Software License Advisory

This document supports software that is proprietary to Honeywell GmbH, Honeywell Control Systems Ltd. and/or to third party software vendors. Before software delivery, the end user must execute a software license agreement that governs software use. Software license agreement provisions include limiting use of the software to equipment furnished, limiting copying, preserving confidentiality, and prohibiting transfer to a third party. Disclosure, use, or reproduction beyond that permitted in the license agreement is prohibited.

Trademark Information

CentraLine and 'close to you' are trademarks of Honeywell Inc.

BACnet and ASHRAE are registered trademarks of American Society of Heating, Refrigerating and Air-Conditioning Engineers. Microsoft and Windows are registered trademarks, and Windows Internet Explorer are trademarks of Microsoft Corporation. Java and other Java-based names are trademarks of Sun Microsystems Inc. and refer to Sun's family of Java-branded technologies. Mozilla and Firefox are trademarks of the Mozilla Foundation. Echelon, LON, LonMark, LonTalk, and LonWorks are registered trademarks of Echelon Corporation.

Tridium, JACE, Niagara Framework, NiagaraNX Framework, Sedona Framework and Vykon are registered trademarks, and Workbench, WorkPlaceAX, and NXSupervisor, are trademarks of Tridium Inc. All other product names and services mentioned in this publication that is known to be trademarks, registered trademarks, or service marks are the property of their respective owners.
Teaching and Learning................................................................. 94
Check Application Status .............................................................. 95
Teach To Controller ................................................................. 95
Unconditional Application Teaching To All Controllers ................. 97
Learn From Controller .............................................................. 97
Clear Controller ............................................................................ 98
Check Hardware Compatibility .................................................... 98
Take Snapshot ............................................................................. 100
Restore Snapshot ........................................................................ 101
Swap IRM Program ........................................................................ 101
Master Sync.................................................................................... 103
Cloning Application ..................................................................... 104
Applying Master Sync .................................................................. 106
Excluding Function Block Items from Master Sync......................... 108
Splitting-Off Application ............................................................ 109
Controller MAC Address Assignment ........................................... 112
Setting MAC Address of Controller .............................................. 113
Synchronizing Device Time ......................................................... 114

FIRMWARE DOWLOAD................................................................... 115

ALARMING..................................................................................... 116

PRINTING...................................................................................... 116

MISCELLANEOUS ...................................................................... 117
Value Updates after Device Power Failure .................................... 117
Hiding Slots ................................................................................ 117
Sylk Wall Module Usage .............................................................. 119
Bulk Operations ........................................................................... 121
Changing Datapoint Names ......................................................... 121
Changing Max Master of Devices ................................................. 125
Reset to Factory Delivery ........................................................... 128
ABOUT THIS USER GUIDE

This user guide describes the configuration and management of IRM (MERLIN, CPO) controllers connected to a BACnet MS/TP network via an EAGLEHAWK controller. Configuration and Management is done using the Niagara IRM Engineering Tool based on the Niagara NX framework.

NOTE: The procedures and screenshots included in this user guide apply in the same way to other brands and their products if applied.

Applicable Technical Literature

- NIAGARA IRM FUNCTION BLOCKS User Guide: EN2B-0415GE51
- NIAGARA IRM APPLICATION Guide: EN2B-0416GE51
- NX General Security Best Practices: EN0Z-1040GE51
- NIAGARA Hardening Guide: EN2Z-0985GE51
- MERLIN N4 Product Data: EN0Z-1035GE51
- MERLIN N4 Installation Instructions: EN1Z-1035GE51
- EAGLEHAWK NX Product Data: EN0Z-1039GE51
- EAGLEHAWK NX Installation & Commissioning Instructions: EN1Z-1039GE51
- EAGLEHAWK NX Onboard IO Driver User Guide: EN2Z-1044GE51
- CENTRALINE NX BACNET UTILITIES DRIVER User Guide: EN2Z-1020GE51
- ARENA NX / COACH NX Printout User Guide: EN2Z-1022GE51
- HAWK 8000 User Guide: EN2Z-1030GE51

IRM related technical literature can be downloaded at:

- IRM on CentralLine Docu Server
- Product Catalog

Common literature can be downloaded at:

- CentralLine Docuserver

SYSTEM REQUIREMENTS

<table>
<thead>
<tr>
<th>Niagara</th>
<th>Niagara 4.4.94.xx and higher</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firmware</td>
<td>IRMN_FW_Ver_1.0.2.2</td>
</tr>
<tr>
<td>Tools</td>
<td>IRMN_TOOL_Ver_1.0.1.7</td>
</tr>
<tr>
<td>Application</td>
<td>IRMN_H_0001_Ver_1.0.1.5</td>
</tr>
</tbody>
</table>

Firmware and Software Downloads

The firmware and the software can be downloaded from the CentralLine partner web at:

- www.centraline.com

Restrictions and Recommendations

For successful and seamless engineering, it is recommended to note the following internal system restrictions:

Number of BACnet Devices max. 64 devices per channel. Depending on performance needs of the application and bus traffic, it is recommended to keep the number of devices below 64.

Controller Memory Usage max. 80 % (recommended 70 %)
Function Blocks Usage
- max. 32 IRM folders overall
- max. 100 function blocks per folder
- max. 2000 function blocks overall
- max 1 wall module per device

Baudrate
9600 through max. 76800 (default = 38400)

Security Best Practices

This section provides the necessary information about the requirements for configuring and managing the security when installing and maintaining a product or system.

Honeywell hereby expressly states that its controllers are not inherently protected against cyber attacks from the Internet and that they are therefore intended solely for use in private networks. However, even private networks can still be subject to malicious cyber attacks by skilled and equipped IT individuals and thus require protection. Customers should therefore adopt the installation and security best practices guidelines for CentraLine / Honeywell BACnet MS/TP-based products to mitigate the risk posed by such attacks.

The following check list describe the General Security Best Practices for CentraLine / Honeywell BACnet MS/TP-based products. They are listed in order of increasing mitigation. The exact requirements of each site should be assessed on a case-by-case basis. The vast majority of installations implementing all of the mitigation levels described here will be far in excess of that required for satisfactory system security.

Since the attached Modbus network is implemented via SERIAL interface (RS485) no security protocol (SSL/TLS) is available. Any vulnerabilities based on the open Modbus protocol might damage the application in the IRM controller.

Incorporating the security check list items 1-5 will generally meet the requirements for most automation control network installations.

Additional information can be obtained from:
- NX General Security Best Practices (EN0Z-1040GE51)
- Niagara 4 Hardening Guide (EN2Z-0985GE51)

Security Check List

1. Use the latest version of IRM software including firmware and software modules.
2. Include the CentraLine NX installation files, configuration files (including station backup), certificates and licenses in the disaster recovery plan.
3. Make sure that the PC running CentraLine NX, where possible, is secured against unauthorized physical access.
4. Make sure that the local Ethernet network that the PC is connected to is secured, e.g. by the use of firewalls and intrusion detection systems.
5. The PC is running the latest version of the Windows operating system, with all updates and service packs.
6. The PC is running virus protection software.
7. Appropriate user accounts are set up on PC and access to files is restricted to only those who are authorized.
8. CentraLine NX is configured to use HTTPS using a certificate from a trusted Certificate Authority.
9. CentraLine NX users are configured as required.
10. CentraLine NX is configured to backup data regularly to a secure location as per your company's backup policy.
11. Ensure that complete commissioning is carried out in a closed local area network without connecting to the internet to avoid unauthorized sniffing of BACnet message packets.
12. Use encrypted communication between the Niagara Tool and the controller. See chapter “Secure Communication”.
INTRODUCTION

The Niagara IRM Engineering Tool provides the following functions:

- Setting up the IRM and EAGLEHAWK controllers for usage in a BACnet MS/TP system architecture
- Creating BACnet devices offline and online
- Creating applications for IRM controllers
- Synchronizing applications between project and controllers
- Adjusting terminal layout deviations (hardware compatibility)
- Creating Master Sync groups
- Splitting applications
- Cloning applications
- Firmware download
- Alarming
- Printout
- Secure communication between project and controllers

Basic Concepts

Engineering Modes

There are two kinds of engineering you can use for engineering an IRM project:

- Offline Engineering
  In this mode, you create an empty BACnet device manually, add an application (optional), and match it afterwards to a device discovered on the bus by using the service pin. This is normally applied when doing the engineering in the office without having the hardware available but knowing the hardware specification of the devices to be used later at the plant.

- Online Engineering
  In this mode, you discover the devices on the BACnet network in the first step and use the devices instantly for application engineering. This is recommended when doing the engineering directly at the plant with the devices already installed on the BACnet bus.

Synchronization Status of Application

The current content of the applications engineered in the project and running in the connected IRM controller is permanently monitored by the control manager (Online engineering only). Modifications can be any of the following:

- Control strategy icons (function blocks)
- Links between function blocks
- Notes
- Annotations
- Author
- Description
- Terminals
- Notification classes
- Etc.

Modifications can be detected in the controller, or in the project, or in both. They can be synchronized by applying the following actions:

- Teach to controller
- Learn from controller
- Clear project (as required)
- Clear controller (as required)

Teaching and Learning

Synchronization of the application in the project and controller application can be performed in two ways:
Teaching to Controller
Downloads the application in the project to the controller

Learning from Controller
Uploads the application from the controller in the project

Teaching Modes
Any modifications in either the project or the controller can be synchronized in one of the following modes:

- on-demand
  will be performed explicitly as desired by the user
- immediate
  occurs instantly when the change is done

Clearing Controller / Project
The application in the project and in the controller, can be cleared independently.

IRM Program
The IRM program includes the control manager, the control strategy (periodic and event programs), the hardware configuration (on board I/O), and the alarming.

Control Manager
The control manager takes care of the major control functions and displays the current status of the IRM program accordingly. The following information is provided:

- Author of the application
- Description
- Application type
- Function block family, version, and numbers
- Number of folders and links
- Memory usage
- Hardware features and compatibility
- Controller connection type
- Engineering units (measurement) type
- Drop of BACnet point settings
- Communication status
- Synchronization status
- Last program change and commissioning dates

Periodic program
Includes the control strategy running in a cyclic manner on a fixed time base.

Event program
Includes event-driven control strategy triggered by particular IO changes.

On board I/O
Shows the hardware I/Os of the controller.

Alarms
Provides the notification classes for establishing alarming.

Hardware Compatibility Check
The hardware configurations of the used physical controller and the hardware defined for the controller in the application can be checked. Any differences are indicated graphically on the terminals on the wire sheet.

Taking / Restoring Snapshot
The current status of an application can be backed up and restored later. This allows restoring a changed application if these changes should be discarded.

Swapping IRM Program
Swapping saves RAM space and reduces processor load and bus traffic. The current state of a swapped device is frozen and saved to an IRM repository on the disk. Then, synchronization is no more possible. In order to synchronize swapped-out devices, the devices must be swapped in again. Swapping can be applied to a single or to all devices per step.

Niagara IRM Engineering Tool
The Niagara IRM Engineering Tool provides the work environment for engineering the controllers.

IRM Operations Monitor / Jobs Sidebar
The IRM Operations Monitor shows all actions in a popup window and all actions are summarized in a list in the Jobs window.

IRM Palette
The following palette is available for creating the application:

- honIrmControl
  provides control function blocks and templates for IRM BACnet devices, IRM programs, and folders.
System Architecture

The following schematic shows an example for a BACnet MS/TP-based system containing an EAGLEHAWK controller as a router and 3 IRM (MERLIN) controllers for room control. The system is engineered using the Niagara IRM Engineering Tool based on the Niagara framework (ver. 4.4 or higher).

Fig. 1. System Architecture based on Niagara IRM Engineering Tool, MERLIN and EAGLEHAWK Controllers

PREREQUISITES

It is assumed that you are familiar with basic Niagara techniques and functions, such as creating platforms, stations, and networks etc.

Make sure that the IRM and EAGLEHAWK controllers are properly connected (see MERLIN N4 Installation Instructions, EN1Z-1035GE51 and EAGLEHAWK NX Installation & Commissioning Instructions, EN1Z-1039GE51).

Make sure that the following steps were done prior of working with the Niagara IRM Engineering Tool.
If not already available in the current Niagara NX installation, copy the following files to the **Modules** folder:

- honIrmAppl-rt.jar
- honIrmAppl-rt.jar.sig
- honIrmConfig-rt.jar
- honIrmConfig-rt.jar.sig
- honIrmConfig-wb.jar
- honIrmConfig-wb.jar.sig
- honIrmControl-rt.jar
- honIrmControl-rt.jar.sig
- honIrmControl-doc.jar
- honIrmControl-doc.jar.sig

---

**Selecting Measurement Type**

Before starting with the application engineering it is important to define the correct measurement type in order to assure that the proper engineering units are used throughout the project.

