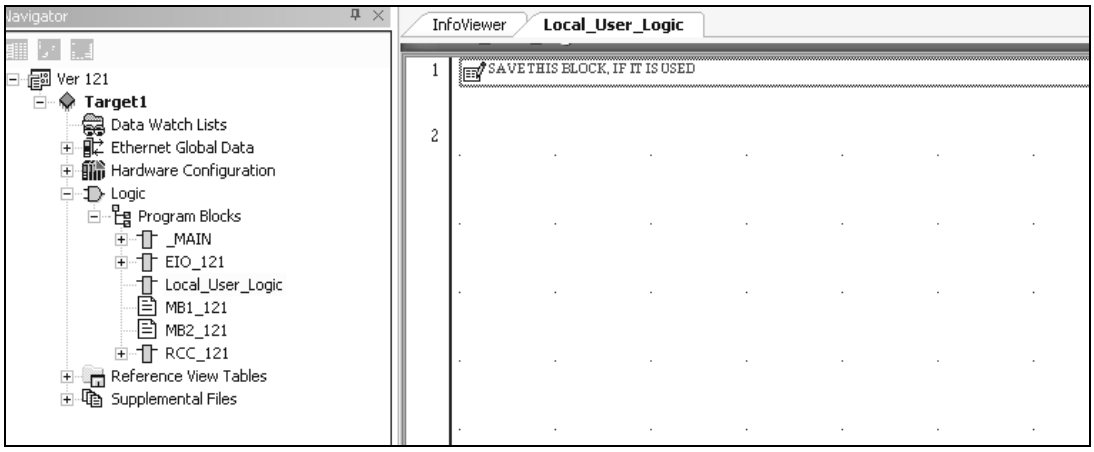
Update the Application if Dual LANs will be Used

Version 1.3x and later Ethernet NIU applications supports dual I/O LANs. The controller also needs this support.

Saving the Existing Application Target Data

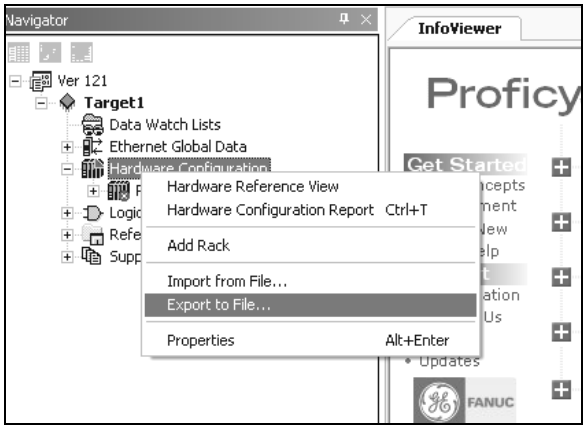
Save the Local_User_Logic Block

If any local logic has been used in the Ethernet NIU target, the *Local_User_Logic* block can be saved by creating a Toolchest Drawer and dragging the block into the drawer; or by copying the *Local_User_Logic* block and pasting it into the new application target.



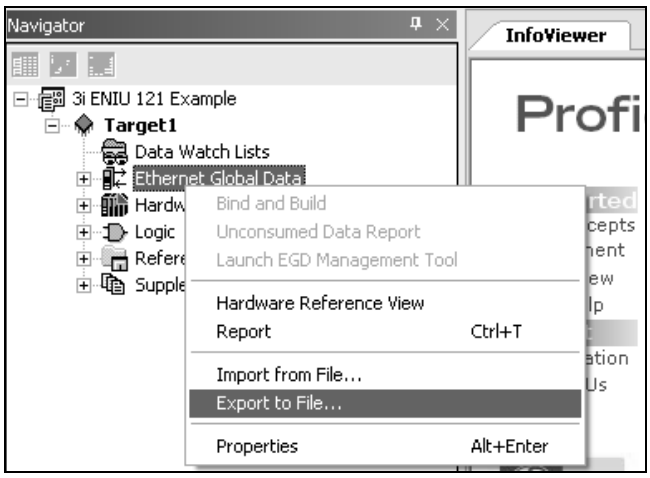
Save the Hardware Configuration

The Hardware Configuration can be saved by Exporting it to a file. Right-click on Hardware Configuration and select Export to File.



Save the Ethernet Global Data Configuration

The Ethernet Global Data can be saved by Exporting it to a file. Right-click on Ethernet Global Data and select Export to File.



Creating a New Version Target

The example screens in this section are for release 1.3x of the Ethernet NIU application. The steps to upgrade to release 1.4x of the Ethernet NIU application are the same.

Using Machine Edition 5.6 or earlier, a version 133 RX3i Ethernet NIU project can be created by using one of the templates for RX3i Ethernet NIU Version 133. This project can have only one RX3i Ethernet NIU target.

These templates pre-populate the target with the Ethernet Global Data exchanges to communicate with a primary controller and a secondary and one or two I/O LANs.

The templates are:

3iENIU_Ver133_1CPU_Template

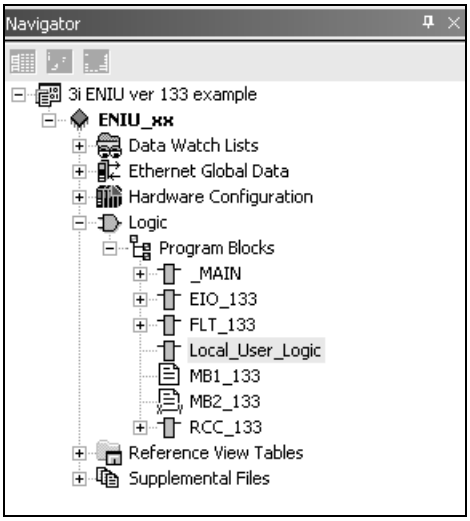
3iENIU_Ver133_2CPU2LAN_Template

Select the template that is suited for the application and create a new Machine Edition project based on that template.

Using Machine Edition 5.7 or later, adding an RX3i Ethernet NIU target (from the Target pulldown menu in Machine Edition) to the existing Ethernet NIU application will incorporate a fully pre-populated RX3i Ethernet NIU into a project. See chapter 3 for instructions on adding an Ethernet NIU Target to an existing application.

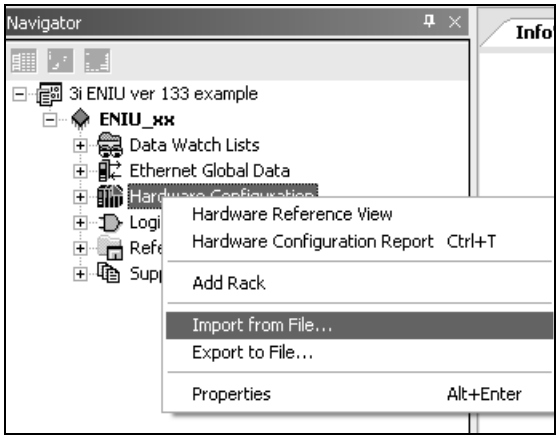
Replace the Local Logic Block

(The example below shows the version 1.3x program blocks).



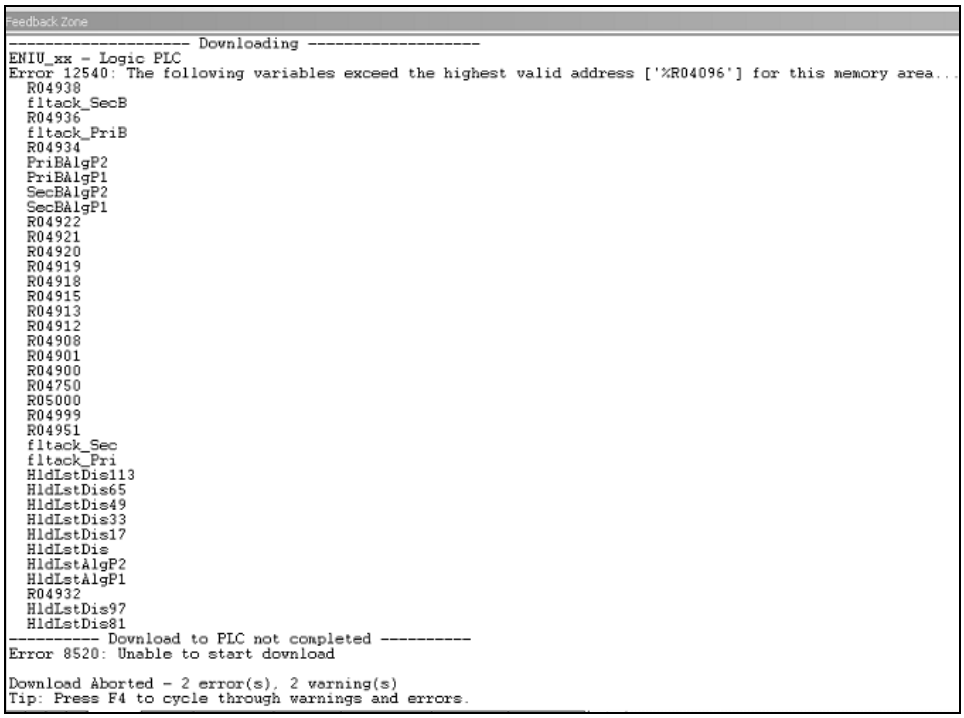
Replace the block by dragging the block saved in the toolchest, or by pasting the block that was copied in the steps above.

