SIEMENS

SIMATIC

Industrial Ethernet Module (IEM)

Manual

Edition 09/2005
A5E00322186-02

Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring to property damage only have no safety alert symbol. The notices shown below are graded according to the degree of danger.

Danger
indicates that death or severe personal injury **will** result if proper precautions are not taken.

Warning
indicates that death or severe personal injury **may** result if proper precautions are not taken.

Caution
with a safety alert symbol indicates that minor personal injury can result if proper precautions are not taken.

Caution
without a safety alert symbol indicates that property damage can result if proper precautions are not taken.

Notice
indicates that an unintended result or situation can occur if the corresponding notice is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel.** Within the context of the safety notices in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

Prescribed Usage

Note the following:

Warning
This device and its components may only be used for the applications described in the catalog or the technical description, and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens.
Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

Trademarks

Disclaimer of Liability

Siemens AG
Automation and Drives
Postfach 4848
90437 NÜRNBERG
GERMANY

Preface

Purpose of the Manual

This manual provides configuration and diagnostic instructions for the Industrial Ethernet Module (IEM).

This manual is intended for software and hardware engineers responsible for upgrading a plant to PCS 7, or for expanding a plant using new S7-400 controllers.

Required Basic Knowledge

Use of this product requires a general knowledge of the field of automation engineering, and some knowledge of PCS 7 and/or APACS+.

In addition, staff should know how to use computers under the Microsoft Windows 2000/XP operating system.

Certification

- UL508

CE Labeling

- CE

Further Support

If you have any technical questions, contact your Siemens representative or distributor.

http://www.siemens.com/automation/partner

Training Centers

Siemens offers a number of training courses to familiarize you with the SIMATIC S7 automation system. Contact your regional training center or our central training center in D 90327 Nuremberg, Germany for details:

Telephone: +49 (911) 895-3200.

Internet: http://www.sitrain.com/

Changes from the previous manual

This manual provides the following additions:

- Adding IEM to IEM functionality for APACS+ communication (using GW_SEND and GW_REC blocks).
- Configuration changes for NIM32.

A&D Technical Support

Worldwide, available 24 hours a day:

United States: Johnson City, TN	**Worldwide:** Nürnberg	**Asia / Australia**: Beijing
Technical Support and Authorization Local time: Monday to Friday 8:00 AM to 5:00 PM Telephone:+1 (423) 262 2522 or +1 (800) 333-7421 (USA only) Fax:+1 (423) 262 2289 Mail to: techsupport.sea@siemens.com GMT: -5:00	**Technical Support** 24 hours a day, 365 days a year Phone:+49 (180) 5050-222 Fax:+49 (180) 5050-223 E-Mail: ad.support@siemens.com GMT:+1:00 **Authorization** Local time: Monday to Friday 8:00 AM to 5:00 PM Phone: +49 (180) 5050-222 Fax: +49 (180) 5050-223 Mail to: ad.support@siemens.com GMT: +1:00	**Technical Support and Authorization** Local time: Monday to Friday 8:00 AM to 5:00 PM Phone:+86 10 64 75 75 75 Fax:+86 10 64 74 74 74 Mail to:ad.support.asia@siemens.com GMT:+8:00
The languages of the SIMATIC Hotlines and the authorization hotline are generally German and English.		

Service & Support on the Internet

In addition to our documentation, we offer our knowledge base online on the Internet at: http://support.automation.siemens.com where you will find the following:

- The newsletter, which constantly provides you with up-to-date information on your products.
- The right documents from our Search function in Service & Support.
- A forum, where users and experts from all over the world exchange their experiences.
- Your local representative for Automation & Drives through our representative's database.
- Information on field service, repairs, spare parts and more under "Services".

Table of Contents

1 Product Description

The Industrial Ethernet Module (IEM) is a configurable communications interface that transfers data between functions blocks of the APACS+™/QUADLOG® system and function blocks of the SIMATIC S7/PCS 7 system, or between function blocks on different APACS+/QUADLOG MBUS segments. The communication function blocks are primarily configured with standard tools of the APACS+/QUADLOG and S7/PCS 7 systems.

Active connections between function blocks in the APACS+/QUADLOG and SIMATIC S7/PCS 7 systems are not affected by the configuration and connection of other function blocks.

The IEM is configured with a single, static IP Address for its single Ethernet interface.

Two IEMs can be used in a redundant pair. The function block configuration (both APACS+ and S7) controls the redundancy.

The S7 CPUs and APACS+ resources can also be redundant. This redundancy has no effect on the function block configuration.

The S7/PCS 7 system provides function blocks that are used to exchange data with the corresponding function blocks in the APACS+/QUADLOG system via the IEM with a minimum of configuration work. S7 function blocks for the following APACS+/QUADLOG standard functions are available: GW_SEND and GW_REC. Each block can transfer 32 REALS, 32 BOOLS, and 16 WORD values, and multiple blocks can be used to give the ability to send more data.

GW_SEND and GW_REC are similar to the S7 BSEND/BRCV blocks. SIMATIC Manager NetPro is used to configure the connections between S7 and the IEM for data to be transferred between systems.

The IEM can also be used as a replacement for the Rack mounted Network Interface (RNI), Rack mounted Industrial Computer (RIC), or Rack mounted Industrial Server stations used as dedicated APACS+ NIM32 MBUS communications interfaces.

1.1 Product Components

The IEM consists of the following:

- SIMATIC Box PC 620 Industrial PC, which does not have a keyboard, mouse, or monitor. For improved reliability, rotating storage media is not used. Instead, the IEM uses flash-memory storage media, which is pre-loaded with the run-time software.
- Bracket for installation.
- MBUS interface card for connection to the MBUS.
- Installation CD, which includes:
  - installation for the NIM32 user interface
  - communication blocks for APACS+, QUADLOG, and S7 controllers
  - user documentation and help files

Configuration of the IEM includes the following:

- IP address and computer name for IP communications. These parameters are configured in the gateway.ini file.
- Node, rack, and slot address as three separate parameters for the MBUS. These parameters are configured in the gateway.ini file.
- NIM32 security files (optional). If the USB holds the files remoteconfigurationsecuritytable.txt and clientaccesstable.txt, then both files will be copied to the IEM folder c:\etc\ (see *SG39NIM32-1*, *Section Client Access Table Operation* on the APACS+ Electronic Manuals CD for details.)
- Communications between IEMs is defined in the iem2iem.txt file in order to establish APACS+/QUADLOG peer-to-peer communications.

Notice

The customer must have access to a USB flash drive, which is a common, pocket-sized, memory device that is plugged directly into a USB port. The USB flash drive will be displayed in Windows Explorer as a removable drive, or by the brand name of the device. An 8 MB or larger USB flash drive will be sufficient.

1.2 Technical Specifications

Component or Category	Description
Stainless Steel Housing	298x301x100mm
Power Supply	110/230 VAC
Weight	Approx. 6kg
Ambient temperatures during operation	5-45ºC
Humidity	5-80% @ 25ºC
Heat Dissipation	Maximum permissible ambient air temperature for the IEM is 45ºC.

2 Installation

The IEM connects to the APACS+ MBUS and to the S7 plant bus Industrial Ethernet, which uses the TCP/IP protocol.

Please refer to the SIMATIC Box PC 620, Getting Started Guide (A5E00131477-03) for complete installation details. This document is packaged with the IEM.

Basic installation procedure:

1. Mount the IEM to a wall or rack using included flanges.
2. Connect AC power.
3. Connect MBUS.
4. Connect Ethernet cable.
5. Install the software by clicking the **setup.exe** file on the installation CD.

   Notice: for APACS+ Function Blocks, you must have *4-mation* installed.

   Notice: For S7 Function Blocks, you must have STEP 7 installed.

2.1 Connection

The IEM uses the following ports. The ports are named in accordance with the labels on the housing.

- Ethernet: Port with 8-pin RJ45 plug that connects to the Industrial Ethernet plant bus (SIMATIC NET).
- MBUS: A 15-pin female port for connection directly to the APACS+ communication bus through a Y cable.
- Power: 110/220 AC power connection.
- USB: The port used for configuring the IEM, using a customer-provided USB Flash Drive.

The connection between the IEM and the MBUS consists of a 15-pin interface connected to an MBUS Y adapter cable (P/N 16137-215 – supplied with the IEM). The other end of the Y connector is attached to local bus segment of the MBUS system. Please note that the total length of a local bus segment in the MBUS system must not exceed 18 meters (60 feet) using standard cables, and up to 168 meters (550 feet) max only when using MBI extender cabling for PCs.

The signal cables of the MBUS are electrically isolated from the IEM. The local bus cable shield is in contact with the IEM housing via the MBUS front panel.

2.2 Replacing an IEM

When replacing an IEM, be sure to use the same settings for all configuration parameters in the new IEM as in the old one.

Notice

The MBUS cards shipped with the IEM have no switches. Configuration is done through the USB flash drive.

Notice

Save all configuration files after configuring an IEM, so they can be reloaded into a new IEM when replacing a module. (Once the IEM time and date have been set, remember to remove the time and date configuration items from the file. Otherwise, the new IEM will have an old time and date.) See section 3.4 Storing the Current Configuration.

3 Configuration – Basic USB Flash Drive Operation

3.1 Startup with Saved Data in the Flash Memory

The operating system, the programs for the IEM functions, the addresses specified for the S7 connections, and the addresses of partner IEMs for APACS+/QUADLOG communication are stored in the flash memory of the IEM.

3.2 Configuring the IEM with a USB Flash Drive

All user configuration and status communication with an IEM is accomplished using a USB flash drive folder that is the same name as the IEM. This allows for configuring several different IEMs using one USB flash drive.

When a USB flash drive is inserted into an IEM, the IEM will look on the root of the USB flash drive for a folder with the same name as the IEM computer name (the default IEM name is **GATEWAY**). If the folder is not found, the IEM will create it, and copy its current configuration files to the new folder.

When changing an IEM computer name, the IEM will create a new folder on the USB flash drive, if necessary, with the same name as the new IEM computer name.

IEMs prior to version 2.0 used the root of the USB flash drive for all configurations.

The IEM will write the following files to the IEM name folder of the USB flash drive,

- oldgateway.ini (previous configuration)
  - The existing configuration parameters will be written to this file, overwriting any existing copy and creating the file if it does not exist.
- Status.txt (status info)
  - Status information will be appended to this file if it already exists. If it does not exist, it will be created.
- Existing NIM32 configuration files are copied to the USB flash drive:
  - The IEM file "c:\etc\remoteconfigurationsecuritytable.txt" is written onto the USB as the file "oldremoteconfigurationsecuritytable.txt"
  - The IEM file "c:\etc\clientaccesstable.txt" is written onto the USB flash drive as the file "oldclientaccesstable.txt".
- The current IEM connection table configuration is copied to the USB flash drive:

- A file named "oldiem2iem.txt" is copied to the USB flash drive. This file identifies the configuration of the IEM to IEM connections.

Notice

Initialize the IEM with proper configuration parameters before connecting it to the Ethernet and MBUS networks.

The settings are specified in a file on the customer-provided USB flash drive (see section IEM Settings).

Notice

A USB hard drive cannot be used as a USB flash drive.

Notice

The first time a new type of USB flash drive (new manufacture or new model) is introduced to an IEM, the IEM will see the device as new hardware. The result is that the IEM will beep three times, then a few seconds later, the IEM will beep three more times. The status.txt will show that the current configuration was copied to the flash drive twice. It is possible that the IEM will then reboot one time.

3.3 Implementing the New Configuration

1. Insert the USB flash drive into the IEM. A folder named GATEWAY will be created which contains the files with the initial, default configuration.
2. Remove the USB flash drive from the IEM and insert it into your other Windows system.
3. Copy and rename the files in the GATEWAY folder.
4. Edit the parameters as required. For example, copy/rename the oldgateway.ini file to gateway.ini, then edit the default settings of the parameters as required to uniquely configure the IEM.
5. With the IEM powered off, plug the USB flash drive with the new gateway.ini file into the IEM.

Notice

Rather than removing power, you can reset the IEM by inserting a pointed object (e.g. an opened-up paper clip) into the **RESET** port on the side of the IEM.

---

6. Power on the IEM. The IEM reads the new configuration and writes the old configuration to a file called oldgateway.ini. The previous configuration of the IEM is stored in this file. Another file, called status.txt, is written containing the results, including errors and warnings, of the configuration change.
7. The IEM beeper will sound after about one minute, indicating that the USB flash drive can be removed.

3.4 Storing the Current Configuration

Retrieve the current configuration of the IEM by plugging in the USB flash drive at any time after the IEM has been powered on for more than a minute. The current configuration is written to the folder with the same name as the IEM computer name on the memory device as a file called **oldgateway.ini**. The IEM beeper will sound, indicating that the USB flash drive can be removed.

4 Configuration - MBUS/Ethernet Network Settings

4.1 Global Configuration by Text File on a USB Flash Drive during Powerup

Currently, there is a configuration program for the NIM32. This will continue to be used only for configuring the NIM32 running on the IEM. The configuration parameters handled by this program are unchanged from the NIM32 configuration parameters available in the ProcessSuite R5 version of the NIM32.

The IEM starts automatically when the power supply is turned on. The IEM operating system boots from the internal flash memory and starts the IEM program. While the IEM program is starting up, the configuration data is read from the USB flash drive. This data is permanently stored in the flash memory of the IEM.

To change the configuration, there must be a file called **gateway.ini** on the USB flash drive, in the folder named for the IEM. This file can be created from scratch, or the oldgateway.ini file can be edited and saved as gateway.ini. The file oldgateway.ini is one of the current configuration files obtained by inserting the USB flash drive in a running IEM. Edit the configuration parameters in this file using a parameter = value format, with one parameter per line. The following parameters are supported:

MBUS Parameters

Node Rack Slot	These parameters define the MBUS address for the IEM. The value of these in combination must form a unique address on the MBUS. The Node can range from 0 to 63, the Rack from 1 through 16, and the Slot from 1 through 16. The IEM will reboot if any of these parameters are changed.

Diagnostics

DiagnosticLevel	These set the diagnostic logging level for the IEM. There are 6 settings, 0 through 5.
DiagLevel	
0	Errors only.
1	Errors only.
2	Medium logging (some warnings and all errors).
3	Medium Logging and periodic statistical tag information.
4	High Logging (and medium logging messages also).
5	High Logging and periodic statistical tag information.

Notice

Setting the diagnostic level above 1 increases the amount of information the IEM outputs, which may result in a slowing of data transfer. Use of level 0 is recommended unless otherwise indicated.

System Settings

IPAddress SubnetMask	These specify the IP address and the subnet mask for the IEM. If IP address is not unique, communcations may not be established with the IEM. Changing this parameter will cause the IEM to reboot.
ComputerName	This parameter specifies the computer name for the IEM. Changing this parameter will cause the IEM to reboot. **Notice**: The Computer Name should be limited to only letters and numbers, and not longer than 15 characters. This syntax restriction is due to a limitation of both Microsoft Windows and the NIM32 software.
TimeZone	This parameter is used to set the time zone for the IEM. It can be one of the following values: International Date Line, Midway Island, Midway Is, Samoa, Hawaii, Alaska, Pacific Time, Tijuana, Arizona, Chihuahua, Mazatlan, Mountain Time, Central America, Central Time, Guadalajara, Mexico City, Monterrey, Saskatchewan, Bogota, Lima, Quito, Eastern Time, Indiana, Atlantic Time, Caracas, Santiago, Newfoundland, Brasilia, Buenos Aires, Georgetown, Greenland, Mid-Atlantic, Azores, Cape Verde Island, Cape Verde Is, Casablanca, Monrovia, Greenwich Mean Time, Dublin, Edinburgh, Lisbon, London, UTC, Zulu, Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna, Belgrade, Bratislava, Budapest, Ljubljana, Prague, Brussels, Copenhagen, Madrid, Paris, Sarajevo, Skopje, Warsaw, Zagreb, West Central Africa, Athens, Istanbul, Minsk, Bucharest, Cairo, Harare, Pretoria, Helsinki, Kyiv, Riga, Sofia, Tallinn, Vilnius, Jerusalem, Baghdad, Kuwait, Riyadh, Moscow, St Petersburg, Volograd, Nairobi, Tehran, Abu Dabi, Muscat, Baku, Tbilisi, Yerevan, Kabul, Ekaterinburg, Islamabad, Karachi, Tashkent, Chennai, Kolkata, Mumbai, New Delhi, Kathmandu, Almaty, Novosibirsk, Astana, Dhaka, Sri Jayawardenepura, Rangoon, Bangkok, Hanoi, Jakarta, Krasnoyarsk, Beijing, Chongqing, Hong Kong, Urumqi, Irkutsk, Ulaan Bataar, Kuala Lumpur, Singapore, Perth, Taipei, Osaka, Sapporo, Tokyo, Seoul, Yakutsk, Adelaide, Darwin, Brisbane, Canberra, Melbourne, Sydney, Guam, Port Moresby, Hobart, Vladivostok, Magadan, Solomon Is., New Caledonia, Auckland, Wellington, Fiji, Kamchatka, Marshall Island, Marshall Is, Nuku'alofa
Time Date	These are used to set the time and date on the IEM. Use 24-hr. format (HH:MM:SS) and date format (MM/DD/YYYY). **Notice:** Seconds must <u>not</u> be omitted from the time. **Notice:** When changing the configuration, the date and time parameters can be removed from the gateway.ini file so as not to reset the IEM Time and Date.

Communication Settings

S7ToAPACS	This parameter takes a non-negative integer value. A value of 0 disables the S7 to APACS communications; any other value enables it.
MaxR_ID	This parameter takes a non-negative integer and is used to specify the maximum value of the R_ID used in the GW_REC and GW_SEND blocks. The larger the number, the more system resources are used. It is recommended to set all IEMs to the same value. The parameter should be set as small as possible to a value that is the anticipated maximum R_ID assigned to any GW_REC/GW_SEND block. You can make the setting in advance of actually implementing the controller programs using the GW_REC/GW_SEND blocks. The default setting is sufficient unless the sytem will be exchanging large amounts of data requiring more GW_REC/GW_SEND pairs.

4.2 Default Configuration

The **gateway.ini** file with the default configuration (except for time/date) appears as shown below:

```
ComputerName = GATEWAY
Date = 3/29/2004
DiagnosticLevel = 0
IPAddress = 192.168.0.1
MaxR_ID = 8
Node = 62
Rack = 16
S7ToAPACS = 0
Slot = 16
SubnetMask = 255.255.255.0
TimeZone = Amsterdam
```

4.3 Configuring the IEM

The first step in configuring peer-to-peer communications is to set up the IEM to establish Ethernet and MBUS communications. Once this has been established, configuring the connection tables for peer-to-peer communications can proceed.

4.3.1 Setting Up the IEM for Ethernet and MBUS Communications

At no time does the IEM require a keyboard or mouse. All configuration operations may be performed by editing the configuration files residing on the USB flash drive, using a text file editor on any USB equipped workstation. After saving changes to files on the USB flash drive, remember to safely remove the USB flash drive by using the **Safely Remove Hardware** icon located on the Windows toolbar. The changes may then be put into effect by introducing the changed files to the IEM.

1. Remove the IEM from the box and connect it (via an RJ45 connector) from the **Ethernet** port (located on the side of the IEM) to an active Ethernet (switch). For further installation instructions, refer to the manual SIMATIC Box PC 620, Getting Started (A5E00131477-03).
2. Insert a USB flash drive (without the gateway.ini file on it) into the IEM, and power up the IEM
3. Upon initial use of the USB flash drive, the IEM will load the USB drivers and beep.
4. After the second set of beeps, it is safe to remove the USB flash drive, which will now have a file called oldgateway.ini in the IEM name folder.
5. Take the USB flash drive to any Windows 2000 or XP PC (Windows NT does not recognize USB) and open the oldgateway.ini file using Notepad or any other text editor. Edit this file using the Default Configuration (see Section *Default Configuration*) as a guide, rename it gateway.ini, and save it in the IEM name directory of the USB flash drive.
6. Insert the USB flash drive with modified gateway.ini file into either IEM USB port, and reboot the IEM.

Notice

Rather than removing power, you can reset the IEM by inserting a pointed object (e.g. an opened-up paper clip) into the **RESET** port on the side of the IEM.

7. Approximately one minute after the reboot, the IEM will beep indicating that the oldgateway.ini file has been written to the USB flash drive, and that the USB flash drive can be removed. The IEM will reboot a second time if parameters such as computer name, IP address, or MBUS address have been changed. The USB flash drive can be left in during this second reboot and removed after the next set of IEM beeps.
8. The IEM will write the following files to the IEM name folder of the USB flash drive,

    - oldgateway.ini (previous configuration)
    - Status.txt (status info)
    - Existing NIM32 configuration files are copied to the USB flash drive:

      The IEM file "c:\etc\remoteconfigurationsecuritytable.txt" is written onto the USB as the file "oldremoteconfigurationsecuritytable.txt"

      The IEM file "c:\etc\clientaccesstable.txt" is written onto the USB flash drive as the file "oldclientaccesstable.txt".
    - The current IEM connection table configuration is copied to the USB flash drive:

      A file named "oldiem2iem.txt" is copied to the USB flash drive. This file identifies the configuration of the IEM to IEM connections.

    New NIM32 configuration files are copied from the USB flash drive:

      If the USB holds a file named "remoteconfigurationsecuritytable.txt", it is copied to the IEM file "c:\etc\remoteconfigurationsecuritytable.txt".

      If the USB holds a file named "clientaccesstable.txt", it is copied to the IEM file "c:\etc\clientaccesstable.txt".

      The content and format of the files is assumed valid. A simple copy is made of the files.

    New IEM connection table configuration (iem2iem.txt) is copied from the USB flash drive:

      If the USB flash drive holds a file named "iem2iem.txt", it is checked for IEM to IEM connection configuration.

      This file holds a line that identifies each of the other IEMs formatted as "NAME,IP". The NAME is the resource name of the other IEM and IP is the IP address of the other IEM. The recommended naming convention is to have the resource name match the computer name of the IEM.

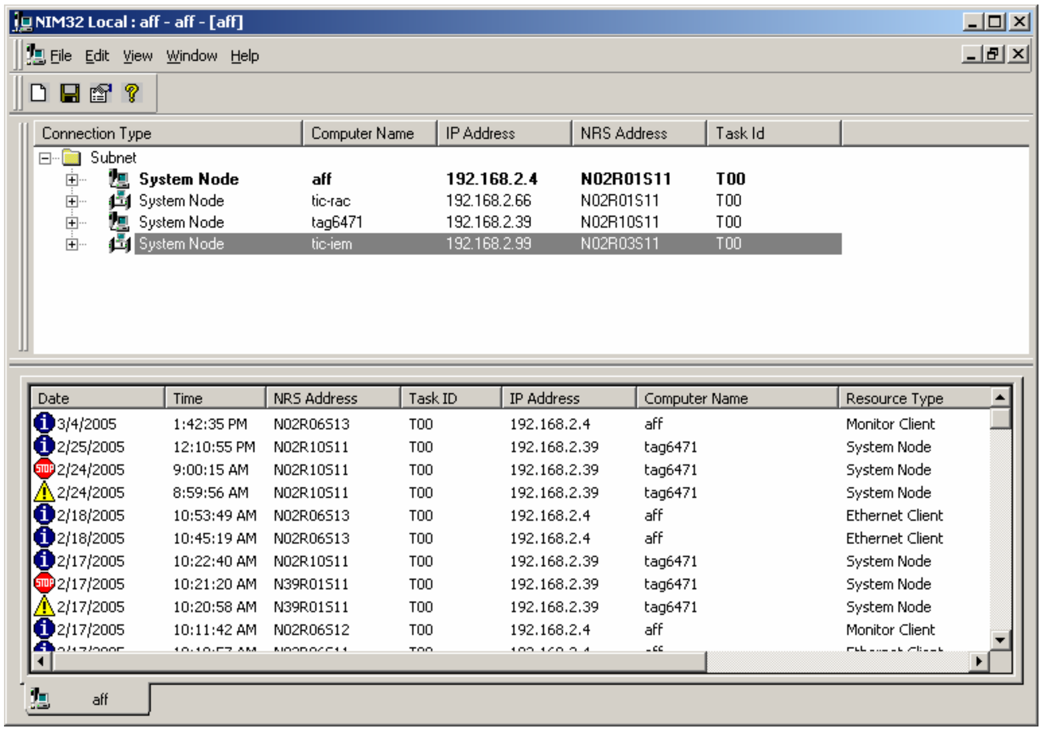
Notice

Immediately after a reboot, the "old" files written back to the USB flash drive relate to the configuration of the IEM before processing the files currently on the USB flash drive. If, after one minute, the USB flash drive is removed and reinserted, it will return "old" files that correspond to the latest USB file configuration.

Notice

The IEM will strip out the Time and Date information from the original Gateway.ini file. This prevents the accidental resetting of the old Time and Date on subsequent reboots using the Gateway.ini file.