**Procedure**

1. In the **Nav** tree, expand the **IRM Program** folder of the controller.
2. Double-click on **Control Manager** icon.

3. On the **Property Sheet**, in **Measurement Type**, select the desired type of engineering units from:
   - SI-metric
   - Imperial

---

EN2B-0414GE51 R1119 12
4. Click Save.

Adding IRM Application Template to Palette

Procedure

1. Make sure that the 'honIrmAppl.jar' file is installed in the Modules folder of the Niagara IRM Engineering Tool installation.
2. Open the Niagara IRM Engineering Tool.
3. On the platform and connect to the station.
4. On the Palette pane, click Open Palette icon.
5. In the Open Palette dialog box, enter ‘honIrmAppl.jar’.

6. If not already selected in the list, select ‘honIrmAppl.jar’, and then click **OK**.

7. In the Palette pane, expand the *honIrmAppl* folder.

8. Right-click IRM Program and click **Copy** in the context menu.
9. In the Nav tree, expand the IRM device and delete the existing IRM Program.

10. Right-click the IRM device and click Paste in the context menu.
SETTING UP EAGLEHAWK CONTROLLER

In order to access the IRM controllers via Niagara IRM Engineering Tool or supervisor, a BACNET IP - MS/TP router must be implemented. For this purpose, it is recommended that you use the EAGLEHAWK NX controller which it can host and that you run a station for IRM engineering in parallel.

NOTE: An EAGLE controller can also be used as router. But since it cannot host a station for IRM engineering, it can be used only as router.

The EAGLEHAWK controller is setup using standard Niagara techniques and functions such as creating platforms, stations, and networks. Hence, only the specialties for the EAGLEHAWK, steps 2 and 5 are described in the following.

NOTE: For detailed information on standard Niagara techniques and functions, please refer to the Niagara online documentation and/or the HAWK 8000 User Guide: EN2Z-1030GE51.

Setting up the EAGLEHAWK controller as router and host for an IRM station (optional) includes the following main steps:

Offline Engineering
1. Open platform (PC)
2. Create EAGLEHAWK station
3. Start and connect to EAGLEHAWK station
4. Create BACnet Network
5. Configure EAGLEHAWK as BACNET IP - MS/TP Router

Online Engineering
6. Commission Controller

(See figure next page.)

For detailed information on offline and online engineering, please refer to the "Engineering modes" section, p. 22.
Fig. 2. Basic Engineering for Typical EAGLEHAWK and IRM Application Scenario
Creating EAGLEHAWK Station

For detailed information on standard Niagara techniques and functions, please refer to the Niagara online documentation and/or the HAWK 8000 User Guide: EN2Z-1030GE51.

Procedure

1. Open the platform on the PC.
2. Start creating the station using the New Station Wizard.
3. In the New Station Wizard dialog, select the 'EaglehawkNX.ntpl' template.
4. Continue with creating the station (standard Niagara procedure).

Configuring EAGLEHAWK Controller as BACNET IP - MS/TP Router

In order to access the IRM controllers via Niagara IRM Engineering Tool or supervisor, a BACNET IP - MS/TP router must be implemented. For this, it is recommended to use the EAGLEHAWK NX plant controller which can host and run a station for IRM engineering in parallel.

NOTE: An EAGLE controller can also be used as router. But since it cannot host a station for IRM engineering, it can be used only as router.
### Prerequisite Steps

1. Open the platform (PC).
2. Create EAGLEHAWK station
3. Start and connect to EAGLEHAWK station

### Procedure

1. Create BACnet Network
2. Select the *Property Sheet* of the BACnet network.

---

**NOTE:** For detailed information on standard Niagara techniques and functions, please refer to the Niagara online documentation and/or the HAWK 8000 User Guide: EN2Z-1030GE51.
3. Open the BACnet palette and expand NetworkPorts.


5. From the BACnet palette, add the MstpPort to Network.
6. Expand **MstpPort** and **Link**.

7. Set/enter the following:
   - Network Number = any, e.g. 56 (must be less or equal than 65535)
   - Port Name = RS485_1
   - Baud Rate = Baud_38400 (see also "Baudrate note" below)
   - Enabled = true

8. Click **Save**.
NOTE: If any of the configuration settings are changed during operation, you must restart the EAGLEHAWK controller (see also Baudrate Note).

9. Commission the EAGLEHAWK station to the EAGLEHAWK controller (standard Niagara procedure).

NOTE: For detailed information on standard Niagara techniques and functions, please refer to the Niagara online documentation and/or the HAWK 8000 User Guide: EN2Z-1030GE51.

**Baudrate Note**

The following baud rates for the MS/TP interface in the IRM N4 controller are supported:

- 9600
- 19200
- 38400 (default)
- 57600
- 76800

The baud rate of each IRM controller on the BACnet MS/TP bus is automatically set by the defined baud rate of the BACNET IP - MS/TP Router (EAGLEHAWK) controller after the IRM controller is powered up and connected to the BACnet MS/TP bus.

Setting/changing the baud rate of a single IRM controller is not possible.

When changing the baud rate of the BACnet MS/TP bus of a running system, any connected IRM controller must be power-cycled to adapt the changed baud rate.

---

**ENGINEERING MODES**

There are two kinds of engineering you can use for engineering an IRM project:

- **Offline Engineering**
  
  In this mode, you create an empty BACnet device manually, add an application, and match it afterwards to a device discovered on the BACnet bus by using the service pin. This usually applies when the engineering is done in the office without having the controller hardware but knowing the hardware specification of the later used devices.

- **Online Engineering**
  
  In this mode, you discover the devices on the BACnet network in the first step and use the devices instantly for application engineering. This is recommended when doing the engineering directly at the site with the devices already installed on the BACnet bus.
Offline Engineering

Offline engineering includes the following steps:

- Create IRM BACnet device
- Create application for the IRM BACnet device (optional)
- Later at the site, do the following:
  - Connect to the BACnet network
  - Discover devices on the BACnet network
  - Match the empty device with a discovered device
  - Synchronize the application, if necessary, by applying any of the following actions appropriately:
    - Learn from controller
    - Teach to controller
    - Clear project
    - Clear controller

For detailed descriptions, please refer to the corresponding sections:

- "Creating IRM BACnet Device", p. 23
- "Working with Applications – The IRM Program", p. 39
- "Matching Devices ", p. 25
- "Teach To Controller", p. 95
- "Learn From Controller", p. 97
- "Clear Project", p. 98
- "Clear Controller", p. 98
- "Check Hardware Compatibility", p. 99.

For further information on offline engineering, please refer also to the "Factory Device Handling" section, p. 31.

Creating IRM BACnet Device

Procedure

1. Double-click the BACnet network folder in the Nav tree, and then select Niagara IRM Engineering Tool view.

2. On the bottom, click New.
3. In the *New* dialog box, select ‘Irm Bacnet Device’, and then click **OK**.  

**RESULT:** The IRM BACnet device is created and added to the *Database* pane and the *BacnetNetwork* tree.

4. Expand the device and create the IRM program by adding control strategy, hardware layout and alarm settings (optional).

To do so, please refer to the section "Working with Applications – The IRM Program", p. 39.

5. After finishing the application engineering, match the offline disconnected device to the desired physical device discovered on the BACnet network (see “Matching Devices ”, p. 25).
6. Teach the application to the controller. If there is an application already running in 
the controller, clear the controller
7. Finally, perform the hardware compatibility check to make sure that the 
application is properly designed for running seamlessly in the created device.
8. If software recommends, remove function blocks in the control logic.

NOTE: Add the physical IO points. Make sure to select the right physical point 
template that matches the online device model.

NOTE: It is allowed matching smaller offline device models with bigger online 
device models. It is strongly recommended not match devices vice versa, 
since hardware compatibility issues may arise. In this case, please execute 
the hardware compatibility check and rework the application accordingly. 
Or, install a bigger device with more hardware I/Os at the site.

For further information on offline engineering, please refer also to the "Factory 
Device Handling" section, p. 31.

Matching Devices

| Purpose       | This function matches an offline-configured, disconnected IRM device to an online 
|               | discovered IRM controller on the BACnet network. The offline-configured, disconnected IRM device will be configured according to the 
|               | properties of the online discovered IRM device. Inconsistencies may occur and can be solved by using the corresponding actions to 
|               | establish synchronicity. |

Procedure

1. Press the service pin at the device you want to match to the offline created IRM 
BACnet device. Do this for all devices, you want to match.
2. On the Discovered pane, click Discover. 
   RESULT: All devices available in the BACnet network are discovered.
   RESULT: In the Service Pin Rank column, the service pin action is indicated 
by consecutive numbers depending on the time when the service 
pin was pressed at the device.
3. On the Discovered pane, select the online device and in the Database pane, 
select the offline created IRM BACnet device.
4. At the bottom, click **Match**.

   RESULT: The *Match* dialog box displays.

5. If desired, you can enter the device ID, Network address, and MAC Address.

6. Enable **Use Cov** by selecting ‘true’ from the drop-down listbox.

7. Click **OK**.

   RESULT: The devices are matched as the properties of the disconnected offline IRM BACnet device indicate in the *Database* pane. The device gets the data of:

   - Device ID
   - Status
     - e.g. Alarm, unacknowledged alarm.
   - Model
   - Firmware Revision
   - Serial No.
   - Irm Family
   - Irm Program Name
The application status is checked by learning the project from the controller. To do that, click the "Learn from controller" button displayed as follows:

RESULT: If, as a result, any inconsistencies of the application appear between the disconnected IRM BACnet device and the discovered IRM controller, a notification message will be displayed and the relevant synchronization status is indicated in the control manager (see section "Synchronicity Check via Control Manager", p. 30).

9. To solve any inconsistencies, synchronize the applications by doing any of the following:

- Clear project
- Clear controller
- Teach to controller
- Learn from controller
- Checking Hardware Compatibility

For detailed descriptions, please refer to the corresponding sections:

- "Clear Project", p. 98
- "Clear Controller", p. 98
- "Teach To Controller", p. 95
- "Learn From Controller", p. 97
- "Check Hardware Compatibility", p. 99

For further information on offline engineering, please refer also to the "Factory Device Handling" section, p. 31.
For detailed descriptions, please refer to the corresponding sections:

- "Working with Applications – The IRM Program", p. 39
- "Teach To Controller", p. 95
- "Learn From Controller", p. 97
- "Clear Project", p. 98
- "Clear Controller", p. 98
- "Check Hardware Compatibility", p. 99

For further information on online engineering, please refer also to the "Factory Device Handling" section, p. 31.

Procedure

1. For virgin factory devices, press the service pin at the devices.
2. On the Discovered pane, click Discover.

RESULT: All devices available in the BACnet network are discovered.

3. On the Discovered pane, select the devices you want to add to the database.

4. Click Add.

RESULT: The Add dialog box displays. You can change properties of the devices before adding them to the database. To do so, select the device and change the desired property from:

- Name
- Device ID
- Network
- Use Cov
- Max Cov Subscriptions
5. Click **OK** in *Add* dialog box.

RESULT: All devices are added to the database. The devices are learned from the project by clicking the "Learn from controller" button displayed as follows:

6. **RESULT:** All devices are added to the database. The following properties are shown:
   - Name
   - Device ID
   - Status
     - e.g. Alarm, unacknowledged alarm.
   - Model
   - Firmware Revision
   - Serial No.
   - IrM Family
   - IrM Program Name
• Irm Application Type
• Irm Sync Status
  synchronized or not synchronized
• Irm Last Change
• Irm Master Sync
  shows whether the device is part of a master sync group
  (source, target, none)
• Password status

A synchronicity check is performed and particular messages are
displayed if non-synchronicity is detected (see section
“Synchronicity Check via Control Manager”, p. 30).

NOTE: Starting from Controller version 1.0.1.9 and Niagara IRM Engineering Tool
version 1.0.1.7, the “Learn From Controller” function is not performed
automatically when a discovered controller is added to the database.
“Learn From Controller” has to be initiated manually when a controller is
added to the database.

Synchronicity Check via Control Manager

Whenever you connect to the BACnet network via device discovery (online mode) or
when matching an empty IRM device (offline mode) to a discovered device of the
BACnet network, the control manager starts working and checks the applications in
the project and in the controller on synchronicity.

An application in the project can be either synchronous or not synchronous with the
application running in the connected controller.
As soon as non-synchronicity is detected, notification messages about the
synchronization status are displayed.

NOTE: Software always give support by displaying appropriate messages which
describe the current status/problem and how to proceed.