Replace the Hardware Configuration

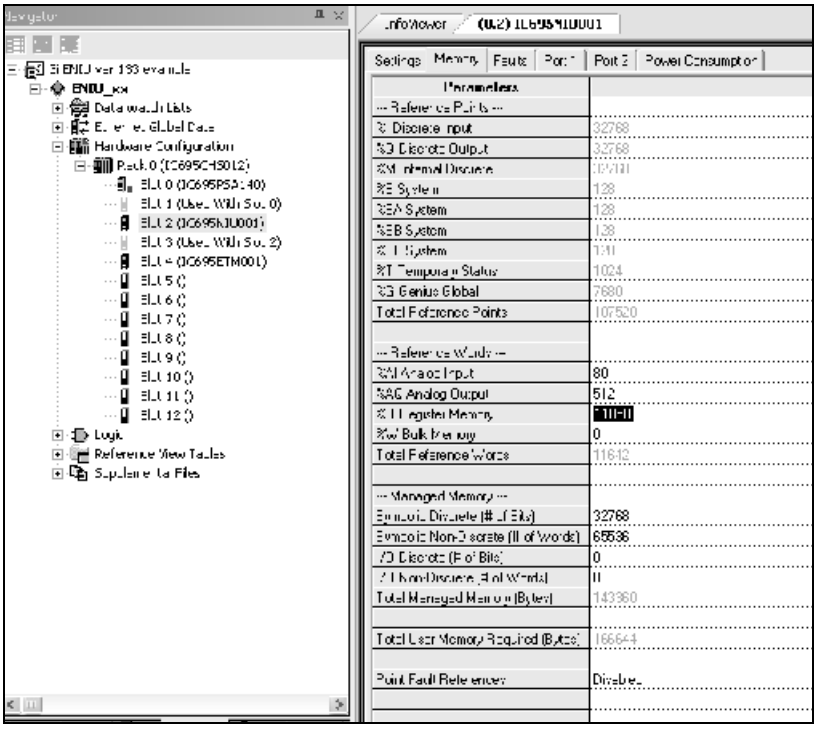


Import the Hardware Configuration that was Exported in steps above.

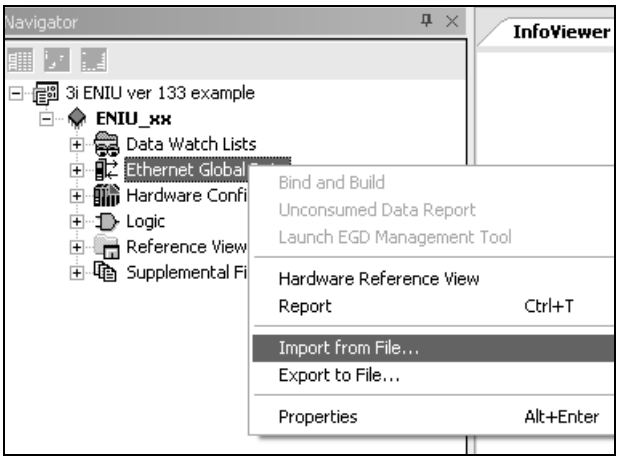
Note: A version 1.3x or later application requires more %R addresses allocated in the Hardware Configuration than previous versions. If less than 11050 %R are allocated in the Hardware Configuration you will see errors like the ones shown below when you download to the Ethernet NIU:



The %R allocation in the Hardware Configuration of the Ethernet NIU needs to be increased to a minimum of 11050, as shown below.



Replace the Ethernet Global Data Configuration



Import the Ethernet Global Data Configuration that was exported in the steps above.

If Fault Reporting or Generic Remote COMMREQ Call Commands are not needed in the application, the conversion is complete.

Adding Enhanced Fault Reporting and/or Generic RCC to a Version 1.2x ENIU Application

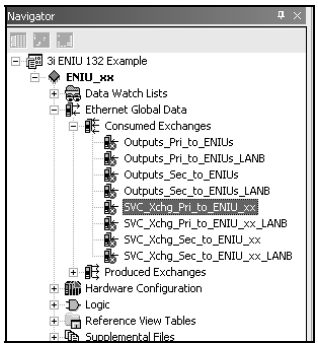
Existing EGD Exchanges

Some version 1.2x Ethernet NIU applications will have existing Ethernet Global Data exchanges configured for Remote COMMREQ Call commands. These are the exchanges labeled *RCC…request/response…* They can be modified for use with Fault Reporting and/or Generic RCC. If they don't exist, they must be created manually. Follow the steps in the next sections to add/modify these exchanges.

Adding/Modifying Ethernet Global Data Exchanges in the ENIU

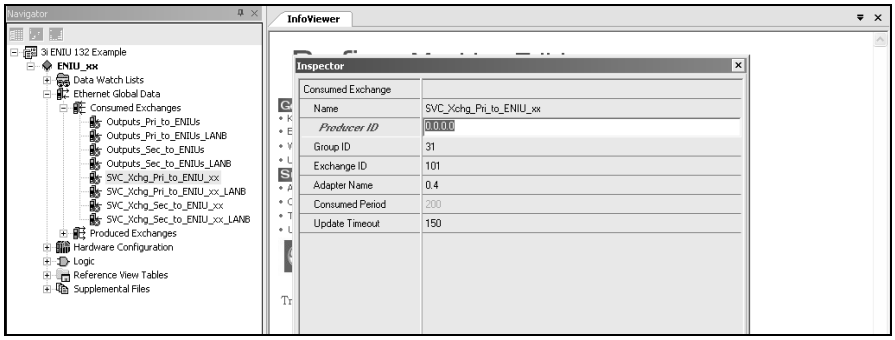
In version 1.3x or later Ethernet NIU targets, both Fault Reporting and Remote COMMREQ Calls are implemented via by the *SVC_Xchg_…* exchanges in the Ethernet Global Data configuration as shown below. If the EGD configuration was imported from a version 1.2x Ethernet NIU application (in the steps above), the exchanges are named *RCC_.*

Consumed Exchanges



There would be one exchange for a single controller/single LAN application and four exchanges for a dual controller/dual LAN configuration. If the application does not already include the exchanges required, they must be added. If Fault Reporting is required, all Ethernet NIUs will need these exchanges. If only RCC is required, only the Ethernet NIUs using RCC need these exchanges.

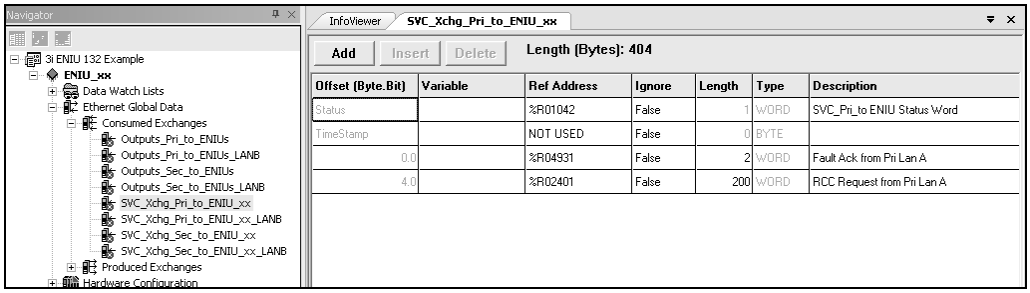
For each added exchange, enter the values for Producer ID, Group ID, Exchange ID, Adapter Name and Update Timeout:



The Producer ID is the Local Producer ID of the controller. The Group ID should be common to all Ethernet NIUs. The Exchange ID must be unique for each Ethernet NIU. The Adapter Name is the rack/slot location of the Ethernet Transmitter Module that is connected to the controller's Ethernet Transmitter Module. The Update Timeout is coordinated with the Production Period of the exchange. See chapter 8 for recommended Production Periods and Update Timeout values.

Exchange Details

The %R04931 data area is used for Enhanced Fault Reporting and the %R02401 data area is used for Remote COMMREQ Call commands. That data area is only needed if the Ethernet NIU will use Remote COMMREQ Calls. If BOTH Enhanced Fault Reporting and Remote COMMREQ Calls will be used, the %R04931 data area MUST be before the %R02401 data area of the consumed exchange details.

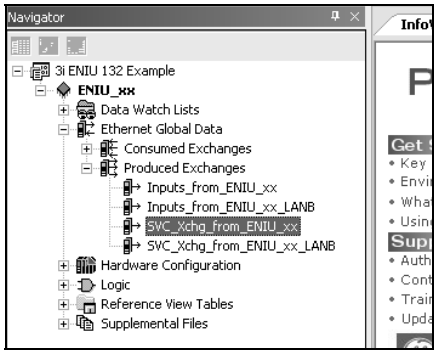


%R4931 and R02401 are specific variables used in the consumed exchange from the primary controller via LAN A and the ladder blocks. The other exchanges use other specific variables.

Exchange Variables for Each Consumed Exchange

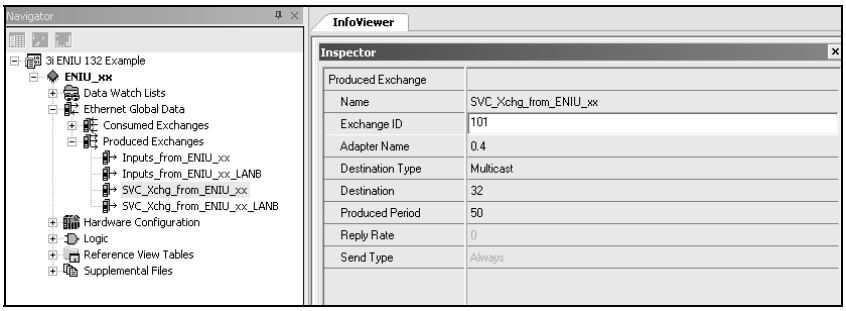
Exchange Name	Fault Variable	RCC Variable
Primary Controller – LAN A	%R04931	%R02401
Primary Controller – LAN B	%R04935	%R02801
Secondary Controller – LAN A	%R04933	%R02601
Secondary Controller – LAN B	%R04937	%R03801

Produced Exchanges



Both Enhanced Fault Reporting and Remote COMMREQ Calls are implemented via the SVC_Xchg_from_… (or RCC_ exchange if you imported the EGD configuration) exchanges in the Ethernet Global Data configuration. There would be one exchange for a single LAN application and two exchanges for a dual LAN. If your application does not already include the exchanges required, they must be added.