9. Review the Status.txt for successful installation message or error message.
10. Go to another PC on the same network and ping the IEM by PC Name and IP address to verify the information. If the IEM does not respond to the ping, review oldgateway.ini to ensure that the parameters are what you expect. Repeat steps 4 through 8 if necessary.
11. Once the IEM IP address and PC name are verified, verify the NRS address using the second PC's *NIM32 User Interface* (Start>Program>ProcessSuite>APACS).
12. When *NIM32 User Interface* starts, select Local and review the top pane.

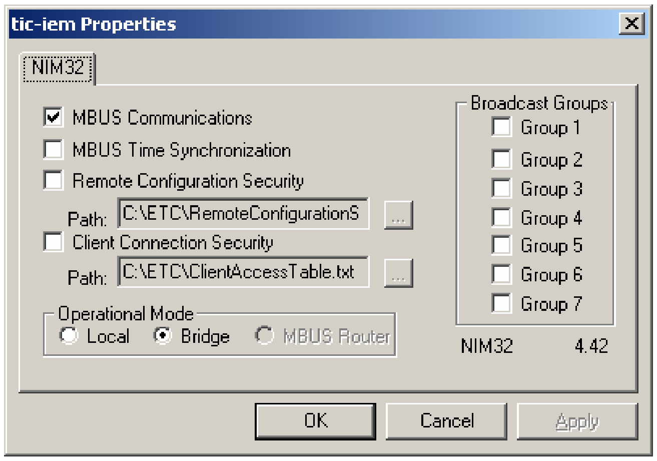


Connection Type	Computer Name	IP Address	NRS Address	Task Id
System Node	**aff**	**192.168.2.4**	**N02R01S11**	**T00**
System Node	tic-rac	192.168.2.66	N02R01S11	T00
System Node	tag6471	192.168.2.39	N02R10S11	T00
System Node	tic-iem	192.168.2.99	N02R03S11	T00

Date	Time	NRS Address	Task ID	IP Address	Computer Name	Resource Type
ℹ 3/4/2005	1:42:35 PM	N02R06S13	T00	192.168.2.4	aff	Monitor Client
ℹ 2/25/2005	12:10:55 PM	N02R10S11	T00	192.168.2.39	tag6471	System Node
🛑 2/24/2005	9:00:15 AM	N02R10S11	T00	192.168.2.39	tag6471	System Node
⚠ 2/24/2005	8:59:56 AM	N02R10S11	T00	192.168.2.39	tag6471	System Node
ℹ 2/18/2005	10:53:49 AM	N02R06S13	T00	192.168.2.4	aff	Ethernet Client
ℹ 2/18/2005	10:45:19 AM	N02R06S13	T00	192.168.2.4	aff	Ethernet Client
ℹ 2/17/2005	10:22:40 AM	N02R10S11	T00	192.168.2.39	tag6471	System Node
🛑 2/17/2005	10:21:20 AM	N39R01S11	T00	192.168.2.39	tag6471	System Node
⚠ 2/17/2005	10:20:58 AM	N39R01S11	T00	192.168.2.39	tag6471	System Node
ℹ 2/17/2005	10:11:42 AM	N02R06S12	T00	192.168.2.4	aff	Monitor Client

Figure 4-1 NIM32 User Interface

13. Select the IEM in the top pane, right-click and select **Properties**. This will open the NIM32 tab.



[✓] MBUS Communications
[ ] MBUS Time Synchronization
[ ] Remote Configuration Security
Path: C:\ETC\RemoteConfigurationS
[ ] Client Connection Security
Path: C:\ETC\ClientAccessTable.txt

Operational Mode: Local [✓ Bridge] MBUS Router

Broadcast Groups: [ ] Group 1, [ ] Group 2, [ ] Group 3, [ ] Group 4, [ ] Group 5, [ ] Group 6, [ ] Group 7

NIM32 4.42

Figure 4-2 NIM32 Configuration

14. Verify that the NIM32 component is in Bridge mode.

Notice

Make sure the security check boxes are not checked, or you will be prevented from using the NIM functionality until the security tables are loaded into the IEM.

5 Configuration - APACS+ Peer-to-Peer Communications

The IEM allows a resource (ACM or CCM) on one MBUS segment to send or receive data values to a resource on another, remote, MBUS segment. The IEM uses standard Ethernet hardware and cabling, and does not need Module Bus expanders (MBX) and Modulnet coaxial cable (MNET).

The IEM uses the GW_SEND and GW_REC function blocks (included with the IEM CD) to transfer arrays of data organized as 16 Word values, 32 Boolean values, and 32 Real values, between resources. Each MBUS segment permits an IEM to communicate with up to eight other MBUS segments. Redundancy is achieved by using a second set of IEMs between segments. The GW_SEND and GW_REC function blocks support the redundant transfer of data over two separate connections.

In the past, a typical multi-node system connected multiple MBUS segments together using MBX modules and MNET cabling. An example system is shown below. This system combines three local MBUS segments called Node2, Node4 and Node6 onto one communications highway called the MNET.

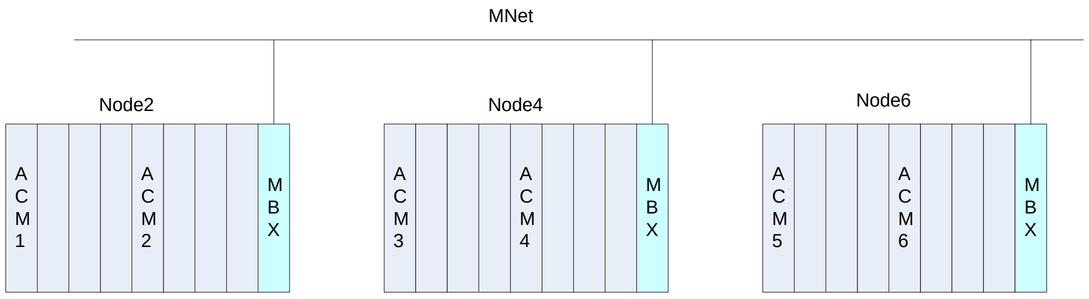


Figure 5-1 APACS+ Multi-node System using MNET with MBXs

The IEM functionality for the Ethernet link is used instead of the MBX/MNET data highway, and can be used to link a total of eight separate IEM devices/local MBUS nodes together. Each IEM has its own IP address and its own local MBUS segment of APACS+ controllers. The APACS+ controllers on each local MBUS segment can now use the IEM Ethernet link functionality to exchange data between remote APACS+ controllers connected to any of up to eight other IEM devices.

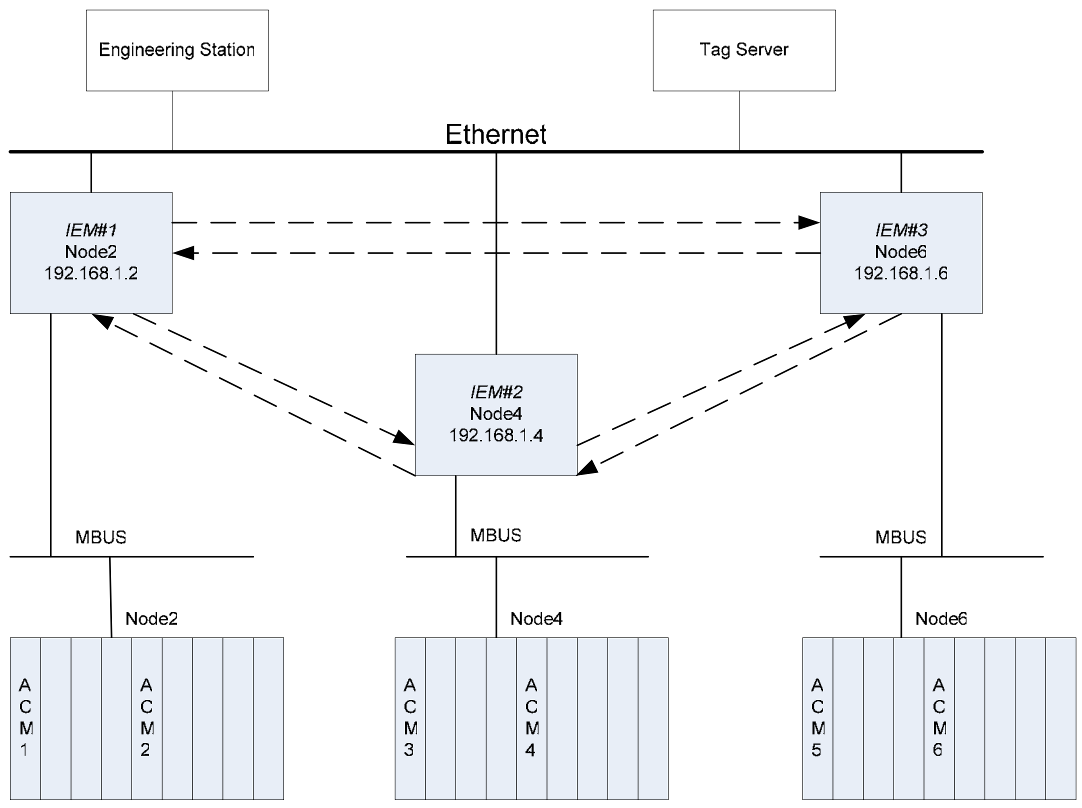


Figure 5-2 APACS+ Multi-node Using Ethernet with IEMs

Controller to controller data exchange across the IEM Ethernet link requires the ACM configuration to use paired GW_REC and GW_SEND function blocks (included on the IEM CD). For an APACS+ controller to send/receive data over the Ethernet link, its *4-mation* configuration requires a CONNECT function block and a GW_SEND paired to a corresponding GW-REC function block at the receiving ACM. The APACS+ controller receiving the data would also include a CONNECT function block for proper operation, (see Section *Configuring the ACM for Peer-to-Peer Communications Using the IEMs*).

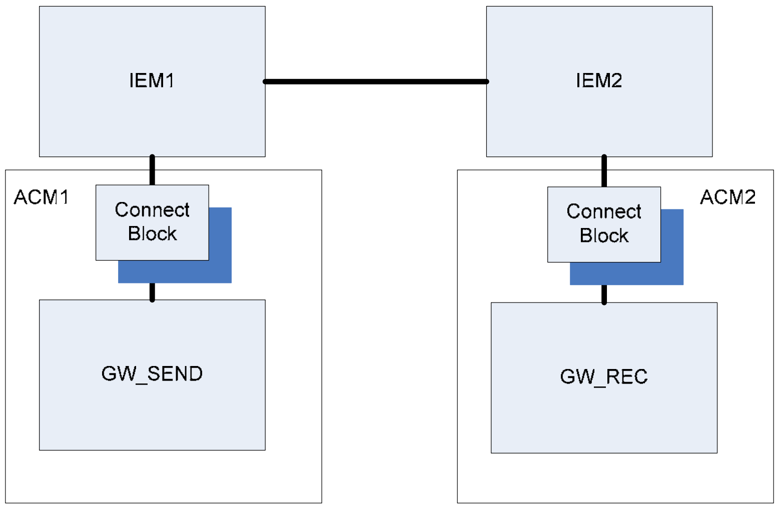


Figure 5-3 APACS+ Communications Blocks for IEM Based Peer-to-Peer Communications

Configuration of peer-to-peer communications is done on two (2) distinct levels.

1. IEM – Configuring the IEM computer name, the IP address, and the node, rack and slot using the gateway.ini file. Also configuring the IEM name and IP address of the IEM with which this IEM is to communicate with using the iem2iem.txt file.
2. APACS+/QUADLOG controller – Configuring the communications blocks (Connect, and the paired GW_SEND & GW_REC function blocks).

5.1 IEM Connection Tables Setup

After establishing a presence on the Ethernet and MBUS, the next step in configuring peer-to-peer communications is to set up the IEM connection tables to determine which IEMs will be used to communicate between controllers. This is accomplished by creating the file “iem2iem.txt” on the USB key.

Notice

The step numbers in the procedure below correspond to the numbers in the graphic.

1. Node 2 controllers (either ACM1 or ACM2 or both) are to communicate with Node 6 controllers (either ACM5 or ACM6 or both). This would require communications using IEM1 and IEM3
2. Node 2 controllers (either ACM1 or ACM2 or both) are also to communicate with Node 4 controllers (either ACM3 or ACM4 or both). This would require communications using IEM1 and IEM2
3. Node 6 controllers (either ACM5 or ACM6 or both) are also to communicate with Node 4 controllers (either ACM3 or ACM4 or both). This would require communications using IEM2 and IEM3

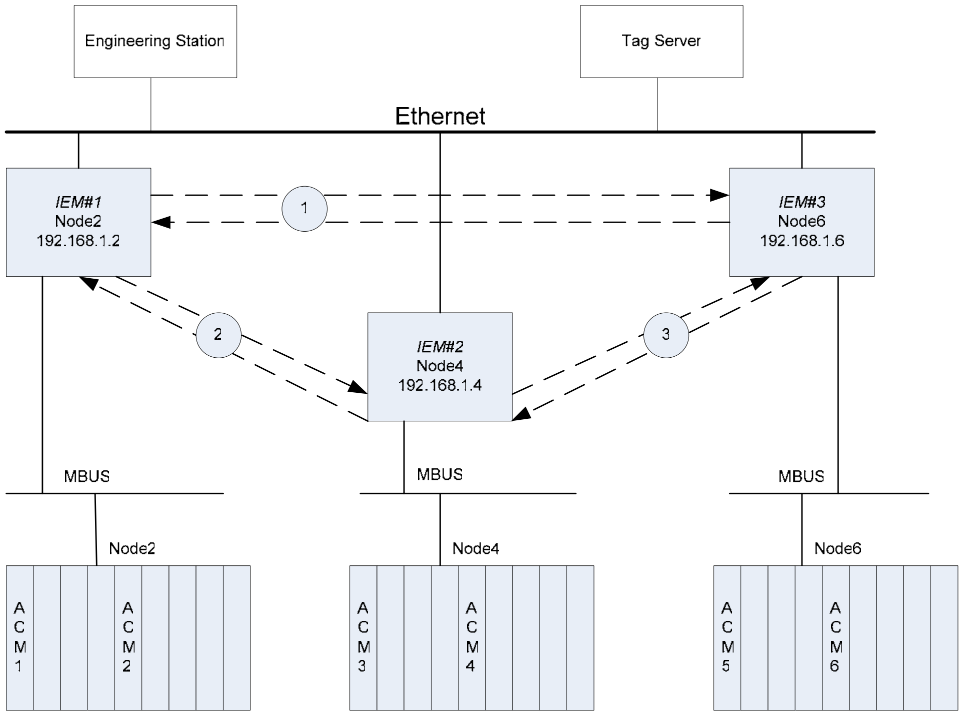


Figure 5-4 IEM Connections for APACS+ Peer-to-Peer Communications

The following connection tables, one for each IEM, are used to configure the intra-IEM communications.

iem2iem.txt for IEM#1	iem2iem.txt for IEM#2	iem2iem.txt for IEM#3
NODE4,192.168.1.4	NODE2, 192.168.1.2	NODE4,192.168.1.4
NODE6,192.168.1.6	NODE6, 192.168.1.6	NODE2,192.168.1.2

Adding IEM connections

1. Place an iem2iem.txt file in the IEM name folder on the USB flash drive. The iem2iem.txt file must contain one connection per line in the form of <remoteIEMname>,IPaddress (no spaces allowed).
   For example, in IEM1 the iem2iem.txt table might be:

   ```
   NODE2,192.168.0.2
   NODE4,192.168.0.4
   NODE6,192.168.0.6
   ```

2. Insert the USB flash drive into the IEM and reboot to update the table.

Notice

Existing IEM connections not listed in the iem2iem.txt file will be removed.

Sending data from ACM3 in Node4 to an ACM5 in Node6

Step	Procedure
1	Configure IEM#2 Node4 with its connection table (iem2iem.txt).
2	Configure IEM#3 Node6 with its connection table (iem2iem.txt).
3	Since the data is to be exchanged with Node6, a connect block is placed in the ACM3 configuration in Node4 and the resource name is set to Node6, since this is the name of the connection from Node4 to Node6.
4	A GW_SEND block is placed in the ACM3 configuration and wired to the connect block.
5	A connect block is placed in the ACM5 configuration in Node6, and the resource name is set to Node4, since it is expecting the data to come from Node4.
6	A GW_REC block is placed in the ACM5 configuration, and wired to the connect block.

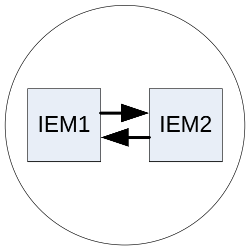
The data flow in the above configured system follows:

Step	Procedure
1	The GW_SEND in ACM3 sends the data to the IEM “Node4”..
2	Node4 sends the data to Node6.
3	The IEM, “Node6”, receives the message. It checks the source of the message, and recognizes that the source is on the other side of its Node4 connection.
4	The GW_REC in ACM5 reads the data from the IEM, NODE6.

Details for configuring the GW_SEND and GW_REC blocks are in the following sections.

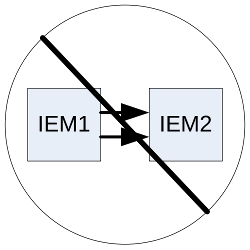
Notice

You must configure each IEM participating in the connection separately. For example, if a connection is made from Node4 to Node2, a corresponding connection must also be made from Node2 to Node4.



The following connection table is invalid because the IEM has two connections to the other IEM. All errors are reported to the status.txt file which is downloaded to the USB flash drive.

Node3	
N02, 192.168.1.102 Node2, 192.168.1.102	⇦ Error



5.2 Using the CONNECT Block with GW_SEND and GW_REC Blocks

- Establishes the communication channel ID between two ACMs.
- Each ACM must contain one CONNECT block for each remote segment with which it is to communicate.

Notice

The APACS+ controller program uses a CONNECT block to make a connection for exchanging data using the GW_REC and GW_SEND blocks. The NAME parameter is used to identify the name of the remote IEM connected to the remote APACS+ controller that will be exchanging data.

The recommended convention is to always use uppercase text for the NAME parameter. The same exact text is used for configuring the IEM using the USB file iem2iem.txt. This file holds the IEM names and their corresponding network IP addresses.

If the APACS+ program uses uppercase text for the CONNECT NAME, then the iem2iem.txt file must also have the same name text in uppercase.

See section 7 for further information on these configuration components.

  - CONNECT blocks are required for all IEM communications using GW_SEND and GW_REC.

- Although multiple GW_SEND / GW_REC blocks can be configured, only one CONNECT block is required for transferring data between a resource and a remote segment. The channel ID displayed at the ID output of the CONNECT block provides the connection. When additional communication is necessary with a different remote segment, an additional CONNECT block is required.

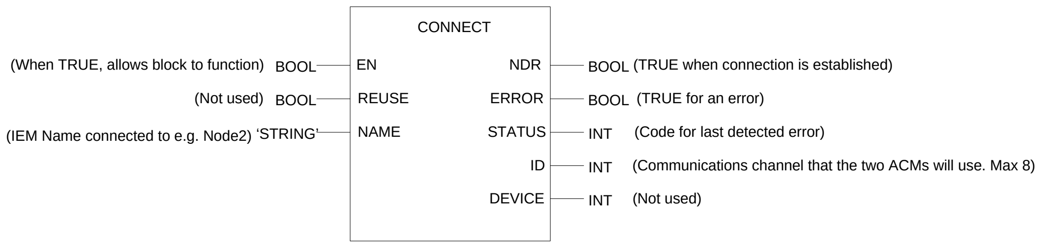

CONNECT
(When TRUE, allows block to function) BOOL EN
(Not used) BOOL REUSE
(IEM Name connected to e.g. Node2) 'STRING' NAME
NDR BOOL (TRUE when connection is established)
ERROR BOOL (TRUE for an error)
STATUS INT (Code for last detected error)
ID INT (Communications channel that the two ACMs will use. Max 8)
DEVICE INT (Not used)

Figure 5-5 APACS+ Connect Block

5.3 GW_SEND and GW_RCV Blocks

- These blocks work as a pair and provide one-way communications between an APACS+/QUADLOG controller and another APACS+/QUADLOG controller or an S7 controller.
- The Remote Identification (R_ID) input to each GW_SEND/GW_RCV pair must be identical. This common linking ID is how the two blocks are linked together
  - The GW_SEND block <u>must</u> exist in one resource, while a paired (common R_ID) GW_RCV block is in the other resource.

See section 7 for further information on these configuration components.

5.3.1 GW_SEND

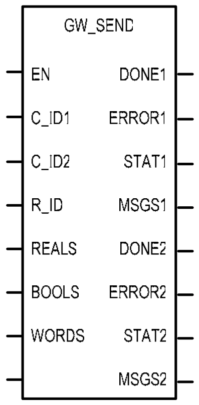


Figure 5-6 GW_SEND Block in APACS+

- Each APACS+/QUADLOG GW_SEND function block is used to send data to one paired GW_REC block in another ACM/CCM or an S7 controller.
- The GW_SEND may be configured to send 32 REAL, 32 BOOLS, and 16 WORD values to its paired GW_REC block in another APACS+/QUADLOG or S7 controller resource. The send values are stored in arrays (REALS, BOOLS, and WORDS) which appear as inputs to the GW_SEND block.
- In the APACS+/QUADLOG or S7 resource, a linked (R_ID) GW_REC block must be configured to receive the data. The identical R_ID input parameter links the two blocks together.

5.3.2 GW_REC

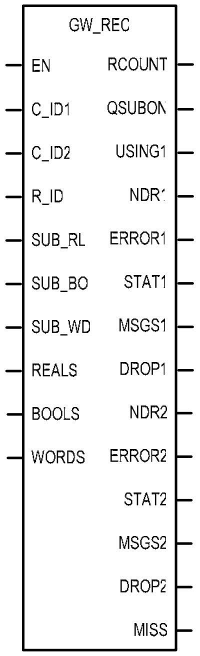


Figure 5-7 GW_REC Block in APACS+

- The APACS+/QUADLOG GW_REC function block is used to receive data from a paired GW_SEND block in another APACS+/QUADLOG or an S7 controller.

  The GW_REC may be configured to receive 32 REALS, 32 BOOLS, and 16 WORD values from another APACS+/QUADLOG or S7 controller resource. The received values are stored in arrays (WORDS, BOOLS, and REALS) which appear as inputs to the GW_REC block.

- In the APACS+/QUADLOG or S7 controller, a corresponding GW_SEND block must be configured to send the data. The identical R_ID input parameter links the two blocks together.

5.4 Example: M-Net to Ethernet Upgrade Procedure

This is a partial implementation example of replacing existing ACM communications using the READ block over MNET with GW_SEND and GW_REC blocks.

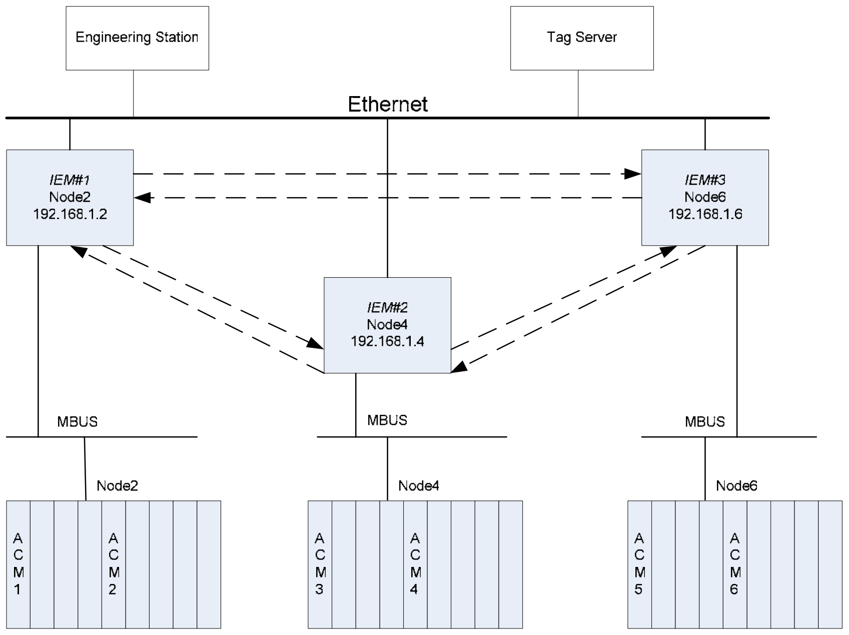


Figure 5-8 Multi-node APACS System Using Ethernet with IEMs

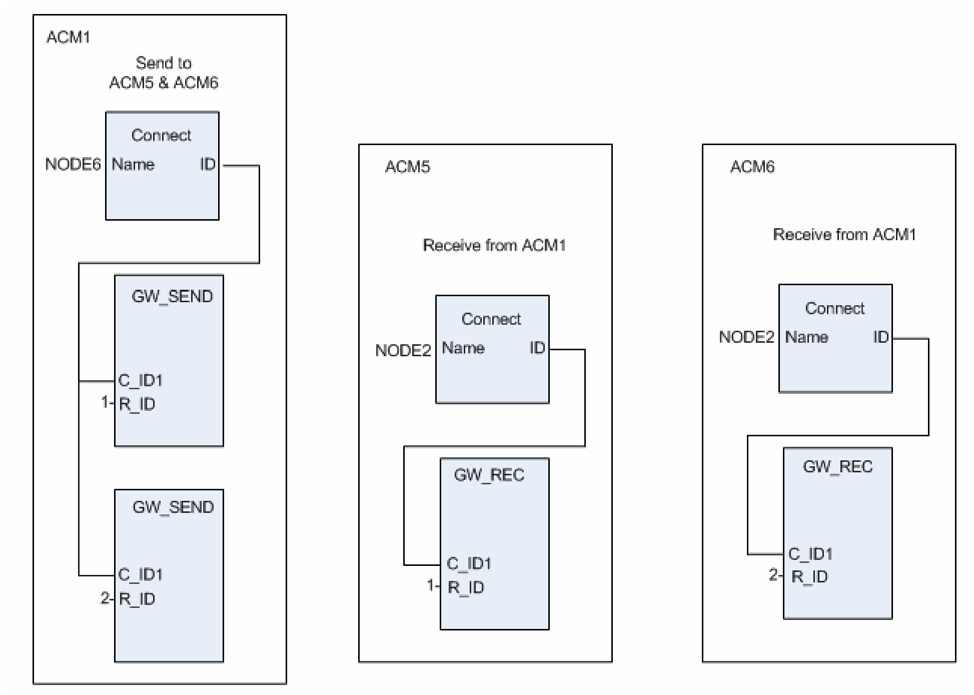


Figure 5-9 Interlocking of Communications Blocks in APACS+

The following procedure corresponds with the two graphics above.

