Contents

Introduction ............................................................................................................. 4

Installation ............................................................................................................ 5

Wiring ...................................................................................................................... 5

Factory Reset ......................................................................................................... 6

RS485 End of Line Resistors ................................................................................. 7

Real Time Clock ..................................................................................................... 9

Configuration ......................................................................................................... 10

Ethernet Connection ............................................................................................ 10

BACnet MSTP to IP Router ................................................................................... 12

Web Browser ........................................................................................................... 12

Sedona Web Browser ........................................................................................... 21

CBMS Engineering Tool ....................................................................................... 22

Date and Time ....................................................................................................... 26

Sedona Histories .................................................................................................. 27

BACnet Time Schedules ..................................................................................... 29

BACnet Trend Logs ............................................................................................... 32

BACnet Alarms ...................................................................................................... 34

Modbus to BACnet Gateway ............................................................................... 37

Introduction .......................................................................................................... 37

Modbus Remote .................................................................................................... 37

Add Points Wizard – Step 1 .................................................................................. 38

Export Slave Profile ............................................................................................... 40

Import Slave Profile ............................................................................................. 40

Tutorial .................................................................................................................... 41

BACnet to Modbus Gateway .................................................................................. 44

Introduction .......................................................................................................... 44

Tutorial ................................................................................................................... 44

Logging ................................................................................................................... 51

Introduction .......................................................................................................... 52

Tutorial ................................................................................................................... 52

Security ................................................................................................................... 53
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firmware Upgrade</td>
<td>55</td>
</tr>
<tr>
<td>Introduction</td>
<td>55</td>
</tr>
<tr>
<td>Tutorial</td>
<td>55</td>
</tr>
<tr>
<td>Creating new kits and components</td>
<td>57</td>
</tr>
<tr>
<td>OBIX Web Service</td>
<td>59</td>
</tr>
</tbody>
</table>
Introduction

The AAC-1000 is a BACnet IP to MSTP router with 2 RS485 ports which can also be configured as a gateway. When configured as a BACnet router it can route messages between its RS-485 (MSTP) port and Ethernet (BACnet/IP) port. When configured as a gateway it can bridge between Modbus and BACnet protocols seamlessly. It is fully programmable via a graphical engineering tool and has support for BACnet Alarms, Schedules and Trendlogs.

There are 3 physical communication ports, 1 Ethernet and 2 RS485 ports on the CBMS gateway. It has drivers for BACnet/IP, BACnet/MSTP, Modbus/RTU and Modbus/TCP. It is powered from a 24 VAC power source and can be mounted on a 35 mm Din rail.

The default application running on the gateway has 1 BACnet/IP port configured as well as 2 BACnet MSTP ports. This document describes the configuration required for operation as a BACnet/IP to BACnet/MSTP router, Modbus to BACnet gateway and BACnet to Modbus gateway.

CBMS Studio has been built using the open source Sedona Framework™.

In a BACnet system, devices can be connected to different physical network such as Ethernet and RS485. In order to communicate across these physical networks, a BACnet router is required and each physical network is given a unique network number in the range 1 to 65535. In addition to the network numbers, each device on the network must be given a unique device ID in the range 0 to 4194303 regardless of the physical network that it is located in. These 3 settings are the bare minimum settings that require to be changed.
Installation

Wiring
The AAC-1000 requires a 24 VAC power supply to be connected in order for it to operate. There are 2 RS485 ports which can be used for either BACnet MSTP or Modbus RTU. The wiring diagram for the AAC-1000 is shown below. If the device is running, the red power LED will turn on to indicate that there is power on the board.

Wiring Diagram
Factory Reset
The AAC-1000 has 2 buttons mounted on the board labelled S1 (Factory Reset) and S2 (Reboot). When the reboot button is pressed the device will reboot immediately. To restore the board back to its factory settings, press and hold button S1 for a period of 5 seconds. This will restore the IP address back to its default of 192.168.1.100. The username will be restored to admin and the password empty. The application files app.sab and kits.scode will be restored back the factory version.

The buttons S1 and S2 are located in the middle of the board as shown by the diagram below.
RS485 End of Line Resistors
The RS485 ports have End of Line resistors which can be used if required by the installation, there are 2 dip switches located next to the RS485 connectors which will be off by default.

The rule of thumb for termination is:

- If the propagation delay of the data line is much less than one bit width, termination is not needed.

This assumes reflections will damp out in several trips back and forth on the data line.

The calculations below show that termination is not normally required at slower serial speeds, e.g. 9600bps.