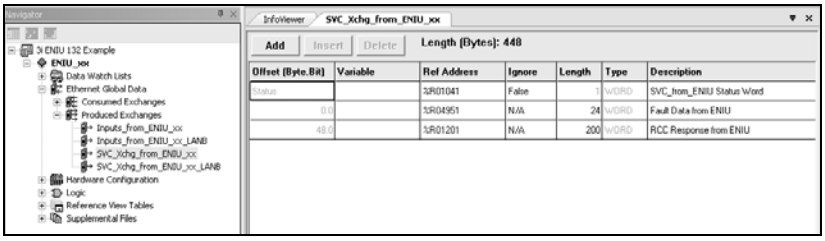
For each exchange, enter values for Exchange ID, Adapter Name, Destination Type, Destination & Produced Period:



The Exchange ID must be unique for each Ethernet NIU. The Adapter Name is the rack/slot of the Ethernet Transmitter Module that connects to the controller. The Destination Type is always Multicast for SVC exchanges. The Destination can be a common number used for all Ethernet NIUs. The Produced Period is coordinated with the Update Timeout for the exchange. See chapter 8 for recommendations for Production Periods and Update Timeout values.

Exchange Details

The %R04951 data area is used for Enhanced Fault Reporting and the %R01201 data area is used for Remote COMMREQ Call Commands. If BOTH Enhanced Fault Reporting and Remote COMMREQ Calls are to be used, the %R04951 data area MUST be before the %R01201 data area in the exchange details.



Adding Fault Logic Blocks for Fault Reporting and/or RCC in the ENIU Application

The ladder blocks for Enhanced Fault Reporting are already present in Version 1.3x and later Ethernet NIU targets. No additional work needs to be done to configure them. If Enhanced Fault Reporting will not be used, set the value of the variable *enable_ph1_flts* to 1. This can be done by setting the Initial Condition value to 1 and then downloading the change to the Ethernet NIU.

This completes the configuration of the Ethernet NIU application.

Adding Enhanced Fault Reporting and/or Generic RCC to the Controller Application

Symbolic Variables for Remote COMMREQ Calls

Using Remote COMMREQ Call commands does NOT require that specific symbolic variables be used in the controller, but it is recommended that the symbolic variables supplied with the templates or csv files be used.

Using one of the project templates will automatically declare the variables in the controller and set up the all exchanges with the same variables.

If the controller is not set up using a template, the variable file *RCCD_1.xx_variables.csv* (xx represents the release number portion of the filename) can be imported into the controller to create appropriate symbolic variables.

Symbolic Variables for the Enhanced Fault Reporting Block

Ethernet NIU Enhanced Fault Reporting requires that specific symbolic variables be used in the controller for all Ethernet Global Data exchanges. If the specified symbolic variables are not used, Enhanced Fault Reporting will not work properly.

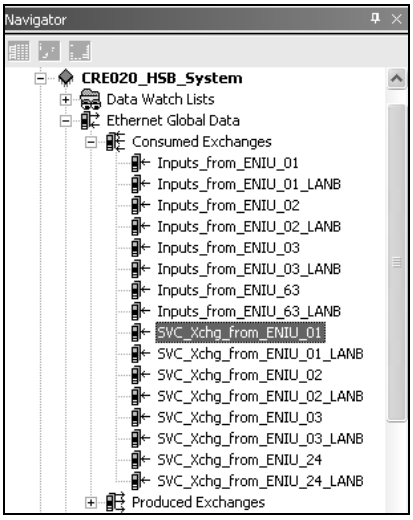
The symbolic variables must be declared as variables in the controller and published either internally or externally. Otherwise, either the controller program will not store to the PLC, or the PLC will fault when it attempts to go into Run mode.

The project templates will automatically declare the variables in the controller and set up the all exchanges with the correct variables. If the controller is not set up using a template, the variable file *ENIU_Faults_1.xx_variables.csv* (xx represents the release number portion of the filename) should be imported into the controller to create the symbolic variables.

Adding/Modifying Ethernet Global Data Exchanges in the Controller

Both Enhanced Fault Reporting and Remote COMMREQ Calls are supported by the *SVC_Xchg_* exchanges in the Ethernet Global Data configuration. There would be one produced exchange and one consumed exchange for a single LAN application and two produced exchanges and two consumed exchanges for a dual LAN configuration for each Ethernet NIU that uses these features. If your controller application does not have the exchanges required, the exchanges must be added to the application. It is not necessary that all Ethernet NIUs have the same features selected.

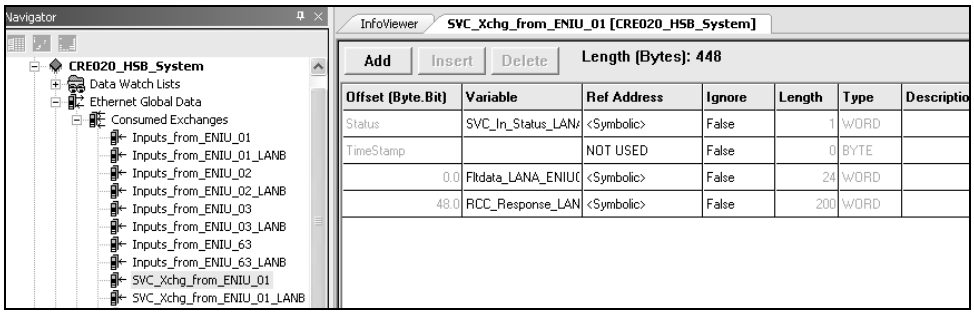
Consumed Exchanges



For each exchange added, enter the values for Group ID and Exchange ID that were used in the configuration of the exchange in the Ethernet NIU target (see steps above). Enter the Local Producer ID of the Ethernet NIU in the field Producer ID. The Adapter Name is the rack/slot of the Ethernet Transmitter Module that connects to the Ethernet NIU. The Update Timeout is coordinated with the Produced Period. See chapter 8 for recommendations for Production Periods and Update Timeout values.

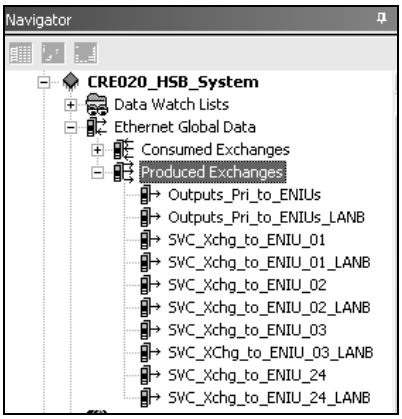
Exchange Details

The *Fltdata_LANx_ENIUyy* data area is used for Enhanced Fault Reporting and the *RCC_Response_LANx_ENIUyy* data area is used for Remote COMMREQ Call commands. If BOTH Enhanced Fault Reporting and Remote COMMREQ Calls are to be used, the *Fltdata_LANx_ENIUyy* data area MUST be before the *RCC_Response_LANx_ENIUyy* data area in the exchange details.



Fltdata_LANx_ENIUyy and *RCC_Response_LANx_ENIUyy* are specific variables used for the *SVC_Xchg_from_ENIUyy* exchange. Where x is A for the LAN A exchange, x is B for the LAN B exchange and yy is the Ethernet NIU number.

Produced Exchanges

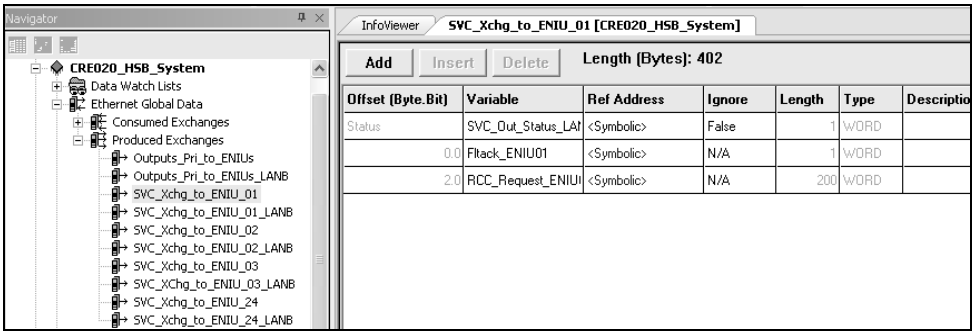


Both Enhanced Fault Reporting and Remote COMMREQ Calls are supported by the *SVC_Xchg_to…* exchanges in the Ethernet Global Data configuration. There would be one exchange for a single LAN application and two exchanges for a dual LAN configuration. If your application does not include the exchanges required, they must be added to the application.

For each exchange, enter values for Exchange ID and Destination used for configuration of the same exchange in the Ethernet NIU (see steps above). The Adapter Name is the rack/slot of the Ethernet Transmitter Module that connects to the Ethernet NIU. The Destination Type is always Multicast for SVC exchanges. The Produced Period is coordinated with the Update Timeout of the exchange. See chapter 8 for recommendations for Production Periods and Update Timeout values.

Exchange Details

The *Fltack_ENIUyy* and *ClearFaults_ENIUyy* data areas are used for Enhanced Fault Reporting and the *RCC_Request_ENIUyy* data area is used for Remote COMMREQ Call commands. If BOTH Enhanced Fault Reporting and Remote COMMREQ Calls are to be used, the *Fltack_ENIUyy* and *ClearFaults_ENIUyy* data areas MUST be before the *RCC_Request_ENIUyy* data area in the exchange details.



Fltack_ENIUyy, *ClearFaults_ENUIUyy* and *RCC_Request_ENIUyy* are specific variables that are used for BOTH the *SVC_Xchg_to_ENIU_xx* (LAN A) exchange and the *SVC_Xchg_to_ENIU_xx_LANB* (LAN B) exchanges.

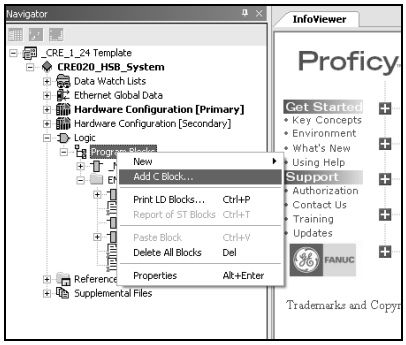
Controller Program Blocks

The controller uses program blocks for Enhanced Fault Reporting and for Remote COMMREQ Call commands.