1. Install the IEM.
2. Configure the IEM.
   - 2.1 Configure the gateway.ini file.
   - 2.2 Configure the connection list.
3. Modify the *4-mation* configuration.
   - 3.1 Establish the IEM connection.
   - 3.2 Configure the GW_REC block.
   - 3.3 Create arrays on the GW_REC block.
   - 3.4 Configure the corresponding GW_SEND block.
   - 3.5 Create arrays on the GW_SEND block.

1. Install the IEM

1	Mount the IEM to wall or rack using included flanges. For further installation information, please refer to the manual SIMATIC Box PC 620, Getting Started (A5E00131477-03).
2	Connect AC power.
3	Connect MBUS.
4	Connect Ethernet cable.

2. Configure the IEM

Notice: In the following example, the gateway.ini file is going to be edited and then the iem2iem.txt file will be edited. It is possible to modify both files and load them onto the IEM in one step.

2.1 Configure the IEM name and network parameters using the gateway.ini file.	
1	Insert the USB flash drive into either USB port on the rear of the IEM.
2	After you hear three beeps, remove the USB flash drive.
3	Insert the USB flash drive into the Engineering Station or other available PC computer.
4	From Windows Explorer, click on the **Removable Disk** drive in the directory with the same name as the IEM computer name. **Notice:** If you change the computer name, the configuration will be placed in the new directory on the USB key in the future. The following files will be displayed: • oldclientaccesstable.txt • oldgateway.txt • oldiem2iem.txt • oldremoteconfigurationsecuritytable.txt • status.txt
5	Open the **oldgateway.ini** file in any text editor, for example Notepad.exe.
6	Modify the following items with the current information: COMPUTERNAME = DATE = DIAGLEVEL = IPADDRESS = MAXR_ID = NODE = RACK = S7TOAPACS = 0 (Note: 0 is the default) SLOT = SUBNETMASK = TIME = (Notice: set the time ahead a minute or two for the delay until the file is installed) TIMEZONE =
7	Save the file, **File>Save As> gateway.ini**
8	Remove the USB flash drive from the computer by selecting the **Safely Remove Hardware** icon on the Windows toolbar.
9	Insert USB flash drive into either IEM USB port.
10	Reset the IEM by inserting the end of a paper clip into the **Reset** port on the left side (facing the IEM) of the IEM. The reset is complete after the three beeps (approximately one minute).

Notice If you changed the Computer Name, IPAddress, Node, Rack or Slot, a second reset will automatically occur in approximately 30 seconds.	
11	The USB flash drive now has the file **oldgateway.ini**, which contains the configuration prior to the recent change. Remove the USB flash drive from the IEM for 10 seconds
12	Re-insert the USB flash drive into the IEM. In a few seconds, the IEM will beep, and the USB flash drive will contain the current information.
13	Remove the USB flash drive from the IEM, and insert it into the work computer.
14	On the USB flash drive; there will be a folder with the same name as the IEM containing the current configuration settings. The **oldgateway.ini** file should contain the new information. The **status.txt** file logs the running list of changes.
15	Save the file **oldgateway.ini** in a safe place for future reference. The file can be saved on the USB flash drive by changing its name (suggest appending a date and time to the file name).
2.2 Configure the iem2iem.txt file holding the connection list.	
1	From the USB flash drive on the Engineering Station or other available PC, open the **oldiem2iem.txt** file. Notice: This is in a directory with the same name as the IEM computer name.
2	Add the IEM <NAME>, <IPADDRESS> for each IEM you are connecting to. The IEM name will be used in the CONNECT block in later steps.
3	Save the file, **File>Save As>iem2iem.txt**
4	Remove the USB flash drive from the computer.
5	Insert the USB flash drive into the IEM.
6	Reset the IEM using the paper clip.
7	When the IEM beeps three times, remove the USB flash drive from the IEM.
8	The USB flash drive will now contain a copy of the **oldiem2iem.txt** file that can be verified.
9	Re-insert the USB flash drive into the IEM to retrieve the current **oldiem2iem.txt** file.
10	Verify the information: • Insert the USB flash drive into the Engineering Station or other available PC computer. From the Removable Disk drive, open the **oldiem2iem.txt** file. Verify the connections. Remove the USB flash drive.

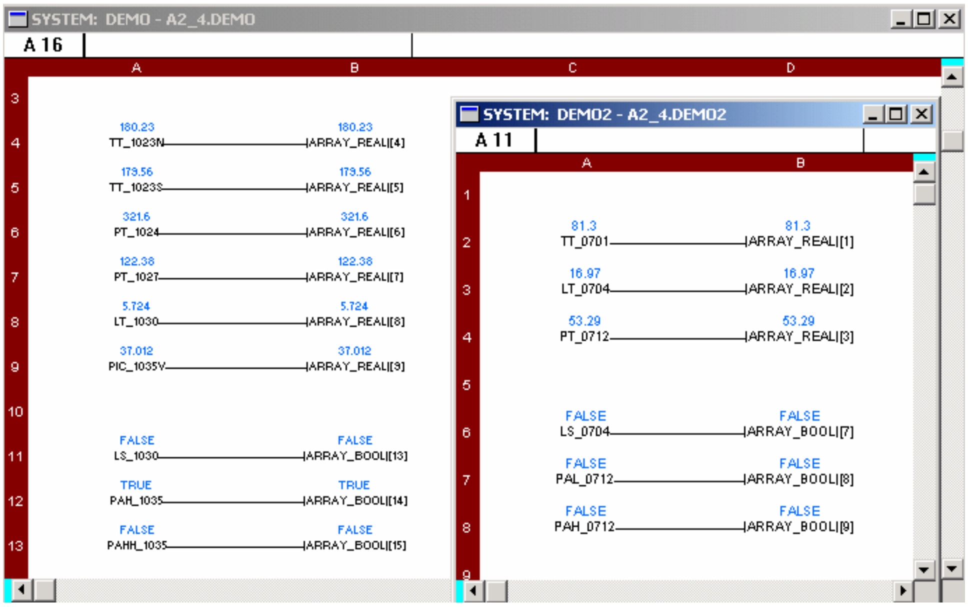
3. Modify the *4-mation* configuration

3.1 Establish the IEM connection by configuring the CONNECT blocks in each resource.	
1	From the controller that contains the current READ block, add a CONNECT block. The NAME input of the CONNECT block will be a string the same name as the IEM connection. When the CONNECT block is enabled with the name of the connection, there will be a non-zero value on the CONNECT block's ID output. This indicates a successful connection.
3.2 Configure the GW_REC block.	
1	Place a GW_REC block on the configuration sheet with the new CONNECT block.
2	Click the **F7 Derived Function** key, and place the cursor on the configuration sheet where you want to place the block. The **Function Block Types** dialog box will open.
3	From the **Categories** area, select the **Standard** radio button.
4	From the **Name/Category** list, select GW_REC. Click the **Select** button. The block should appear on the configuration sheet.
5	Connect the ID output from the CONNECT block to the C_ID1 input of the GW_REC block. You can use a variable, a wire, or a stub. If you have more than one GW_REC connected to the same node, it may be useful to use a Global variable for the ID.
3.3 Create arrays on the GW_REC block.	
1	Create arrays on GW_REC for the REALS (max 32 elements), BOOLS (max 32 elements), and WORDS (max 16 elements) to the input side of the GW_REC block.
2	Fan-out the values of the array using standard left to right assignments in an open area of the configuration sheet.
3.4 Configure the corresponding GW_SEND block	
1	Configure the SEND IEM the same way you configured the REC IEM.
2	Assuming the values from the source controller are on several configuration sheets, create a PROGRAM block to consolidate these values.
3	In the new consolidation PROGRAM block, place a CONNECT block. The NAME input of the CONNECT block will be a string the same name as the destination IEM.
4	Click the **F7 Derived Function** key, and select the configuration sheet where you want to place the block. The **Function Block Types** dialog box will open.
5	From the **Categories** area, select the **Standard** radio button.

6	From the **Name/Category** list, select GW_SEND. Click the **Select** button. The block should appear on the configuration sheet.
7	Connect the CONNECT block ID output to C_ID1 Input. Use the same R_ID as the GW_REC block and enable the SEND block.
3.5 Create arrays on the GW_SEND block.	
1	Create an array for REALS, BOOLS and WORDS. If the variables to send are on more than one configuration sheet, it is advisable to use a global array.
2	Assign the variables to the arrays using standard left to right assignments such as (assuming a global array): <var_name> - |<array_name>|[element] or TANK1_LVL - |SND_REAL|[13]

Notice

Any Engineering Stations or Tag Servers that will be interfacing with these IEMs must have the IEMs added to their APACS.ini current NIM list from the APACS+ Control Panel Applet.



```
180.23                     180.23
TT_1023N ----------------- |ARRAY_REAL|[4]
179.56                     179.56
TT_1023S ----------------- |ARRAY_REAL|[5]
321.6                      321.6
PT_1024 ------------------ |ARRAY_REAL|[6]
122.38                     122.38
PT_1027 ------------------ |ARRAY_REAL|[7]
5.724                      5.724
LT_1030 ------------------ |ARRAY_REAL|[8]
37.012                     37.012
PIC_1035V ---------------- |ARRAY_REAL|[9]

FALSE                      FALSE
LS_1030 ------------------ |ARRAY_BOOL|[13]
TRUE                       TRUE
PAH_1035 ----------------- |ARRAY_BOOL|[14]
FALSE                      FALSE
PAHH_1035 ---------------- |ARRAY_BOOL|[15]
```

```
81.3                       81.3
TT_0701 ------------------ |ARRAY_REAL|[1]
16.97                      16.97
LT_0704 ------------------ |ARRAY_REAL|[2]
53.29                      53.29
PT_0712 ------------------ |ARRAY_REAL|[3]

FALSE                      FALSE
LS_0704 ------------------ |ARRAY_BOOL|[7]
FALSE                      FALSE
PAL_0712 ----------------- |ARRAY_BOOL|[8]
FALSE                      FALSE
PAH_0712 ----------------- |ARRAY_BOOL|[9]
```

Figure 5-10 Writing to an Array Element in APACS

6 Configuration - S7 to APACS+ Communications

After the hardware is installed and the power supply is switched on, three steps are required to put the IEM into operation:

1. In a STEP 7 project using the HW-Config program within STEP 7, configure the IEM as a SIMATIC PC Station and as a participant to the plant bus.
2. Using the NetPro program under STEP 7, configure S7 connections between the IEM and the participating S7 stations.
3. Configure the IEM settings by entering the IP addresses from NetPro.

These tasks are described in the following sections. Once the IEM is in operation, configure the specific function blocks in the APACS+/QUADLOG and SIMATIC S7/PCS7 systems. The function blocks and their configurations are described in Function Blocks section.

6.1 Hardware Component Configuration in a STEP 7 Project

In STEP 7, the S7 stations and the IEMs are configured to be connected to a common Industrial Ethernet plant bus.

6.2 Setting up S7 Stations

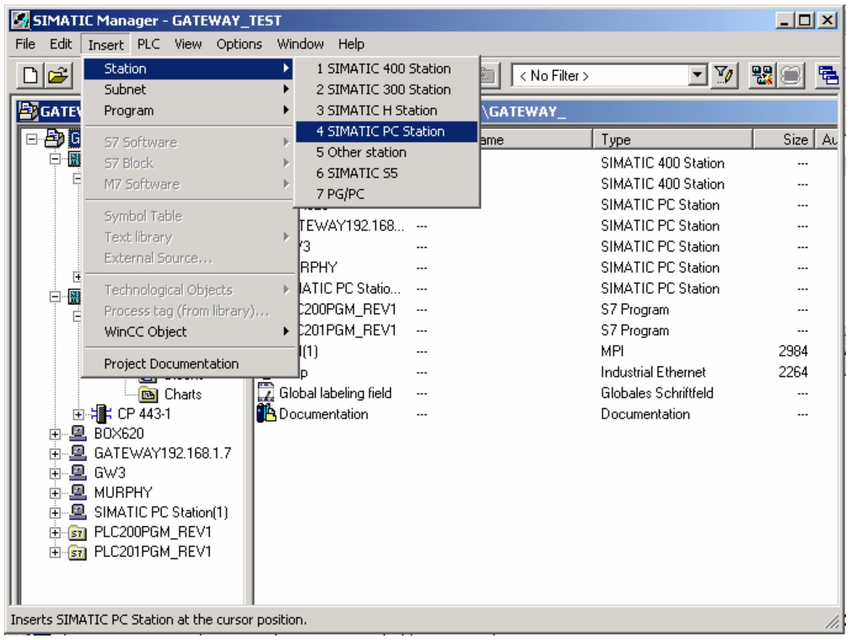
For the IEM, select the station type **SIMATIC PC Station** in STEP 7. Use the **Insert** function to integrate the station into a STEP 7 project.

Using the HW-Config utillity, assign the first two plug-in slots in the SIMATIC PC station (the IEM station) as follows:

- CP/IE General must be in slot one
- User application must be in slot two

6.3 Adding the IEM to a PCS 7 project

1. Open the PCS 7 project in SIMATIC Manager.
2. In the component view, add a **SIMATIC PC Station** at the root of the hierarchy. Give it a meaningful name, then click on the name to open the next level. A **Configuration** object should appear in the right pane.



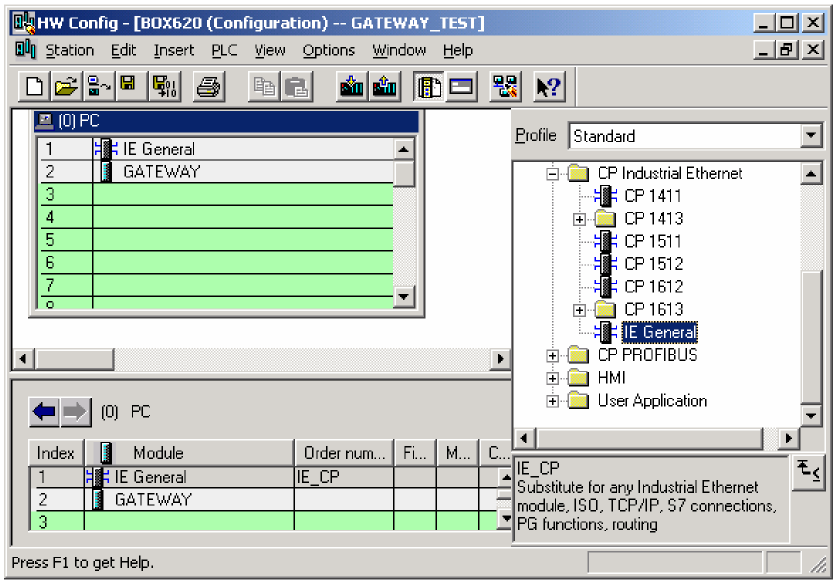
ame		Type	Size
		SIMATIC 400 Station	---
		SIMATIC 400 Station	---
		SIMATIC PC Station	---
TEWAY192.168...	---	SIMATIC PC Station	---
/3	---	SIMATIC PC Station	---
RPHY	---	SIMATIC PC Station	---
ATIC PC Statio...	---	SIMATIC PC Station	---
C200PGM_REV1	---	S7 Program	---
C201PGM_REV1	---	S7 Program	---
I(1)	---	MPI	2984
p	---	Industrial Ethernet	2264
Global labeling field	---	Globales Schriftfeld	---
Documentation	---	Documentation	---

Inserts SIMATIC PC Station at the cursor position.

Figure 6-1 Setting Up in SIMATIC Manager

3. Double-click **Configuration** in the right pane to open the HW Config utility.

4. From the Hardware Catalog, select the generic card network, which appears as either **IE Allgemein** or **IE General**, from the **SIMATIC PC Station/CP-Industrial Ethernet** category. Add the network card to the first slot.



1	IE General
2	GATEWAY
3	
4	
5	
6	
7	

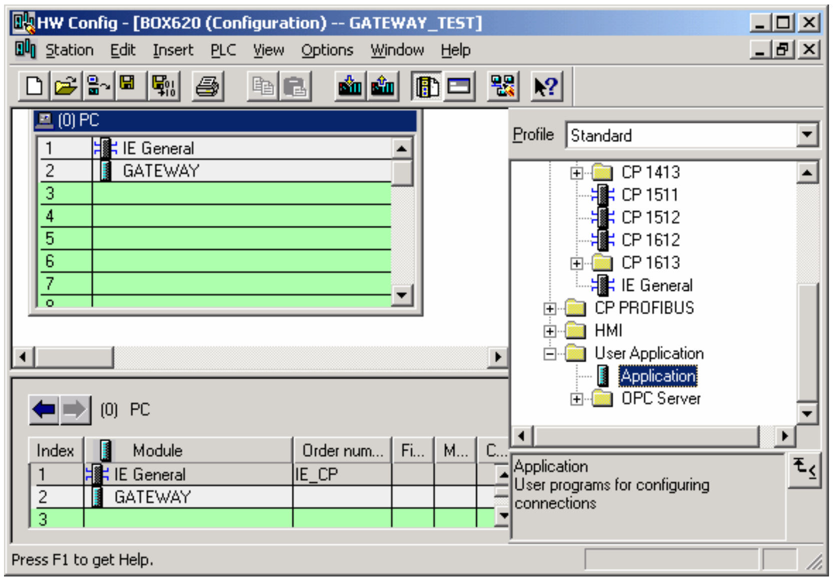
(0) PC

Index	Module	Order num...	Fi...	M...	C...
1	IE General	IE_CP			
2	GATEWAY				
3					

IE_CP
Substitute for any Industrial Ethernet module, ISO, TCP/IP, S7 connections, PG functions, routing

Figure 6-2 HW Configuration in SIMATIC Manager -1

5. Add an application object to the second slot from the **SIMATIC PC Station/User Application** category. The application object instance in the illustration has also been renamed Gateway.



1	IE General
2	GATEWAY
3	
4	
5	
6	
7	

(0) PC

Index	Module	Order num...	Fi...	M...	C...
1	IE General	IE_CP			
2	GATEWAY				
3					

Application
User programs for configuring connections

Figure 6-3 HW Configuration in SIMATIC Manager -2

6. Save the hardware configuration changes. The IEM itself is preconfigured to contain a generic network card and a single application.

Notice

The configuration must match only IE General in slot 1 and the application in slot 2, or NetPro will not be able to download the configuration to the IEM.

6.4 Ethernet Configuration

To set up the bus system to which the IEM station must be connected, follow these steps:

1. Open the PCS 7 project in SIMATIC Manager.
2. In the Component view, open the hardware configuration for the IEM.
3. In the HW-Config utility, right-click **IE General**, and select **Object Properties** to open a window with the bus connection parameters.

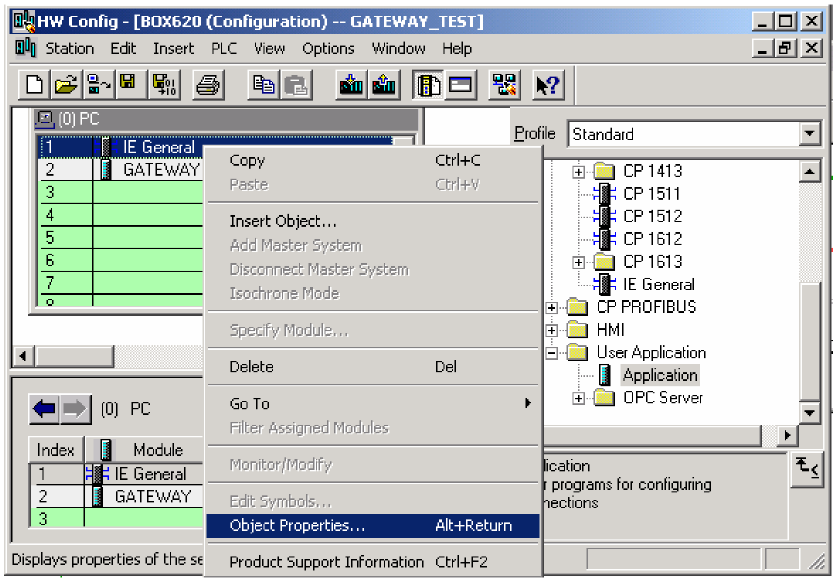
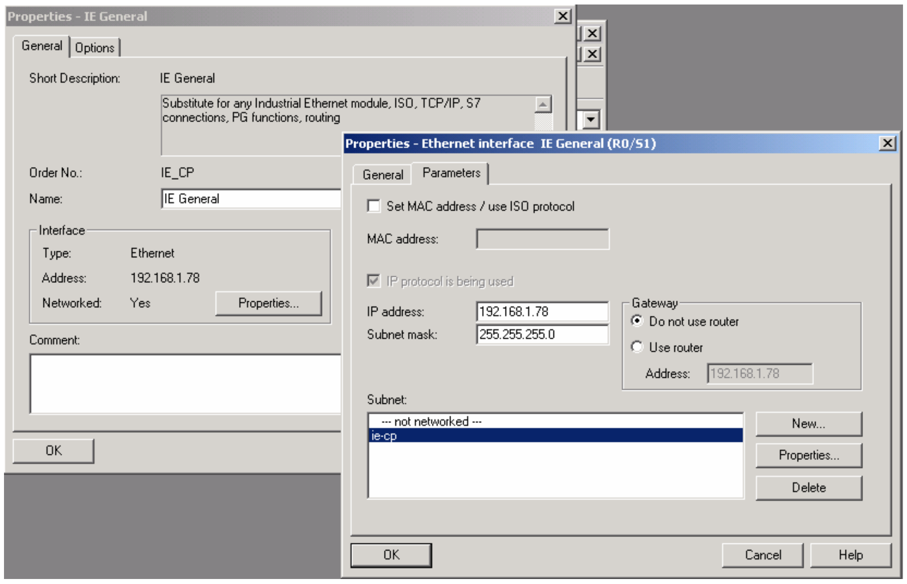


Figure 6-4 HW Configuration in SIMATIC Manager -3

4. The IE General Properties show the IP (Ethernet) address of the IEM station that is automatically assigned by HW-Config. Click **Properties** to display the **General Properties** dialog. Change the IP address and the subnet mask, if necessary.



Short Description: IE General

Substitute for any Industrial Ethernet module, ISO, TCP/IP, S7 connections, PG functions, routing

Order No.: IE_CP

Name: IE General

Interface

Type: Ethernet

Address: 192.168.1.78

Networked: Yes

Comment:

Set MAC address / use ISO protocol

MAC address:

IP protocol is being used

IP address: 192.168.1.78

Subnet mask: 255.255.255.0

Gateway

Do not use router

Use router

Address: 192.168.1.78

Subnet:

--- not networked ---

ie-cp

Figure 6-5 Setting General Properties in SIMATIC Manager

5. Make note of the IP address for the IEM station and the subnet mask specified in HW-Config. These are required for IEM configuration later. **Set MAC address / use ISO protocol** must be unchecked.
6. In the lower part of the dialog, under Subnet, enter the bus system parameters. Make sure that the IEM station is connected to the same bus as the S7 CPUs that are to communicate with the IEM.