Here’s how to calculate the propagation delay:

- Propagation delay = length of cable / propagation velocity

Propagation velocity is typically 66 to 75% of the speed of light (c). Here is an example calculation for a 2000 ft data line at 9600bps with a propagation velocity of 0.66 c:

- 2000 ft of transmission line for a round trip of 4000 ft.
- $c = 186,000 \text{ miles/s} \times 5280 \text{ ft in a mile or } 982,080,000 \text{ ft/s}.$
- Propagation velocity = $0.66 \times 982,080,000$ or $648,172,800 \text{ ft/s}.$
- Propagation time = $4000 \text{ ft} / 648,172,800 \text{ ft/s}$ or about $6.2 \text{ µs}.$

If the reflections damp out in three round trips on the cable the signal will stabilize in about $18.6 \text{ µs}.$ At 9600 bps one bit is $104 \text{ µs}$ wide. The bit will be sampled at the center of the bit (after $52 \text{ µs}$) so in this case termination is not required as the signal will have stabilized after about $20 \text{ µs}.$

So for cables 2000ft or less termination is not needed at 9600bps. Since termination adds to the complexity of the installation and can potentially load down RS485 devices, Lantronix recommends only using termination if it’s required.

If termination is required set the dip switch to on, and no more than 2 termination resistors should be used, one at each end of the RS485 transmission line.
**Real Time Clock**
The AAC-1000 comes with a battery backed real time clock. When there is no power on the board the real time clock will keep it’s time using the power supplied from the battery. After a prolonged period of time the battery may need to be changed, it is located in the position shown by the diagram below.
Configuration
The configuration of the router can be done with a Web Browser, Engineering Tool or the Niagara Workstation. This manual will describe the configuration using both a web browser and the Engineering Tool from CBMS Studio.

Ethernet Connection
When the device is shipped it is configured with a static Ethernet address of 192.168.1.100. If your PC is already connected to the Sub domain 192.168.1.x then you can connect the AAC-1000 directly to this network and skip the following instructions for connecting an Ethernet cable directly to the AAC-1000.

If you are using a different subnet then you can connect an Ethernet cable from your PC directly to the Ethernet port of the AAC-1000. Once you have connected the 24 VAC power and Ethernet cable, go to your PC settings and change your IP address to a static address of 192.168.1.1.

- For Windows 8, right click on Windows Icon at the lower left of your screen and select “Network Connections”
- From the Network Connections dialog, find your Ethernet adapter right click and then select “Properties”
- This will display the “Ethernet Properties” dialog box as shown below.
Select Internet Protocol Version 4 and then select “Properties” to display the following dialog box.

![Internet Protocol Version 4 (TCP/IPv4) Properties](image)

Change the IP address to use a static IP address of 192.168.1.1

After these changes have been made you will be able to use an Ethernet cable connected directly to the AAC-1000 and connect to it from a web browser or the Engineering Tool.
BACnet MSTP to IP Router

Web Browser

Using the web browser on your PC, open up the AAC-1000 web configuration screen with the IP address of 192.168.1.100

Click the “Settings” menu tab to bring up the login dialog and enter your username and password. The default username is admin, the password is empty.

If the login is successful you will be taken to the general settings page of the AAC-1000 where you can change the basic settings of the BACnet IP to MSTP router.
General Settings
The basic settings of the BACnet IP to MSTP router can be changed from this page. For most systems, these are the only settings that will need to change.

The following settings can be changed from the general settings page. The save button must be selected after changing any of the values in order to save the settings to the device.

- **Device ID.** This is the BACnet device ID corresponding to the AAC-1000, it must be changed to a unique number on the BACnet network. The default value is 9999.
- **BACnet IP Enable.** This should be left as enabled and it should only be deactivated if the BACnet IP to MSTP routing is not required.
- **BACnet IP network number (1-65535).** This value represents the BACnet IP network number for all devices connected to the IP network. All devices on the IP network should use the same network number. This setting defaults to 1 and can be left unchanged on most systems.
- **BACnet IP Port number.** The default setting is 47808 (0xBAC0) and can be left unchanged on most systems.

The following settings are common to both MSTP networks.

- **MSTP Enable.** When the port is enabled, the AAC-1000 will act as a BACnet IP to MSTP router on this port. If the MSTP port is not used then the port should be disabled.
- **MSTP network number (1-65535).** This value represents the BACnet MSTP network number for all devices connected to this MSTP port. This setting defaults to 11 and each MSTP port
requires a unique network number. If there are multiple AAC-1000 devices used, then the network number on each MSTP port will need to be changed with a unique number.

- **MSTP Baud Rate.** The AAC-1000 baud rate does not auto select and the value used should correspond to the value required by the MSTP devices.

- **MSTP MAC address.** Each MSTP device requires a unique MAC address in the range 0-127. It is common for the MSTP router to have a MAC address of zero, so this value can be left unchanged for most systems.