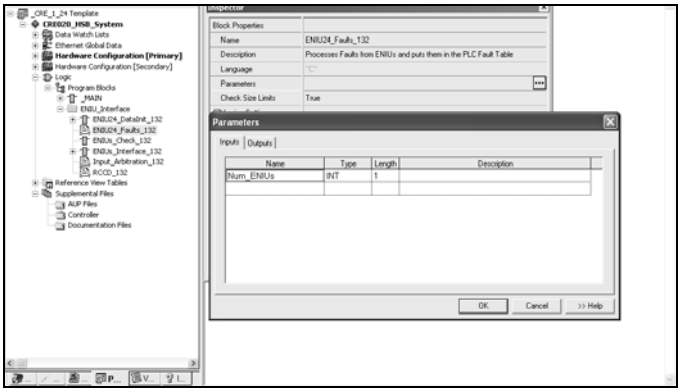
The block required for Enhanced Fault Reporting is *ENIUxx_Faults_xxx* and the block required for Remote COMMREQ Call commands is *RCCD_xxx*. These are C Blocks, so they are only available for use in PACSystems controllers.

Adding the C Blocks to the Controller Application

If your application does not already have them, you can add them by selecting Add C Block if you have the .gefElf file or by copying and pasting them from an application that already has the C blocks.

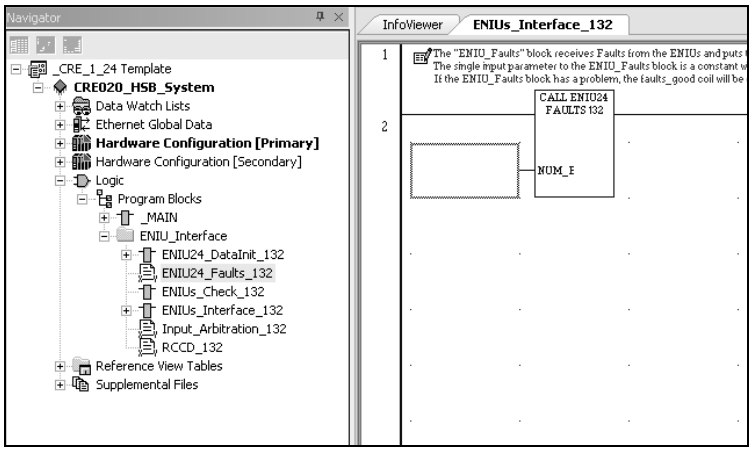


The single input parameter to the *ENIU_Faults* block is a constant which is the number of Ethernet NIUs in the system.



Using Fault Reporting in the Control Program

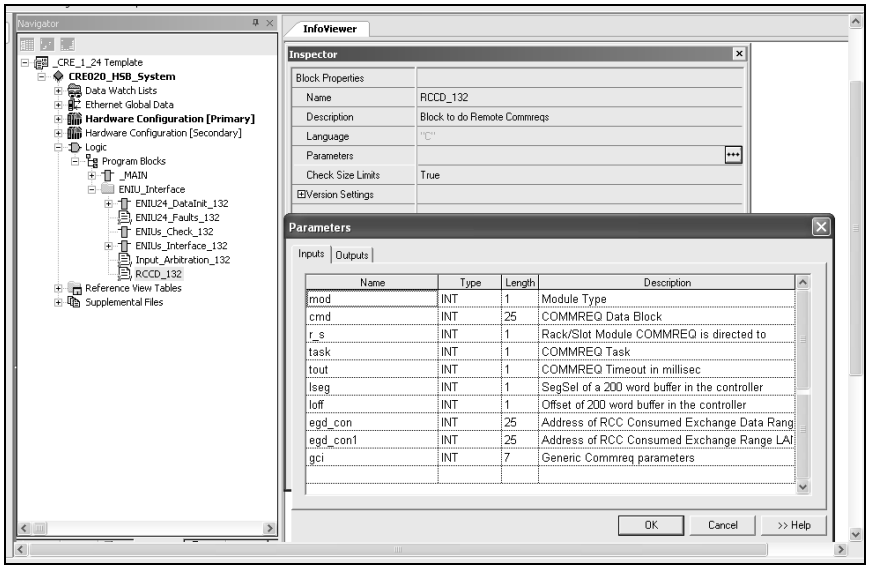
The *Fault* C Block needs to be called in the controller application and needs the number of Ethernet NIUs that are using Enhanced Fault Reporting entered on its input parameter *Num_ENIUs.*



Calling the block is all that is needed to enable Enhanced Fault Reporting. See chapter 10 for more details on Enhanced Fault Reporting.

Using Remote COMMREQ Calls in the Control Program

The input parameters to the C block must be set up as shown.



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).

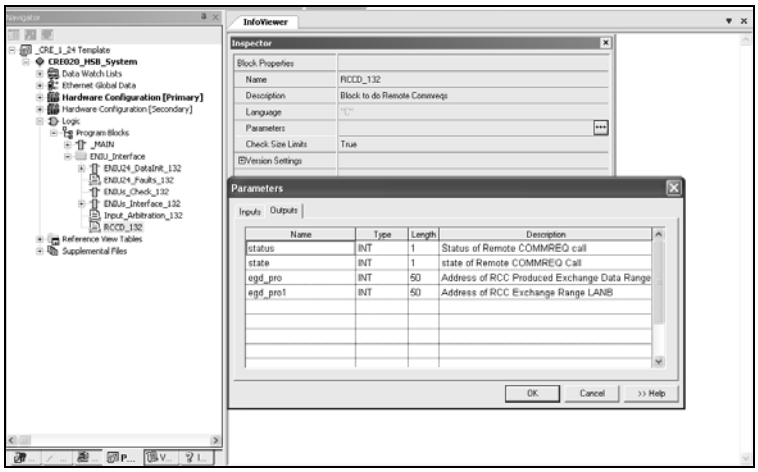
loff Starting reference number of the buffer. Enter a constant, i.e. 7001.

egd_con Pointer to the RCC Data Area in *SVC_Xchg_from_ENIU_xx* Exchange. Enter the starting register reference of the exchange data range. It must have an array length of 25.

egd_con1 Pointer to the RCC Data Area in *SVC_Xchg_from_ENIU_xx_LANB* Exchange. Enter the starting register reference of the exchange data range. It must have an array length of 25.

gci Pointer to the starting address of the gci parameter data.

The output parameters to the C block must be set up as shown below.



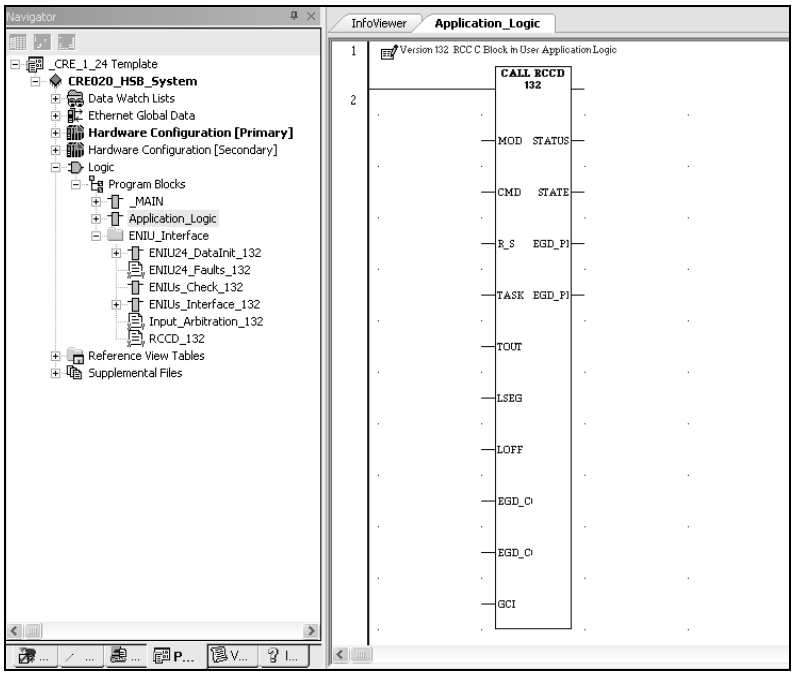
status Status of the Remote COMMREQ Call command. Enter a register reference. This is monitored to determine completion and success of the Remote COMMREQ Call command.

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

egd_pro Pointer to the data area in *SVC_Xchg_to_ENIU_xx* exchange. Enter the starting register reference of the exchange data. The range must have an array length of 50.

egd_pro1 Pointer to the data area in *SVC_Xchg_to_ENIU_xx* exchange. Enter the starting register reference of the exchange data. The range must have an array length of 50.

The Remote COMMREQ Call C Block needs to be called in the controller application. A separate copy of the block needs to be called for each Ethernet NIU that uses Remote COMMREQ Call commands (so that they can be controlled independently). This block requires input parameters and control logic to implement an RCC Command. See Chapter 12 for more details on implementing logic for Remote COMMREQ Call commands.



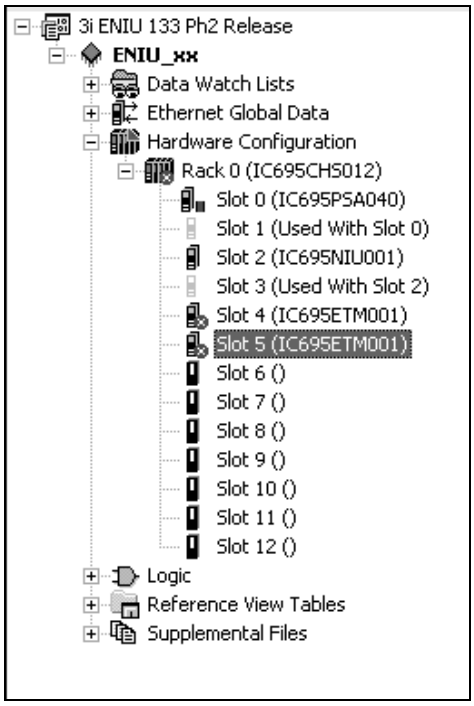
Adding a New Target to a Version 1.2x Application

If a newer version Ethernet NIU target is added to a version 1.2x application, the target must be modified as described in this section.