WARNING!
It is strongly recommended to read and note any messages in order to
avoid derived misleading actions afterwards which may result in the
deletion or loss of application data.

To solve any inconsistencies, synchronize the applications by doing any of the
following:

• Clear project
• Clear controller
• Teach to controller
• Learn from controller
• Checking Hardware Compatibility

For detailed descriptions, please refer to the corresponding sections:

• "Clear Project", p. 98
• "Clear Controller", p. 98
• "Teach To Controller", p. 95
• "Learn From Controller", p. 97
• "Check Hardware Compatibility", p. 99

For detailed description of the control manager, see section “The Control Manager”,
p. 42).
Default Factory Device ID Settings

For successful communication via BACnet, each controller must have a unique device ID. By default, each virgin IRM controller has assigned to it the factory device ID = 4194302. But when multiple controllers have the same device ID assigned, they cannot talk via the BACnet protocol (Who is? - I am broadcast messages).

As a result, virgin controllers on the MS/TP bus will not be visible after discovery.

IRM Configuration Settings

For engineering IRM controllers on a BACnet network, particular settings must be configured as default settings.

Procedure

1. To do this, expand the BacnetNetwork folder in the Nav tree.
2. Double-click the irmConfig folder in the BACnet network to display the property sheet.

3. On the property sheet, do any of the following:

   - **Id Range Min**
     Enter the value for the lowest device ID that is assigned to a device during discovery.

   - **Id Range Max**
     Enter the value for the highest device ID that is assigned to a device during discovery.

   Devices will get device IDs automatically assigned within the range of ID Range Min and ID Range Max settings

   NOTES: Actual device ID assignment is executed not until the corentroller is connected to the network.

   Then when adding the discovered controllers to the database, in the Edit dialog to each controller a unique device ID is issued within the range of the min and max ID limits configured in the IRM Config Properties (see above).

   - **Teach Modified Controller**

     **IMPORTANT!**
     
     It is strongly recommended not to change this setting.

     NOTE: Starting from Controller version 1.0.1.9 and Niagara IRM Engineering Tool version 1.0.1.7, the “Learn From Controller” function is not performed automatically when a discovered controller is added to the database. "Learn From Controller" has to be initiated manually when a controller is added to the database.
- Inconsistent Program Check
  If true (default), software checks the application for inconsistencies and shows a message if this is the case. Inconsistencies may occur if you have moved/copied the application to an external location, e.g. to the palette, changed it there and then include it again.

  NOTE: Copying and pasting parts of an application from an external location such another controller does not generate inconsistencies in the application.

**Assignment of Unique Device IDs**

Depending on the requirements/scenarios (online at the site, offline in the office), virgin IRM controllers need to be engineered either by using or by NOT using the service pin button.

**Offline Mode (service pin not accessible)**

Create a device (controller) by entering the device name and the serial number for the controller (use a barcode scanner software if available or enter the data manually). Do this for all controllers.

Discover the controllers. This will list all offline devices. Objects will come as ???.

At the site, go online by connecting to the network.

Discover the controllers on the network. Now software searches for controllers that have the same serial numbers issued offline.

Once those controllers are added to the network, they will start communicating with the physical online controllers.

Then when adding the discovered controllers to the database, to each controller a unique device ID is issued within the range of the min and max ID limits configured in the IRM Config Properties (see above).

  NOTE: In the Add dialog box, you can change the pre-defined device ID before the database addition.

**Online Mode (Service pin accessible)**

For each controller, press the service pin at the controller before executing the Discover command. The controllers will be discovered, but they still show the default factory ID.

Then when adding the discovered controllers to the database, to each controller a unique device ID is issued within the range of the min and max ID limits configured in the IRM Config Properties (see above).

  NOTE: In the Add dialog, you can change the pre-defined device ID before the database addition.

In both cases, device IDs will be assigned within the range of the min and max ID limits configured in the IRM Config Properties (see above).

---

**Detailed Offline Procedure**

1. Expand the BacnetNetwork folder in the Nav tree.
2. Right-click the IrmConfig folder and then click Add Serial Number in the context menu.

  **RESULT:** The Add Serial Number dialog box displays.
3. Enter the **Device Name**, **Serial No** and **Device Id**. Use a barcode scanner software if available or enter the data manually.

![Add Serial Number](image)

NOTE: Caused by the controller reboot after the firmware download the device status changes to `{unackedAlarm}`. In order to change the device name, this alarm has to be acknowledged.

4. Confirm by clicking **OK**.
   
   RESULT: The device is added to the *IrMConfig* Property Sheet on the right pane.

![Property Sheet](image)

5. Repeat steps 2 through 4 for all controllers.

6. Discover the controllers. This will list all offline devices. Objects will come as ???.

![Discovered](image)

7. At the site, go online by connecting to the network.

8. Discover the controllers on the network.
9. Now software searches for controllers that have the same serial numbers issued offline.

10. Add the controllers to the database.

RESULT: Once those controllers are added to the network, they will start communicating with the physical online controllers.

Then when adding the discovered controllers to the database, each controller a unique device ID is issued within the range of the min and max ID limits configured in the IRM Config Properties (see above).

NOTE: In the Add dialog, you can change the pre-defined device ID before the database addition.

11. Finish by matching the offline device with the online device in the database (see "Matching Devices", p. 25).
1. **IRM BACnet Device** point discovery should be done with the **IRM BACnet Point Manager**. The BACnet Point Manager is used by default. The **IRM BACnet Point Manager** must be selected manually.

1. **Detailed Online Procedure**

   1. Double-click the *BacnetNetwork* folder in the *Nav* tree.
   2. At the controller, press the service pin for 15 sec. Then click **Discover**.

      **RESULT:** The controller will be discovered, but it still shows the default factory ID '4194302'.
3. Repeat step 2 for each controller.
4. Select the controllers.

5. Add them to the database via drag&drop or by clicking Add.

RESULT: The Add dialog box displays.
To each controller, a unique device ID is assigned within the range of the min and max ID limits configured in the IRM Config Properties.

NOTE: In the Add dialog box, you can change the pre-defined device ID before the database addition.

6. Click OK.

RESULT: The controllers are added to the database.

For further information, please refer also to the "Engineering modes" section, p. 22.

7. **IRM BACnet Device** point discovery should be done with the **IRM BACnet Point Manager**.
   The BACnet Point Manager is used by default. The **IRM BACnet Point Manager** must be selected manually.
WORKING WITH APPLICATIONS – THE IRM PROGRAM

Application engineering includes the creation of:

- periodic control strategy and/or
- event driven control strategy
- hardware layout (onboard IO)
- alarming settings.

Preparing Niagara Work Environment

For working conveniently during application engineering, prepare the work environment by enabling the following functions in the Niagara IRM Engineering Tool:

**BACnet Device Manager View**

Double-click the BACnet network folder in the Nav tree, and then select the IRM BACnet Device Manager view.

**IRM Palette**

The following palette is available for creating the application:

- honIrmControl
  
  It provides control function blocks and templates for IRM BACnet devices, IRM programs, and folders

Open the palette in the Palette pane

**IRM Operations Monitor / Jobs Sidebar**

The IRM Operations Monitor shows all actions in a popup window and all actions are summarized in a list in the Jobs window.
In the **Windows** menu, click **Side Bars**, and then click **Jobs**.

When actions are executed, they will be shown in the temporary **IRM Operations Monitor** popup window and listed permanently in the **Jobs** window.

---

**IRM Program Components**

**IRM Program in Nav tree**

The IRM Program is part of the device and includes the following components:

- Control Manager
- Periodic program
- Event program
- Onboard IO
- Alarms
Wire Sheet and Property Sheet View

The component's work space can be shown by double-clicking it in the tree and selecting the **Wire Sheet** view. For the Control Manager, the **Property Sheet** displays after double-clicking on the item.

For detailed functional descriptions, please refer to the following sections:

- "The Control Manager", p. 42
- "Periodic Program", p. 46
- "Event Program", p. 47
- "Onboard IO", p. 48
- "Alarms", p. 50

Control Logic Source / Palette

For creating the application, the "honIrmControl" palette is used which includes all necessary components such as logic function blocks, terminals, and templates for IRM BACnet devices, IRM programs, and folders.

---

**Fig. 3. Palette showing physical Point template, Control Loop, Logic, BACnet objects and Arithmetic function blocks**

**Modifications and Consistency Check**

Whenever working on the application, e.g. periodic program, on board I/O etc., every modification is detected by the control manager and indicated graphically by a yellow warning symbol that replaces the original symbols at the modified item on the wire sheet, and in the tree. This applies to teaching mode 'on demand' only. When working in teaching mode 'immediate', changes are not indicated graphically since they are written instantly to the controller.
Example:

In addition, the IRM Program icon and the component's icon gets a dark square background.

The control manager icon switches from synchronous state to non-synchronous state.

On the property sheet of the controller manager, the synchronization status switches accordingly:

![Fig. 1. Synchronization Status Information in Control Manager before Application Change]

![Fig. 2. Synchronization Status Information in Control Manager after Application Change]

Depending on where the application is created (project) or changed (project or controller), the application must be taught to the controller, or must be learned from the controller to achieve synchronicity. In both cases, if meaningful and required, the controller or the project must first be cleared.

The Control Manager

Procedure

8. In the Nav tree, expand the IRM Program folder.
9. Double-click on **Control Manager** icon.

RESULT: On the **Property Sheet**, the following information is displayed.

- Author of the application
- Description
- Application Type
- Function Block Family
- Function Block Version
- Number of Folders
- Number of Function Blocks
- Number of Links
- Memory Usage shows the memory usage of the device, application and parameters in percentage and graphically
- Controller Hardware Features displays the I/O configuration (online available in Online mode)
- Hardware Compatibility indicates whether the terminal layout of the application is compatible with the hardware layout of the physical controller
- Controller Connection allows selecting the connection type from: network, offline, and simulation
- Teaching Mode allows selecting the teaching mode from on demand and immediate
- Measurement Type allows selecting the type of engineering units from SI-metric or Imperial
- Drop of BACnet Output defines which type of reference point, reference input or output is created when dropping a BACnet output from another device
- Drop of BACnet Value defines which type of reference point, reference input or output is created when dropping a BACnet value point from another device
- Communication Status displays the communication status of the device: online or offline
- Is Synchronized indicates whether the project and the controller are synchronized (yes) or not (no)
• Synchronization status
displays a message describing the cause of the current
synchronization status
• Last program change
displays the date of the last program change
• Last Commissioned
displays the date of the last commissioning

10. Do any of the following if desired:

a. In Controller Connection, select the connection type for the devices from:
   – Network
     connects to the physical BACnet network
   – Offline
     disables the connection to the network. This is useful in case you want to
     reduce bus traffic if the application has not been swapped out.

b. In Teaching Mode, select how the application changes in the project are
   written to the controller from:
   – Immediate
     changes are written to the controller automatically and are effective
     immediately
   – on demand
     changes are written to the controller manually and explicitly by the Teach
to Controller action.

   NOTE: For both teaching modes, only the changes are written to the device,
   hence the process is very fast

c. In Measurement Type, select the type of engineering units from:
   – SI-metric
   – Imperial

11. Click Save.

---

### Memory Usage

The memory of the controller’s RAM is max. 344 KB and is consumed by the:

- device
- application
- parameters

The Niagara IRM Engineering Tool provides a memory consumption check of the
current application in the tool. This is an approximate calculation of the application in
the tool but not a real-time consumption within the controller.

The calculated memory usage is shown in the IRM Manager as percentage and
graphically via traffic light symbol.

---

#### Memory Usage and Its Display

The max. number of memory in the controller is 344 KB.

Each folder added consumes 8 KB of memory. The max. number of folders is 36.
4 folders are reserved by default. Hence, user-defined individual folders can be 32
at max.

The calculated memory consumption will be indicated graphically and as percentage
in the traffic light symbol as follows:

- Green < 80 %
- Yellow > 80 %
The memory usage can be viewed on the property sheets of the following different levels:

- IRM Program (percentage)
- Control Manager (percentage and graphically)
- Periodic and Event Program and its subfolders (percentage)

Recommendations

It is recommended to monitor the memory usage on the Control Manager level and to avoid to exceed the limit of 80%.

To keep the memory usage as low as possible, it is recommended to optimize application engineering by keeping the number of folders as low as possible and the number of added function blocks per folder as high as possible in relation to the required result of the logic.

The max. number of function blocks per folder is 150. If you want to use more than 150 function blocks, please split them among two or more folders.
Periodic Program

Creates the control strategy running in a cyclic manner on a fixed time base. All function blocks in the root folder and its subfolders are processed. The periodic program is executed every 500 msec within the controller.