6.5 Configuration of Connections in STEP 7

The NetPro utility is used to configure S7 connections between the IEM and an S7 CPU. The NetPro utility is a STEP 7 component.

NetPro defines and centrally administers all connections. The parameters of the S7 connections are imported into the S7 CPUs and the IEM (it is necessary to download connections to the IEM) using standard procedures of the S7 configuration.

NetPro shows all the stations of an S7 project. Use NetPro to select a participant to the Ethernet and insert a connection to another participant to the Ethernet. The IEM station must be selected to establish an active connection. It is important, however, that the connection between the S7 CPU and IEM is of the S7 connection type.

NetPro assigns the active role to the IEM when establishing an online connection. When the IEM station is selected, the following value appears in the connection parameters: **Actively establish connection = yes**. The value is **no** when the CPU is selected.

It is possible to configure a non-redundant connection to the IEM from any CPU of a redundant S7 station. Each CPU has its own communication module, which provides access to the Ethernet. S7 and APACS+ function blocks with redundant connections use all available configured connections to the IEM and automatically coordinate the data traffic using these connections. Each of the two connections can handle the entire communication between the IEM and the CPU. Communication is not affected if one of the two connections (one of the two redundant S7 station subsystems) fails.

Notice

The IEM does not support fault tolerant connections with H systems. Use regular connections, and select unique names.

Notice

When the connection is made in NetPro, you will see it appear as a database style record.

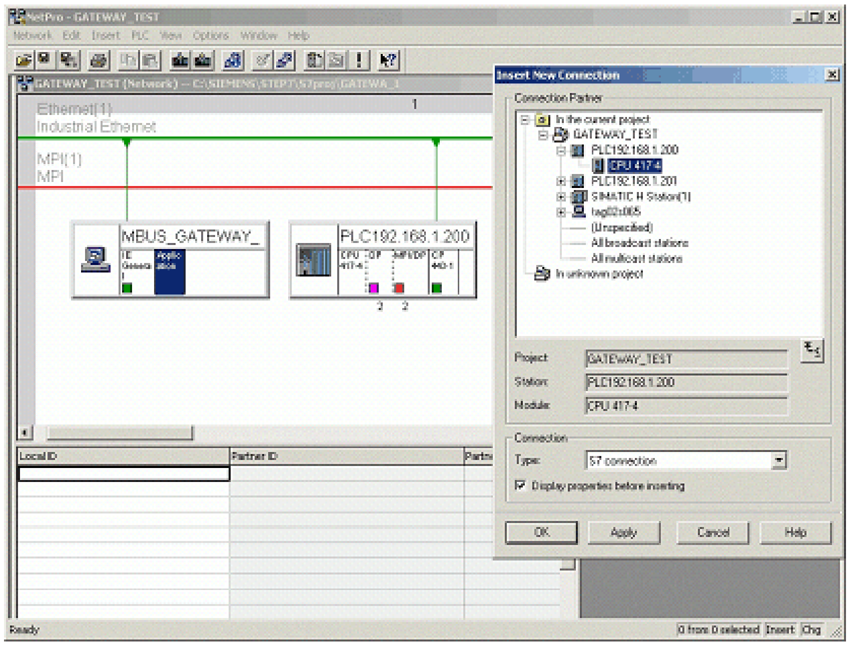
When viewed in NetPro with the focus placed on the IEM application, the first two fields from the left Local_ID and Partner_ID are of importance:

Local_ID holds the connection name string value the APACS+ communication block will use.

Partner_ID holds the parameter value to be used in the associated S7 Gateway CFC block’s C_ID nub.

6.6 Configuring the system to exchange data between APACS+/ QUADLOG and PCS 7

1. Open the PCS 7 project in SIMATIC Manager.
2. Open **Network Configuration**. NetPro displays all of the controllers in a network view.
3. Select the application in the IEM that will send the data to the S7-400 controller.
4. Select **Insert > New Connection** from the NetPro menu bar.
5. In the Insert New Connection dialog, select the S7 controller and the CPU. The Connection Type is **S7 connection**.



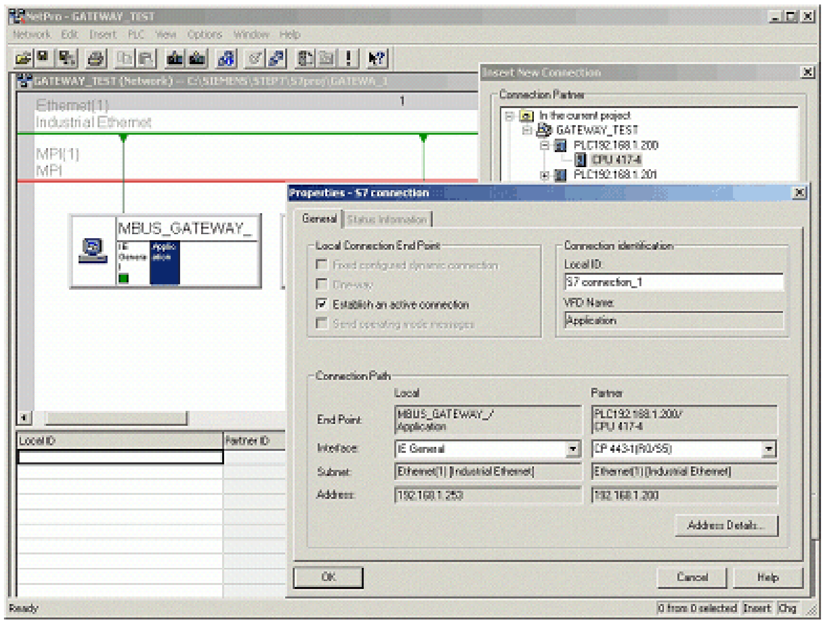
Project: GATEWAY_TEST
Station: PLC192.168.1.200
Module: CPU 417-4
Connection
Type: S7 connection
[✓] Display properties before inserting

Figure 6-6 NetPro Setup -1

6. Click **Apply**, and then edit the text in the Local ID field to give it a proper connection name. Give the connection the same (or similar) name as the controller. (For a redundant system, it is recommended to append an “A” to the name on the primary path, and a “B” for the backup path.)

Notice

Since the connection name will be used as the virtual APACS+ resource name, the connection name must be a valid and unique APACS+ resource name. Therefore, the resource name must be limited to 16 alphanumeric characters and the user must ensure it is unique name on the MBUS.



Local Connection End Point
- [ ] Fixed configured dynamic connection
- [ ] One-way
- [x] Establish an active connection
- [ ] Send operating mode messages

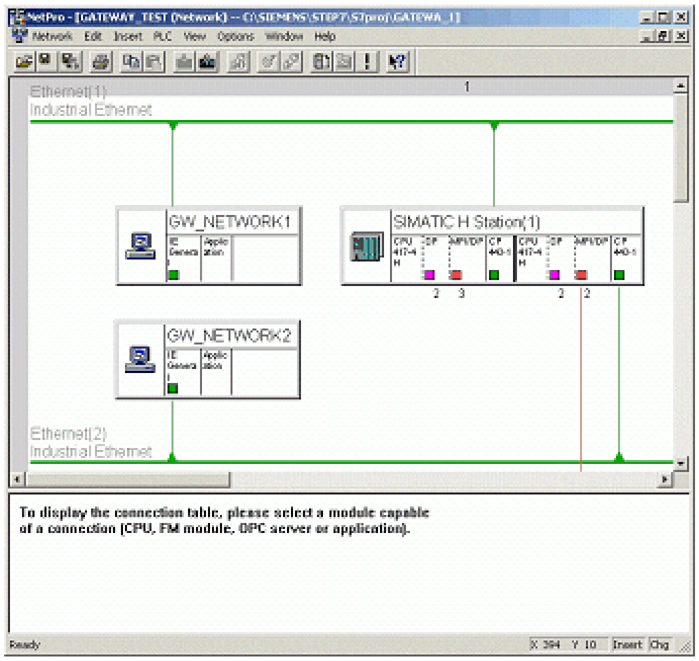
Connection identification
Local ID: S7 connection_1
VFD Name: Application

Connection Path

	Local	Partner
End Point:	MBUS_GATEWAY_/ Application	PLC192.168.1.200/ CPU 417-4
Interface:	IE General	CP 443-1(R0/S5)
Subnet:	Ethernet(1) [Industrial Ethernet]	Ethernet(1) [Industrial Ethernet]
Address:	192.168.1.253	192.168.1.200

Figure 6-7 NetPro Setup - 2

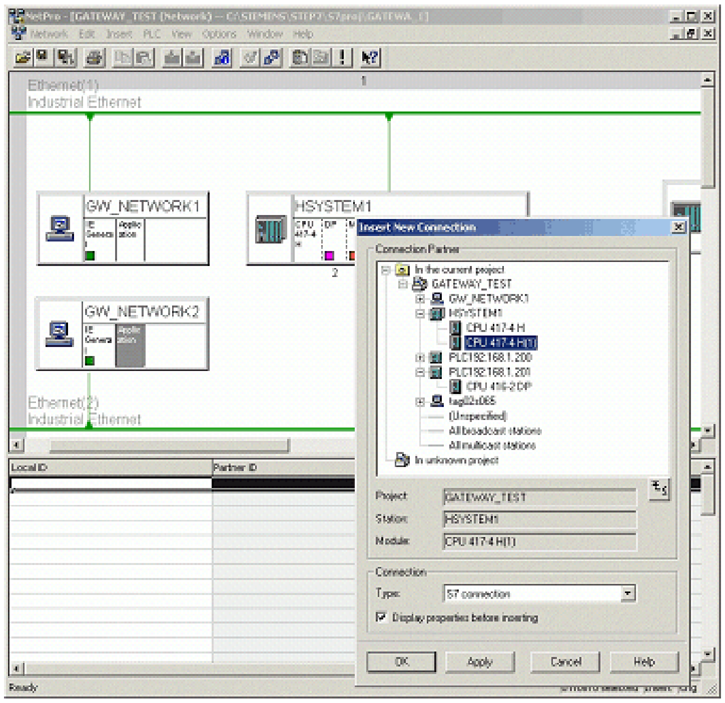
A redundant configuration is shown below. Here there are two separate Ethernet IEMs, each with a separate Ethernet network. Likewise, the S7 H-system has each of its two CP443 modules on the separate networks.


NetPro - [GATEWAY_TEST (Network) -- C:\SIEMENS\STEP7\S7proj\GATEWA_1]
Network Edit Insert PLC View Options Window Help
Ethernet(1)
Industrial Ethernet
GW_NETWORK1
SIMATIC H Station(1)
GW_NETWORK2
Ethernet(2)
Industrial Ethernet
Ready
Insert Chg

To display the connection table, please select a module capable of a connection (CPU, FM module, OPC server or application).

Figure 6-8 NetPro Setup -2

7. Select the CPU associated with the CP443 module on the same network.



Project: GATEWAY_TEST
Station: HSYSTEM1
Module: CPU 417-4 H(1)

Connection
Type: S7 connection
☑ Display properties before inserting

Figure 6-9 NetPro Setup -3

8. Compile and download the connection data to the IEM. Be sure to choose the **Download Selected and Partner Stations** Option. The IEM will create virtual APACS+ controllers using the names given to the connections in the previous step.

Notice

When connection information is downloaded to the IEM, all communication connections between S7 and APACS+ will be stopped.

9. Open the CFC configuration for the controller.
10. Add GW_REC and GW_SEND blocks to the S7-400 controllers as needed. For the ID1 and ID2 (redundant) connections, use the values assigned to the connections by NetPro.
11. Assign R_ID values to the blocks that are unique for the connection. Start with "1" for the first value, and assign additional values consecutively.

Notice

Failure to assign unique R_IDs may result in indeterminate behavior.

12. Configure the GW_SEND blocks with the desired Real, Boolean, and Word data to send to APACS+/QUADLOG.
13. Compile and download changes to the controller.
14. Open the APACS+/QUADLOG configuration in *4-mation*.

15. Using *4-mation*, add a connect block in APACS+ to connect to the virtual APACS+ controller created by the IEM from the connection data in step 8. If redundant data paths are used, two connect blocks are required.
16. Configure the name for the controller input in the connect block using the name given to the connection in step 6 above.
17. Using *4-mation*, add the appropriate GW_SEND and GW_REC blocks to the APACS+ configuration. Each GW_SEND block in APACS+ is used to send data to a corresponding GW_REC block in an S7-400 controller. Each GW_REC block in APACS+ receives data from a corresponding GW_SEND block in an S7-400 controller. Set the EN input to TRUE to enable the blocks.

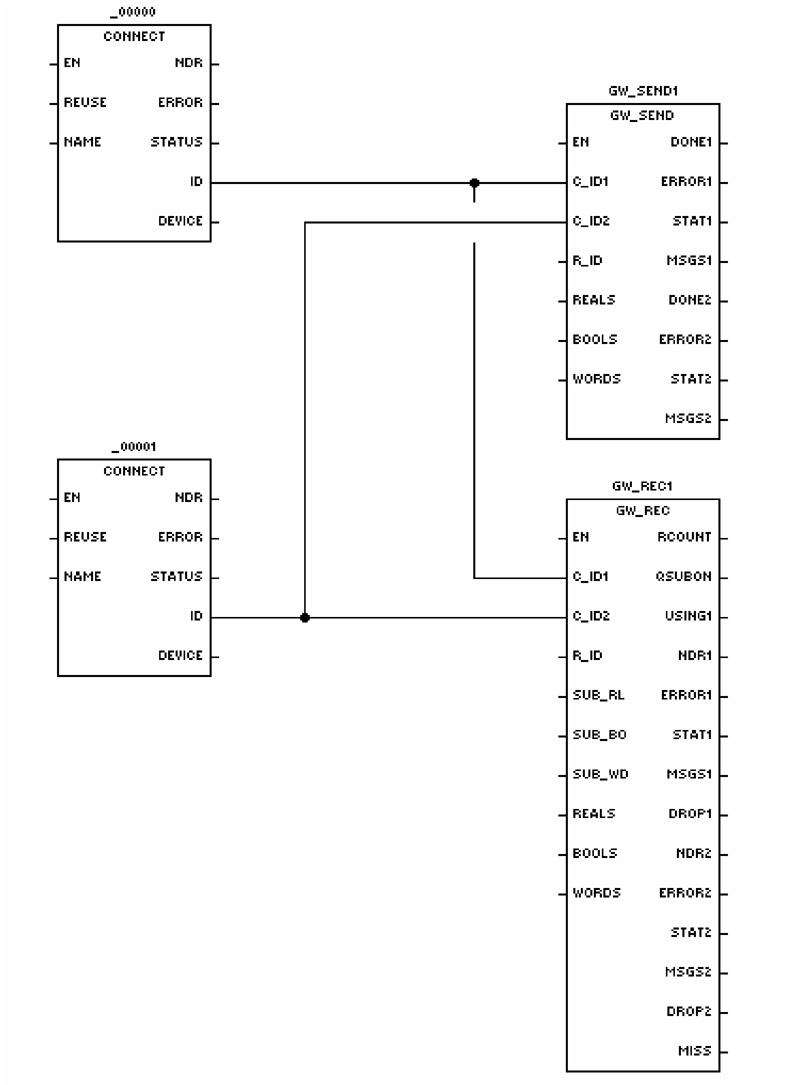


Figure 6-10 Communication Blocks in *4-mation*

18. Connect the ID output of the connect blocks to the ID1 and ID2 inputs of the GW_REC and GW_SEND blocks in APACS+. ID1 is the primary communication channel, and ID2 is the backup channel. (See Figure 2-22.)
19. Configure the R_ID input for the GW_SEND and GW_REC blocks. The R_ID value should match the R_ID value on the corresponding block on the S7-400 controller.
20. Configure the REALS, BOOLS, and WORDS inputs of the GW_SEND blocks with arrays containing the desired data to send to S7. Configure the REALS, BOOLS, and WORDS inputs of the GW_REC blocks with arrays to receive the data sent from S7.
21. Download the APACS+ configuration to the APACS+ resource. APACS+/ QUADLOG should now be able to exchange data with the S7-400 controller.

Figure 2–23 shows the completed configuration. The circled numbers with arrows correspond to the configuration steps above.

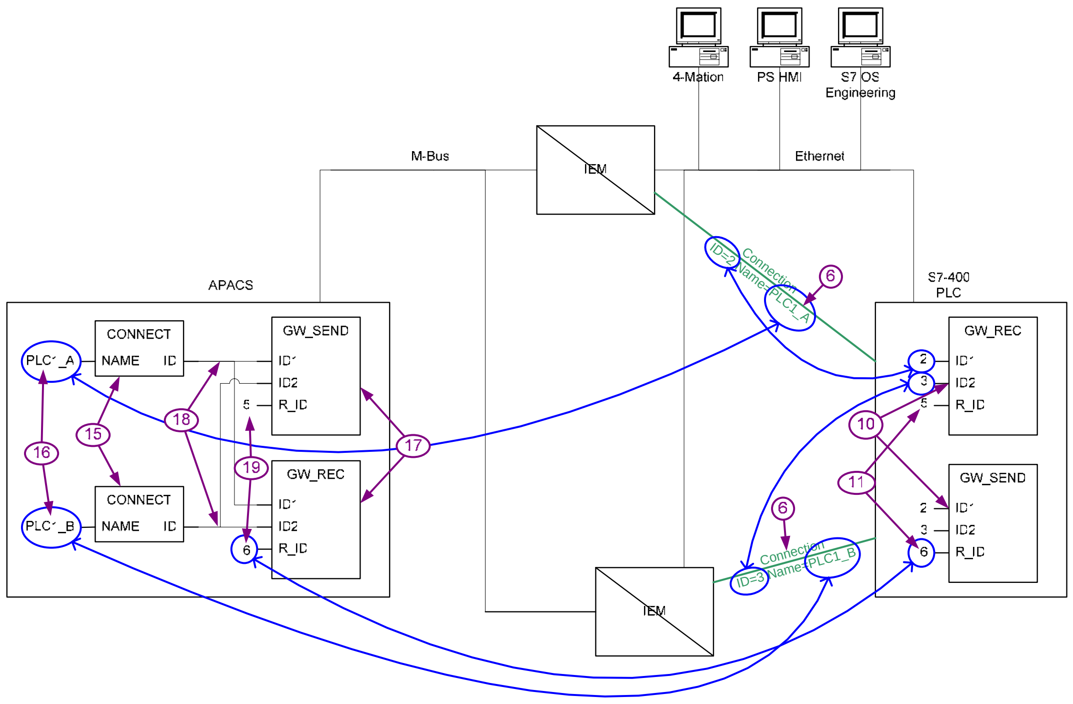


Figure 6-11 STEP 7 / APACS+ Communications Overview

7 Communications Blocks - GW_SEND and GW_REC

7.1 Overview

The APACS+/QUADLOG IEM Communication Function Block Library adds specialized function blocks to *4-mation* to permit the configuration of communications between APACS+/QUADLOG controller resources and S7 Controller resources via the Industrial Ethernet Module (IEM). The IEM is connected to both the MODULBUS network and Industrial Ethernet to serve as a gateway between APACS+/ QUADLOG and S7. The blocks are designed to pass data between an S7 controller through the IEM to an APACS+/QUADLOG resource. In addition, the blocks provide communications between APACS+/QUADLOG resources on different MBUS segments.

The APACS+/QUADLOG library blocks are used in conjunction with the S7 IEM communication blocks as send-receive pairs to provide a data transmission capability. The library contains two function blocks: GW_SEND and GW_REC.

The S7 function blocks for communication with APACS+/QUADLOG are copied from a provided function block library into an S7 project using the SIMATIC Manager. These blocks will be used to communicate with APACS+ (ACM, ACM+, ACMx), QUADLOG (CCM, CCM+, CCMx). The interface is the same in all environments.

7.1.1 Online Configuration

APACS+/QUADLOG blocks can be configured online. After a change to any of the S7 communication block's Remote ID (R_ID) inputs, restart the S7 controller.

All of the other parameters required for communication between APACS+/QUADLOG and SIMATIC PCS 7 are contained in the APACS+/QUADLOG or S7 function blocks. When changing, adding, or removing connected function blocks in either system, no changes are required on the IEM. Active connections between function blocks in the APACS+/QUADLOG and PCS 7 systems are not affected by the configuration and connection of more function blocks.

Notice

If an R_ID value is used that is greater than the MAX R_ID configuration parameter in the IEM, then the MAX R_ID must be reconfigured in the IEM.

Once the IEM Library is properly installed in APACS+/QUADLOG, the *4-mation*™ configuration software will automatically include the IEM GW_SEND and GW_REC function blocks after a new configuration has been initialized. These function blocks can then be selected from *4-mation*’s Standard function block list (they do not appear on the icon bar).

1. Use the DERIVED key and then select the Standard option button to display the list of function blocks in *4-mation*.
2. Highlight the GW_SEND or GW_REC block and choose the SELECT button to place the block on the sheet.

7.1.2 Common Features of the Send Function Block

7.1.2.1 Data Sources

The GW_SEND block sends 32 REAL, 32 BOOL, and 16 WORD values. An R_ID input parameter identifies the block with the corresponding GW_REC block.

In the S7 GW_SEND block, each REAL, BOOL, and WORD value is configured as an individual input parameter of the block. In the APACS+/QUADLOG GW_REC, there is one array input parameter for the REALS, one array input parameter for the BOOLs and one array input parameter for the WORDs.

7.1.2.2 Connection to Receive Function Blocks

The GW_SEND block has two ID inputs specifying the communication paths. These are input from the ID outputs from connect blocks in APACS+.

7.1.3 Common Features of the Receive Function Block

7.1.3.1 Data Destination

The GW_REC block receives 32 REAL, 32 BOOL, and 16 WORD values. An R_ID input parameter identifies the block with the corresponding GW_SEND block.

In the S7 GW_REC block, each received REAL, BOOL, and WORD value is shown as an individual output parameter of the block. In the APACS+/QUADLOG GW_REC, there is one array parameter for the recieved REALS, one array parameter for the received BOOLs and one array parameter for the received WORDs. Note that although these arrays function as outputs, they appear on the input side of the block due to an APACS+ restriction which prevents arrays from appearing on the output side.

In the S7 GW_REC block, each substitute REAL, BOOL, and WORD value is shown as an individual input parameter of the block. In the APACS+/QUADLOG GW_REC, there is one array parameter for the substitute REALS, one array input parameter for the substitute BOOLs and one array input parameter for the substitute WORDs. If the SUBS_ON parameter is true and the block encounters an error, then the configured subsitute values will be copied to the REAL, BOOL and WORD outputs.

7.1.3.2 Connection to Send Function Blocks

The GW_REC block has two ID inputs specifying the communication paths. These are input from the ID outputs from connect blocks in APACS+.

7.2 APACS+/QUADLOG GW_SEND Function Block

The APACS+/QUADLOG GW_SEND function block is used to send data to an S7 controller, or another APACS+ resource. The GW_SEND may be configured to send 32 REALS, 32 BOOLS, and 16 WORD values to an S7 controller resource. The send values are stored in arrays (REALS, BOOLS, and WORD) which appear as inputs to the GW_SEND block. In the S7 or APACS+ resource, a corresponding S7 GW_REC block must be configured to receive the data. An R_ID input parameter identifies the block with the corresponding S7 GW_REC block.

The GW_SEND block has two C_ID inputs specifying the communication paths. These are input from the ID outputs from CONNECT blocks. The CONNECT block must be configured with the connection name of the corresponding S7 Netpro connection to the IEM, or the connection name of the remote APACS+/QUADLOG IEM. For non-redundant communication, configure only C_ID1. For redundant communication, configure both C_ID1 and C_ID2 inputs. When used redundantly, the GW_SEND block transmits the same data to two separate IEMs.

When the EN input is TRUE, the block sends the configured data every other scan. Therefore if the scan rate of the controller is 100ms, the data will be sent at a rate of 200ms. The GW_SEND block transmits at each opportunity, even if the data values are unchanged.

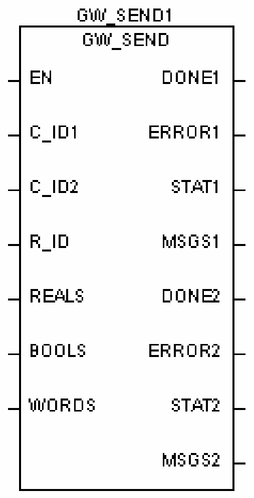


Figure 7-1 APACS+ GW_SEND Block

Notice

The APACS+/QUADLOG GW_SEND block is unable to detect if the S7 controller, containing the GW_REC block is not running.