As described earlier, version 1.3x or later functionality of the Ethernet NIU application is not available in a version 1.2x application. If the additional functionality is required, the application and its targets must be upgraded. See previous section of this chapter for details on upgrading a version 1.2x application.

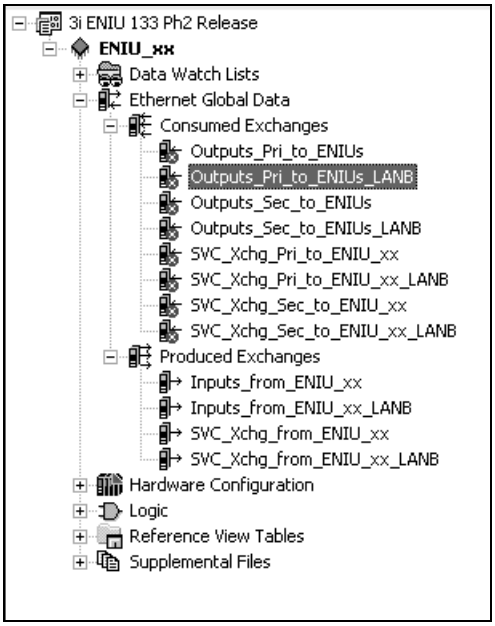
Modification to Hardware Configuration

Version 1.2x applications only support single LAN systems. In the version 1.3x or later Ethernet NIU target, delete the second Ethernet Transmitter Module in the hardware configuration.

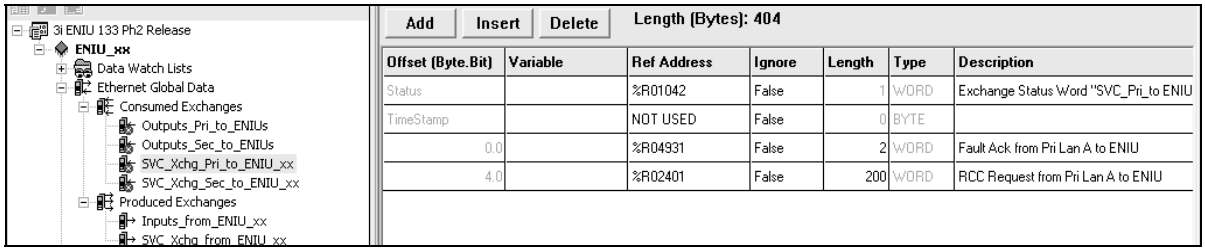


Modification to Ethernet Global Data Exchanges

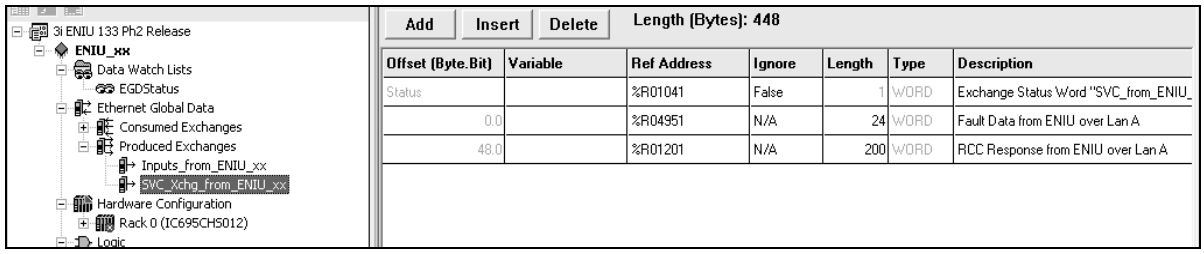
Delete ALL exchanges configured for the second LAN (LANB) for both the Consumed and Produced Exchanges.



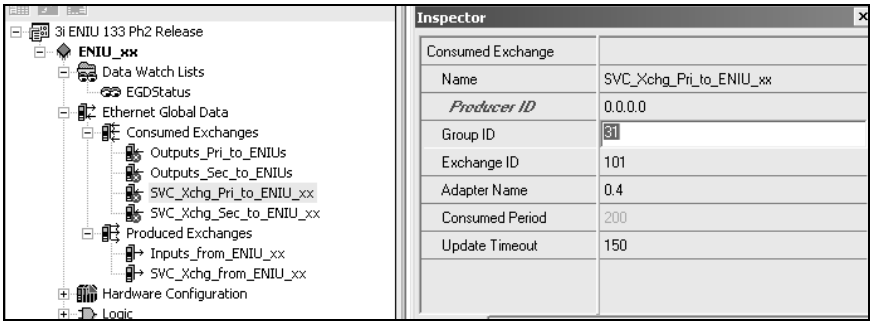
Version 1.2x applications do not support Enhanced Fault Reporting to the controller or clearing of faults to individual Ethernet NIUs. Delete the Data Area for Fault Reporting in each SVC exchange. Do this for both the *SVC_Xchg_Pri_to_ENIU_xx* and *SVC_Xchg_Sec_to_ENIU_xx* exchanges of the Consumed Exchanges.



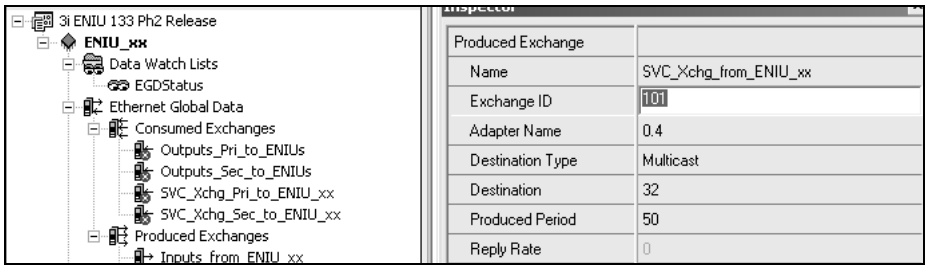
And the *SVC_Xchg_from_ENIU_x*x exchange of the Produced Exchanges:



In the Exchange's Properties of the SVC exchanges:



For both the *SVC_Xchg_Pri_to_ENIU_xx* and *SVC_Xchg_Sec_to_ENIU_xx* exchanges change the Group ID from 31 to 0 and the Exchange ID to 91 (for exchange from Primary controller) or 92 (for exchange from Secondary Controller).



For the *SVC_Xchg_from_ENIU_xx* exchange, change the Exchange ID from 101 to 90.

These examples assume that the default Exchange IDs and Group Numbers were used in the application. The Exchange IDs and Group Numbers used for the Ethernet NIU target must match those used in the controller target.

Appendix A Configuration Worksheets

This appendix contains configuration worksheets. Printed copies of these worksheets can be used to record configuration parameters for:

- The Primary Controller
- An Optional Secondary Controller
- Each Ethernet NIU I/O Station
- I/O Modules in the I/O Station
- *Inputs_from_ENIU*, the Ethernet Global Data exchange for the inputs sent from the Ethernet NIU to the controller(s).
- *Outputs_Pri_to_ENIUs*, the Ethernet Global Data exchange for the outputs sent to the Ethernet NIU from the primary controller.

Primary Controller

Parameters of the Ethernet Global Data

EGD Local Producer ID:	_____	NOTE: The Local Producer ID is set up as part of the template set to 10.10.10.101

Parameters of the Ethernet Transmitter Module for I/O LAN (for LAN A if dual LANs)

IP Address:	_____	IP Address of the Ethernet Transmitter Module
Subnet Mask:	_____	Subnet Mask of the Ethernet Transmitter Module
Gateway:	_____	If used

Parameters of the Ethernet Transmitter Module for Optional LAN B

IP Address:	_____	IP Address of the Ethernet Transmitter Module
Subnet Mask:	_____	Subnet Mask of the Ethernet Transmitter Module
Gateway:	_____	If used

Parameters of the Ethernet Transmitter Module for Other Communications

IP Address:	_____	IP Address of the Ethernet Transmitter Module
Subnet Mask:	_____	Subnet Mask of the Ethernet Transmitter Module.
Gateway:	_____	If used

Secondary Controller

Parameters of the Ethernet Global Data

EGD Local Producer ID:	_____	NOTE: The Local Producer ID is set up as part of the template set to 10.10.10.101

Parameters of the Ethernet Transmitter Module for I/O LAN (for LAN A if dual LANs)

IP Address:	_____	IP Address of the Ethernet Transmitter Module
Subnet Mask:	_____	Subnet Mask of the Ethernet Transmitter Module
Gateway:	_____	If used

Parameters of the Ethernet Transmitter Module for Optional LAN B

IP Address:	_____	IP Address of the Ethernet Transmitter Module
Subnet Mask:	_____	Subnet Mask of the Ethernet Transmitter Module
Gateway:	_____	If used

Parameters of the Ethernet Transmitter Module for Other Communications

IP Address:	_____	IP Address of the Ethernet Transmitter Module
Subnet Mask:	_____	Subnet Mask of the Ethernet Transmitter Module.
Gateway:	_____	If used

Ethernet NIU (Complete for Each ENIU I/O Station)

Parameters of the Ethernet Global Data

EGD Local Producer ID: _____

NOTE: The Local Producer ID is set up as part of the template set to 10.10.10.xx (with xx being the Ethernet NIU number)

Parameters of the Ethernet Transmitter Module for I/O LAN A

IP Address: _____ IP Address of the Ethernet Transmitter Module

Subnet Mask: _____ Subnet Mask of the Ethernet Transmitter Module

Gateway: _____ If used

Parameters of the Ethernet Transmitter Module for I/O LAN B

IP Address: _____ IP Address of the Ethernet Transmitter Module

Subnet Mask: _____ Subnet Mask of the Ethernet Transmitter Module

Gateway: _____ If used

Parameters of the Ethernet Transmitter Module for Other Communications (if used)

IP Address: _____ IP Address of the Ethernet Transmitter Module

Subnet Mask: _____ Subnet Mask of the Ethernet Transmitter Module.

Gateway: _____ If used

I/O Modules in the Ethernet NIU I/O Station

Slot	Module Type	Ref Addresses	COMMREQs

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). Defaults are shown in parentheses.