Procedure

1. Double-click **Periodic Program** in the tree, and then select the **Wire sheet view**.
2. Open the **hnonIrmControl** palette.
3. From the palette, add control items (see "IRM Program Components", p. 40) to the wire sheet via drag&drop.
4. Create/change the control logic by applying desired steps such as connecting, adding, deleting, moving icons, and/or adding and deleting connections.

Synchronization Check

Any modifications on the Periodic Program wire sheet are detected by the control manager. Modifications can be any of the following:

- Added item
- Deleted item
- Moved item
- Deleted connection
- Added connection
- etc.

As result, all modified items will be not in sync with the application in the controller and hence indicated by a yellow "warning" symbol on the item.
The application can be synchronized according to the set teaching mode, on demand or immediate.

After synchronization, the warning symbols are removed and the items are indicated in its synchronized state.

See also the description of the "The Control Manager", p. 42.

Event Program

Creates event-driven control strategy. All function blocks in the root folder and its subfolders are processed.

The event program is executed:

- when the time interval of 1000 msec has elapsed
- whenever the state of a hardware point configured as binary input that is used in the event program, changes and when this binary input point is used as an input slot to a function block

You can configure BI and UI as binary inputs.

Procedure

1. Double-click Event Program in the tree, and then select the Wire sheet view.
2. Open the honIrmControl palette.
3. From the palette, add control items (see "IRM Program Components", p. 40) to the wire sheet via drag&drop.
4. Create/change the control logic by applying desired steps such as connecting, adding, deleting, and moving icons, and/or adding and deleting connections.

Synchronization Check

Any modifications on the Event Program wire sheet are detected by the control manager. Modifications can be any of the following:

- Added item
- Deleted item
- Moved item
- Added connection
- Deleted connection
- etc.
As result, if working in ‘on demand' teaching mode, all modified items will be not in sync with the application in the controller and hence indicated by a yellow “warning” symbol on the item.

The application can be synchronized according to the active teaching mode, ‘on demand' or ‘immediate'. After synchronization, the warning symbols are removed and the items are indicated in its synchronized state. When working in teaching mode 'immediate' this happens immediately.

See also the description of the “The Control Manager”, p. 42.

NOTE: In order to achieve the best performance for the operation of light and blind, the light and blind function block has to be put into the Event Program folder. In addition, the UIs of the buttons used must be connected from the On Board IO folder to the Event Program folder, e.g. to a PassThru function block which is not used anymore.

NOTE: In order to achieve the best performance for the operation of light and blind, the light and blind function block has to be put into the Event Program folder. In addition, the UIs of the buttons used must be connected from the On Board IO folder to the Event Program folder, e.g. to a PassThru function block which is not used anymore.

Make a connection from the “On board IO” Wiresheet to the Event Program to a Pass_Thru Functionblock

Onboard IO

Shows the hardware configuration to which the application is designed for. The hardware displayed here can differ from the real hardware layout of the physical controller. For proper operation, the hardware of the project does not have to fit 100% to the physical hardware of the controller used later. But, if the difference is too much, hardware compatibility is not guaranteed. The control manager will show incompatibilities and software give support and tells what to do for synchronicity.
Procedure

1. Double-click On Board I/O in the tree, and then select the Wire sheet view.
2. Open the honirmControl palette.
3. From the palette, add single physical terminals manually or a pre-defined template to the wire sheet via drag&drop. Pre-defined templates are matched to particular controller models. The following physical terminals and templates are available:

4. Create/change the layout by applying desired steps such as connecting, adding, deleting, and moving terminals, and/or adding and deleting connections.

NOTE: When discovering datapoints on the BACnet network via IRM point manager, the I/O hardware points of the IRM controllers are not detected and visible by default. To expose I/O hardware points on the BACnet network, please refer to the "Exposing Hardware I/O Points on BACnet" section, p. 58.

Synchronization Check

Any modifications on the On Board I/O wire sheet are detected by the control manager. Modifications can be any of the following:

- Added item
- Deleted item
- Moved item
- Added connection
- Deleted connection
- etc.
As result, all modified items will be not in sync with the application in the controller and hence indicated by a yellow “warning” symbol on the item.

The application can be synchronized according to the active teaching mode, 'on demand' or 'immediate'. After synchronization, the warning symbols are removed and the items are indicated in its synchronized state.

See also the description of the "The Control Manager", p. 42.

### Alarms

**Alarming**

Provides the notification classes for establishing alarming
Synchronization Check

Any modifications such as adding, deleting or copying & pasting items on the Alarms wire sheet are not allowed.

You can only change the properties of the pre-defined notification class objects on the Property Sheet after you have double-clicked the icon.

Secure Commissioning Communication

Starting from Controller version 1.0.1.9 and Niagara IRM Engineering Tool version 1.0.1.7 Secure Commissioning Communication is implemented for IRM Controller commissioning.

Secure Commissioning Communication means that only commissioning communication messages between the Niagara Engineering Tool and the IRM Controller are sent encrypted. All further communication is not affected.
In the initial state (factory setting), the IRM controller and the Niagara IRM Engineering Tool use a non-visible default password. This default password is not exposed to the outside world.

If a user-defined password has been entered, this password is sent to the IRM controller and from then the controller cannot be accessed without this password.

Resetting a controller password can only be done by conducting a Reset to factory. Refer to "Reset to Factory Delivery", page 128.

**IMPORTANT:**
Resetting the controller password means conducting a manually factory reset for each single controller.

To understand each other, the Niagara IRM Engineering Tool and the IRM Controllers must both use the same password. When changing a project or when discovering IRM controllers, it may be necessary to re-enter the passwords on one or both sides. Password setting can be conducted via the **Set Password for** function in the Niagara IRM Engineering Tool. It is possible to apply the password only for the tool or for the tool and the controller.

### Setting a Password for a Single Controller

The user defined password setting for single controller is done in **IRM Program** via the **Action** menu entry **Set Controller Password**.

Password change is done via the same procedure with additional old password input.

### Setting a Password for All Controllers

Bulk user-defined password setting for all controllers in the BACnet is done via right click at **IRMConfig** select **Action** menu entry **Set Controller Password**.
The user-defined password in the Niagara IRM Engineering Tool will also be set.

To ensure that the password in the IRM Controller and the Niagara IRM Tool is the same, **Set Password for** can be used to select where the password should be changed.

Change password is done via the same procedure with additional old password input.

**NOTE:** Old password should be empty if the password is set at first time.

---

**Resetting the User Password to the Default Password in Tool**

Right click at **IRMConfig Action** menu entry **Reset Tool Password**
Resetting a controller password can only be done by conducting a Reset to factory. Refer to "Reset to Factory Delivery", page 128.

**Use Cases**

The table below shows different use cases for setting password to controller.

<table>
<thead>
<tr>
<th>Use Case</th>
<th>IRM Controller ≥ 1.0.1.9</th>
<th>Niagara IRM Engineering Tool ≥ 1.0.1.7</th>
<th>How it works/Solution</th>
</tr>
</thead>
</table>
| **First time user commissioning the controller** | New Controller connected to BACnet network                   | First time, device is discovered and added to IRM device manager view | Default password is used for Teach/Learn/Ping/Clear controller. Usage of user defined password.  
**Recommendation:** Use default password for commissioning and enter user defined password after commissioning. Keep the user defined password in safe place for future commissioning.  
**NOTE:** There is no way to read/learn the application logic from controller without user defined password.  
User defined password setting via **Set Controller Password** in **Actions** menu of **IRM Program**.  
Reset user defined password to default password via **Set Controller Password** in **Actions** menu of **IRM Program**. Enter additionally the old user defined password. |
| **Set common User Defined Password for all controller** | New Controllers connected in BACnet network.  
Default Password is set from factory | First time, devices are discovered and added to IRM device manager view | Default password is used for Teach/Learn/Ping/Clear controller.  
**Solution:** User can set user defined password in Engineering Tool and in all controller. |
<table>
<thead>
<tr>
<th>Use Case</th>
<th>IRM Controller ≥ 1.0.1.9</th>
<th>Niagara IRM Engineering Tool ≥ 1.0.1.7</th>
<th>How it works/Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change common User Defined Password for all controllers</td>
<td>Existing controllers.</td>
<td>Engineering Tool has the same user defined password as all controller. User want to change the user defined password in all controllers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>User defined password is already set by user during commissioning</td>
<td></td>
<td>Existing user defined password is used for Teach/Learn/Ping/Clear controller.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Solution:</strong> User can change user defined password in all controllers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use <strong>Set Controller Password</strong> in <strong>Actions</strong> menu at IrmConfig in Bacnet Network.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Old and new password has to be entered.</td>
</tr>
<tr>
<td>Modify application in existing controller by another user</td>
<td>Existing controllers.</td>
<td>User2 discovered the devices and add those devices into Irm device manager</td>
<td>User defined password is set by User1.</td>
</tr>
<tr>
<td></td>
<td>User defined password is set during commissioning</td>
<td></td>
<td>User2 uses different user defined password and gets the below error message:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>There is an error in controller communication. It could be password mismatch in controller and station.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>User2 can’t Teach/Learn/Ping/Clear the controller until the user defined password from user1 is set in the Engineering Tool.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Having entered User1 user defined password all engineering operations can be conducted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>IMPORTANT:</strong> If user defined password is not communicated no engineering operations are possible. If it is lost, the controller has to have a factory reset and has to be commission again.</td>
</tr>
<tr>
<td>Modify application in existing controller with same station</td>
<td>Existing controllers.</td>
<td>Existing station.</td>
<td>All engineering operations can be conducted.</td>
</tr>
<tr>
<td></td>
<td>User defined password is already set</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modify application in existing controller with other station</td>
<td>Existing controllers.</td>
<td>Another station.</td>
<td>User gets error in job log and in dialog during Teach/Ping/Learn.</td>
</tr>
<tr>
<td></td>
<td>User defined password is already set</td>
<td></td>
<td>There is an error in controller communication. It could be password mismatch in controller and station.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Solution:</strong> Reset user defined password in Engineering Tool. Set user defined password used in the controller also in the Engineering Tool.</td>
</tr>
<tr>
<td>Modify application with new station</td>
<td>Existing controllers.</td>
<td>New station.</td>
<td>User gets error in job log and in dialog during Teach/Ping/Learn.</td>
</tr>
<tr>
<td></td>
<td>User defined password is already set</td>
<td></td>
<td>There is an error in controller communication. It could be password mismatch in controller and station.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Devices are discovered and added into station DB. Default user</td>
<td><strong>Solution:</strong> Reset user defined password in Engineering Tool. Default user password will be used in the Engineering Tool.</td>
</tr>
</tbody>
</table>
Use Case | IRM Controller ≥ 1.0.1.9 | Niagara IRM Engineering Tool ≥ 1.0.1.7 | How it works/Solution
--- | --- | --- | ---
Replace the damaged controller with new controller from factory. | New controller Default user password is set from factory | User use the same station to configure the controller again. In Station, user defined password is set. | User cannot do Teach/Learn/Ping/Clear controller
Solution1: Reset the user defined password in the Engineering Tool. Default user password will be used in the Engineering Tool and new controller added in the network. Set user defined password again? Solution2: Backup of the application logic with the Engineering Tool. Next delete the controller from the data base in the within the Engineering Tool. Add the new discovered device into the station DB and then restore the backup.

User want to reset controller to factory-default | Existing controllers. User Defined Password is set | Existing station. User defined password is set. | Conduct a factory reset at the controller Reset the user defined password in the engineering Tool Discover and match with controller in the Engineering Tool Set user defined password and do the commissioning.

Copy device for new controller in the network | New Controller added to the network Default user password is set from factory | Copy from controller in station having user defined password set. User matching the cloned device with new controller added in the network. | Copy will copy the password from the existing device. Solution: Reset user defined password in Engineering Tool. Default user password will be used. Set the user defined password used in the old controller.

Clone Application from one controller to another controller | Clone application logic from one controller to multiple controllers or single controller in IRM Device Manager view | Clone operation in IRM Device manager view will not change the password in target controllers. Also, it won’t copy the password from source controller to target controller.

Password Status

The Password Status column has been added in the IRM Device manager to help the user easily check the status and identify the problem in setting a password for the IRM controller.

Below the highlighted text, the following will be shown in the Password Status column:

- Not supported
  If the IRM controller firmware version is below 1.0.2.2.

- Default password
  If no password has been set in the IRM controller.

- User password
  The user can set or change the password in the IRM controller.

- Password mismatch
  If the password set for the tool does not match the password set in the IRM controller.
Application Engineering Guidelines

Commissioning and Station Copier Usage

Before a controller can be used for the first time after initial installation or upgrade, it must be commissioned by loading the following components from the PC (localhost) into the controller (remote host):

- Niagara software
- license
- application / station

When changes will be done later to the application on the PC, the station can be copied into the controller by using the Station Copier function. In this case, a new commissioning is not necessary.