Table 7–1 GW_SEND Parameters

Parameter	Data Type	Description/Comments
Inputs		
EN	BOOL	When TRUE, the block writes data to the S7 partner GW_REC block (through the IEM) every other scan.
C_ID1	INT	Connection ID - identifies the communication channel between two resources. Obtained from the ID output of the CONNECT block.
C_ID2	INT	Connection ID - identifies an additional communication channel between two resources to be used for redundancy. Obtained from the ID output of another CONNECT block.
R_ID	DINT	Remote Message ID – Must match the R_ID of the corresponding REC block.
REALS	ARRAY	Send Array of REALS. A maximum of 32 elements will be sent. Only a 1 dimensional array of type REAL is permitted.
BOOLS	ARRAY	Send Array of BOOLS. A maximum of 32 elements will be sent. Only a 1 dimensional array of type BOOL is permitted.
WORDS	ARRAY	Send Array of WORDS. A maximum of 16 elements will be sent. Only a 1 dimensional array of type WORD is permitted.
Outputs		
DONE1	BOOL	TRUE = Send command completed successfully this scan on connection ID1.
ERROR1	BOOL	Error indicator. TRUE = problem writing the data on connection ID1 or configuration error.
STAT1	STRING	Contains a string description of the last detected error code for connection ID1. Empty if no error.
MSGS1	UDINT	Number of Messages output for connection ID1. Indicates the number of complete send (write) operations since the function block was last initialized. Disabling and re-enabling the CONNECT block used in conjunction with this function block zeros this value. This number is also reset when it reaches 1000.
DONE2	BOOL	TRUE = Send command completed successfully this scan on connection ID2.
ERROR2	BOOL	Error indicator. TRUE = problem writing the data on connection ID2 or configuration error.
STAT2	STRING	Contains a string description of the last detected error code for connection ID2. Empty if no error.

MSGS2	UDINT	Number of Messages output for connection ID2. Indicates the number of complete send (write) operations since the function block was last initialized. Disabling and re-enabling the CONNECT block used in conjunction with this function block zeros this value. This number is also reset when it reaches 1000.

Softlist		
Timeout	UINT	Block Timeout parameter in milliseconds. If the block does not receive a response to a write message within the duration established by this timeout value, the block indicates an error on its STATUS output. Default = 3000
Version	STRING	Contains the version string of the function block for easy identification.

7.2.1 Error Codes

The STATUS output of the GW_SEND block can provide the error codes/messages shown in Table 7–2.

Table 7–2 GW_SEND Error Codes and Messages

ERR_CODE	Explanation
38 Dest Database Rev Changed	Indicates the configuration in the destination resource (the IEM) has changed and this function block must reinitialize.
39 Destination Did Not Respond	Indicates the destination resource did not respond within the timeout period determined by the Timeout softlist parameter with an acknowledge signal.
40 Tag Name Not Found in Dest	Indicates the function block has failed to find the desired tag name in the destination resource. This error is typically seen if the R_ID exceeds the maximum allowed by the IEM.
45 Invalid ID Input	Indicates that this function block is not connected to the Resource Name specified by the CONNECT function block. The CONNECT block's ID output must be connected to this block's ID input, and the connection ID must be non-zero.
46 Initializing Block	Indicates that this function block is performing initialization because of one of the following reasons: The configuration has changed A communications error has occurred A configuration error has occurred
47 Array Input Not Valid	Indicates that one or more of the input send arrays (REALS, BOOLS, WORDS) is not valid. If an array is connected, it must have the expected data type and one dimension. At least one of these arrays must be configured for the block to send data, but it is not necessary to configure ones that are not needed.
48 Block Out Of Memory	Indicates the function block has run out of memory for the communications resources required to send messages. It may be necessary to reduce communication function block configuration.
49 Waiting For Response	Indicates the function block is waiting for the destination resource to respond with either an acknowledge signal or the data being read.
52 Invalid R_ID	The R_ID input is invalid for one of the following reasons: the input is not connected or the R_ID exceeds the maximum allowed by the Gateway.

7.3 APACS+/QUADLOG GW_REC Function Block

The APACS+/QUADLOG GW_REC function block is used to receive data from an S7 controller. The GW_REC may be configured to receive 32 REALS, 32 BOOLS, and 16 WORD values from an S7 controller resource. The received values are stored in arrays (REALS, BOOLS, and WORDS) which appear as inputs to the GW_REC block. In the S7 controller, a corresponding S7 GW_SEND block must be configured to send the data. An R_ID input parameter identifies the block with the corresponding S7 GW_SEND block.

The GW_REC block has two C_ID inputs specifying the communication paths. These are input from the ID outputs from CONNECT blocks. The CONNECT block must be configured with the connection name of the corresponding S7 Netpro connection to the IEM, or the connection name of the remote APACS+/QUADLOG IEM. For non-redundant communication, configure only C_ID1. For redundant communication, configure both C_ID1 and C_ID2 inputs. If operating redundantly, the data received on C_ID1 will always be used in the receive arrays when no error occurs. If an error occurs on C_ID1 and data is received successfully on C_ID2, then the data from C_ID2 will be used.

The receive arrays are updated after a 0-to-1 transition of the signal at output NDR1 or NDR2 ("New Data Ready" on C_ID1 or C_ID2, respectively). Block outputs are provided on each connection for NDR, ERROR, STAT, MSGS, and DROP. These outputs indicate "New Data Received", ERROR received, STAT value for the receive, number of MSGS received, and number of dropped messages. For example, connection one has the outputs NDR1, ERROR1, STAT1, MSGS1, and DROP1.

The block output for RCOUNT indicates the number of received REAL values and the block output QSUBS_ON indicates that the output values for the BOOLS, WORDS, and REALS are not received values, but instead are the substitute values.

The GW_SEND block transmits its data with its own internal incrementing "reference count" (REFCNT). The GW_REC block checks for the data received on each connection to determine if a "reference count" has been missed. The DROP value is incremented whenever the data received on the connection has a "reference count" exceeding the next expected value. For example, if the last data had REFCNT=1, then the next expected REFCNT=2. However, if the next data had REFCNT=3, then DROPn would be incremented. A DROP can occur if the GW_SEND block is sending at a faster rate then the GW_REC can receive. A DROP can also occur due to transient network upsets/congestion or if programming, debugging, or downloading is occurring to either the GW_SEND or GW_REC controller.

Even though one connection might see a DROP condition, the data will most likely come over the redundant connection. When used redundantly, the GW_SEND block transmits the same data (same REFCNT) to two separate IEMs. If the DROP is seen on one connection, it might not occur on the other. Only when GW_REC misses the next REFCNT entirely is the MISS output set to TRUE.

The GW_REC block also permits using substitute values for the 32 REALS, 32 BOOLS, and 16 WORD values when an error occurs. Two settings are required to enable value substitution. Unless the soflist parameter SUBS_ON is true, no substitution occurs. When SUBS_ON=TRUE, then substitution occurs after REC_MON cycles have elapsed and no new data has been received or another error has occurred. The absence of new data could be due to an error locally or remotely. When SUBS_ON=FALSE, the block will report “Stale Data” in the absence of new data after REC_MON cycles have elapsed.

The REC_MON should be set to a value outside of the normal receive behavior for the block. For instance, if the GW_REC is placed in a controller with a fast scan rate, then many scans might occur before any new data is received. When GW_REC is used with a slow scan rate then fewer "no data" scans would occur between receives. The setting made for REC_MON is typically made once the update behavior of the GW_SEND/ GW_REC data exchange has been observed. However, a large setting can be made based on the expected performance. For example, if the expectation is to receive data every second, then the REC_MON setting can be configured to use the value substitution after 10 "idle" seconds. If the GW_REC is placed in a controller running at 100ms cycle, then 10 seconds corresponds to 100 calls (make REC_MON=100).

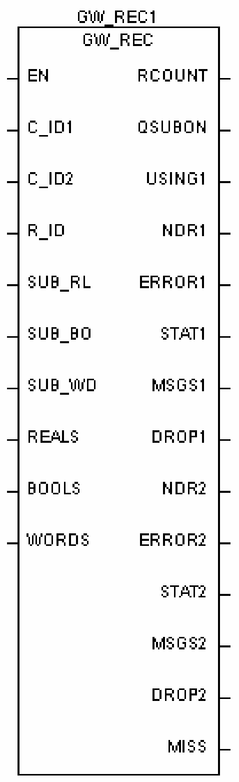


Figure 7-2 GW_REC Block in APACS+

Table 7–3 GW_REC Parameters

Parameter	Data Type	Description/Comments
Inputs		
EN	BOOL	When TRUE, the block reads data from the S7 partner GW_SEND block (through the IEM) each scan.
C_ID1	INT	Connection ID - identifies the communication channel between two resources. It is to be obtained from the ID output of the CONNECT block.
C_ID2	INT	Connection ID - identifies an additional communication channel between two resources to be used for redundancy. Obtained from the ID output of another CONNECT block. The data from this connection will not be used unless a problem is detected with connection ID1.
R_ID	DINT	Remote Message ID – Must match the R_ID of the corresponding SEND block.
SUB_RL	ARRAY	Substitute Array of REALS. These values, if configured, will be copied to the RD_RL array if the block does not receive data and the SUBON parameter is TRUE. Only a 1 dimensional array of type REAL is permitted.
SUB_BO	ARRAY	Substitute Array of BOOLS.These values, if configured, will be copied to the RD_BO array if the block does not receive data and the SUBON parameter is TRUE. Only a 1 dimensional array of type BOOL is permitted.
SUB_WD	ARRAY	Substitute Array of WORDS. These values, if configured, will be copied to the RD_WD array if the block does not receive data and the SUBON parameter is TRUE. Only a 1 dimensional array of type WORD is permitted.
REALS	ARRAY	Received Array of REALS. This array holds the REAL values received from S7. A maximum of 32 REALS may be received. Only a 1 dimensional array of type REAL is permitted.
BOOLS	ARRAY	Received Array of BOOLS. This array holds the BOOL values received from S7. A maximum of 32 BOOLS may be received. Only a 1 dimensional array of type BOOL is permitted.
WORDS	ARRAY	Received Array of WORDS. This array holds the WORD values received from S7. A maximum of 16 WORDS may be received. Only a 1 dimensional array of type WORD is permitted.
Outputs		
RCOUNT	INT	Indicates how many REAL values were received from S7. [The S7 GW_SEND allows the user to configure how many REALS to send]

Parameter	Data Type	Description/Comments
QSUBON	BOOL	Indicates when substitute values are being used. True = Substitute values used
USING1	BOOL	Indicates True when the data is being used from Connection 1. False Otherwise.
NDR1	BOOL	New Data Ready True = New Data Received on connection ID1
ERROR1	BOOL	Error indicator for connection ID1 True=Communication or Configuration Error
STAT1	STRING	Contains a string description of the last detected error code on connection ID1.
MSGS1	UDINT	Number of Messages output for connection ID1. Indicates the number of complete receive (read) operations since the function block was last initialized. Disabling and re-enabling the CONNECT block used in conjunction with this function block zeros the NUMMSG value.
DROP1	UDINT	Indicates how many times a missed message was detected on connection ID1. Value is reset when MSG1 is reset.
NDR2	BOOL	New Data Ready True = New Data Received on connection ID2
ERROR2	BOOL	Error indicator for connection ID2 True=Communication Error
STAT2	STRING	Contains a string description of the last detected error code on connection ID2.
MSGS2	UDINT	Number of Messages output for connection ID2. Indicates the number of complete receive (read) operations since the function block was last initialized. Disabling and re-enabling the CONNECT block used in conjunction with this function block zeros the NUMMSG value.
DROP2	UDINT	Indicates how many times a missed message was detected on connection ID2. Value is reset when MSG2 is reset.
MISS	BOOL	Indicates whether data was missed on one connection when operating non-redundantly, or on both connections when operating redundantly.
Softlist		
Timeout	UINT	Block Timeout parameter in milliseconds. If the block does not receive a response to a read message within the duration established by this timeout value, the block writes ‘Timeout’ to its STATUS output. Default = 3000

Parameter	Data Type	Description/Comments
REC_MON	UINT	Receive monitoring (cycles) Max number of cycles that block will tolerate stale data before posting an error and using substitute values (if SUB_ON = True). Default = 3. A value of 0 will cause the receive monitoring functionality to be effectively turned off.
SUBS_ON	BOOL	Substitution On. If TRUE, substitute values will be used if error occurred while receiving data or if no data received. If SUBS_ON is TRUE, the the substitution arrays which correspond to the connected send arrays must be configured; otherwise an error will occur. For example, if the block is configured to send REALS (the REALS input is connected) and SUBS_ON is TRUE, then the SD_RL array input must be configured.
Version	STRING	Contains the version string of the function block for easy identification

Notice

If the received number of REALS, BOOLS, and WORDS exceeds the size of the corresponding connected receive array, the remaining elements are ignored, and no error is indicated.

7.3.1 Error Codes

The STATUS output of the GW_REC block provides the error codes/messages shown in Table 7–4.

Table 7-4 GW_REC Error Codes and Messages

ERR_CODE	Explanation
38 Dest Database Rev Changed	Indicates the configuration in the destination resource (the IEM) has changed and this function block must reinitialize.
39 Destination Did Not Respond	Indicates the destination resource did not respond within the timeout period determined by the Timeout softlist parameter with an acknowledge signal.
40 Tag Name Not Found in Dest	Indicates the function block has failed to find the desired tag name in the destination resource. This error is typically seen if the R_ID exceeds the maximum allowed by the IEM.
45 Invalid ID Input	Indicates that this function block is not connected to the Resource Name specified by the CONNECT function block. The CONNECT block's ID output must be connected to this block's ID input, and the connection ID must be non-zero. **Note:** This can also occur when the sending controller is stopped.
46 Initializing Block	Indicates that this function block is performing initialization because of one of the following reasons: • The configuration has changed • A communications error has occurred • A configuration error has occurred
47 Array Input Not Valid	Indicates that one or more of the array inputs is not valid. At least one input receive array must be connected and valid for the block to operate. If an array is connected, it must be of the expected data type and it must be one dimensional.
48 Block Out Of Memory	Indicates the function block has run out of memory for the communications resources required to send messages. It may be necessary to reduce communication function block configuration.
49 Waiting For Response	Indicates the function block is waiting for the destination resource to respond with either an acknowledge signal or the data being read. This may result if the S7 controller is shut down.

ERR_CODE	Explanation
50 Incompatible Msg Format	Block is receiving a message from an S7 GW_SEND block which is incompatible with this version of the GW_REC.
51 Stale Data Received	Block is receiving data which is stale – the same data is being received more than once, possibly because the partner GW_SEND block or the IEM is experiencing communications problems. The scan rate of the GW_SEND block may need to be adjusted.
52 Invalid R_ID	The R_ID input is invalid for one of the following reasons: the input is not connected, the R_ID exceeds the maximum allowed by the Gateway, the corresponding R_ID is not configured in S7, or the S7 controller is stopped.

7.4 S7 GW_SEND Function Block (FB 598)

7.4.1 Description

The S7 GW_SEND function block forms a simple user interface to the block SFB12 BSEND. It is designed to provide redundant Ethernet transmission of the data to two separate IEMs. In redundant mode, two separate S7 connections are used. The block can also be used in non-redundant mode using only a single S7 connection.

The block sends 32 BOOL values, 16 WORD values, and up to 32 REAL values. A corresponding GW_REC block is used to receive the data. The IEM is the interface to the MBUS network of APACS+ controllers. The APACS+ controllers also have equivalent versions of the GW_SEND and GW_REC blocks.

The S7 GW_SEND block looks like the picture below. The parameters for the GW_SEND block are described in the tables below.

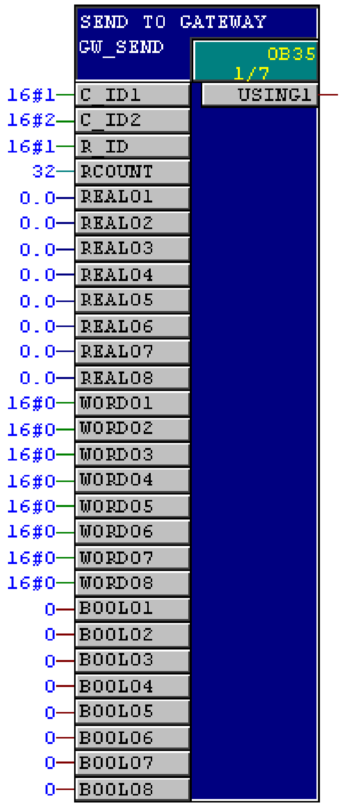


Figure 7-3 S7 GW_SEND Block

Table 7–5 GW_SEND Default Visible Parameters

Parameter	Declaration	Data Type	Description
C_ID1,C_ID2	INPUT	WORD	Connection ID
R_ID	INPUT	DWORD	Remote ID
RCOUNT	INPUT	INT	Number of REAL values
BOOL01..BOOL08	INPUT	BOOL	BOOLEAN values
WORD01..WORD08	INPUT	WORD	WORD values
REAL01…REAL08	INPUT	REAL	REAL values
DONE1, DONE2	OUTPUT	BOOL	Transmission complete
USING1	OUTPUT	BOOL	Connection 1 sending OK

Table 7–6 GW_SEND Additional Parameters

Parameter	Declaration	Data Type	Description
BOOL09..BOOL32	INPUT	BOOL	BOOLEAN values
WORD09..WORD16	INPUT	WORD	WORD values
REAL09…REAL32	INPUT	REAL	REAL values
DONE1, DONE2	OUTPUT	BOOL	Transmission complete
ERROR1, ERROR2	OUTPUT	BOOL	Send job had error
STAT1, STAT2	OUTPUT	WORD	Status of send job
MSGS1, MSGS2	OUTPUT	WORD	Messages sent

7.4.2 Operating Principle

The internal SFB12 BSEND is used to exchange the data values between the communication partners. The ERROR and STAT outputs for GW_SEND come directly from the BSEND block (refer to its Help documentation). Each connection has its own ERROR and STAT output (ERROR1, STAT1 and ERROR2, STAT2).

The output USING1 indicates that the primary connection (C_ID1) is actively sending the data. USING1 is false only when C_ID1 has some error, but C_ID2 does not.

Each connection also has outputs for DONE and MSGS. The output “DONE” indicates that the transmission was completed and a new transmission can begin. The GW_SEND block is capable of sending data on every second call (provided that the last data sent was received by its partner). The MSGS value is incremented on each send and resets to zero on any ERROR or when the value reaches 1000. The MSGS value is intended to show the current behavior of the block. For example, comparison of MSGS1 and MSGS2 can be done to confirm that both of the connections are sending without error. To accumulate message counts over different durations, the DONE1 and DONE2 signals can be used to drive other logic (such as an ADD operation to increment by one).

When new data is ready to be sent, a REFCNT (reference count) is incremented internal to the block (and also transmitted). The transmitted block holds an internal “format” value, the RCOUNT value, the REFCNT value followed by the 32 BOOL, the 16 WORD, and the number of REAL values indicated by the RCOUNT. The GW_REC block accepts the data provided its “format” value is correct, and its REFCNT exceeds its previous received REFCNT (and roll-over of this 32-bit value is taken into account).

7.4.3 Performance Considerations

The GW_SEND block checks and attempts a new send on each second call. The cyclic block (such as OB35) holding the GW_SEND determines the cycle time between block calls. For instance, if GW_SEND is placed in OB35, which has a 100ms cycle, then the GW_SEND will send every 200ms at most.

The GW_REC block also takes two calls to complete a new receive. While GW_REC and GW_SEND can both be placed in the same OB cycle, best performance is possible when the GW_REC is called at a rate more than twice as fast as the GW_SEND block. For example, if the GW_SEND block is sent every 200ms then the corresponding GW_REC block should be placed in an OB with a cycle smaller than 100ms. As a general guideline, the GW_REC block(s) should just be placed in the fastest OB cycle suitable for the program.

Each GW_SEND/GW_REC block uses a unique Remote ID (R_ID). For each connection, the Remote ID should begin at “1” and be assigned consecutively.

In this case, the data for each Remote ID is effectively sent and received in sequence (each R_ID in turn) with a correspondingly longer time to transmit and process. In general, the processing cycles chosen for best performance for a single GW_SEND/GW_REC should be multiplied by the number of GW_SEND/GW_REC resources used over the same connection. For example, if two resources, R_ID=1 and R_ID=2 are used, then the two GW_SEND calls should be made every 400ms.

For the connections using the GW_SEND and GW_REC blocks to transfer data, do not intermix calls to BSEND/BRCV or other communications blocks with different R_ID values over the same connections used by GW_SEND/GW_REC.

In the S7 controller, a receive block (GW_REC) is capable of receiving provided it has processed a previously received data block. When GW_REC is idle it can receive during one cycle. On its next call it will process the data and be ready to receive again. It basically takes two cycles of GW_REC to get new data. For this reason and to account for the overhead of transmission to/from the IEM and the APACS+ controllers, the GW_REC block needs to be called on a cycle more than twice as fast as GW_SEND.

GW_SEND always sends the 32 BOOLS, and the 16 WORD values. An input parameter “RCOUNT” defines the number of REAL values to send. RCOUNT can be set to a value of 0 up to 32.

Notice

In the S7-400 the settings for the block connection numbers (C_ID1, C_ID2) and the Remote ID (R_ID) are bound at the time of Cold-Start. If these are changed during runtime, the block continues to function with the settings present at the time of ColdStart. These values should be set once when the program is compiled and left alone during program execution.

7.4.4 H-Systems Considerations

For the S7-400 H system, use NetPro to assign each connection to a specific S7 CPU. Since the IEM has a single Ethernet interface, there will be a separate IEM for each network (and each connection). The connection assignment is made for a standard “**S7 connection**” (not for the “S7 connection fault tolerant connection”). However, the hardware configuration for the S7 CP443 module should have the setting selected for **Activate Fast switchover of the connections** (appearing on the configuration page for Options/Ethernet Profile for Fault Tolerant Connections). With this configuration, both connections are active when both the master and slave CPUs are running and both connections will remain running with a fault in one of the CP443s. In the case of a CPU failure (or while in STOP), one connection will still be running in the master CPU. With a separate IEM on each network, this configuration maintains the data transmission in the case of a single failure point during operation.

7.4.5 Error Handling

All ERROR and STAT output settings correspond directly to the values defined by the BSEND block except for one. The GW_SEND block also defines the case of ERROR=1 with STAT=255 as an invalid configuration (C_ID1, C_ID2, or R_ID are invalid).

Additonal error codes and descriptions are listed in Table 7–7.

Table 7–7 GW_SEND Error Codes

Error	Status	Description
0	11	Warning: new job is not effective since the previous job is not yet completed.
0	25	Communication has started. The job is being processed.
1	1	Communications problems.
1	2	Negative acknowledgement from the partner SFB/FB. The function cannot be executed.
1	3	R_ID is unknown on the connection specified by the ID or the receive block not called.
1	4	Error in the send area pointer SD_1 or the data length LEN.
1	5	Reset request was executed.
1	6	Partner SFB/FB is in the DISABLED state (EN_R has the value 0).
1	7	Partner SFB/FB is in the wrong state(not redy to receive again).
1	8	Access to remote object in the user memory was rejected.
1	10	Access to the local user memory not possible (for example, access to a deleted DB).
1	12	No instance DB found (loading a new instance DB from the PG).
1	18	R_ID already exists in the connection ID.
1	20	Insufficient memory. H-System: SFB first called while update in progress.
1	255	Incorrect settings for C_ID1, C_ID2, R_ID or RCOUNT. Values not in range.

7.5 S7 GW_REC Function Block (FB 599)

7.5.1 Description

The GW_REC block represents a simple user interface to SFB13 BRCV. It is designed for use with the IEM product, which provides the connectivity to APACS+ controllers on the MBUS network. The block is designed for redundant communications over two standard S7 connections, each on a separate Ethernet network having a separate IEM for each connection.

The GW_REC block receives 32 BOOL, 16 WORD, and 32 REAL values from its corresponding GW_SEND block. It is designed for redundant communications using two S7 connections. The data transmitted by the GW_SEND block in the remote controller is received in the S7 with a corresponding GW_REC block that uses the same connection(s) and Remote ID setting (R_ID). If used non-redundantly, the C_ID1 setting specifies the connection number.

Data is only available after the job is completed, and after a 0-to-1 transition of the signal at output NDR1 or NDR2 (“New Data Ready” on connection 1 or connection 2, respectively).

The S7 GW_REC block looks like the picture below. The parameters for the GW_REC block are described in the following tables.