Parameters of the Ethernet NIU's Produced Exchange

Exchange Property Inspector

Exchange ID:	(1)	Match the Exchange ID of the Consumed (in controller) exchange, 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 five Ethernet NIUs.

Parameters of the Controller's Consumed Exchange

Exchange Property Inspector

Producer ID:		Ethernet NIU Producer ID.
Group ID:	2	Set to 2 for system with multiple Ethernet NIUs. Must match the Destination field in the ENIUs produced exchange
Exchange ID:	1	Do not change.
Adapter Name:		Ethernet Transmitter Module location in Controller configuration.
Update Timeout:	32	Based on the Production Period of the exchange.

Parameters of the Exchange

Offset	*Variable*	*Ref Address*	*Ignore*	*Length*	*Type*	*Description*
Status	*InEx_Status_LANA_ENIU_xx*		False	1	Word	EGD Status
Timestamp			False	0	Byte	
0.0	*StatusWords_LANA_ENIU_xx*		False	10	Word	ENIU Status
2.0			False		Bool	Discrete Inputs
xx.0			False		Int	Analog Inputs
	Note: Variable names or Reference Addresses can be used for discrete inputs and analog inputs			Enter the length of discrete inputs and analog inputs		

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. Defaults are shown in parentheses.

Parameters of the Ethernet NIU's Consumed Exchange

Exchange Property Inspector

Producer ID:		Producer ID of the controller
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

Exchange Property Inspector

Exchange ID:	(1)	Change only if the controller will produce more than one exchange).
Adapter Name:		ETM location in controller configuration
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	OutEx_Status_LANA		False	1	Word	EGD Status
0.0	ControlWords_LANA_B		N/A	10	Word	Control Data
20.0		%Q0001	N/A	2048	Bit	Discrete Outputs
276.0		%AQ0001	N/A	512	Word	Analog Outputs

Appendix B Input_Arbitration C Block

When dual LANs are used, two sets of inputs from the Ethernet NIUs are available to each of the redundant controllers. In a dual LAN system that is developed using Proficy Machine Edition, the C block Input_Arbration must be included in the controller application programs to determine which inputs should be used.

If version 1.3 or earlier templates have been used, the *Input_Arbitration* C block is automatically included and configured, so no additional work is needed.

If a template set was not used to create the application, the *Input_Processing* C block must be added to the application and set up as described below. The block will be a toolchest drawer export. The download needs to be imported into the toolchest. The block can then be copied from the Toolchest into the project. To copy the C block, select and drag the block while holding down the Ctrl key. Drop the block into the project.

This chapter provides information on configuring the C block in addition to the Point Fault/Data Quality features. It covers:

- *Input_* C Block
  - Adding the C Block to the Controller Logic
  - Symbolic Variable Use
  - Input Processing Error Codes
- Redundant Controller, Dual LAN
  - Switching Logic
  - Dedicated Signals
- Input Data Features
  - Point Fault Data
  - Data Quality

Point Fault/Data Quality Feature

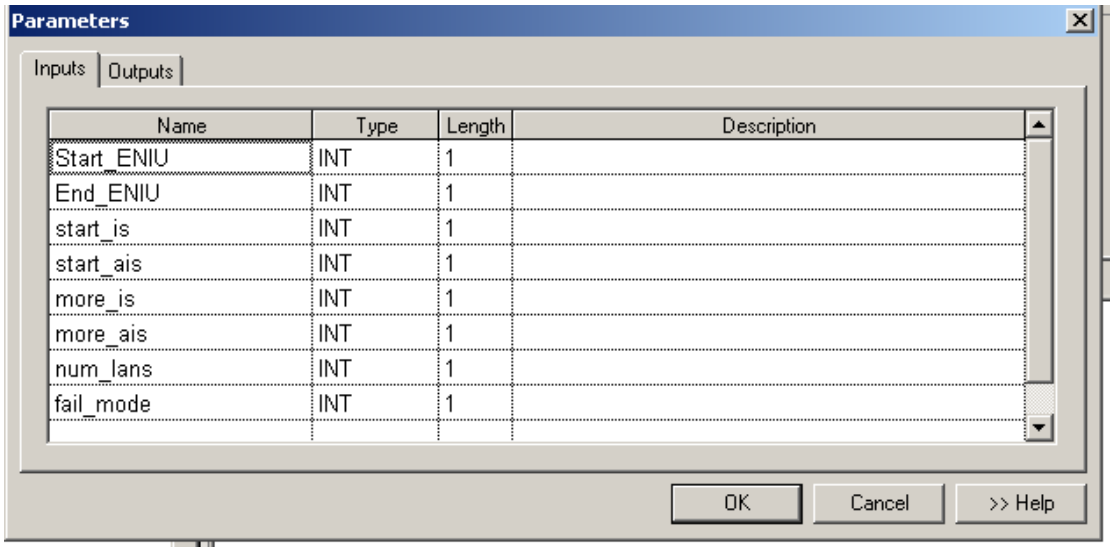
PPS Systems provide a Data Quality feature by using the Point Fault feature of the PACSystems controller. Point Faults are enabled by default in a PPS System. In a non-PPS System, the Point Faults feature is disabled and if needed must be enabled.

Operation is the same for a single LAN or dual LANs. For Dual LAN systems, the CPU sets Point Faults if communication to an Ethernet NIU is lost on both LANs. When communication to an Ethernet NIU is lost, the CPU sets Point Faults for all %I and %AI inputs. When there are dual LANs, two sets of inputs from the Ethernet NIUs are available to each of the redundant controllers. The setting/clearing of Point Fault data is performed by the *Input_Arbitration* block, which should be included in the controller application programs to determine which inputs should be used.

Configuring the Input Arbitration C Block

The inputs and outputs of the C block are described below.

Inputs for the C Block



Start_ENIU — Constant that is the Ethernet NIU number of the first Ethernet NIU, typically: 1

End_ENIU — Constant that is the number of the last Ethernet NIU. Ethernet NIUs should be numbered consecutively.

start_is — Constant that is the offset in %I memory where the first discrete input from the Start ENIU will be written. Inputs are packed consecutively in the controller. It does not need to start at 1, but must be on a byte boundary (33, 129 etc…).

start_ais — Constant that is the offset in %AI memory where the first analog input from the Start ENIU will be written. Inputs are packed consecutively in the controller.

more_is — Constant that is the offset in %I memory where additional discrete inputs from the Start ENIU will be written. Inputs are packed consecutively in the controller. It does not need to start at 1, but must be on a byte boundary (33, 129 etc…). Must not overlap with start_is range.

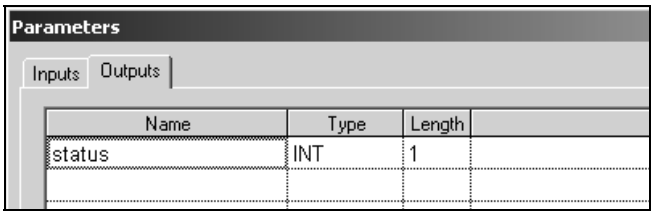
more_ais — Constant that is the offset in %AI memory where additional analog input from the Start ENIU will be written. Must not overlap with start_ais range.

*num_lans — 1= single LAN, 2= (not applicable), 3 = dual LAN with dual interfaces (2 Ethernet Transmitter Modules)

*fail_mode — specifies mode when there is communication fault with the Ethernet NIU, 0 = zero inputs, any other value (non-zero) = hold last state.

* Only available in *Input_Processing* C block

Output of the C Block



status — Status of the *Input_Arbitration* C block call. Enter a Variable of type Integer. This location will be monitored to determine completion and success of the configuration of the *Input_Arbitration* block. *Input_Arbitration* block will check all inputs to the block the first time it is executed, and will output an error of the form xxyy (in hex) if an error is found. xx is the Ethernet NIU number and yy is the Error Code. If the xx is 00, the error code is a common parameter. If no error is found the block returns a value of 1 and the inputs are provided to the controller. If an error is returned, the error must be corrected and the controller must be restarted to reset the *Input_Arbitration* block. If the *Input_Arbitration* block indicates an error in the status output, the inputs are not updated in the controller. See page 4-6 for the list of error codes.

Symbolic Variables for the Input Arbitration Function

The input arbitration function of the C block requires that specific symbolic variables be used in the controller for all Ethernet Global Data exchanges. If the specified symbolic variables are not used, input arbitration will not work properly.

The symbolic variables must be declared as variables in the controller and published either internally or externally. Otherwise, either the controller program will not store to the PLC, or the PLC will have a fault when it attempts to go into Run mode.

The PPS project templates automatically declare the variables in the controller and set up all the exchanges with the correct variables.

The variables that must be used on the *Input_from_ENIU_xx* exchanges are:

- *ENIUxx_LAN_y_InputsDiscrete* – Discrete Inputs from Ethernet NIU xx on LAN y.
- *ENIUxx_LAN_y_InputsAnalog* – Analog Inputs from Ethernet NIU xx on LAN y.
- *ENIUxx_LAN_y_InputsRegister* – Optional word inputs from Ethernet NIU xx on LAN y. These input are placed in the symbolic variable *ENIUxx_Register_Data* after input arbitration.
- *ENIUxx_LAN_y_Xtra_InputsDiscrete* – Optional additional discrete inputs from Ethernet NIU xx on LAN y, only used if all discrete inputs are all used and more inputs are needed.
- *ENIUxx_LAN_y_Xtra_InputsAnalog* – Optional additional analog inputs from Ethernet NIU xx on LAN y, only used if all analog inputs are all used and more are needed.

If the controller is not set up using the template, the variable file *Input_Arbitrationxxx_Variables.csv* should be imported into the controller to create the symbolic variables. All exchanges must be set up as described in chapter 3.

Input Processing – Error Codes

The input processing function requires configuration of the I/O parameters. The status variable on the output of the block contains a value of 0001 when the configuration is correct. Most error codes identify the number of the Ethernet NIU in the upper byte and the error code in the lower byte. Error codes that do not include an Ethernet NIU number have 00 in the upper byte and report the error code in the lower byte. It is best to display the error code in Hex.

Note: Inputs are not passed to the %I and %AI reference tables of the controller until the Input_Processing block is configured correctly and returning a status of 0001.