For detailed description, please refer to the HAWK 8000 User Guide: EN2Z-1030GE51.

Final Step after Application Engineering

IMPORTANT!

It is strongly recommended to swap out the application in any case after finishing the engineering in online or offline mode.

Swapping out the application avoids additional communication between Niagara IRM Engineering Tool and controller and reduces bus traffic.

For details about Swapping, please refer to the "Swap IRM Program" section, p. 101.

Memory and Folder Usage

It is recommended to monitor the memory usage on the Control Manager level and to avoid to exceed the limit of 80 %.

To keep the memory usage as low as possible, it is recommended to optimize application engineering by keeping the number of folders as low as possible and the number of added function blocks per folder as high as possible in relation to the required result of the logic.

The max. number of function blocks per folder is 150. If you want to use more than 150 function blocks, please split them into two or more folders.

For details, please refer to the "Memory Usage" section, p. 44.

IRM Function Blocks and External Application Components

The IRM program accepts IRM control function blocks only. Direct connection between IRM control function blocks with Niagara components by using the Niagara Link function does not work. External communication should be established via BACnet components which link the IRM function blocks to the Niagara components.

NOTE: Usage of SystemA function block provided in Niagara IRM Engineering Tool version 1.0.1.7 and higher require controller firmware version 1.0.1.9 or higher. If SystemA function block is not used, controllers with firmware versions lower than version 1.0.1.9 can be engineered with actual IRM Engineering Tool version.
Application Templates Usage

For quick engineering and avoiding hardware compatibility problems, use pre-defined onboard I/O configurations (templates) provided in the PhysicalPoints folder of the honirmControl palette. When using any of these, make sure that they fit to the physical device model.

Exposing Hardware I/O Points on BACnet

When discovering datapoints on the BACnet network via IRM point manager, the I/O hardware points of the IRM controllers are not detected and visible by default. To expose hardware I/O points on the BACnet network, they must be prepared individually to make them discoverable. This is done by adding a BACnet function block, e.g. a BACnet numeric input, to the periodic or event driven program and link it to the hardware I/O point function block you want to expose on the BACnet network.

Procedure

1. Double-click On Board I/O in the tree, then select the Wire sheet view.
2. On the wire sheet, identify the I/O hardware point you want to expose to the BACnet network and note its type and name, e.g. an analog output with the name 'AO4 – HwAoFanSpeedCtl'.
3. On the On Board I/O wire sheet, right-click the I/O hardware datapoint, then click Link Mark in the context menu.
4. Double-click **Periodic Program** in the tree, and then select the **Wire sheet view**.
5. From the **honirmControl** palette, add a 'BACnet Numeric Input' BACnet function block to the wire sheet.
6. Change the default name to the name of the hardware I/O point you want to expose, e.g. 'AO4 – HwAoFanSpeedCtl'.
7. Click **OK**. The BACnet Numeric Input is added to the wire sheet.
8. Right-click the BACnet Numeric Input, then click **Link Mark** in the context menu.
Modbus Interface Engineering

The Modbus protocol is described in the following two documents, which can be found at www.modbus.org:
- Modbus_over_serial_line_V1_02.pdf
- Modbus_Application_Protocol_V1_1b.pdf

Addressing and Wiring Rules

1. Use only one IRM controller per Modbus network.
2. Maximum number of devices: 32, including the IRM controller.
3. Address: Can be from 1 to 247.
   Note: Never use address 0, because 0 is used by the IRM controller exclusively for broadcast addressing!
   • Address must be unique on the Modbus.
   • Dip switch setting (when available) must be correct (depending on its physical connection) for every Modbus device address.
4. Termination and Bias: Must be correct for every IRM Modbus Device.
5. Naming of Modbus terminals. Modbus devices may use different naming and descriptions for the wire connections. Possible synonyms can be as follows:
   • Transmit Data: TxD, Tx, T+/R+, D1, B
   • Receive Data: RxD, Rx, R-/T-, D0, A
   • Signal Ground: GND, COMMON

NOTE: A and B are not clearly defined. Depending on the manufacturer, A and B have different meanings.

Engineering Workflow

The following order of investigation and checks is recommended:
1. Get Modbus RTU slave documentation
   It is mandatory to have the vendor’s documentation of the Modbus functionality for every Modbus device that is to be connected to the IRM Modbus Device. This will allow obtaining information about most of the required information, but most likely not all of it. Information that is missing can be obtained from the support function of the vendor.

2. Verify usage and activation of the correct Modbus device interface
   Modbus devices may not have the Modbus interface activated by default. Activation may be necessary via software, e.g., setting a parameter via the Modbus device’s user interface. In addition, activation may be necessary via hardware, e.g., shorting an input of the Modbus device.

   When using a MVC or W858xxx controller as a Modbus device, verify that you have the correct hardware variant which supports Modbus on the RS485 interface as expected. Consult the MVC documentation for this purpose.

3. Verify RTU Transmission Mode
   All Modbus devices must use transmission mode RTU. No other transmission mode (e.g., ASCII) is supported.

4. There is only one Modbus master allowed on the bus, which is the IRM Controller. It does not support “Multi-Master” nor “Modbus plus” devices nor does it support such configurations on the Modbus. As per the Modbus standard, the IRM Controller (as the master device) does not have a Modbus address. The IRM Controller has to be added to the Bacnet Network.

   - Drag and drop IrmBacnetDevice from honIrmControl palette to BacnetNetwork

   - Add Modbus Device to Periodic/Event Program wire sheet. In the Nav tree, expand the IRM Program folder of the controller. Double-click on Periodic program icon
• Drag and drop ModbuDevice from honIrmControl palette to wire sheet

• Change name in dialog box
To enter ModbusDevice settings double click on ModbusDevice function block in wire sheet

5. Check Modbus Channel Communication Parameters

**Baud Rate:**
All Modbus devices must have an identical Baud rate, an identical Parity, and an identical number of stop bits.
Typically, the Baud rates of Modbus devices are set and changed via their user interfaces or via their separate configuration software.

**Parity:**
Modbus devices may have a configurable Parity, most likely, they will not have a configurable number of stop bits.
If a Modbus device has an un-matching number of stop bits, this device cannot be used!
NOTE: Not all Modbus products comply with the Modbus standard. Modbus devices that have implemented “no parity” may have 1 or 2 stop bits.
In most cases, select Modbus Parity = “Even”.

**Stop Bits:**

If Modbus Parity = “None”, then select Modbus Stop Bits = “2”.

6. Check Modbus Device Settings

**Device Address:**
Modbus devices can have addresses between 1 and 247. All Modbus devices must have a different address. Typically, the Modbus addresses of 3rd-party Modbus devices are set and changed via their user interfaces or via their separate configuration software.

**Byte Order:**

The Modbus standard defines the serial bit order to be Least Significant Bit First and Most Significant Bit Last.

However, the Modbus standard does not define the Byte and Word order of messages. Hence, there are Modbus devices that will transmit Most Significant Byte First or Lower Significant Byte First.

For 32-bit values, Most Significant Word First or Lower Significant Word first may be implemented.

The Modbus standard defines the serial bit order to be Least Significant Bit First and Most Significant Bit Last.
However, the Modbus standard does not define the Byte and Word order of messages. Hence, there are Modbus devices that will transmit Most Significant Byte First or Lower Significant Byte First.

For 32-bit values, Most Significant Word First or Lower Significant Word first may be implemented.

<table>
<thead>
<tr>
<th>BigEndian</th>
<th>LittleEndian</th>
<th>LittleEndianWithWordswap</th>
<th>BigEndianWithWordswap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int32, Unit32, Float</td>
<td>Int16, Unit16</td>
<td>Int16, Unit16</td>
<td>Int32, Unit32, Float</td>
</tr>
<tr>
<td>Byte1 Most significant Byte</td>
<td>Byte1 Most significant Byte</td>
<td>Byte1 Most significant Byte</td>
<td>Byte1 Most significant Byte</td>
</tr>
<tr>
<td>Byte2 Second most...</td>
<td>Byte2 Least Significant Byte</td>
<td>Byte2 Least Significant Byte</td>
<td>Byte2 Least Significant Byte</td>
</tr>
<tr>
<td>Byte3 Third most...</td>
<td>Byte3 Most Significant Byte</td>
<td>Byte3 Most Significant Byte</td>
<td>Byte3 Most Significant Byte</td>
</tr>
<tr>
<td>Byte4 Least Significant Byte</td>
<td></td>
<td></td>
<td>Byte4 Third most...</td>
</tr>
</tbody>
</table>

Max Read Points Counts:

![Max Read Points](image)

Via Modbus, the controller can request a single register of a Modbus device and then gets back a single value as response or it requests several registers (start register, number of registers) and gets back several values in one response.

This method results in less data traffic on the bus and thus it accelerates the communication.

The controller recognizes successive registers with the same operating mode and then automatically asks for multiple registers in a message.

Often, however, the manufacturer of the Modbus device has limited the number of registers that can be requested with a single message. This limitation can be entered here. The IRM controller can query up to 20 consecutive registers in a single message.

7. Add Read & Write points

Select Read Point tab and Add
Read Point Name:
User defined name

Read Point Address:
Address range is 0-9998
Against device manufactures Modbus specification an address offset of -1 might be required, because address range starts at 0

The Read Point Address is the data address and NOT the Coil/Register Number of the Modbus device.

Information is stored in the Modbus devices in four different tables. Two tables store on/off discrete values (coils) and two store numerical values (registers). The coils and registers each have a read-only table and read-write table. Each table has 9999 values.

Each coil or contact is 1 bit and assigned a data address between 0000 and 9998. Each register is 1 word = 16 bits = 2 bytes and also has data address between 0000 and 9998.

<table>
<thead>
<tr>
<th>Coil/Register Numbers</th>
<th>Data Addresses</th>
<th>Type</th>
<th>Table Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-9999</td>
<td>0000 to 270E</td>
<td>Read-Write</td>
<td>Discrete Output Coils</td>
</tr>
<tr>
<td>10001-19999</td>
<td>0000 to 270E</td>
<td>Read-Only</td>
<td>Discrete Input Contacts</td>
</tr>
<tr>
<td>30001-39999</td>
<td>0000 to 270E</td>
<td>Read-Only</td>
<td>Analog Input Registers</td>
</tr>
<tr>
<td>40001-49999</td>
<td>0000 to 270E</td>
<td>Read-Write</td>
<td>Analog Output Holding Registers</td>
</tr>
</tbody>
</table>

Example:
The first Holding Register, number 40001, has the Data Address 0000. The difference between these two values is the offset. Each table has a different offset. 1, 10001, 30001 and 40001.

Read Point Type:

Operation Mode (Poll Rate):
This parameter allows adapting to the communication and processing performance of the Modbus device. If the Modbus device does not respond fast enough, increase the poll rate.
DoNotRead - Output is NULL
Low - Register is read after multiple of 5 s one register at a time.
Medium - Register is read after all high priority registers are read, one register at a time.
High - Register is read every 50 ms

NOTE: If no high priority polling rate is assigned to any register, medium priority register will be read every 50 milliseconds one register at a time if present.

Scaling Factor:
Read value is multiplied by 10^Scaling Factor
E.g. Read value=1500. With Scaling Factor -3 the calculated value = 1.5
     Read value=1500. With Scaling Factor +2 the calculated value = 150000

Data Format:

Data Format

SingleBit
Int16 -32768 to 32767
Uint16 0 to 65535
Int32 -2147483648 to 2147483647
Uint32 0 to 4294967295
Float -3.4E+38 to +3.4E+38 (32 Bit)

Event Bit Masking:

Event Bit Masking (Hex)

If the Read Point is read in the Event Program, the value can be masked via the Event Bit Mask. Whenever the masked value changes, the logic of the Event Program Wire Sheet view is executed immediately. Masked value means logical AND with the value received from the Modbus.

Example for a 16 bit value

Occupey Sensor
Int16 mask for Occupancy Sensor 00 02
Binary 0000 0000 0000 0010 = Hex 0002 = Decimal 2
The Event Program Wire Sheet logic is executed periodically. Additionally, it can be executed immediately (event driven), such an event can be triggered by a Modbus value.

Masked Occupancy Sensor: The execution of the Event Program Wire Sheet logic will immediately be executed when the Occupancy Sensor Bit changes.

None masked Occupancy Sensor (0xFFFF): The execution of the Event Program Wire Sheet logic will be executed immediately when the Occupancy Sensor or the Brightness Sensor changes.