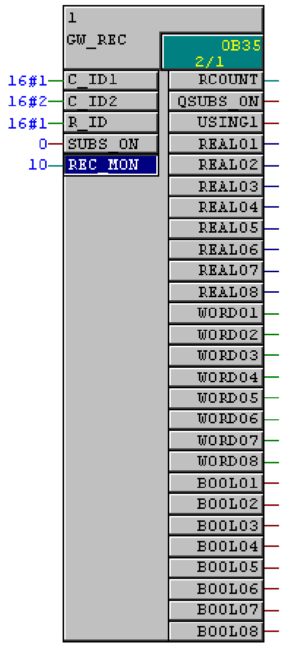


Figure 7-4 S7 GW_REC Block

Table 7–8 GW_REC Default Visible Parameters

Parameter	Declaration	Data Type	Description
C_ID1,C_ID2	INPUT	WORD	Connection ID
R_ID	INPUT	DWORD	Remote ID
SUBS_ON	INPUT	BOOL	Enable substitution
REC_MON	INPUT	INT	Number of calls before substitution done.
RCOUNT	OUTPUT	INT	Number of REAL values
QSUBS_ON	OUTPUT	BOOL	Substitute values set
USING1	OUTPUT	BOOL	Connection 1 OK
BOOL01..BOOL08	OUTPUT	BOOL	BOOLEAN values
WORD01..WORD08	OUTPUT	WORD	WORD values
REAL01…REAL08	OUTPUT	REAL	REAL values

Table 7–9 GW_REC Additional Parameters

Parameter	Declaration	Data Type	Description
SUB_BO01… SUB_BO32	INPUT	BOOL	Substitute BOOL values
SUB_WD01… SUB_WD16	INPUT	WORD	Substitute WORD values
SUB_RD01… SUB_RD32	INPUT	REAL	Substitute REAL values
BOOL09..BOOL32	OUTPUT	BOOL	BOOLEAN values
WORD09..WORD16	OUTPUT	WORD	WORD values
REAL09…REAL32	OUTPUT	REAL	REAL values
NDR1, NDR2	OUTPUT	BOOL	New DataReady
ERROR1, ERROR2	OUTPUT	BOOL	Receive error
STAT1, STAT2	OUTPUT	WORD	Status of job
MSGS1, MSGS2	OUTPUT	WORD	Messages received
DROP1, DROP2	OUTPUT	WORD	Messages missed
MISS	OUTPUT	WORD	Data missed

7.5.2 Operating Principle

The GW_REC block takes two calls to complete a new receive. Best performance is possible when the GW_REC is called at a rate more than twice as fast as the rate at which the data is being transmitted. For example, if the GW_SEND block sends data every 200ms, then the corresponding GW_REC block should be placed in an S7 OB with a cycle smaller than 100ms. As a general guideline, the GW_REC block(s) should just be placed in the fastest OB cycle suitable for the program.

Each GW_SEND/GW_REC block uses a unique Remote ID (R_ID). For each connection, the Remote ID should begin at “1” and be assigned consecutively.

The block inputs for connection configuration are the parameters for C_ID1 (connection 1 identifier), C_ID2 (connection 2 identifier), and the R_ID setting (Remote ID). The R_ID is a unique setting having a value of 1 or greater and must be assigned consecutively for each connection. The R_ID should match the value given in the corresponding GW_SEND block.

Notice

In the S7-400 the settings for the block connection numbers (C_ID1, C_ID2) and the Remote ID (R_ID) are bound at the time of Cold-Start. If these are changed during runtime, the block continues to function with the settings present at the time of ColdStart. These values should be set once when the program is compiled and left alone during program execution.

Block outputs are provided on each connection for NDR, ERROR, STAT, MSGS, and DROP. These outputs indicate “New Data Received”, ERROR received, STAT value for the receive, number of MSGS received, and number of dropped messages. For example, connection one has the outputs NDR1, ERROR1, STAT1, MSGS1, and DROP1.

The block output for RCOUNT indicates the number of received REAL values and the block output QSUBS_ON indicates that the output values for the BOOLS, WORDS, and REALS are not received values, but instead are the substitute values.

The internal SFB13 BRCV is used to exchange the data values between the communication partners. The ERROR and STAT outputs for GW_REC come directly from the BRCV block (refer to its Help documentation). Each connection has its own ERROR and STAT output (e.g. ERROR1, STAT1 and ERROR2 and STAT2).

However, GW_REC also generates its own error STAT value of 255 if the inputs C_ID1, C_ID2, R_ID are misconfigured (e.g., R_ID1=0). GW_REC also generates the error STAT value of 254 whenever data is “stale” and not being received. When SUBS_ON is false, but REC_MON is set greater than zero, then the “stale data” condition is made (ERROR=true, STAT=254) when GW_REC is called REC_MON number of times and no new data is received. If REC_MON (record monitoring) is not required, then set REC_MON to zero. The “stale data” condition is also made (ERROR=true, STAT=254) whenever substitute values are being used. If SUBS_ON=true and REC_MON are greater than zero, then the substitute values are used whenever REC_MON number of calls have been made and no data was received. The output QSUBS_ON is true whenever the substitute values are used to set the output values.

Once new data is received, the “stale data” condition is reset (ERROR=false and STAT=0) and QSUBS_ON is set to false. The setting of STAT=254 is made when there is not some other error already present for the connection (STAT will be set to 254 only when it is currently set to zero).

The output USING1 indicates that the primary connection (C_ID1) is actively receiving the data. USING1 is false only when C_ID1 has some error, but C_ID2 does not.

The output NDR indicates that a new data instance has been received. The GW_REC block is capable of receiving data on every second call of each connection. The MSGS value is incremented on each receive and resets to zero on any ERROR or when the value reaches 1000. The MSGS value is intended to show the current behavior of the block. For example, comparison of MSGS1 and MSGS2 can be done to confirm that both of the connections are receiving without error. To accumulate message counts over different durations, the NDR1 and NDR2 signals can be used to drive other user-programmed logic blocks (such as an ADD operation to increment by one).

The GW_SEND block transmits its data with its own internal incrementing “reference count” (REFCNT). The GW_REC block checks for the data received on each connection to determine if a “reference count” has been missed. The DROP value is incremented whenever the data received on the connection has a “reference count” exceeding the next expected value. For example, if the last data had REFCNT=1, then the next expected REFCNT=2. However, if the next data had REFCNT=3, then DROPn would be incremented. A DROP can occur if the GW_SEND program is sending at a faster rate then the GW_REC can receive. A DROP can also occur due to transient network upsets/congestion or if programming, debugging, or downloading is occurring to either the GW_SEND or GW_REC controller.

Even though one connection might see a DROP condition, the data will most likely come over the redundant connection. When used redundantly, the GW_SEND block transmits the same data (same REFCNT) to two separate IEMs. If the DROP is seen on one connection, it might not occur on the other. Only when GW_REC misses the next REFCNT entirely is the MISS output set to true.

The GW_SEND block transmits at each opportunity, even if the data values are unchanged. The DROP and MISS outputs are intended for use in setting up and commissioning the operation of the data exchange between the GW_SEND and GW_REC controllers. That is, to determine if the data exchange is performing as expected. In actual practice, the data values themselves should be used in the controller programs to “handshake” or otherwise control and/or influence behavior between the controllers.

The block also permits using substitute values for the 32 BOOL, 16 WORD, and the (up to) 32 REAL values. Two input settings are required to enable value substitution. Unless the input SUBS_ON is true, no substitution occurs. When SUBS_ON=true, then substitution occurs after REC_MON calls have elapsed and no new data has been received. The absence of new data could be due to an error locally or remotely.

The REC_MON should be set to a value outside of the normal receive behavior for the block. For instance, if the GW_REC is placed in a fast cyclic OB, then many calls might occur before any new data is received. When GW_REC is placed in a slower OB, then fewer “no data” calls would occur between receives. The setting made for REC_MON is typically made once the update behavior of the GW_SEND/ GW_REC data exchange has been observed. However, a large setting can be made based on the expected performance. For example, if the expectation is to receive data every second, then the REC_MON setting can be used to do the value subsitution after 10 “idle” seconds. If the GW_REC is placed in an OB running at 100ms cycle, then 10 seconds corresponds to 100 calls (make REC_MON=100).

7.5.3 Error Handling

All ERROR and STAT output settings correspond directly to the values defined by the BRCV block except for one. The GW_REC block also defines the case of ERROR=1 with STAT=255 as an invalid configuration (C_ID1, C_ID2, or R_ID are invalid).

The GW_REC error codes and descriptions are listed in Table 7–10.

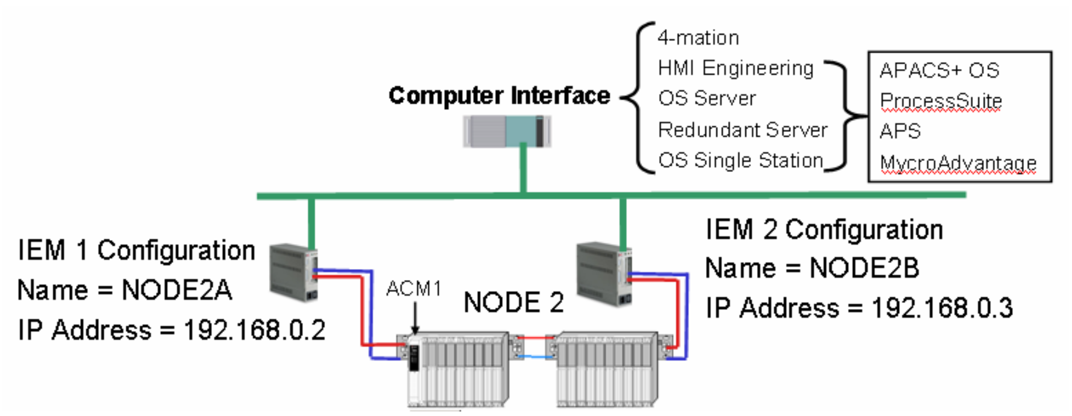
Table 7–10 GW_REC Error Codes

Error	Status	Description
0	11	Warning: New job is not effective since the previous job is not yet completed
0	17	Warning: Block receiving data asynchronously
0	25	Data being received
1	1	Communications problems
1	2	Function cannot be executed (protocol error).
1	4	Error in the receive area pointer RD_1 regarding the data length or data type.
1	5	Reset request received, incomplete transfer.
1	8	Access error in the corresponding SFB/FB 12 "BSEND".
1	10	Access to the local user memory not possible (for example, access to a deleted DB)
1	12	No instance DB found (loading a new instance DB from the PG).
1	18	R_ID already exists in the connection ID.
1	20	Insufficient memory. H-System: SFB first called while update in progress·
1	254	Data has not come in REC_MON calls: “stale data“ or substitute values in use
1	255	Incorrect settings for C_ID1, C_ID2 or R_ID

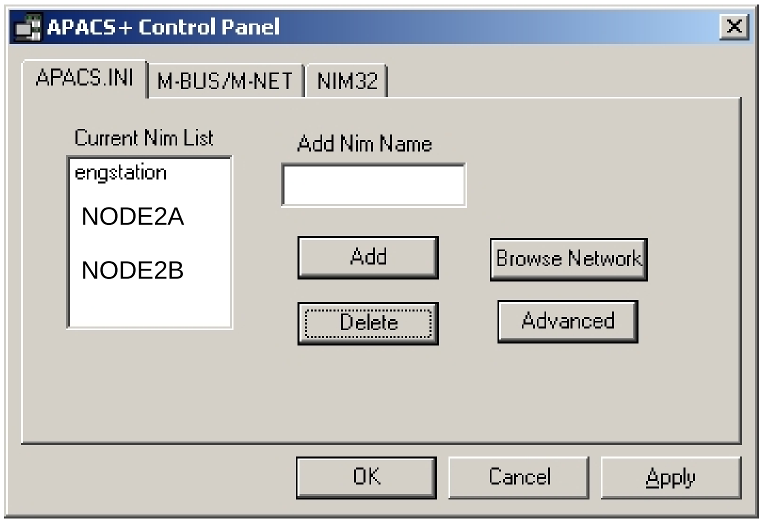
8 Redundancy

8.1 Configuring Redundancy for NIM32 Operation

The IEM Name and IP Address must be unique.



Add the IEM Names to the Nim List on the Engineering Station. APACS+ communications is non-deterministic in that a connection will be made to both NODE2A and NODE2B, however the data may come from either at any give time.

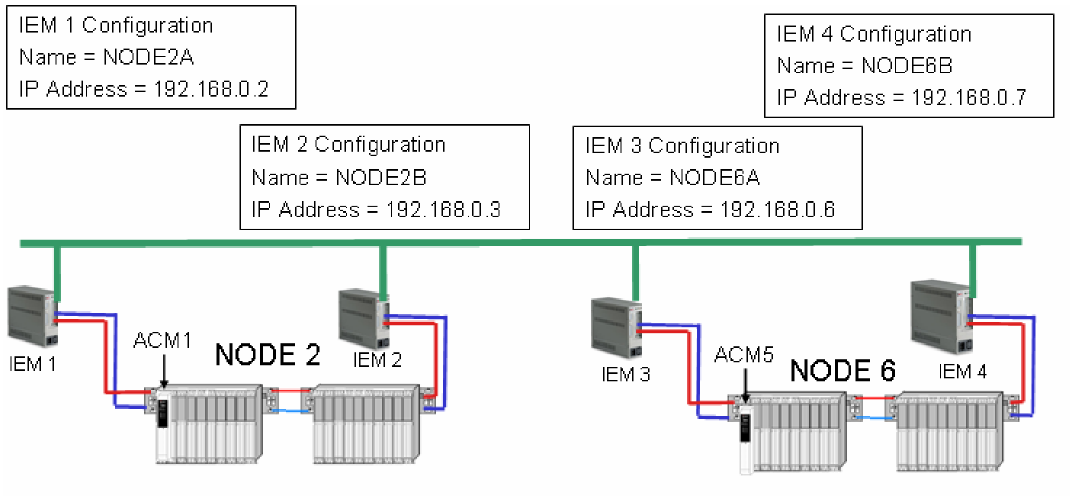


Current Nim List
engstation
NODE2A
NODE2B

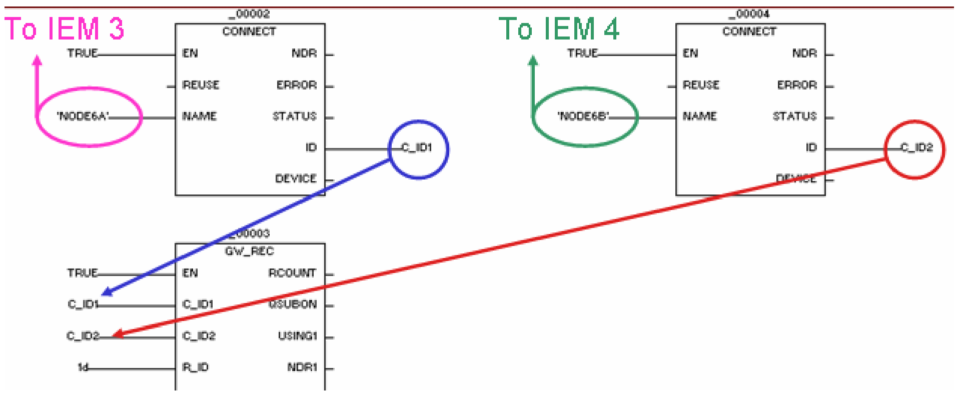
Add Nim Name

8.2 Configuring Redundancy for IEM2IEM Operation

Configure the IEM2IEM.txt file such that IEM 1 passes data to IEM 3 and IEM 2 passes data to IEM4. Due to the fact that IEM 1 and IEM 2 are connected to the same MBUS, they both cannot pass data to IEM 3.

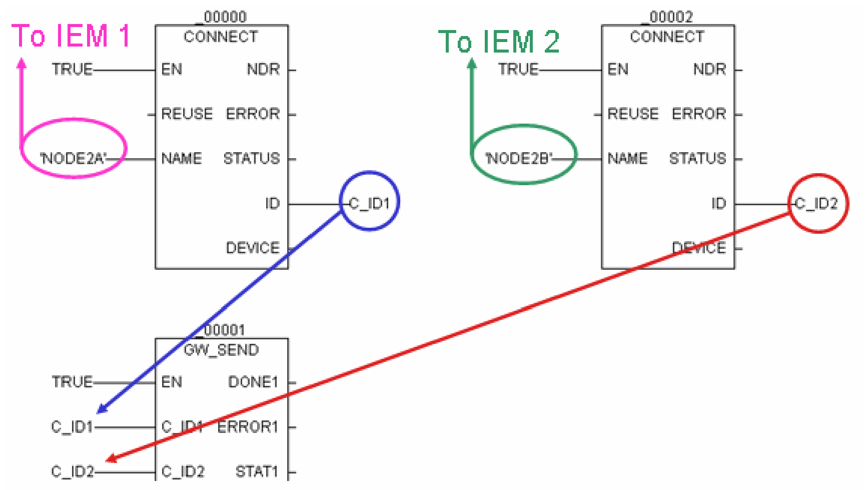


8.3 Configure the GW_SEND and GW_REC Blocks for Redundancy in ACM1



1. Add a second CONNECT block, and configure it to communicate to the redundant IEM.
2. Connect the ID nub to the C_ID2 nub of the GW_REC block. No other configuration is required.

8.4 Configure the GW_SEND and GW_REC Blocks for Redundancy in ACM5



1. Add a second CONNECT block, and configure it to communicate to the redundant IEM.
2. Connect the ID nub to the C_ID2 nub of the GW_SEND block. No other configuration is required.

9 NIM32 Operation

This chapter describes the operation, configuration, user interface, and diagnostics of the NIM32 software. The chapter is divided into the following sections:

- **Section 9.1 Introduction** – Provides product description, product support information, and related literature.
- **Section 9.2 Operation** – Describes the functional requirements of the NIM32 software, including its operation as an NT service, its modes of operation, and its role in MBUS redundancy.
- **Section 9.3 Configuration** – Discusses the requirements for Ethernet-based network connections that use the NIM32 software, such as the APACS+ control panel applet settings, requirement for host files, APACS.INI file settings, TCP/IP logical ports, and hardware/operating system requirements.
- **Section 9.4 User Interface** – Discusses the NIM32 user interface, including status window displays and the Connection List.
- **Section 9.5 Diagnostics and Error Reporting** – Discusses the NIM32 support of an interface to the NT Event Log and the APACS+ Diagnostic Logger, and the use of priority criteria to resolve address conflicts. This section also provides and defines all messages displayed in the NIM32 status window.

9.1 Introduction

9.1.1 Product Description

The APACS+ Network Interface Manager 32-bit (NIM32) software provides an Ethernet-to-MBUS interface within an APACS+ system. This software-based solution eliminates the need for hardware-based interfaces. NIM32 software provides Ethernet and MBUS access to control and I/O modules, and data contained in an APACS+ system.

9.1.1.1 Capacity

In addition to the eight possible S7 connections, each NIM32 supports a maximum of 256 simultaneous connections (not including NIM32s and NIM32 user interfaces) to APACS+ communication sessions. Any control resource (ACM, CCM, Control or Data Acquisition Engine, or Control Simulator) or standard client application (*4-mation*, API, Diagnostic Logger, APACS+ I/O Server, SOE Viewer, or the Time Sync Utility) counts as a connection. If the maximum limit of 256 connections occurs, remote connections that have been inactive for at least 20 seconds are de-activated to allow new connections to be made. Each NIM32 supports connections to 100 NIM32s, including itself. Each NIM32 supports a maximum of 100 NIM32 User Interface connections.

9.1.1.2 Compatibility

The NIM32 software supports communications to the following:

- APACS+ Ethernet client applications: *4-mation* (Version 4.40), API (Version 4.37), I/O Server (Version 4.40), and Diagnostic Logger (Version 4.41) or later versions
- ACM System Software version 3.xx and later
- CCM/ACM+ System Software version 3.xx and later
- CCMx System Software version 3.51 and later
- ACMx System Software version 4.50 and later

9.2 Operation

This section describes the functional operation of the Network Interface Manager 32-bit (NIM32) software. The NIM32 software provides MBUS connectivity to Ethernet.

Prior to the introduction of NIM32 software, there were two versions of each of the APACS+ standard client applications (API, I/O Server, Diagnostic Logger, SOE Viewer, and the Time Sync Utility) to interface with both MODULBUS (MBUS) and Ethernet. All applications can now use just Ethernet versions.

The Bridge Mode provides an Ethernet to MBUS bridging capability, allowing remote standard client applications to communicate with resources on the local MBUS through the IEM. This is the same functionality that is provided by the DOS NIM software.

The NIM32 provides a direct Ethernet connection between APACS+ client applications and the Control Engine operating on remote computer nodes. No MBUS communication software or MBUS adapter cards are required.

9.2.1 Redundancy

9.2.1.1 Controller Redundancy

The NIM32 software supports controller redundancy. Any standard APACS+ client application and the NIM32 software communicate with a redundant pair of controllers operating as a single resource. The standard client and the NIM32 will automatically continue to communicate with the new primary controller when a switchover occurs. Controller redundancy is supported for both peer-to-peer (module-to-module) and node-to-node (rack-to-rack) redundancy.

9.2.1.2 Ethernet/Server Redundancy

In this architecture, each client node has two Ethernet connections. One connection is to Server Node A of a redundant pair of Tag Servers, and the second connection is to Server Node B of the redundant pair. Each server pair additionally has a dedicated Ethernet connection for redundancy synchronization and switchover.

The NIM32 software can operate on multiple IEMs to create redundant connection paths between client nodes and MBUS networks. If multiple connection paths exist between a client node and an MBUS resource, the initial connection path is arbitrarily chosen. If this connection path fails, a timeout is detected at the client, and after a timeout period of less than 10 seconds, a new connection is established using an alternate connection path.

Once the connection is established, the client node and resource continue to use the same NIM32 for all subsequent communications. If the server fails, the client times out, and a connection is automatically established with the other server running the NIM32 software. The new connection happens automatically within 10 seconds. The user is not required to restart any applications or to manually select the new path for the new connection to be established. This backup mechanism requires the client node to identify both server nodes in either a HOSTS file or in the APACS.INI file.

9.3 Configuration

The configuration of the NIM32 software is performed by the use of the NIM32 tab of the APACS+ Control Panel applet. Refer to the *APACS+ Control Panel Software Guide* (document SG39-15). The settings of the APACS+ Control Panel are made for the IEM using the Node/Rack/Slot settings in the USB file, **gateway.ini**.

Notice

Use *NIM32 User Interface* on another computer to connect remotely to the NIM32 on the IEM.

9.3.1 Hosts File

A HOSTS file is a static table that is used on each computer node to resolve host names (or computer names) to IP addresses. The advantage of HOSTS files is that they can be used to optimize connection times. The disadvantage is that they can be more difficult to administer than NT services, such as WINS and DNS, because the HOSTS file must be manually maintained on each node. A HOSTS file is required for certain types of networked connections using the NIM32 software. The HOSTS file must be located in the appropriate directory (**\systemroot\System32\Drivers\Etc** for Windows NT and Windows 2000). The HOSTS file can be modified with any standard text editor, such as Windows Notepad.

HOSTS file entries are required for connections to client nodes that do not use name resolution (DNS or WINS). The client node HOSTS file contains the IP address and computer name of a remote computer node that is running the NIM32 software in Bridge Mode.

HOSTS file entries are not required for client nodes that are configured to use DNS or WINS and connect to a remote NIM32 that is operating in Bridge Mode.

Notice

The HOSTS file cannot be edited on the IEM. The HOSTS file should only be changed on client computers that need to use the NIM32 on the IEM, and the client computer does not use name resolution (DNS or WINS).

Although a HOSTS file is not required for client nodes that are configured to use DNS or WINS, a HOSTS file can be used to improve connection times for standard client applications, such as *4-mation* Configuration Software. Windows name resolution first searches a HOSTS file for name resolution before using DNS or WINS servers.

If a user edits the HOSTS file, the client applications need to be restarted for the change to take effect; however, the associated computer node does not need to be rebooted.

9.3.1.1 Use of NIM Names

With DOS NIM software, NIM names are aliases that are used in the HOSTS file. NIM names can be used optionally in the HOSTS file of a computer node for client application connections to remote NIM32s. The use of NIM names and the search settings in the APACS.INI file can be used to optimize connection times for client applications. For more details on the use of NIM names, refer to section 3.2 of *Communication Considerations for Versions 4.20 and Higher* (document number CG39-13).

Notice

HOSTS file cannot be edited on IEM.

9.3.2 LMHOSTS File

The LMHOSTS file is a Microsoft-modified enhanced version of the HOSTS file. This file has the same functionality that a HOSTS file does, with additional capabilities that are not needed by the NIM32 software. An LMHOSTS file can be used as a replacement to the HOSTS file. For use with the NIM32 software, the format of entries in the LMHOSTS file is the same as that for a HOSTS file.

Notice

LMHOSTS file cannot be modified on an IEM. Modifications can only be made on client computers.

9.3.3 Settings for Client Nodes

The settings in the APACS.INI file required for client nodes that connect to an instance of NIM32 software running in Bridge Mode are provided in Table 9–2.

Notice

MIN_NIM_NAME_SEARCH and MAX_NIM_NAME_SEARCH are included for backward compatibility. The NIM_LIST is the preferred implementation.

For more details regarding APACS.INI settings, refer to section 3.2 of the *Communication Considerations for Versions 4.20 and Higher Configuration Guide* (document CG39-13).

Notice

These settings are only for client computers, not the IEM.