Error	*Error Code*	*Error*	*Error Code*
BAD_I_DATATYPE_LANA	xx10	I_MORE_DATASIZE_A_B_DIFFERENT	xx5c
BAD_AI_DATATYPE_LANA	xx11	AI_MORE_DATASIZE_A_B_DIFFERENT	xx5d
BAD_EX_DATATYPE_LANA	xx12	EX_MORE_DATASIZE_A_B_DIFFERENT	xx5e
BAD_I_DATATYPE_LANB	xx13		
BAD_AI_DATATYPE_LANB	xx14	I_MORE_DATA_SIZE_NOT_BYTE_MULTIPLE	xx60
BAD_EX_DATATYPE_LANB	xx15	HIGHEST_MORE_I_EXCEEDS_32000	xx61
BAD_I_DATASIZE_LANA	xx16	HIGHEST_MORE_AI_EXCEEDS_TABLE	xx62
BAD_AI_DATASIZE_LANA	xx17		
BAD_EX_DATASIZE_LANA	xx18	I_MORE_DATA_START_NOT_BYTE_BOUNDARY	0064
BAD_I_DATASIZE_LANB	xx19	START_ADDR_ZERO_WITH_I_MORE_DATA_LEN	xx66
BAD_AI_DATASIZE_LANB	xx1a	START_ADDR_ZERO_WITH_AI_MORE_DATA_LEN	xx67
BAD_EX_DATASIZE_LANB	xx1b	I_DATASIZE_PRI_XFER_DIFFERENT	xx71
I_DATASIZE_A_B_DIFFERENT	xx1c	I_DATASIZE_SEC_XFER_DIFFERENT	xx72
AI_DATASIZE_A_B_DIFFERENT	xx1d	AI_DATASIZE_PRI_XFER_DIFFERENT	xx73
EX_DATASIZE_A_B_DIFFERENT	xx1e	AI_DATASIZE_SEC_XFER_DIFFERENT	xx74
		EX_DATASIZE_PRI_XFER_DIFFERENT	xx75
		EX_DATASIZE_SEC_XFER_DIFFERENT	xx76
HIGHEST_I_EXCEEDS_32000	xx21	I_MORE_DATASIZE_PRI_XFER_DIFFERENT	xx77
HIGHEST_AI_EXCEEDS_TABLE	xx22	I_MORE_DATASIZE_SEC_XFER_DIFFERENT	xx78
		AI_MORE_DATASIZE_PRI_XFER_DIFFERENT	xx79
I_DATA_START_NOT_BYTE_BOUNDARY	0024	AI_MORE_DATASIZE_SEC_XFER_DIFFERENT	xx7a
		I_DATA_OVERWRITES_XTRA_I_DATA	0xa0
BAD_I_DATATYPE_PRI_XFER	xx41	AI_DATA_OVERWRITES_XTRA_AI_DATA	0xa1
BAD_I_DATATYPE_SEC_XFER	xx42		
BAD_AI_DATATYPE_PRI_XFER	xx43	I_XTRA_DATA_OVERWRITES_I_DATA	0xa3
BAD_AI_DATATYPE_SEC_XFER	xx44	AI_XTRA_DATA_OVERWRITES_AI_DATA	0xa4
BAD_EX_DATATYPE_PRI_XFER	xx45	START_ENIU_INPUT_BAD	0xb0
BAD_EX_DATATYPE_SEC_XFER	xx46	END_ENIU_LESS_THAN_START_ENIU	0xb1
BAD_I_MORE_DATATYPE_PRI_XFER	xx47	ENIU_GREATER_THAN_63	0xb2
BAD_I_MORE_DATATYPE_SEC_XFER	xx48	I_DATA_START_ADDRESS_BAD	00c1
BAD_AI_MORE_DATATYPE_PRI_XFER	xx49	AI_DATA_START_ADDRESS_BAD	00c2

Error	Error Code	Error	Error Code
BAD_AI_MORE_DATATYPE_SEC_XFER	xx4a	I_XTRA_DATA_START_ADDRESS_BAD	00c4
BAD_I_MORE_DATATYPE_LANA	xx50	AI_XTRA_DATA_START_ADDRESS_BAD	00c5
BAD_AI_MORE_DATATYPE_LANA	xx51		
BAD_EX_MORE_DATATYPE_LANA	xx52		
BAD_I_MORE_DATATYPE_LANB	xx53		
BAD_AI_MORE_DATATYPE_LANB	xx54	IO_LAN_STATUS_BAD	0xf2
BAD_EX_MORE_DATATYPE_LANB	xx55	ETM_LANA_BAD	0xf3
BAD_I_MORE_DATASIZE_LANA	xx56	ETM_LANB_BAD	0xf4
BAD_AI_MORE_DATASIZE_LANA	xx57	REGISTER_DATA_TOO_SMALL	0x31
BAD_EX_MORE_DATASIZE_LANA	xx58		
BAD_I_MORE_DATASIZE_LANB	xx59		
BAD_AI_MORE_DATASIZE_LANB	xx5a		
BAD_EX_MORE_DATASIZE_LANB	xx5b		

Redundant Controller (CRE), Dual LAN Considerations

Switching Logic

Switching logic is provided in CRE Controller templates. Switching conditions are generated in the ladder block *ENIUs_CRE_Check.* The switching conditions are solved in the block *ENIUs_Interface*. The basic operation of the switching logic is:

- When an Ethernet NIU is powered up, the Ethernet NIU defaults to LAN A. The application can use the switching logic to request that the Ethernet NIUs use LAN B.
- Reestablishment or loss of the inactive LAN does not cause any switching action to the inactive LAN.
- Transfer of controller causes the Ethernet NIU to follow the other controller using the same LAN as used before.

The switching logic can be customized for the application by revising the logic in the block *ENIUs_Interface*.

There are up to four rungs with similar structure, which are used for switching to primary controller LAN A, primary controller LAN B, secondary controller LAN A or secondary controller LAN B. The following control signals control switching between the LANs:

- *ControlWords_LANA_B[0].X[3]*,
- *ControlWords_LANA_B[0].X[13]*,
- *ControlWords_LANA_B[0].X[4],* and
- *ControlWords_LANA_B[0].X[14]*

The logic controlling these coils can be modified as needed.

Predefined Signals for Custom Switching Logic

The following predefined signals are available for use in any custom switching logic or HMI application. These signals are in addition to the standard system variables (# variables) that are available in controller applications. See the *PACSystems CPU Reference Manual,* GFK-2222 and *PACSystems Hot Standby CPU Redundancy Manual,* GFK-2308 for more details on system variables that are available for control or HMI animation.

Signal Name	*Description*
ENIUxx_on_PriA	ENIUxx is being controlled by primary controller on LAN A
ENIUxx_on_SecA	ENIUxx is being controlled by secondary controller on LAN A
ENIUxx_on_PriB	Dual LAN only ENIUxx is being controlled by primary controller on LAN A
ENIUxx_on_SecB	Dual LAN only ENIUxx is being controlled by secondary controller on LAN A
ENIUxx_CommOK_PrimaryA	ENIUxx communication good to primary controller on LAN A
ENIUxx_CommOK_SecondaryA	ENIUxx communication good to secondary controller on LAN A
ENIUxx_CommOK_PrimaryB	Dual LAN only ENIUxx comm good to primary controller on LAN A
ENIUxx_CommOK_SecondaryA	Dual LAN only ENIUxx comm good to secondary controller on LAN A
ENIUxx_Flt	No communication to ENIUxx
Ctl_by_Standby	Standby controller only – Standby controller in controlling one or more Ethernet NIUs
Pri_Ctl_On_A_B	Primary controller is controlling some Ethernet NIUs on LAN A and some on LAN B
Sec_Ctl_On_A_B	Secondary controller is controlling some Ethernet NIUs on LAN A and some on LAN B
NoENIUonPriLANA	No Ethernet NIUs are being controlled by primary controller on LAN A
NoENIUonPriLANB	No Ethernet NIUs are being controlled by primary controller on LAN B
NoENIUonSecLANA	No Ethernet NIUs are being controlled by secondary controller on LAN A
NoENIUonSecLANB	No Ethernet NIUs are being controlled by secondary controller on LAN B
ENIUonPriLANA	At least one Ethernet NIU is being controlled using primary controller on LAN A
ENIUonPriLANB	At least one Ethernet NIU is being controlled using primary controller on LAN B
ENIUonSecLANA	At least one Ethernet NIU is being controlled using secondary c-Controller on LAN A

Signal Name	*Description*
ENIUonSecLANB	At least one Ethernet NIU is being controlled using secondary controller on LAN B
Pri_New_Mstr	Operation has switched to the primary controller
Sec_New_Mstr	Operation has switched to the secondary controller
StatusWords_LANy_ENIU_xx	10 words of status data sent by Ethernet NIU to controller(s) **

** See chapter 9 for information on the ten words of status data sent by the Ethernet NIU to the controller(s)

Dedicated Signals

Certain signals should not be either deleted or renamed. These signals must be marked as Publish in their properties. Renaming, deleting or marking Publish as false will cause a failure when the program is downloaded to the controller. These signals are available for use in any custom switching logic or HMI annunciation.