Select Write Point tab and Add

ADD MODBUS WRITE POINT

Write Point Name
Write point Address
Write Point Type

Data Format

Significant Change

NOTE: Write Point Address range is 0-9998

Write Point Name:
User defined name

Write Point Address:
For details refer to Read Point Address

Write Point Type:
Write Single Coil
Write Multiple Registers
Data Format:

<table>
<thead>
<tr>
<th>Data Format</th>
<th>SingleBit</th>
<th>Int16</th>
<th>-32768 to 32767</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Uint16</td>
<td>0 to 65535</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Int32</td>
<td>-2147483648 to 2147483647</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uint32</td>
<td>0 to 4294967295</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Float</td>
<td>-3.4E+38 to +3.4E+38 (32 Bit)</td>
</tr>
</tbody>
</table>

Significant Change:
To reduce bus load a significant change for analog values is recommended. This ensures that values are only sent when the value has changed significantly from the value previously sent.

To finish the Modbus configuration press save and then click in the Nav Tree on Periodic/Event program.

Delete a Modbus Device in Wire Sheet View

A complete ModbusDevice function block with configured Read and Write Points can only be deleted via right click Action menu Delete Modbus Device menu entry. The standard Delete function will not work.
Duplicate Modbus Device for further usage

**Save Function Block as a Template**
In the Periodic Program and Event Program Wire Sheet view the right click duplicate function only copies the device without the connected function blocks. To have a complete Modbus Device configuration as a template, please use the **Save As Template** function.

**Procedure**
Right click in the Wire Sheet View on the item you want to reuse and select **Save As Template**

![Wire Sheet View](image)

Select Location where to save template

![File Chooser](image)

Template will be saved in a `.palette` file. Default name will be the name of the marked function block.

If you want to add a function block to a existing template just mark it in the wire sheet click on **Save As Template** and select an already existing `.palette` file and choose **Append**

![File Exists](image)
Insert Function Block saved as a Template

Open .palette file.

Choose the template file

Double click on ModbusDevice in Palette panel shows in template saved function blocks

Now mark in the Wire sheet the components you want to use (Ctrl+A, Ctrl+left Mouse Click or via frame). Copy the selection via Ctrl+C and paste it to your target Wire Sheet (e.g. Periodic Program) via Ctrl+V.
Verify Modbus Communication (Trouble Shooting)

For a first test of the communication, connect only one Modbus device using a short cable (i.e. cable less than 3 meters in length).

Create one Read Point; this Read Point should have a known value which can be verified.

Verify that the T2 and R2 LEDs are flickering (indicating data communication). Use the IRM Engineering Tool to verify that the Modbus data can be read and written. T2 LED indicates sending a request message to a Modbus Device and R2 LED indicates receiving an answer message from the Modbus Device.

Possible error causes could be wrong settings for Baudrate, Parity and Stopp bits.

The function block Modbus Read Point has two outputs:
- **Out** = received value from the Modbus Device
- **Out Cause** = error code. 0 = no error; Further value are listed in the Niagara Guide Help of the Modbus Read Point function block. Out Cause pin slot is not visible by default and needs to be enabled.

Possible error causes:
- Reverse Modbus polarity
- Polling interval defined in Operation Mode too short, (Out Cause = Timeout = 8) Some devices do not support fast polling and therefore might not be able to response in time.
- Read/Write Point Address Offset +1/-1
- Device Setting Byte Order

Once this is working properly, add the next Modbus device.

Reference Datapoints Usage

**General**

If the control and monitoring system contains more than one BACnet device (controller), the devices communicate with one another via the BACnet MS/TP protocol. This enables one controller to read values from other controllers and set values on other controllers.

This data communication is realized via so-called reference input and reference output points which are assigned to physical or value BACnet points.

Assignments can be done manually (manual creation and mapping) or automatically (automatic creation via drag&drop).

For manual creation and mapping, the device instance, object instance and object ID of the physical and value BACnet points are entered manually. For automatic creation via drag&drop, the device instance, object instance and object ID of the physical and value BACnet points of the physical and value BACnet points are carried over automatically.

**NOTE:** The reference input and output function provided by the Niagara IRM Engineering Tool is a proprietary Honeywell BACnet function.
BACnet - MSTP

**Value (Read)**

- **SOURCE**
  - BACnet Device A
  - Physical Point
  - Value Point

- **Mapping**
  - Device Instance
  - Object Instance
  - Object ID
  - (Manually)

- **TARGET**
  - BACnet Device B
  - REFERENCE INPUT

Fig. 4. Data Exchange via Manual Mapping of Reference Input Point to Physical / Value Point

**Value (Write)**

- **SOURCE**
  - BACnet Device B
  - Physical Point
  - Value Point

- **Mapping**
  - Device Instance
  - Object Instance
  - Object ID
  - (Manually)

- **TARGET**
  - BACnet Device A
  - REFERENCE OUTPUT

Fig. 5. Data Exchange via Manual Mapping of Reference Output Point to Physical / Value Point

**Value (Read)**

- **SOURCE**
  - BACnet Device A
  - Physical Input Point

- **Dropping**
  - Device Instance
  - Object Instance
  - Object ID
  - (Automatically)

- **TARGET**
  - BACnet Device B
  - REFERENCE INPUT
  - always*

*physical input always creates reference input
Fig. 6. Data Exchange via Dropping of Physical Input Point

Fig. 7. Data Exchange via Dropping of Physical Output Point or Value Point

Examples: There are two controllers (A, and B) in two different rooms on the BACnet MS/TP bus. Controller A has an outside air temperature sensor connected. The outside air temperature value sensed by controller A should be provided to controller B.

There is a 3rd party BACnet MS/TP wall module connected to an IRM controller which should process values coming from the wall module.

Read / Write Directions

A reference input point (target) is used for reading a value from a physical or value BACnet point (source). A reference output point (source) is used for writing a value to a physical or value point (target).

There are two ways to apply reference points in the application:

- Manual creation and mapping of reference points to BACnet points (A)
- Automatic creation and mapping of reference points via Drag&drop of BACnet points (B)

COV and Polling

The controller containing the reference point subscribes to the mapped physical or value point if COV reporting is possible. If COV reporting is not possible, it polls the mapped physical or value point.

Manual Reference Point Creation and Mapping

The reference input-BACnet point connection is established by manual creation and mapping of the reference input point (target) to the physical or value BACnet point (source).

The reference output connection is established by manual creation mapping of the reference output point (source) to the physical or value BACnet point (target).

In both cases, the following device and object information from the BACnet source or target points must be entered during the mapping:
Automatic Reference Point Creation and Mapping via Drag&Drop

Reference points can be automatically created and mapped to physical or value BACnet points by dragging & dropping the physical or value BACnet points onto the wisheet. In this case, the corresponding device and object information comes with the corresponding BACnet point and must not be entered manually.

Master – Slave Configurations

A typical application scenario for the usage of reference points is a master-slave configuration consisting of one master controller connected to multiple slave controllers and/or 3rd Party BACnet devices. The master incorporates the main control logic, writes values to the slave controllers and/or 3rd Party BACnet devices, and receives values from the slave controllers and/or 3rd Party BACnet devices.

Example:

- The master controller:
  - reads the window contact status [EffWindow] from the slave controller via RefIn
  - reads the room temperature [ExtWmRmTemp] from the external BACnet Wall Module via RefIn
  - writes the occupancy status [EffOccMd] to an external BACnet Wall Module via RefOut

- The slave controller:
  - reads the fan output status [FanStage] of the staged fan (slow, medium, high) from the master controller via RefIn

(See figure next page.)
Fig. 8. Master-Slave Configuration using Reference Points

= Analog Input / Output Point

= Binary Input / Output Point
Manual Creation and Mapping of Reference Points (A)

Example: Two IRM controllers and a wall module are on the MS/TP bus in master-slave configuration:

- CLMERL6N (Master)
- RS4N (Slave)
- External BACnet wall module

- The CLMERL6N master controller:
  - reads the window contact status [EffWindow] from the slave controller via RefIn
  - reads the room temperature [ExtWmRmTemp] from the BACnet Wall Module via RefIn
  - writes the occupancy status [EffOccMd] to an external BACnet Wall Module via RefOut

- The RS4N slave controller:
  - reads the fan output status [FanStage] of the staged fan (slow, medium, high) from the master controller via RefIn

Reference Inputs

Based on the example above, the reference input functions of the CLMERL6N master controller are described in the following.

The CLMERL6N master controller
- reads the window contact status [EffWindow] from the slave controller via RefIn
- reads the room temperature [ExtWmRmTemp] from the BACnet Wall Module via RefIn

NOTE: The read function of the RS4N slave controller is described using the automatic creation via drag & drop (see "Automatic Creation and Mapping of Reference Points via Drag & Drop (B)" section, p. 89).

Procedure

1. Note the device IDs of the CLMERL6N master and RS4N slave controllers, in this case '5003' and '5001' (displayed in the Discovered and Database pane).

2. In the Nav tree, expand the BACnet network and browse to the Points folder of the RS4N slave controller.
3. Double-click the Points folder and discover the points by clicking Discover on the bottom.

4. Sort the discovered points by clicking on the Object Name column.
5. In the point list, scroll to the ‘EffWindow’ BACnet point.
6. Note the Object ID, in this case ‘multiStateValue:8’.
7. In the Nav tree, browse to the CLMERL6N master controller.
8. Expand the IRM Program folder.
9. Double-click the control program folder, Periodic Program or Event Program, to which you want to add the reference input point.
10. In the **honIrmControl** palette, expand BacnetObjects, and then drag&drop the **Refln** BACnet object to the wire sheet.
11. In the Name dialog box, change the name to 'EffWindow', and then click OK.

The reference input is added to the wire sheet.

12. Double-click the symbol to display the property sheet for the reference input.
13. On the Property Sheet, expand **Device Instance**, **Object Type**, and **Object Instance**.

14. Enter the values of the wall module and the 'EffWindow' multistate value point as shown in step 1 and 6.
   - Device Instance: device ID = 5001
   - Object Type: object ID = multistateValue
   - Object Instance: object ID = 8
15. Click Save.

16. The reference input in the CLMERL6N master controller is mapped to the multistate value point 'EffWindow' in the RS4N controller which provides the window contact status.

17. Apply the procedure in the same way for:
   - Reading the room temperature by mapping a reference input from the master controller to the analog value [ExtWmRmTemp] of the external BACnet Wall Module
   - Reading the fan stage by mapping a reference input from the RS4N slave controller to the multistate output [FanStage] of the CLMERL6N master controller.

Alternatively, you can use the "automatic reference point creation and mapping via drag&drop" method which is described in the section
"Automatic Creation and Mapping of Reference Points via Drag&drop (B)", p. 89. This section uses fan stage reading as an example.

18. For writing the occupancy status from the master controller to the external BACnet wall module, please refer to the "Reference Outputs" section, p. 84.

19. If desired, you can change the setting for master sync or other settings.

**Master-Slave Synchronization**

If Master Sync Enabled is true, this property will be synchronized between master and slaves in case a master sync operation is performed. The source device setting will be synced to a target device during master sync operation. If the source device has set the flag to false, then this parameter will not be synced to all target devices.

**Reference Outputs**

**Example:** Two IRM controllers and a wall module are on the MS/TP bus in master-slave configuration:

- CLMERL6N (Master)
- RS4N (Slave)
- External BACnet wall module

In the following procedure, the reference output function of the CLMERL6N master controller is described.

- The CLMERL6N master controller writes the occupancy status [EffOccMd] to an external BACnet wall module via RefOut

**Procedure**

1. Note the device ID of the external BACnet wall module, in this case, e.g. '5008' (displayed in the Discovered and Database pane).
2. In the Nav tree, expand the BACnet network and browse to the Points folder of the wall module.
3. Double-click the Points folder and discover the points by clicking Discover on the bottom.
4. Sort the discovered points by clicking on the Object Name column.
5. In the point list, scroll to the 'EffOccMd' BACnet point.
6. Note the Object ID, in this case, e.g. 'multiStateValue:13'.
7. In the Nav tree, browse to the CLMERL6N master controller.
8. Expand the IRM Program folder.
9. Double-click the control program folder, Periodic Program or Event Program, to which you want to add the reference output point.
10. In the *honIrmControl* palette, expand BacnetObjects, and then drag&drop the *RefOut* BACnet object to the wire sheet.
11. In the Name dialog box, change the name to 'EffOccMd', and then click **OK**.

The reference output is added to the wire sheet.

12. Double-click the symbol to display the property sheet for the reference output.
13. On the Property Sheet, expand **Device Instance**, **Object Type**, and **Object Instance**.