Table 9–1 Parameter Settings in the NETWORK Section of the APACS.INI File

Parameter Name	Description
MIN_NIM_NAME_SEARCH (obsolete see Important below)	Used to limit computer name to IP address resolution for a client that is establishing a connection to a remote instance of NIM32 software. Valid range is 1 to 64 (for NIM1 to NIM64).
MAX_NIM_NAME_SEARC H (obsolete see Important below)	Used to limit computer name to IP address resolution for a client that is establishing a connection to a remote instance of NIM32 software. Valid range is 1 to 64 (for NIM1 to NIM64).
NIM_LIST (prefered for NIM32)	Used to establish a connection list of specific instances of NIM32 software. If name resolution is used, this parameter specifies the specific computer names to which the client node connects. (The list is limited to 160 characters.) Using this parameter along with WINS or DNS eliminates the need for a HOSTS file at the client node for connection to a remote instance of NIM32 software. This is the preferred use. If this parameter is used, MIN_NIM_NAME_SEARCH and MAX_NIM_NAME_SEARCH will not be used.

Notice

MIN_NIM_NAME_SEARCH and MAX_NIM_NAME_SEARCH both require that all PC Names be in the format of NIMxx where xx is a number between 1 and 64 (e.g., NIM1, NIM 12, etc.). This scheme only used with older – pre-NIM32 – systems and has been replaced with the NIM_LIST command which places no restrictions on PC names.

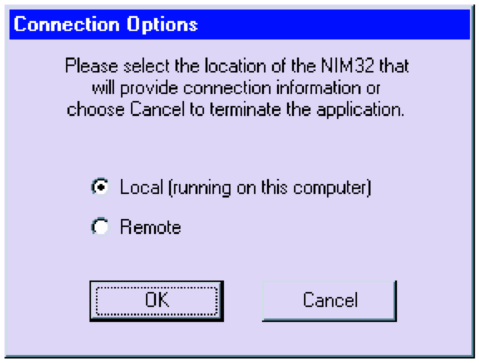
9.4 User Interface

This section describes the NIM32 software's user interface. While the NIM32 is an NT service that may run independently of a user interface, this user interface is available as a separate application to allow configuration of the NIM32 and real-time visual status of NIM32 operations and diagnostic information.

The user interface will be installed on a separate computer that may also have a NIM32 installed on it.

9.4.1 User Interface Startup

When the NIM32 User Interface is started, a dialog box is displayed. The dialog box provides a choice to connect to either a local or remote NIM32. This NIM32 is identified as the “Primary NIM32.” The dialog box is shown in Figure 9-1.


Connection Options

Please select the location of the NIM32 that will provide connection information or choose Cancel to terminate the application.

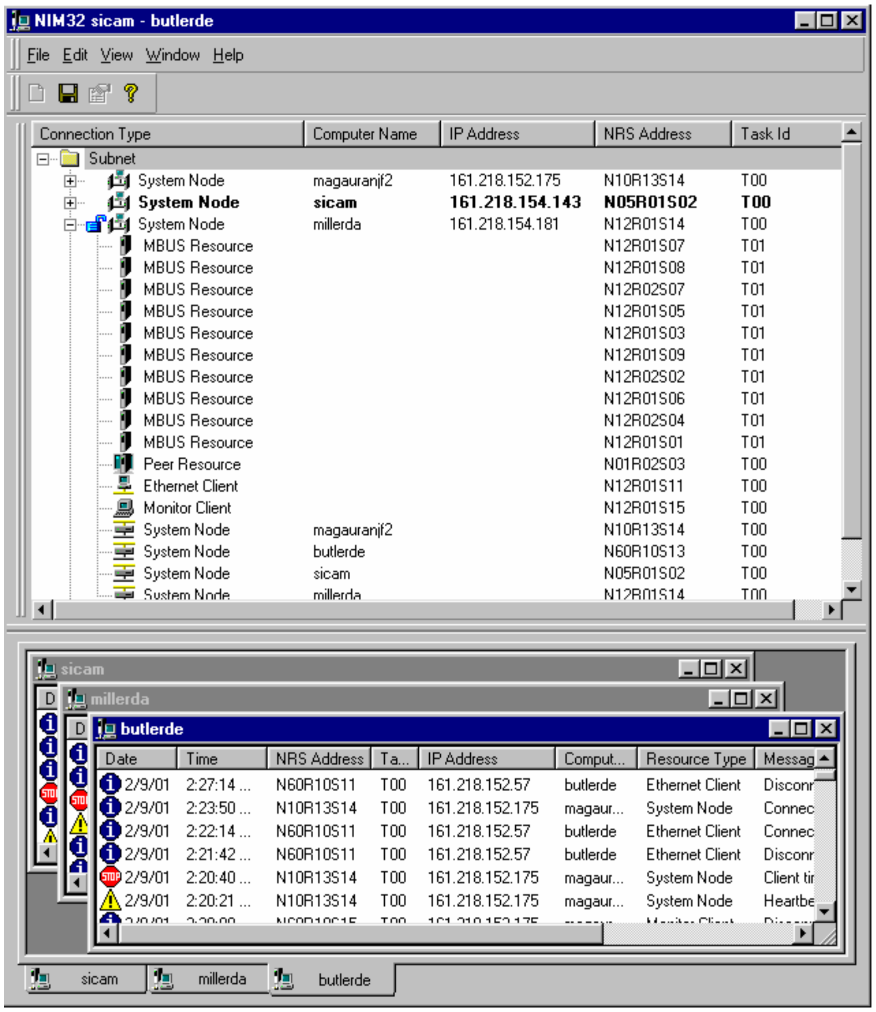
[✓ Local (running on this computer)]
Remote

Figure 9–1 NIM32 User Interface Start-up (Connection Options) Dialog Box

If **Remote is** selected, it is possible to enter the name of a remote PC or browse to select the desired PC. Enter a computer name that is running a NIM32, or select **Cancel** to shut down the user interface. If the user interface is started from a shortcut that specifies a particular NIM32 to connect to, the dialog box functionality described previously is bypassed (for example, by adding the NIM32 name to the command line in the shortcut in the Start menu).

9.4.2 User Display

This section describes general aspects of the User Display window (as shown in Figure 9-2). The primary NIM32 is shown in boldface in the Connection List.



Connection Type	Computer Name	IP Address	NRS Address	Task Id
Subnet				
System Node	magauranjf2	161.218.152.175	N10R13S14	T00
System Node	**sicam**	**161.218.154.143**	**N05R01S02**	**T00**
System Node	millerda	161.218.154.181	N12R01S14	T00
MBUS Resource			N12R01S07	T01
MBUS Resource			N12R01S08	T01
MBUS Resource			N12R02S07	T01
MBUS Resource			N12R01S05	T01
MBUS Resource			N12R01S03	T01
MBUS Resource			N12R01S09	T01
MBUS Resource			N12R02S02	T01
MBUS Resource			N12R01S06	T01
MBUS Resource			N12R02S04	T01
MBUS Resource			N12R01S01	T01
Peer Resource			N01R02S03	T00
Ethernet Client			N12R01S11	T00
Monitor Client			N12R01S15	T00
System Node	magauranjf2		N10R13S14	T00
System Node	butlerde		N60R10S13	T00
System Node	sicam		N05R01S02	T00
System Node	millerda		N12R01S14	T00

Date	Time	NRS Address	Ta...	IP Address	Comput...	Resource Type	Messag
2/9/01	2:27:14 ...	N60R10S11	T00	161.218.152.57	butlerde	Ethernet Client	Disconr
2/9/01	2:23:50 ...	N10R13S14	T00	161.218.152.175	magaur...	System Node	Connec
2/9/01	2:22:14 ...	N60R10S11	T00	161.218.152.57	butlerde	Ethernet Client	Connec
2/9/01	2:21:42 ...	N60R10S11	T00	161.218.152.57	butlerde	Ethernet Client	Disconr
2/9/01	2:20:40 ...	N10R13S14	T00	161.218.152.175	magaur...	System Node	Client tir
2/9/01	2:20:21 ...	N10R13S14	T00	161.218.152.175	magaur...	System Node	Heartbe
2/9/01	[illegible]	N60R10S15	T00	161.218.152.175	[illegible]	Monitor Client	[illegible]

Figure 9–2 NIM32 User Display Window

The NIM32 User Display consists of menu bars, toolbars, a Connection List window, and one or more Status History windows. The Connection List occupies the upper portion of the window, and the Status History windows are located beneath it.

The NIM32 User Display supports resize, minimize, maximize, move, and scroll bar functions. The relative sizes of the Connection List window and the Status History window are adjustable within the confines of the User Display window. Tiling of windows is available within the Status History window area.

The background color of all windows and the color of all text are system colors. Colors are shown on icons (as noted in subsequent sections).

The application title bar is entitled **NIM32**, and includes the name of the NIM32 to which it is connected. When the User Interface is opened to the local NIM32 using the **Local** option of the initial dialog box, then the word **Local:** appears before the computer name. If the local NIM32 is selected using the computer name or browse options, the computer name is shown without **Local:**.

The User Interface shuts down with the workspace (state) saved and re-opens to the same size and position; it does not start automatically.

9.4.2.1 User Display Menu Choices

The following User Display menus are provided:

File

Edit

View

Window

Help

The File menu includes the following choices:

New - Opens a new status window for the NIM32 that is currently selected in the Connection List.

Close - Closes the status window on top.

Close All - Closes all the status windows.

Save - Saves the contents of the current Status History window.

Log – Enables or disables logging of the active status window.

Exit - Exits the User Interface application. This does not stop the NIM32 service.

The **Edit** menu includes the following choice:

Clear - Clears a status window. This choice prompts the user with an “Are you sure?” warning dialog box to avoid the loss of all status history.

The **View** menu includes the following choices:

Columns – Selects which columns are visible in the Connection List.

Default Columns – Returns the Connection List column settings to their default state.

Toolbar – Displays the toolbar.

Properties - Shows the property page for the NIM32 highlighted in the Connection List.

The **Window** menu includes the following choices:

Cascade

Tile Horizontal

Tile Vertical

The **Help** menu includes the following choices:

Contents - Displays the NIM32 help file.

About – Opens the About dialog box. The About dialog box includes versions of each of the two User Interface components. In addition, this dialog box displays the versions of the NIM32 UI Container and the NIM32 UI Feature Set components.

The **System** menu options are standard (**Move**, **Size**, **Maximize**, **Minimize**, **Close**).

9.4.2.2 User Display Toolbar

The following menu items have tool bar buttons:

File: New

File: Save

View: Properties

Help: Contents

When the mouse pointer pauses over a tool bar button, its name appears in a tool tip (balloon).

9.4.3 Connection List

The upper portion of the User Interface window contains the Connection List window, which lists the active connections to the NIM32. The Connection List includes standard clients, standard resources, and remote NIM32 computer nodes. The Connection List does not include DOS NIMs.

9.4.3.1 Window Layout and Content

Tree Hierarchy

The Connection List opens in a Windows Explorer style format, with all of its contents shown in a hierarchical tree structure. The hierarchy is as follows:

Top Level:	Subnet (local or remote)
Second Level:	NIM32 connections
Third Level:	Local connections to the NIM32 (resources and clients)

The Top Level category in the hierarchy of the Connection List is the subnet, which includes all NIM32 connections on that subnet. The User Interface program starts up with a single subnet, which is called “Subnet” by default, but is user-editable:

- The edited name doesn’t persist if the NIM32UI is restarted.
- It is possible to expand or collapse any portion of the hierarchy by clicking on the +/- box to the left of each object.

Each NIM32 connection opens as a subcategory directly beneath a subnet folder. The primary NIM32 in the system is shown in bold. Directly beneath each NIM32 connection are subcategories that include all local connections to the NIM32.

Notice

The NIM32 is aware of other NIM32s, and each NIM32 shows the others of which it is aware in its Connection List. If two NIM32s do not list each other, it may be because of an NRS address conflict between the two.

When the status of a connection to a NIM32 is not good, an overlay (displayed as a question mark) appears over that connection in the Connection List.

Table 9-2 NIM32 Connection Types

Connection Type	Examples
MBUS resource	ACM, CCM
Peer resource	Control Engine
Ethernet peer resource	Control Simulator
MBUS client	MBUS version of *4-mation*
Ethernet client	Ethernet version of *4-mation*
Monitor client	NIM32 User Interface
System node	NIM32 service (Local and Bridge Modes)

Columns of Data

Each entry in the Connection List includes status information in a column format. The columns in Table 9-3 are available by default in the Connection List window.

Table 9–3 Default Columns In Connection List

Column Name	Description
Connection Type	Name or type of NIM32, resource, or client connection
Computer Name	Computer name for NIM32, client, or resource node
IP Address	IP Address of NIM32, client, or resource node
Node/Rack/Slot	MBUS node, rack, and slot address (see Notice below)
Task ID	Task identification number. This is the unique identifier for a specific MBUS application, providing, in addition to Node/Rack/Slot, addressing information such that the full address appears as NxxRyySzzTaa. For Ethernet applications, the Task ID is zero.

Notice

The NIM32 uses the Node/Rack/Slot address of other NIM32s and NIM32 client interfaces to track all connections to the NIM32. Since Ethernet clients do not have a Node/Rack/Slot assignment, the NIM32 to which the client connects assigns an arbitrary, but unique, Node/Rack/Slot address based on the NIM32’s own Node/Rack/Slot address. This Node/Rack/Slot address does not correspond to a physical address, but is used by the NIM32 only to distinguish multiple Ethernet clients. This address is displayed in the Connection list and is reported in the Status History window.

The Connection List window supports display or hiding any of the columns using a menu selection or dialog box; however, the first column, Connection Type, is always retained. It is possible to re-order or re-size columns by dragging the column headings. The current column format will be saved and used the next time the Connection List window is opened.

Icons

Each connection type (e.g. NIM32, MBUS resource, peer resource, Ethernet client, etc) is distinguished by its own unique icon. These icons are displayed to the left of each object in the Connection List. A different icon is available for each mode of the NIM32 software (Local and Bridge) to indicate the operating mode of each instance of NIM32 software. An additional security icon is displayed to the left of a NIM32 node, indicating the local client’s access privileges to the NIM32. The icons used to identify connection types in the Connection List are shown in Table 9-4.

Table 9–4 Connection Type Icons

Description	Icon
Subnet (closed)	📁
Subnet (open)	📂
Connection Type	**Icon**
MBUS resource	▮
Peer resource	▮▮
Ethernet peer resource	▮▮
MBUS client	🖥
Ethernet client	🖥
Monitor client	💻
System node (Local mode)	▮🖥
System node (Bridge Mode)	▮🖥▮
System node (mode not identified)	▬
Security	**Icon**
Client connection security enabled, access denied	🔒
Client connection security enabled, access allowed	🔓
Client connection security disabled	(None)
Status	**Icon**
Status is OK	(None)
Status is not OK	❓

9.4.3.2 Functionality of the Connection List

The Connection List of the NIM32 User Display contains very useful diagnostic information. A detailed discussion of the functionality of the Connection List is provided in the following paragraphs.

The Connection List displays all active connections to each instance of NIM32 software. Double-clicking on a NIM32 object expands its tree, thereby showing all active connections for that instance of NIM32 software and opening the Status History window for that instance of NIM32 software. The types of connections that are listed differ based on the operational mode of the NIM32 software.

The first subcategory beneath the Local Subnet includes the NIM32 connections. Each of these top-level NIM32 connections displays all possible connection types beneath it (regardless of mode).

These connection types include:

- MBUS resource (ACM, CCM)
- Peer resource (Control Engine)
- Ethernet peer resource (Control Simulator)
- MBUS client (such as MBUS version of *4-mation*)
- Ethernet client (such as Ethernet version of *4-mation*)
- Monitor client (NIM32 User Interface)
- Remote NIM32 node

Notice

It is possible to expand the tree hierarchies of multiple top-level instances of NIM32 software at the same time. Therefore, a particular node type (e.g. Resource or NIM32) may simultaneously appear in the trees of more than one instance of NIM32 software. This occurs because each instance of NIM32 software senses its own connection to the same device. In this way, the Connection List of each instance of NIM32 software is accurately shown.

9.4.3.3 Access to Other Displays

It is possible to access other displays from the Connection List window that relate to the currently selected instance of NIM32 software. Right-clicking on an instance of NIM32 software that appears in the Connection List provides two menu choices (see Table 9-5).

Table 9-5 NIM32 Connection List Menu Items

Menu Item	Description
New Status History Window	Opens a new Status History window for the selected instance of NIM32 software in the Status History window area of the User Display.
Properties	Displays the NIM32 Properties tab of the APACS+ Control Panel applet, allowing configuration of the selected instance of NIM32 software.

9.4.3.4 Security

The Connection List shows all NIM32 nodes and their active connections, regardless of how the Client Access table is configured in the NIM32 software. However, each instance of NIM32 software with Client Connection Security enabled may prevent a client from communicating with the software if the NIM32 software's Client Access table does not contain the client's computer name or IP address.

A padlock-shaped icon is used to indicate the security access privileges of a local client for the given instance of NIM32 software, as follows:

Closed padlock: Indicates security is enabled on the NIM32 software. Standard client applications on the node running the User Interface do not have access to the NIM32 because this node does not have access privileges to communicate with the NIM32 software.

Open padlock: Indicates security is enabled on the NIM32 software. Standard client applications on the node running the User Interface have access to the NIM32 because this node has access privileges to communicate with the NIM32 software.

No padlock: Standard client applications have access to the NIM32 because Client Connection Security is disabled on the instance of NIM32 software.

9.4.4 Status History Window

The lower portion of the User Interface window contains the Status History window. The Status History window area displays events and various status and error messages that can occur during operation of the NIM32 software. Each message includes the date and time of occurrence.

At startup of the User Interface, a single Status History window area is displayed to reflect the list of events for the primary NIM32. A tab beneath the Status History window area indicates the computer name of the node running the associated NIM32 software.

It is possible to open additional Status History windows for other NIM32 nodes by right-clicking on the desired instance of NIM32 software in the Connection List and making a menu selection to open another Status History window (or by using the menu selection **File > New**). Multiple Status History windows may be open at the same time and are overlaid by default. A Status History window area may be brought to the front by selecting its tab beneath the window. The Status History windows may be cascaded or tiled in the lower viewing area using standard Windows functions.

Each entry in the Status History window area contains the following fields. Column headings identify each field.

- Date and time stamp that the event was posted
- Node/Rack/Slot address of the client or resource causing the message to be posted
- Task ID of the application that caused the message to be posted
- IP address of the client or resource that caused the message to be posted
- Computer name of the client or resource that caused the message to be posted
- Resource type
- Text of the posted message

Icons are displayed next to each message, with the following coloring scheme:

- **Blue** – Informational messages that require no user action. These messages typically involve connection status
- **Yellow** – Warning messages
- **Red** – Error messages that alert to either a severe system problem or a client (or resource) problem.

The NIM32 Status History window displays up to 1,000 entries.

Notice

When a NIM32 connects to a NIM32 User Interface, it sends the most recent 100 status events. New entries are posted to the top of the list. When the Status History window is full, the oldest entries are deleted as new entries are placed in the window. New entries in the Status History window do not automatically scroll the window to the most recent entry in the window. However, when the open Status History Window is displaying the top of the list, it stays at the top as new entries are posted to the list.

A menu item allows a user to open a dialog box for saving the contents of the Status History window as a text (**.csv**) file. When multiple Status History windows are open, the dialog box saves the contents of the active Status History window. The dialog box supports standard browse capability to select the file.

A menu item enables or disables the logging of events from the active status window to a file. Enabling this option brings up a dialog box for specifying the file name and location. The log file is a **.csv** file with the same information as the displayed status window.

9.4.5 Configuration

The NIM32 configuration display is accessed by selecting the APACS+ Control Panel application or by right-clicking on a NIM32 object in the Connection List window and selecting **Properties**, which opens the NIM32 tab of the APACS+ Control Panel. When selected from the Connection List, the configuration parameters shown in the Control Panel apply only to the currently selected instance of NIM32 software. The other Control Panel tabs are not displayed.

Clicking the **Apply** or **OK** buttons after making configuration changes results in a dialog box that provides an opportunity to accept or cancel the changes. When the configuration changes are accepted, any changes to configuration parameters are immediately sent to the NIM32. The changes take effect immediately, without manually restarting the NIM32 or rebooting the node.

For complete details, please refer to the *APACS+ Control Panel Version 4.40 or Higher Software Guide* (document SG39-15).

9.5 Diagnostics and Error Reporting

9.5.1 Startup Errors

Any errors that prevent the NIM32 from starting normally are reported to the NT Event Viewer. For example, a TCP/IP port conflict prevents the NIM32 from starting.

After startup, all errors are reported to the User Interface's Status History window, and most are reported to the Diagnostic Logger application. The errors that can be reported are identified in Table 9–6 through 9-9.

9.5.2 Name and Address Conflict Resolution

All computers on a network must have unique names. When configuring the network setup, assigning two PCs the same name causes the NIM32 to post an error message, **Computer name resolver has detected a computer name conflict**, once per second. If this occurs, stop the NIM32, change one of the computer names to remove the conflict, and then restart the NIM32.

The NIM32 requires that all computer nodes and resources on a network each have a unique Node, Rack, and Slot (NRS) address. If a conflict exists, the NIM32 determines which entity has rights to use the address by using the priority criteria shown in the paragraph below. Entities that are not granted access to the address are blocked from communicating with the NIM32, and an event is posted to the status window and the diagnostic logger to indicate the conflict.

Each instance of NIM32 software running on a computer node controls its own message routing and resolves any address conflicts. A NIM32 prioritizes NRS addressing as follows (from highest to lowest):

1. Nodes (MBUS resources) that are connected to a local MBUS
2. Control Simulator and Control Engine running on the local computer node (including the Open Application Server Control Engine)
3. Remote connections to NIM32s that are running on separate computer nodes
4. Multiple NIM32s on one local MBUS (Since these NIM32s do not actually connect to each other, this is not a problem. The NIM32 user interface in one node does not display the other NIM32 and its resources in the event of a conflict.)
5. Connections to local client applications (such as *4-mation* and the Diagnostic Logger)

9.5.3 Status History Window Messages

The NIM32 software's Status History window is capable of displaying any of the messages shown in Table 9–7 through 9-10. Additionally, initialization, informational, warning, and error messages from Tables 9-7 through 9-10 with a System Service Code (SSC) are logged by the Diagnostic Logger Utility if it is operating on the system.

When an uninitialized Advanced Control Module (ACM) is present in an APACS+ system, the NIM32 may post a connection-established message for the ACM. If the ACM remains uninitialized, the NIM32 subsequently posts a **Heartbeat Missed** message, followed by an **Abnormal Disconnect** message. These messages should be ignored. To avoid this situation, initialize the ACM.

9.5.3.1 Initialization Messages

Messages that can be displayed when the NIM32 software initializes are provided in Table 9–6.

9.5.3.2 Runtime Informational Messages

Informational messages that can be displayed by the NIM32 software at runtime are provided in Table 9–7. These messages can appear during normal operation and are not an indication of problem conditions. No user actions are required.

9.5.3.3 Warning Messages

Warning messages that can be displayed during an instance of NIM32 software operation are provided in Table 9–8. These messages are an indication of a potential problem. NIM32 software operation continues. Warning messages are additionally posted to the Diagnostic Logger Utility.

9.5.3.4 Error Messages

Error messages that can be displayed during NIM32 software operation are provided in Table 9-9. These messages are an indication of an error condition that can prevent operation of the NIM32 software. Error messages are additionally posted to the Diagnostic Logger Utility.