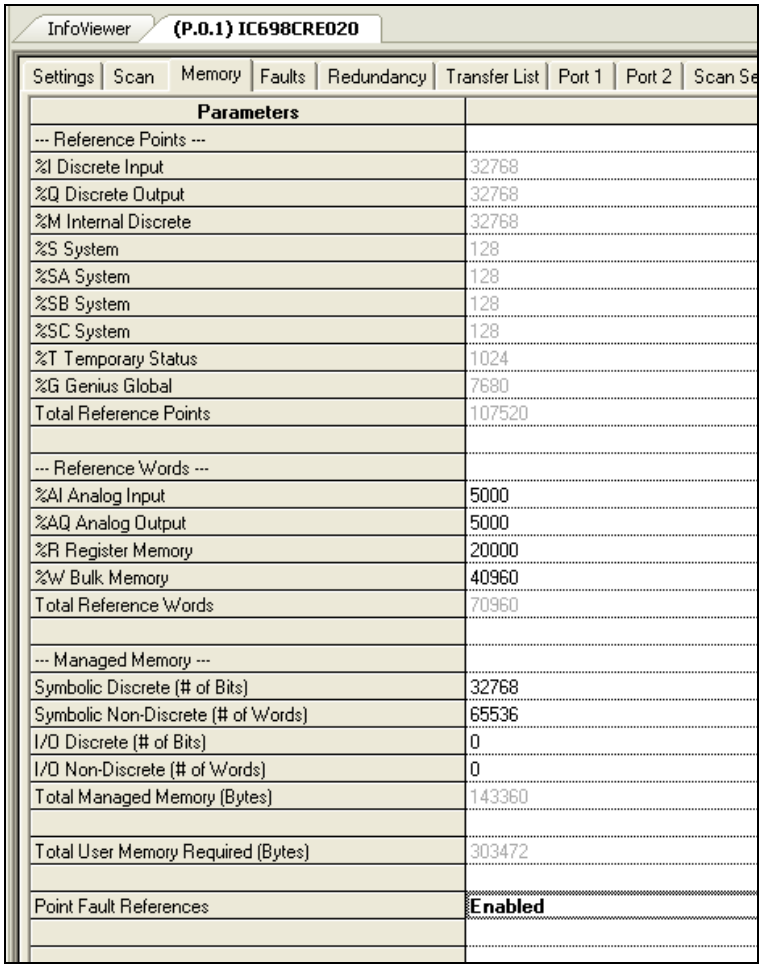
Signal Name	Signal Name
Signals for each Ethernet NIU (indicated by xx) and both LANs (indicated by y)	
ENIUxx_LAN_y_InputsDiscrete	*InEx_Status_LANy_ENIU_xx*
ENIUxx_LAN_y_InputsAnalog	*StatusWords_LANy_ENIU_xx*
ENIUxx_LAN_y_InputsRegister	*SVC_In_Status_LANy_ENIU_xx*
ENIUxx_LAN_y_Xtra_InputsAnalog	*SVC_Out_Status_LANy_ENIU_xx*
ENIUxx_LAN_y_Xtra_InputsDiscrete	*Fltdata_LANy_ENIUxx*
ENIUxx_LAN_y_Xtra_InputsRegister	*RCC_Response_LANy_ENIUxx*
Signals for each Ethernet NIU (indicated by xx)	
RCC_Request_ENIUxx	*fltbuf_eniu_xx*
Fltack_ENIUxx	*fltptr_eniu_xx*
ENIUxx_Register_Data	*ENIUxx_Register_Xtra_Data*
Signals for both LANs (indicated by y)	
OutEx_Status_LANy	
Signals common to all Ethernet NIUs and both LANs	
ControlWords_LANA_B	*Sw_Pri_A*
Eniuoffsetarray	*Sw_Pri_B*
LANA_LSW	*Sw_Sec_A*
LANB_LSW	*Sw_Sec_B*

Input Data Features

Point Fault Data

Discrete and Analog inputs have a property known as Point Faults References. The *Input_Arbitration* C block will set the Point Fault reference for all input points that are sent by an Ethernet NIU that has lost communications with the controller. When communications are lost to an Ethernet NIU, ALL the input points from that Ethernet NIU will have their Point Fault reference set to true.

Point Fault References must be enabled in the Hardware Configuration of the CPU module (on the Memory tab) for this feature to work. PPS controllers have the Point Fault Reference set to true by default. Inputs that are sent by I/O other than Ethernet NIUs are not affected by this feature.



Parameters	
--- Reference Points ---	
%I Discrete Input	32768
%Q Discrete Output	32768
%M Internal Discrete	32768
%S System	128
%SA System	128
%SB System	128
%SC System	128
%T Temporary Status	1024
%G Genius Global	7680
Total Reference Points	107520
--- Reference Words ---	
%AI Analog Input	5000
%AQ Analog Output	5000
%R Register Memory	20000
%W Bulk Memory	40960
Total Reference Words	70960
--- Managed Memory ---	
Symbolic Discrete (# of Bits)	32768
Symbolic Non-Discrete (# of Words)	65536
I/O Discrete (# of Bits)	0
I/O Non-Discrete (# of Words)	0
Total Managed Memory (Bytes)	143360
Total User Memory Required (Bytes)	303472
Point Fault References	**Enabled**

Data Quality

Refer to Proficy Process System documentation for use of Data Quality feature with in PPS Controller Function Block.

A

Additional error codes, 13-19

Addresses written to by EGD Exchanges, 11-2

B

Backplane locations for ENIU, 2-3

Battery, 1-6

Battery installation, 2-3

C

Cable length, 1-10

Cable Lengths and Shielding, 2-6

CE Mark installation requirements, 2-2

Clearing Faults, 10-6

COMMREQ Error Codes by Module Type, 13-45

COMMREQ Status Word, 12-24

COMMREQ Status Word error codes, 12-29

COMMREQs, 1-11
- 1, Get Device Status, 13-32
- 1, Get HART Device Information, 13-27
- 1, Send Device Explicit, 13-3
- 13, Dequeue Datagram, 13-21
- 14, Send Datagram Command, 13-24
- 15, Request Datagram Reply, 13-25
- 2, Get Master Status, 13-34
- 2, Send HART Pass-Thru Command, 13-29
- 4, Get Detailed Device Status, 13-7
- 4, Get Device Diagnostics, 13-37
- 5, Get Status Information, 13-9
- 5, Read Module Header, 13-38
- 6, Clear Counters, 13-40
- 6, Get Input Status from a Device, 13-11
- 7, Send Device Explicit Extended, 13-3, 13-13
- 8, Enable/Disable Outputs, 13-20
- 9, Read Module Header, 13-16
- E201, Send Data Command, 13-42
- E501, Parameter Load, 13-41
- for an RX3i Profibus Master Module, 13-32
- for DeviceNet Master Modules, 13-3
- for Genius Bus Controller Modules, 13-20
- for HART Communications, 13-26
- for High-Speed Counter Modules, 13-42
- for Modbus RTU Master, 13-43
- for RX3i and Series 90-30 Motion Controller Modules, 13-41
- Supported by Remote COMMREQ Calls, 13-2

COMMREQs in the Local Logic, 11-3

COMMREQs, general information
- memory type codes, 13-47

Configuring the Ethernet NIU, 5-2

Configuring the Ethernet NIU without templates, 6-1, 7-1

Configuring the PC-Based Controller, 5-3

Controllers on the Network, 1-14

D

Documentation, 1-3

Dual Ethernet LAN, 1-18

E

EGD Exchanges, 8-2

EGD Exchanges for Remote COMMREQ Calls, 12-5

EGD Timing for the Project Templates, 8-11

Embedded switches, 2-8

ESD protection
- CE Mark requirements, 2-2

Ethernet Cable, 2-8

Ethernet Global Data, 1-20

Ethernet NIU, 1-5

Ethernet ports, 1-10

Ethernet Transmitter Module, 1-8

Expansion baseplates, 1-12

F

Fatal Error codes, 13-19

Fault Data, 10-2

Fault Handling, 10-5

Fault Reporting, 16-3

Fault Tables, 10-3

Firmware upgrades, 2-7

Firmware Upgrades
- Ethernet Transmitter Module, 1-11

Function Block for Modbus Master, 15-4

G

Generic RCC, 16-3

H

Hardware Configuration for Modbus Master, 15-3
Hazardous Locations, 2-2
HMI and External Communication Devices, 2-4

I

I/O Station, 1-4
I/O support, 1-12
IC693CHS392, 1-12
IC693CHS398, 1-12
IC695ETM001, 1-8
IC695NIU001, 1-5
IC698ACC701, 1-6
Inputs for the C Block, 12-18
Inputs_from_ENIU, A-6
Inputs_from_ENIU_xx, 8-2
Installation, 2-1
IP Address
  checking for duplicated, 10-9
IP Address of the Ethernet NIU, 10-9

L

LEDs, 2-15
Local Logic Block, 11-1
Local User Block, 1-19
LOG command, 10-11

M

Max-ON CPU Controllers, 2-11
MB_P1, 15-3
MB_P2, 15-3
Modbus Master, 12-21
Modbus Master for the Ethernet NIU, 15-2
Modbus Master Function Codes, 15-11
Modbus Query 8002, 15-11
Modbus Query 8006, 15-11
Modules and Baseplates in the I/O Station, 1-12

N

Network connection, 10-17
Number of Ethernet Interfaces, 1-15

O

Operation, 1-19
Outputs of the C Block, 12-19
Outputs_Pri_to_ENIU, 8-2
Outputs_Pri_to_ENIUs, A-8
Outputs_Sec_to_ENIU, 8-2

P

PING command, 10-10
Producer Period and Consumer Update Timeout Settings, 8-11
Programmer connection, 2-7

R

RCC “C” Block, 12-16
RCC_Pri_request_to_ENIU_xx, 12-6, 12-7
RCC_response_from_ENIU_xx, 12-6, 12-8
RCC_Sec_request_to_ENIU_xx, 12-6
Read Diagnostics, 12-22
Read RCC Command at Switchover, 12-27
Redundant Controllers, 2-11
Redundant Controllers using Network Switch Devices, 2-10, 2-13
Redundant Controllers with Multiple I/O Stations, 2-12
Redundant Ethernet Cable Connections, 2-14
Remote COMMREQ Call, 12-2
Restricted Addresses, 11-2
Return Status, 12-20

S

Serial ports, 2-5
SNTP Time Synchronization, 8-13
Specifications
  Ethernet NIU, 1-7
  Ethernet Transmitter Module, 1-10

STAT Command, 10-12
STAT LED, 10-13
Station Manager, 10-8
  Ethernet Transmitter Module, 1-11
Surge protection, 2-2
SVC (Service Exchange) Operation, 8-9
Symbolic Variables for Fault Handling, 10-7

T

TALLY command, 10-17
Templates
  for RX3i Ethernet NIUs, 1-16
  sets, 1-17
    using, 3-1
Testing the Network using the PING command, 10-10

U

Upgrading, 16-1