14. Enter the values of the external BACnet wall module and the "EffOccMd" multistate value point as shown in step 1 and 4.
   - **Device Instance**: device ID = 5008
   - **Object Type**: object ID = multistateValue
   - **Object Instance**: object ID = 13

15. Click **Save**.
16. The reference input in the CLMERL6N master controller is mapped to the multistate value point 'EffOccMd' in the wall module.

17. If desired, you can change the setting for master sync or other settings.

**Master-Slave Synchronization**

If Master Sync Enabled is true, this property will be synchronized between master and slaves in case a master sync operation is performed. The source device setting will be synced to a target device during master sync operation. If the source device has set the flag to false, then this parameter will not be synced to all target devices.
Automatic Creation and Mapping of Reference Points via Drag&drop (B)

Niagara allows the automatic creation of reference points when BACnet points are dragged&dropped onto the wiresheet. Thus, you do not have to enter the values for device instance, object type, and object instance of the target controller and point manually, since they come with the BACnet point automatically.

This automatic creation is always executed whenever a BACnet point is dragged&dropped onto the wiresheet of another controller or external BACnet device.

The drop result, that is, which type of reference point, input or output, is to be created, can be defined in the control manager as described in the following.

Drop Settings in Control Manager

When using reference points in the application via drag&drop, the dropping result depends on the pre-setting for drop in the control manager of the device to which the BACnet point is dragged&dropped.

In the control manager, the result for dropping a BACnet output and a BACnet value point can be selected. For both point types, either a reference output or a reference input can be selected as drop result.

NOTE: Input points cannot be overridden by another controller. Hence when dropping an input onto the wiresheet, always a reference input is created and a drop setting for inputs in the control manager is not necessary.

Example:

Two IRM controllers and a wall module are on the MS/TP bus in master-slave configuration:

- CLMERL6N (Master)
- RS4N (Slave)
- External BACnet wall module

- The CLMERL6N master controller:
  - reads the window contact status [EffWindow] from the slave controller via Refin

Example:

Two IRM controllers and a wall module are on the MS/TP bus in master-slave configuration:

- CLMERL6N (Master)
- RS4N (Slave)
- External BACnet wall module

- The CLMERL6N master controller:
  - reads the window contact status [EffWindow] from the slave controller via Refin
– reads the room temperature [ExtWmRmTemp] from the BACnet Wall Module via RefIn
– writes the occupancy status [EffOccMd] to an external BACnet Wall Module via RefOut

• The RS4N slave controller:
  – reads the fan output status [FanStage] of the staged fan (slow, medium, high) from the master controller via RefIn

Based on the example above, the reference input function of the RS4N slave controller is described.

• The RS4N slave controller:
  – reads the fan output status [FanStage] of the staged fan (slow, medium, high) from the master controller via RefIn.

Procedure

1. In the Nav tree, expand the BACnet network and browse to the Points folder of the CLMERL6N master controller.
2. Double-click the Points folder and discover the points by clicking Discover on the bottom.
3. Sort the discovered points by clicking on the Object Name column.
4. In the point list, scroll to the ‘FanStage’ BACnet point.
5. Add the 'FanStage' BACnet point to the database by clicking **Add** at the bottom.

6. Click **OK** in the *Add* dialog box. The point will be added to the *Database* and the *Points* folder.
7. In the Nav tree, browse to the RS4N slave controller.
8. Expand the IRM Program folder.
9. In the control manager of the RS4N slave controller, set the drop option for BACnet output dropping (mandatory for outputs and value points, but not necessary for input points, see previous subsection).
   Based on the example, select 'Create RefInput' in Drop of BACnet Output.
10. Double-click the control program folder, Periodic Program or Event Program, to which you want to add the ‘FanStage’ BACnet point.
11. From the Points folder of the CLMERL6N master controller, drag & drop the 'FanStage' BACnet point to the wiresheet of the RS4N slave controller.

12. In the Name dialog box, click OK.

A reference input point with the name of the dropped BACnet point, in this case 'FanStage', is created.

13. Double-click the symbol of the reference input point to display the property sheet.
14. On the Property Sheet, you can see that the reference input is already mapped to the dropped BACnet multistate output point indicated by the values for device instance, object type, and object instance.

15. If desired, you can change the setting for master sync or other settings.

**Master-Slave Synchronization**

If Master Sync Enabled is true, this property will be synchronized between master and slaves in case a master sync operation is performed. The source device setting will be synced to a target device during master sync operation. If the source device has set the flag to false, then this parameter will not be synced to all target devices.

**Application Management**

**Teaching and Learning**

Synchronization can be performed in two ways:

- **Teaching to Controller**
  - Downloads the changed application in the project to the controller
- **Learning from Controller**
  - Uploads the changed application from the controller in the project

When in teaching mode, application changes of the project can be written to the controller in two ways:
- Immediate
  - changes are written to the controller automatically and are effective immediately
- on demand
  - changes are written to the controller manually and explicitly by the Teach to Controller action.

The explicit usage of the teaching and learning actions and the time when a particular action is used depends on the application status in the project and the connected controller, and the result you want to achieve.

Controllers can:

- be empty (factory delivery, cleared, no application)
- have a history (engineered with application)

**Recommendations**

Please be sure, which result you want to achieve and ask the following:
• Do you want to keep the application in the controller, or in the project?
• Do you want to change the application in the project and teach it to the controller?
• Do you want to learn an application from the controller to the project?

If you want to keep the application in the project, learning the application from the controller should be avoided for controllers with history. In this case, the controller should be cleared before in order to avoid the destruction of the application in the project.

If you want to keep the application in the controller, teaching the application to the controller would not destroy the application. Only the changes are taught.

NOTE: If there are too many changes in the project or in the controller, learning from and teaching to the controller does not work.

NOTE: Software always give support by displaying appropriate messages which describe the current status/problem and how to proceed.

Check Application Status

To check the current Status of the Application Software see the Property Sheet view of the IrmBacnetDevice

Application Software Status:

• Controller is Empty
  Result of the Flush Controller command. Download Application software via Teach command

• Incomplete Application due to failed teach
  Application software was not downloaded correctly and is not running

• Application teach is succeed and running
  Application software was downloaded correctly and is running

Teach To Controller

"Teach to Controller" downloads the changed application of the project to the controller. This can happen in two ways:
  – Immediate
    changes are written to the controller automatically and are effective immediately
on demand changes are written to the controller manually and explicitly by the “Teach to Controller” action.

NOTE: When working in teaching mode 'immediate', no messages display and no changes are indicated graphically when changing the application. The following procedure can be applied at any time when working in teaching mode 'on demand'.

Procedure
1. Right-click on IRM Program in the tree, then click Actions, and then click Teach To Controller in the context menu.

RESULT: The changes are written to the controller. The applications are synchronized. The successful action is displayed via Irm Operations Monitor.

Application Download Failure
Due to various reasons the download of an application can fail. This is shown via message in the Jobs Sidebar. In case of a failed application download, however parts of the application may have already been downloaded to the controller, and such application parts could start and operate the IRM controller. In this case, damage of the controlled equipment may occur.

IMPORTANT!
To avoid damaging the controlled equipment or environment due to partial application download failure, it is strongly recommended to clear the controller and perform the application download again.
Unconditional Application Teaching To All Controllers

Overwriting of an application can be permitted for the complete network

Teach Modified Controller:
- **Deny** Default value. Don’t allow overwriting of an existing Application in the Controller. When trying to teach a different application to a controller an error message is generated which requires to clear the controller.
- **Allow** Allows the unconditional overwriting of an existing application.

Learn From Controller

"Learn From Controller" uploads the current application from the controller in the project. The teaching mode does not care in this case.

**WARNING!**  
When learning from the controller, the application in the project will be deleted.

If the synchronization status is different, a message displays recommending either to clear the project (modifications are discarded) or to teach the controller (modifications in the controller are discarded).

Procedure

1. Right-click on IRM Program in the tree, then click Actions, and then click Learn From Controller in the context menu.
RESULT: The application is to uploaded into the project. The applications are synchronized. The successful action is displayed via **Irm Operations Monitor**.

**Clear Controller**

"Clear Controller" deletes the application in the controller.

**Procedure**

1. Right-click on IRM Program in the tree, then click **Actions**, and then click **Clear Controller** in the context menu.

RESULT: The application is deleted in the controller. The successful action is displayed via **Irm Operations Monitor**.

**Clear Project**

"Clear Project" deletes the application in the project.

**Procedure**
1. Right-click on IRM Program in the tree, then click Actions, and then click Clear Project in the context menu.

RESULT: The application is deleted in the project. The successful action is displayed via Irm Operations Monitor.

Check Hardware Compatibility

The hardware configurations of the used physical controller and the hardware defined for the controller in the application can be checked. Any differences are indicated graphically via yellow warning symbol on the terminals in the wire sheet. In addition, a message displays that lists the affected terminals.

Procedure

1. Right-click on IRM Program in the tree, then click Actions, and then click Check Hw Compatibility in the context menu.

RESULT: In case of incompatibilities, as message box displays that lists the affected terminals:
On the wire sheet, the terminals are marked as "dirty" by a yellow warning symbol.

2. Confirm the message box by clicking OK, and then remove the dirty terminals.
3. Teach the controller.
   
   RESULT: The successful action is displayed via Irm Operations Monitor.

---

Take Snapshot

The current status of an application can be backed up and restored later. This allows restoring a changed application if these changes should be discarded.

Procedure

1. Right-click on IRM Program in the tree, then click Actions, and then click Take Snapshot in the context menu.
RESULT: The application is backed up and the successful action is displayed via Irm Operations Monitor.

---

**Restore Snapshot**

The current status of an application can be backed up and restored later. This allows restoring a changed application if these changes should be discarded.

**Procedure**

1. Right-click on IRM Program in the tree, then click Actions, and then click Restore Snapshot in the context menu.

RESULT: The application is restored in the project and the successful restore is displayed via Irm Operations Monitor.

---

**Swap IRM Program**

**Purpose**

Swapping saves RAM space and reduces processor load and bus traffic. The current state of a swapped device is frozen and saved to an IRM repository on the disk. The swapped device is indicated by a proxy which inherits and shows the minimum information of the device necessary for swapping-in the device. For swapped-out devices, synchronization is no more possible. In order to synchronize swapped-out devices, the devices must be swapped in again.

You can swap single devices one after another or multiple devices in one step.

**IMPORTANT!**

*It is strongly recommended to swap out the application in any case after finishing the engineering in online or offline mode.*

**Procedure (Single Device)**

1. To swap-out the device, expand the device.
2. Right-click on the IRM Program folder in the tree, then click Actions, and then click Swap in the context menu.
RESULT: The device is swapped out. On the Property Sheet the proxy is displayed. In the tree, the IRM program is shown without any subfolders.

The successful swap-out is displayed in the IRM Operations Monitor.

3. To swap-in the device, right-click on the IRM Program folder in the tree, then click Actions, and then click Swap in the context menu.

RESULT: The device is swapped in. In the tree, the IRM program is restored showing all subfolders. The successful swap-in is displayed in IRM Operations Monitor.
Procedure (Multiple Devices)

1. To swap-out the devices, expand the BACnet network.
2. Right-click on the IRM Config folder in the tree, then click Actions, and then click Swap Out in the context menu.

RESULT: The devices are swapped out. The successful swap-out is displayed in the Irm Operations Monitor.

3. To swap-in the devices, right-click on the IRM Config folder in the tree, then click Actions, and then click Swap In in the context menu.

RESULT: The devices are swapped in. The successful swap-in is displayed in Irm Operations Monitor.
The "Master Sync" function is based on a group of multiple devices that must have the same application type. This is called a master sync group. The master sync group is established via the "Clone Application" function which clones the application of one selected device (template) to multiple devices.

Among all devices, one device is defined as the source and the other devices are defined as targets. Then the master sync command allows rolling out the current application (changes) of the source device to all target devices per one step.

The master sync function works in online and offline mode on project level, but not in the devices. Hence, the applications in the project do not have to be synchronous with the applications in the devices when working in teaching mode 'on demand'.

### Fig. 9. Master Sync Group including Source and Target Devices

**NOTE:** Cloning of the application and the source-target definition for the devices can be done independently. That is, the source-target definition of the devices can be done before or after cloning of the device application and vice versa.

To start with cloning the application, see section "Cloning Application", p. 104.

### Cloning Application

Clones the device application based on a selectable template (device) to selectable devices. As result, all devices will receive the same application type which is the basis of a master sync group.

**NOTE:** Even if all devices will have the same application type after cloning, the master sync group is not finally established. This is done before or after the cloning via the >>Master Sync function (see "Applying Master Sync" section, p. 106).