Table 9–6 NIM32 Initialization Messages

Message	Description	User Action
Local MBUS communications disabled	MBUS communications could not be started, and the NIM32 cannot communicate with local MBUS resources. This is a normal condition if an MBUS adapter card is not installed in the computer node. If an MBUS adapter card is installed, the condition is possibly due to improper installation of MBUS driver software or malfunctioning MBUS adapter card.	If an MBUS adapter is installed, re-install MBUS driver software and verify proper operation of MBUS.
Local MBUS communications enabled	Initialization message confirming proper operation of MBUS communications	No action required
Client Access Table found at <pathname>	Lists the location of the Client Access Table	No action required
Remote Configuration Security Table found at <pathname>	Lists the location of the Remote Configuration Security Table	No action required
Client Connection Security is enabled.	Client Connection Security has been configured, and a Client Access Table has been found.	No action required
Client Connection Security is disabled.	Client Connection Security is disabled, or a Client Access Table has not been found.	No action required
Remote Configuration Security is enabled.	Remote Configuration Security has been configured, and a Remote Configuration Security Table has been found.	No action required
Remote Configuration Security is disabled.	Remote Configuration Security is disabled, or a Remote Configuration Security Table has not been found.	No action required

Table 9–7 NIM32 Software's Runtime Informational Messages

SSC	Message	Description	Detected	User Action	Icon Color
	Client Connection Aborted	A client application abnormally terminated, and the connection to the NIM32 has been broken.	A client application abnormally terminated, and the connection to the NIM32 has been broken.	No action required	Blue
	Unexpected TCP Receive Failed	An error has occurred in receiving an Ethernet TCP message.	When receiving Ethernet TCP messages	No action required	Red
	An unexpected client not connected	A client is trying to use a closed connection.	When a message is received, a check is performed to validate the connection.	No action required	Red
	Record discarded	The NIM32 is having problems sending messages to a NIM32 user interface.	When the number of retries to send the message has been exceeded	No action required	Red
	Driver Not Loaded	The MBUS drive is not loaded. This is a normal condition if an MBUS adapter card is not installed in the computer node. If an MBUS adapter card is installed, the condition is possibly due to improper installation of MBUS driver software or malfunctioning MBUS adapter card.	At startup	If an MBUS adapter is installed, re-install MBUS driver software and verify proper operation of MBUS.	Red
	Configuration record file not found – attempt to create file	Configuration file not found. Creating a new one. Configuration changes will not be saved.	At startup	Check for full or write-protected disk.	Red

SSC	Message	Description	Detected	User Action	Icon Color
	Can’t create a configuration file	The creation of the configuration file failed. Configuration changes will not be saved.	At startup	Check for full or write-protected disk.	Red
	Failure to create configuration record file	The creation of the configuration file failed. Configuration changes will not be saved.	At startup	Check for full or write-protected disk.	Red
1	Multiple connection paths detected; communication now occurring on MBUS	An MBUS connection path has now been identified between the NIM32 and the resource. This connection path is now used , and the Ethernet connection path serves as a backup path.	This is detected at startup and when an additional path is connected during runtime.	No action required	Blue
11	Start up completed. This node is online.	NIM32 software initialization has been successfully completed.	Upon completion of initialization procedures	No action required	Blue
12	Shutdown completed. This node is offline.	NIM32 software termination has been successfully completed.	Upon completion of shutdown procedures	No action required	Blue
13	Connection established	A user has just established a connection to the NIM32 software.	When a connection attempt is successful	No action required	Blue
14	Heartbeat received	After posting a "Heartbeat missed" message, this message is posted when the NIM32 software has resumed receiving the periodic heartbeat messages.	After posting a "Heartbeat missed" message, this message is posted when the NIM32 software has resumed receiving the periodic heartbeat messages.	No action required	Blue
19	Client connection closed	A user application has closed its connection to the NIM32.	When a user application has closed its connection to the NIM32	No action required	Blue

SSC	Message	Description	Detected	User Action	Icon Color
24	Remote Configuration Access Table could not be found at the configured location. Remote Security has been disabled.	The Remote Configuration Access Table could not be found at the configured location. Remote Security has been disabled.	At startup, the `REMOTECONFIGURATION SECURITYTABLE.TXT` file was not found.	No action required	Blue
30	NIM32 configuration changed	A NIM32 configuration change has occurred.	When the user makes changes to the configuration	No action required	Blue

Table 9–8 NIM32 Software's Warning Messages

SSC	Message	Description	Detected	User Action	Icon Color
2	Heartbeat missed	The periodic heartbeat message was not received by the NIM32 software.	During runtime when a client or resource fails to send a heartbeat (“Here I am message”) in the expected timeframe	Check MBUS communicatio n path	Yellow
3	Heartbeat missed; NIM32 has switched to alternate communication path.	A heartbeat message was dropped. Since an alternate communications path exists, the NIM32 software has switched to the use of the alternate communication path.	When a primary communications path has faltered, and the secondary path was enabled	Check MBUS communicatio n path	Yellow
4	Client connection aborted	A client application was abnormally terminated, and the connection to the NIM32 software has been broken.	When a client application is abnormally terminated, and the connection to the NIM32 software has been broken	No action required	Yellow
5	Source of message not recognized	A message was received from a source that isn't recognized. This is probably due to improper registration of the client with the NIM32 software. The message was discarded.	When a client's or resource's name is not properly registered with the NIM32 software, and that client or resource has just attempted to pass a message through the NIM32 software.	Client Connection Security enabled, but source not listed in the Client Access Table. Possible software failure. Monitor and contact supplier.	Yellow
6	Permission to connect is denied	The Client Access Table is enabled, and a client that is not identified in the table attempts to connect to the NIM32 software.	As soon as an invalid client attempts to connect to the NIM32 software.	No action required	Yellow

SSC	Message	Description	Detected	User Action	Icon Color
7	This node is not licensed for remote connections.	The NIM32 software is licensed for Local Mode, and a remote computer node attempts to connect. The connection is not granted.	When a remote computer node attempts to connect, and the connection is not granted.	No action required	Yellow
8	Client Access Table not found; security disabled.	The Client Access Table cannot be found at the configured location. Security has been disabled, and all connections to the NIM32 software are allowed.	At startup the `SECURITY_TABLE.TXT` file is not found. Its location is defined by `NIM32_SECURITY_ TABLE_PATH`.	No action required	Yellow
9	Not all UDP messages processed during this scan	The NIM32 software can- not complete processing all UDP messages in its queue. Communications may be degraded. No messages are deleted from the queue.	When the number of messages processed in any one port per scan reaches a predetermined limit, the NIM32 software informs users that significant loading is occurring.	Check the MBUS/ Ethernet communicatio ns path. Reduce the number of applications running on the local PC.	Yellow
10	Licensing Violation	Software has detected that the `APACS.LIC` file has expired, or the hardkey was physically removed from the PC.	These tests are run periodically in the background.	Install valid license file or replace hard key.	Yellow

Table 9–9 NIM32 Software's Error Messages

SSC	Message	Description	Detected	User Action	Icon Color
	Local MBUS Communica -tions Startup failed	Startup of the MBUS interface has failed.	At startup	If an MBUS adapter is installed, re-install MBUS driver software and verify proper operation of MBUS.	Red
	MBUS Driver not functional	The MBUS driver is not functioning correctly.	At startup	If an MBUS adapter is installed, re-install MBUS driver software and verify proper operation of MBUS.	Red
20	Maximum number of connections exceeded	The connection cannot be made. The maximum number of connections is currently in use.	When a client tries to establish a connection. The current number of connections is checked. When that limit is exceeded, no more connections can be established.	Close any unneeded connections.	Red
21	Client timeout; connection aborted	A communication timeout occurs between the NIM32 software and a client application or a resource. Possible causes include abnormal termination of client application or communications problems.	After an extended period of time, no messages are received from a client or resource.	Check the MBUS/ Ethernet communications path. Reduce the number of applications running on the local PC.	Red

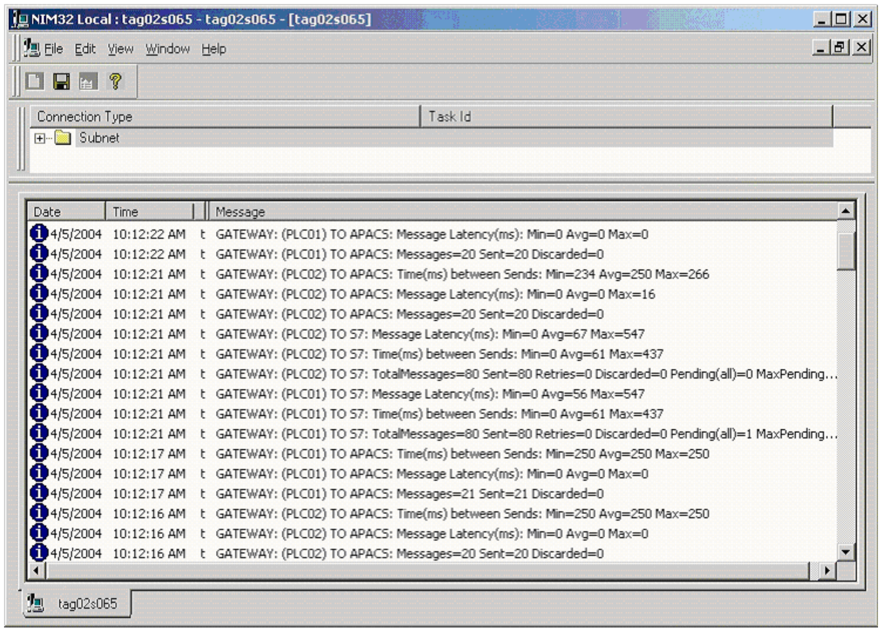
SSC	Message	Description	Detected	User Action	Icon Color
22	Computer name not resolved	Security is enabled, and the computer name cannot be resolved to an IP address using a HOSTS file, WINS or DNS.	At startup and at client/resource connection time. Can also occur after a client name is added to the Client Access Table.	Verify that computer node is configured for WINS or DNS. Consult network administrator. As an alternative, use static IP addresses in a HOSTS file and Client Access Table.	Red
23	Local computer name not found	The NIM32 software cannot read the local computer name from the operating system.	At startup	If an Ethernet network card is not installed, verify that TCP/IP loopback software is properly configured and installed. Otherwise, possible software failure. Monitor and contact supplier.	Red
25	NRS address conflict; lower priority connection disconnecte d	A higher priority connection is detected using an NRS address that is already in use. The lower priority connection is disconnected.	When a new client or resource comes online	Change the node, rack, and slot address of the lower priority conflicting connection.	
26	NRS address conflict; connection blocked.	A client attempted to connect using a node, rack, slot address that is already in use.	When a new client or resource, at a lower priority, attempts to come online	Change the node, rack, and slot address of the conflicting client.	Red

SSC	Message	Description	Detected	User Action	Icon Color
27	Internal error; TCP/IP communica-tion error.	A TCP/IP communication error occurs. Normally, when this error occurs, it is an indication that the PC that is running the NIM32 software is overloaded (typically with multiple high-loading applications) and may experience difficulty in processing the full queue of messages.	During startup, shutdown, and normal transmission of TCP/IP messages, the NIM32 software monitors interaction with the communications stack.	Possible software failure. Monitor and contact supplier	Red
28	TCP/IP port is already in use. Terminating	During startup, the NIM32 attempts to use a TCP/IP logical port that is in use by another application. The NIM32 software terminates.	At startup	Modify port assignments in the Services file to eliminate the conflict and restart the NIM32 software.	Red
29	Internal error; MBUS com-munication runtime error.	An MBUS communication error occurs.	At startup or when sending or receiving MBUS messages.	Possible software failure. Monitor and contact supplier	Red
32	Internal error; attempts to disconnect from invalid path.	The NIM32 software attempts to disconnect from a nonexistent client.	When an unregistered client attempts to disconnect	Possible software failure. Monitor and contact supplier	Red
33	Internal error; software has entered an invalid internal state.	The NIM32 software has internally entered an undefined state.	When the NIM32 software detects invalid internal logical state	Possible software failure. Monitor and contact supplier	Red

SSC	Message	Description	Detected	User Action	Icon Color
34	Internal error; internal state changed unexpectedly.	An internal error has occurred during software execution.	When the NIM32 software detects invalid internal logical state	Possible software failure. Monitor and contact supplier	Red
35	Internal error; NULL pointer encountered.	An internal error has occurred during software execution.	When the NIM32 software detects an internal pointer that is not properly set up for use.	Possible software failure. Monitor and contact supplier.	Red
36	Internal error - Out of Memory.	The NIM32 software cannot allocate additional memory for internal use.	During runtime, when client connections are being requested, memory is allocated for each connection. If the allocation fails, this message is displayed.	Possible software failure. Monitor and contact supplier	Red

10 Industrial Ethernet Module Diagnostics

This section describes the messages logged from the Industrial Ethernet Module (IEM) to the ProcessSuite NIM32. The IEM logs the diagnostics messages based on its configuration setting for DiagnosticLevel. When diagnostics are enabled, the IEM also computes the statistics for its APACS+ Diagnostic tags. The diagnostic messages are logged every 60 seconds when the setting for DiagnosticLevel has a value of either 3 or 5. The IEM messages logged to the NIM32 are easily recognized since each message begins with the text "GATEWAY."


NIM32 Local : tag02s065 - tag02s065 - [tag02s065]
File Edit View Window Help
Connection Type
Task Id
Subnet
tag02s065

```
Date       Time            Message
4/5/2004   10:12:22 AM  t  GATEWAY: (PLC01) TO APACS: Message Latency(ms): Min=0 Avg=0 Max=0
4/5/2004   10:12:22 AM  t  GATEWAY: (PLC01) TO APACS: Messages=20 Sent=20 Discarded=0
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC02) TO APACS: Time(ms) between Sends: Min=234 Avg=250 Max=266
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC02) TO APACS: Message Latency(ms): Min=0 Avg=0 Max=16
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC02) TO APACS: Messages=20 Sent=20 Discarded=0
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC02) TO S7: Message Latency(ms): Min=0 Avg=67 Max=547
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC02) TO S7: Time(ms) between Sends: Min=0 Avg=61 Max=437
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC02) TO S7: TotalMessages=80 Sent=80 Retries=0 Discarded=0 Pending(all)=0 MaxPending...
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC01) TO S7: Message Latency(ms): Min=0 Avg=56 Max=547
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC01) TO S7: Time(ms) between Sends: Min=0 Avg=61 Max=437
4/5/2004   10:12:21 AM  t  GATEWAY: (PLC01) TO S7: TotalMessages=80 Sent=80 Retries=0 Discarded=0 Pending(all)=1 MaxPending...
4/5/2004   10:12:17 AM  t  GATEWAY: (PLC01) TO APACS: Time(ms) between Sends: Min=250 Avg=250 Max=250
4/5/2004   10:12:17 AM  t  GATEWAY: (PLC01) TO APACS: Message Latency(ms): Min=0 Avg=0 Max=0
4/5/2004   10:12:17 AM  t  GATEWAY: (PLC01) TO APACS: Messages=21 Sent=21 Discarded=0
4/5/2004   10:12:16 AM  t  GATEWAY: (PLC02) TO APACS: Time(ms) between Sends: Min=250 Avg=250 Max=250
4/5/2004   10:12:16 AM  t  GATEWAY: (PLC02) TO APACS: Message Latency(ms): Min=0 Avg=0 Max=0
4/5/2004   10:12:16 AM  t  GATEWAY: (PLC02) TO APACS: Messages=20 Sent=20 Discarded=0
```

Figure 10-1 IEM Diagnostics Logging in NIM32 User Interface

The messages logged to the NIM32 are either Statistics messages or Operational messages. The statistics messages can be used to determine if the IEM send and receive blocks (GW_SEND and GW_REC) have been optimally matched such that the rate that the messages are sent does not exceed the rate at which they are received.

The operational messages indicate the start and stop of the IEM whenever it gets a new configuration. After the new configuration is processed, the IEM restarts and logs a message indicating the number of connections in use. Other operational messages are logged in the case of certain unexpected events. These messages are logged to give detailed information in case a call for Siemens product support is necessary.

10.1 Statistics Messages

Table 10-1 Statistics Messages

Message	Description
(CONNECTION_NAME) RID=nnn not in S7, messages=nnn	For the indicated connection, the RID (Remote ID) is not used in any GW_REC block. Even though the APACS+ GW_SEND block is sending the message to the IEM, the message is not being received by the S7 program. In the S7 program, add a GW_REC block for the connection and make the assignment to the indicated RID. This **not in S7** message might reappear with a different RID. That is, it only logs the most recently seen RID that is not programmed in the S7.

Message	Description
(CONNECTION_NAME) TO S7: Messages=nnn Sent=nnn Retries=nnn Discarded=nnn Pending(all)=nnn MaxPending(RID=nnn)=nnn RIDs=nnn	This message lists the activity for the connection in the last Diagnostic Interval (for example, the last 60 seconds). The message values indicate the activity since this message last occurred. The **Messages** indicates the number of data messages from the APACS+ GW_SEND block that have been received by the IEM. The value for **Sent** indicates the number of data messages transmitted to the S7. If the S7 program was not ready to receive messages (for example, when offline, in Stop, or the GW_REC block has not been called in a fast enough cycle), then the **Retries** value indicates the number of times a message has been resent. The **Discarded** value indicates the number of messages intended for the S7 that were not sent. This value indicates messages not sent due to errors, retry attempts exhausted, or time expiration. Each message received by the IEM gets a receive timestamp. If the message is held by the IEM for more than one second, it is considered too old and discarded. This insures that the S7 is sent current messages. This implementation also insures that a backlog of messages is prevented. A backlog could occur if the APACS+ GW_SEND is sending at a faster rate than the S7 GW_REC is capable of receiving. The **Pending(all)** value indicates the number of messages from all of the APACS+ GW_REC instances that are still in the IEM awaiting transmission to the S7. The value is the total for all RID values used for the connection. Ideally, this number should be less than or equal to the number of different RID values used on the connection. If it is larger, then either the APACS+ GW_SEND call should be made less frequently or else the S7 GW_REC should be made more frequently. The **MaxPending(RID=nnn)** value indicates the RID that had the most messages pending. Ideally this value should never be more than a few messages. There typically could occur temporary conditions over the sampling interval where several messages for a given RID might be awaiting transmission. However, if this value is consistently high, then optimization of the GW_SEND and GW_REC calls should be made. The final value for **RIDs=nnn** indicates the total number of RIDs used (in an S7 GW_REC block) on the connection. It is an indication of the load over a given connection. Since each RID is separately transmitted to the S7, a higher number of RID values equates to longer time to transmit each RID message to the S7.

Message	Description
(CONNECTION_NAME) TO S7: Time(ms) between Sends: Min=nnn Avg=nnn Max=nnn	This message indicates the rate of message transmission (in milliseconds) over this S7 connection. If the connection is using only a single RID, then the **Avg** (Average) value should approximate the rate that the APACS+ GW_SEND is sending each message. When more than one RID is used on the connection, this value indicates the rate for all RIDs. The **Max** value can indicate temporary conditions where it might occasionally take longer to transmit the message (e.g. TCP/IP retry). The rate of message transmission is also influenced by the rate that the GW_REC call is made in the S7 program.
(CONNECTION_NAME) TO S7: Message Latency (ms): Min=nnn Avg=nnn Max=nnn	This message for **Message Latency** indicates how long a message is held in the IEM. Whenever the APACS+ GW_SEND and S7 GW_REC have been optimized so that the message can be received as fast as it is sent, the values for **Min** and **Avg** will typically indicate small milliseconds value. If messages have been backlogged (more than one is pending), then the **Max** times will increase. A backlog occurs when multiple messages for the same RID or else multiple messages for different RIDs are awaiting transmission. Temporary conditions (such as transmission errors, retries, or S7 not ready to receive) can cause the **Max** value to reach higher values.
(CONNECTION_NAME) TO APACS: Messages=nnn Sent=nnn Discarded=nnn	The messages coming from the S7 GW_SEND blocks are indicated. These messages are sent to APACS variables (in the APACS API). The totals for **Messages** and **Sent** will match, and the total for **Discarded** will be zero unless some problem occurs in the IEM. This log message indicates that the messages from the S7 GW_SEND have been sent to the APACS API. The GW_REC blocks programmed in the APACS+ controllers provide the outputs for indicating whether the block received the messages.
(CONNECTION_NAME) TO APACS: Message Latency (ms): Min=nnn Avg=nnn Max=nnn	Each message from the S7 GW_SEND is time-stamped when received by the IEM. When the message is given to the APACS API, it is considered sent and is time-stamped again. The time difference is the **Latency**, indicating the amount of time the message spent in the IEM.
(CONNECTION_NAME) TO APACS: Time(ms) between Sends: Min=nnn Avg=nnn Max=nnn	This message indicates the rate that messages are coming from the S7 GW_SEND blocks programmed for the S7 connection. When only a single GW_SEND is programmed in the S7, the average time (**Avg**) in milliseconds should corresepond to the rate that the GW_SEND is transmitting the message. When there are multiple GW_SENDs on the same connection, these values indicate the total throughput for all RIDs used on the connection.

10.1.1 Diagnostic and Statistics Data Tags available through any virtual APACS resource

Each virtual APACS resource will maintain data tags for statistic and diagnostic purposes. These tags will be available using the tag read capability of the APACS API.

Notice

Each tag name must be preceded by the virtual resource name (i.e. connection name). For example, to read the number of packets sent OK from APACS+ to S7 using connection PC200_ID1, the tagname would be:

PLC200_ID1. APACStoS7.PacketsSentOK

Table 10-2 Diagnostic and Statistical Data

Tag Name	Data Type	Description
S7.Connection.Online	BOOL	Indicates whether the S7 connection is alive
S7.InvalidR_ID	INT	If >0 then this RID is not used in the S7 program
APACStoS7.Messages.Processed	DWORD	Count of messages sent to S7 from APACS
APACStoS7.Messages.SentOK	DWORD	Number of messages sent to the S7 successfully
APACStoS7.Messages.RetryCount	DWORD	Number of retry attempts when sending messages
APACStoS7.Messages.Deleted	DWORD	Number of messages not sent (error or too old)
APACStoS7.Messages.Pending	DWORD	Total number of messages now waiting to be sent
APACStoS7.MaxPendingAnyRID	DWORD	Most pending messages at any time for any RID
APACStoS7.MinSendInterval	DWORD	Min Interval between message sends to the S7 from APACS
APACStoS7.MaxSendInterval	DWORD	Max Interval between message sends to the S7 from APACS
APACStoS7.AvgSendInterval	DWORD	Avg Interval between message sends to the S7 from APACS
APACStoS7.MinTimePending	DWORD	Min latency (time message was pending)
APACStoS7.AvgTimePending	DWORD	Avg latency (time message was pending)
APACStoS7.MaxTimePending	DWORD	Max latency (time message was pending)

Tag Name	Data Type	Description
Gateway.Connections.TotalConfigured	USINT	Total number of connections currently configured by the IEM.
Gateway.Connections.TotalInUse	USINT	Total number of connections in use - the number of connections that will be serviced.
Gateway.Connections.TotalOnline	USINT	Total number of connections online
Gateway.Messages.PerSecond	USINT	Number of messages processed per second since start
Gateway.Messages.PerSecond5s	USINT	Number of messages processed per second in the last 5 seconds
Gateway.Version	STRING	Gateway Version
S7toAPACS.Messages.Processed	DWORD	Count of messages sent to APACS from S7
S7toAPACS.Messages.SentOK	DWORD	Number of messages sent to APACS successfully
S7toAPACS.Messages.Deleted	DWORD	Number of messages discarded
S7toAPACS.MinSendInterval	DWORD	Min Interval between message sends to APACS from S7
S7toAPACS.MaxSendInterval	DWORD	Max Interval between message sends to APACS from S7
S7toAPACS.AvgSendInterval	DWORD	Avg Interval between message sends to APACS from S7
S7toAPACS.MinTimePending	DWORD	Min latency (time message was pending)
S7toAPACS.MaxTimePending	DWORD	Max latency (time message was pending)
S7toAPACS.AvgTimePending	DWORD	Avg latency (time message was pending)

10.2 Operational Messages

Table 10-3 Operqational Messages

Message	Description
S7 API: Gateway not configured	The IEM has not yet had any connections downloaded from NETPRO. Use SIMATIC Manager, Hardware Configuration, to assign the IEM to the S7 project, and use NETPRO to assign and download the configuration. The download to the IEM is performed with the S7ONLINE connection set to the TCP/IP interface. Use SIMATIC Manager, Options, and select **Set PG/PC Interface** to make this setting.
S7 API: Could not initialize	The IEM does not have a current NETPRO configuration. Use NETPRO to download the configuration to the IEM.
S7 API: No Connections Configured	The IEM does not have a current NETPRO configuration. Use NETPRO to download the configuration to the IEM.
S7 API: TOO MANY CONNECTIONS CONFIGURED!! Only nnn of nnn are used.	NETPRO was used to assign more connections to the IEM than the 8 permitted. The IEM performance capacity was exceeded. The message indicates the number of connections that will be serviced. Use NETPRO to remove the extra connections and download the corrected configuration to the IEM.
S7 API: S7 Communications started. Connections Configured, count=nnn	This message identifies the number of connections currently used by the IEM. It should match the configuration created in NETPRO.
S7 API: S7 Communications stopped	This is an informative message stating that the communications to the S7 were stopped. This can occur if a new NETPRO configuration has been downloaded to the IEM. When this happens, the S7 communications are stopped, the new configuration is accepted, and then S7 communications are restarted.
ACP (CONNECTION_NAME) FATAL Error...	The APACS+ Communications Process (ACP) has encountered some type of fatal error (from the APACS API). This condition prevents the IEM from running. Contact Siemens product support if a check of the complete configuration (IEM, GW_SEND and GW_REC programming, or physical cabling) does not solve the indicated problem. Further checks of the IEM hardware and operating system might be required.
ACP (CONNECTION_NAME) Error updating statistics...	The APACS tags used for the statistics could not be refreshed. An error code from the APACS API is logged. Although not fatal, this error should be investigated and Siemens product support contacted. Further checks of the IEM hardware and operating system might be required.