**Example:** The following schematic shows a master sync group consisting of 4 devices with the same application A which has the program name A and application type A. The application is cloned using device 1 as template and the devices 2 through n are selected for receiving this application. They are forming a master group defined by application Type A.
Fig. 10. Cloning Application

Procedure

1. In the Database pane, select all devices which should be included in the clone. In the next step, the application of one selected device will be used as template. All others will receive the application type of the selected template device.

RESULT: The following message box displays.

2. Confirm the message by clicking OK.

RESULT: The Select Template dialog box displays.

3. Select the template (device) from the drop-down listbox.

4. Confirm the message by clicking OK.
RESULT: The selected template will be used for all devices selected in the Database pane. The Irm Application Type column in the Database pane indicates that all devices have now the same application type. The master sync group is formed.

NOTE: Cloning of the device application can be done before or after the source-target definition of the devices and vice versa.

Applying Master Sync

Allows rolling out the current application (changes) of the source device to all target devices per one step. The differences in all target device applications are synchronized with the application of the source device.

NOTES: The source-target definition of the devices can be done before or after cloning of the device application and vice versa.

At least one device must include an application in order establish a master sync group via cloning. If not already done, clone the application as described in the section "Cloning Application", p. 104.

The master sync function can be applied in online and offline mode.

Example:

The following procedure shows an offline engineering example with 6 Irm devices created manually using the standard Niagara New command. The application has been cloned.

Procedure

1. Invoke the Database pane to display the offline devices. In the Irm Application Type column, the same application type is shown for all devices forming the master sync group.

2. Select the device you want to define as the source, and then click Set as Source.
3. Select the devices you want to define as the targets, and then click **Set as Target**.

4. Select the source device, and then click **>>Master Sync**.

RESULT: The *Synchronize differences* dialog box displays.
5. Confirm the message by clicking **OK**.

**RESULT:** The differences in all target device applications are synchronized with the application of the source device.

---

### Excluding Function Block Items from Master Sync

As desired, particular values of function blocks can be excluded from the Master Sync function. Then, when performing the master sync command, the excluded values of the function block will be kept and not affected by the updated application. This applies to the periodic and event programs.

Excluding function block values from master sync updates can be applied in the master device and in the target device(s).

When excluding a function block value in the master device, this particular value will be kept in all target devices.

When excluding a function block value in a target device, only this particular target device will be excluded from the value update.

---

**Fig. 11. "Master Sync Enabled" Configuration and Function Block Value Updates**

**Procedure**

1. Double-click the item in the *Periodic program* or *Event program* to display the Property Sheet.
2. Expand the values area by clicking 📊. The Master Sync Enabled option is set to 'true' by default.

3. To exclude a function block value from master sync updates, set the option to 'false'.

4. Click Save.

Splitting-Off Application

Splits the unique application of the master sync group into a new application and keeps the existing application. For the new application, you can enter a different IRM program name. A new IRM application type is issued automatically by the software. The new application can then be cloned to form the new master sync group.

This function can be applied in order to extend an existing application with new features for the usage in a similar environment, e.g. the application of a small-sized office will be used as basis for creating an application for a mid-sized office.
(See figure next page.)

**Fig. 12. Splitting-Off and Cloning Application**

**Procedure**

1. In the Database pane, select the device you want to split-off, and then click Split-Off Application.
RESULT: The following message box displays.

![Split-off IRM Application type]

2. Confirm the message by clicking OK.

RESULT: The following dialog box displays.

![Name for Program (empty keeps existing n...]

3. Enter a new name for the IRM program.

![Name for Program (empty keeps existing n...]

4. Confirm by clicking OK.

RESULT: In the Database pane, in the Irm Program Name column the defined program name is displayed and in the Irm Application Type column the new application type is displayed.

With the device based on this split-off application type, you can now form a new master sync group by cloning the application and defining the source and targets.
NOTE: Cloning of the device application can be done before or after the source-target definition of the devices and vice versa.

### Controller MAC Address Assignment

#### Automatic MAC Address Assignment

The factory setting of the MAC address of an IRM controller is 0xFF by default. On first power-up, a controller will automatically assign itself a unique MAC address within the range of assignable MAC addresses. The range of assignable MAC addresses is defined by the Min_Mac and Max_Mac settings within the controller. For virgin factory controllers, the default value for Min_Mac is 1 and for Max_Mac it is 32 (= Max_Master setting in Niagara).

NOTE: The Max_Master setting can be extended in Niagara to 127 at maximum and saved into the controller.

Once the controller has found a valid MAC address, it is saved in the controller permanently. From now on and on every power-up or system reset, this MAC address is used for MS/TP communication.

The permanent MAC address changes automatically if any of the following conditions happen:

- The Auto Mac process is re-triggered by Niagara via "You are" command using the MAC address 255 (0xFF)
- There is a conflict caused by devices in the network with the same MAC address.

Reset to factory defaults by pressing the service pin during power-on.

#### Manually Changing MAC Addresses

In some scenarios, you might intend to change the MAC addresses:

**Example:** If a small number of controllers are connected to a network, the Auto MAC function executed in the controller may result in huge gaps of the assigned MAC addresses, e.g. 2, 7, 16, 23 and 31. Such a MS/TP network is not optimized in terms of bandwidth usage.

To optimize the network, you can manually change the MAC addresses of the controllers in Niagara (see also section "Setting MAC Address of Controller", p. 113). Make sure that unique MAC addresses are assigned.

**IMPORTANT!**

Once you have changed all desired MAC addresses for the controllers, it is recommended to change the Max_Master setting of the controllers to the highest MAC address (=Max_Mac) among all controllers. This results in an optimum bus performance.

NOTE: If the Max_Master setting is not set to the Max_Mac value, it will not impact the MS/TP functionality, but only the MS/TP performance.

#### Swapping MAC Addresses

If you want to assign to a controller (A) a MAC address which is already assigned to another controller (B), the following steps must be applied:

1. Re-assign a free MAC address in the range of 1-32 to the controller B which blocks the needed MAC address.
2. Now, assign the freed MAC address of controller B (step 1) to the controller A.
3. Assign the next needed MAC address to the controller B (optional, e.g. in case of particular MAC address requirements).
4. Above steps can be continuously applied for assigning desired MAC addresses to further controllers.
5. Change the Max_Master setting to the highest MAC address (=Max_Mac) among all controllers (see Important Note above).

---

**Setting MAC Address of Controller**

At any point, the automatically or manually assigned MAC address of an IRM controller and the Max_Master setting can be changed in Niagara.

**Procedure**

1. Open the *Property Sheet* of the IRM controller.

2. In *Address*, change the MAC Address as desired.

3. Confirm by clicking *Save*.

4. If the changed address is the highest address on the MS/TP network, change the Max_Master setting as follows.

5. Expand the *Config / Device Object* area.
6. In maxMaster, enter the highest MAC address of all controllers of the network.

7. Confirm by clicking Save.

8. Enter the same number of ‘maxMaster’ in all controllers of the network.

Synchronizing Device Time

Synchronizes the time of the devices with the time either from the work bench PC that is used for device engineering or from the supervisor. This depends from where the function is performed.

Procedure

1. In the Database pane, select TSync.

RESULT: The Synchronize Time dialog box displays.
2. Confirm the message by clicking OK.

RESULT: The time of the devices are set to time of the supervisor.

---

**FIRMWARE DOWLOAD**

Allows updating the firmware in the controller via download.

**Procedure**

1. In the Database pane, select the controller.

2. Click **Download Firmware** button.

RESULT: The *File Chooser* dialog box displays.
3. Navigate to the folder where the firmware file is located, then select the firmware file (.BIN), and then click the Open button.

RESULT: The firmware download process is started as indicated on the top in the Firmware Download progress bar. When the firmware download is finished successfully, it is indicated by the 'Success' message.

NOTE: Caused by the controller reboot after the firmware download the device status changes to {unackedAlarm}. In order to change the device name, this alarm has to be acknowledged.

ALARMING

To use the alarming function, please refer to the CENTRALINE NX BACNET UTILITIES DRIVER User Guide: EN2Z-1020GE51.

PRINTING

To use the printing function, please refer to the ARENA NX / COACH NX Printout User Guide: EN2Z-1022GE51.
MISCELLANEOUS

Value Updates after Device Power Failure

If a device had a power failure and the device is restarting, it will take 3 through 5 minutes until the values will be updated in the wire sheet. To accelerate the value update on the wire sheet, please manually refresh the wire sheet by navigating to another page and returning to the wire sheet.

Hiding Slots

Procedure

1. Right-click the function block icon, of which you want to hide slots.

2. In the context menu, click Pin Slots.

RESULT: The Pin Slots dialog box is displayed.
3. Click the slot you want to be hidden.

RESULT: The pin icon at the selected slot will be removed.

4. Click OK.

RESULT: The function block symbol redispalyes. The hidden slots are removed from display.

NOTE: Any slots manually hidden in the Pin Slots dialog are not stored in the controller. After clearing the project and learning the application back from the controller, the hidden slots will be visible again.
Sylk Wall Module Usage

When using a Sylk or external wall module, only one wall module per controller can be added and its address is fixed to 1.

Procedure

1. Double-click Event Program or Periodic Program in the tree.
2. Open the honIrmControl palette.
3. Scroll down to the Wallmodule group, and then expand it.
4. Select the ‘WmConfigHvacA’ entry and drag&drop it onto the wireshet.

5. In the Name dialog box, change the name if desired and then click OK.

6. Double-click the WmConfigHvacA icon to display the Property Sheet.
7. On the property sheet, select the Sylk wall module type in **Wm Model**.

RESULT: Under **SylkWallmodule**, the fixed Wm Address can be viewed.

For detailed information on configuration of wall modules, please refer the Niagara IRM Application User Guide: EN2B-0416GE51.
Bulk Operations

Changing Datapoint Names

**Purpose**

Quickly edit multiple datapoint names of the same type per bulk procedure.

**Procedure**

1. In the **Nav** tree, expand the Services folder and double-click **ProgramService**.
2. Click the **Find Objects** button.
   The *Bqi Query Builder* dialog box displays.

3. In **In:** click the magnifier search symbol to search for the device.
   The *Choose Root* dialog box displays.

4. Select the device, and then click **OK**.
   The *Bqi Query Builder* dialog box redisplay and shows the selected device name.
5. From the **Of type** drop-down listbox, select 'Custom Type'.

6. From the drop-down listbox added right to the **Of type** drop-down listbox, select 'honIrmControl'.

7. From the next drop-down listbox, added right to the **Of type** drop-down listbox, select the datapoint type, e.g. 'BACnet Boolean Input'.
8. Click **OK**.
9. The datapoints found are listed on the **Batch Editor** pane.

10. On the bottom of the **Batch Editor** pane, click the **Rename** button. The **Rename** dialog box displays.
11. In the Replace field, enter the name you want to use for replacing the current datapoint names, e.g. 'D017' should be added as prefix to the existing datapoint name 'BBI'. If desired, check additional options Match case and/or Match whole word.

12. Click OK. The replace results are displayed in the Batch Editor Results message box.

13. On the wire sheet, the datapoint names are changed accordingly.

14. Teach the application to the controller.
Changing Max Master of Devices

**Purpose**  
Quickly change the max master value of all devices.

**Procedure**

1. In the **Nav** tree, expand the Services folder and double-click **ProgramService**.

2. Click the **Find Objects** button.  
The **Bqi Query Builder** dialog box displays.

3. From the **Of type** drop-down listbox, select 'Custom Type'.
4. From the drop-down listbox added right to the **Of type** drop-down listbox, select 'bacnet'.

5. From the second drop-down listbox, added right to the **Of type** drop-down listbox, select the datapoint type, e.g. 'BACnetDeviceObject'.

   ![Diagram showing BACnet query builder]

This will list down all device objects of all devices below the network.
6. On the bottom of the Batch Editor pane, click Edit Slot button.
7. In the Edit Slot dialog box, select 'maxMaster' from the Property dialog box.
8. In New Value, enter the new max master value, then click OK.

In the Batch Editor Results dialog box, all devices are shown having now the new max master value.
During normal operation of the controller, a short press (< 1 sec) of the Service Button will cause a Service Pin Message (BACnet WhoAmI as a Private Transfer, SerialNo. = 130) to be sent.

If the service pin is pressed and then the controller is switched on (while the service pin is still pressed), a reset to factory delivery is performed. The service button has to be pressed until the green power LED goes out at least twice and is switched on again. The reset results in the following:

- The application is cleared from the controller.
- The MAC address will be set to 0xFF, meaning that the controller will now search for a new mac address. Auto MAC will be automatically triggered after a controller power-up.
- The maxMaster setting will revert to its default value of 32.
- The Max info frames will revert to 10.
- The device instance will revert to its default of 4194302.
- The device name will revert.
- The values of automac min_mac and max_mac will be reset to 1 and maxMaster, respectively.