**IP Address**

The IP Address settings can be changed on this page. After changing any of these settings a reboot will be required before those changes take effect.

The following settings are available on this page.

- **IP Address.** This is the static IP address used by the AAC-1000

- **Netmask.** The netmask is used by the AAC-1000 when broadcasting messages on the network. For a class A network, this will be set to 255.255.255.0. If in doubt please consult your network administrator.

- **Default Gateway.** This setting is used by the AAC-1000 when it needs to connect to the internet. In most systems an internet connection is not required and this setting is not used.

- **Broadcast Address.** This setting is read only and is derived from the IP address and netmask.
Device

From this page, the BACnet device settings for the AAC-1000 can be configured. These are the values that are displayed on the BACnet device object and will be readable from a BACnet client.

- **Device ID.** This is the BACnet device ID corresponding to the AAC-1000, it must be changed to a unique number on the BACnet network. The default value is 9999.
- **Object Name.** This property, of type CharacterString, shall represent a name for the object that is unique internetwork-wide. The minimum length of the string shall be one character. The set of characters used in the Object_Name shall be restricted to printable characters.
- **Description.** This property, of type CharacterString, is a string of printable characters that may be used to describe the application being carried out by the BACnet Device or other locally desired descriptive information.
- **Location.** This property, of type CharacterString, indicates the physical location of the BACnet Device.
- **Application Software Version.** This property, of type CharacterString, identifies the version of application software installed in the machine. The content of this string is a local matter, but it could be a date-and-time stamp, a programmer’s name, a host file version number.
- **APDU Timeout.** This property, of type Unsigned, shall indicate the amount of time in milliseconds between retransmissions of an APDU requiring acknowledgment for which no acknowledgment has been received. A suggested value for this property is 10,000 milliseconds for devices that permit modification of this parameter. Otherwise, the default value shall be 60,000 milliseconds. This value shall be non-zero if the Device object property called Number_Of_APDU_Retries is non-zero. In order to achieve reliable communication, it is recommended that the values of the APDU_Timeout properties of the Device objects of all intercommunicating devices should contain the same value.
- **APDU Segment Timeout.** This property, of type Unsigned, shall indicate the amount of time in milliseconds between retransmission of an APDU segment. A suggested value for this property is 5000 milliseconds. This value shall be non-zero if the Device object property called Number_Of_APDU_Retries is non-zero. If segmentation of any kind is supported, then the APDU_Segment_Timeout property shall be present. In order to achieve reliable communication, it is recommended that the values of the APDU_Segment_Timeout properties of the Device objects of all intercommunicating devices should contain the same value.

- **Number of APDU Retries.** This property, of type Unsigned, shall indicate the maximum number of times that an APDU shall be retransmitted. A suggested value for this property is. If this device does not perform retries, then this property shall be set to zero. If the value of this property is greater than zero, a non-zero value shall be placed in the Device object APDU_Timeout property.

- **Max Segments Accepted.** This property, of type Unsigned, shall indicate the maximum number of segments of an APDU that this device will accept.

**BACnet IP**

From the BACnet IP page, the settings of the BACnet IP driver can be changed.

- **Enable.** This should be left as enabled and it should only be deactivated if the BACnet IP to MSTP routing is not required.
- **Status.** This is a read only value that displays the status of the port. If it is working correctly then it will display the text “Alive”.
- **BACnet IP network number (1-65535).** This value represents the BACnet IP network number for all devices connected to the IP network. All devices on the IP network should use the same network number. This setting defaults to 1 and can be left unchanged on most systems.
- **BACnet IP Port number.** The default setting is 47808 (0xBAC0) and can be left unchanged on most systems. **Status**
- **Max Qty Foreign Devices.** If the router is operating as a BBMD, then this value is used to specify the maximum number of foreign device connections that the AAC-1000 will accept.
- **Error message.** The error message for the BACnet IP port if there is an error condition.
- **NIC Adapter.** The NIC adapter used by the BACnet IP port.
- **MAC Address.** The BACnet MAC Address associated with this port.
- **MAX Apdu Length Accepted.** A read only value indicating the maximum APDU length accepted by this port.
- **Packets Sent.** The number of IP packets sent.
- **Packets Received.** The number of IP packets received.
- **Select Error Count.** The number of select errors for this port.
- **Receive Error Count.** The number of receive errors for this port.
- **Error Count.** The total number of errors for this port.
**BBMD**

The AAC-1000 can operate as a BBMD, to allow it to connect 2 different IP subnets together. When there are 2 subnets, then 1 AAC-1000 on each subnet should be configured as a BBMD and both BBMD’s should have 2 entries containing for the IP address of each BBMD.

A maximum of 5 entries can be added to the Broadcast Distribution Table (BDT) of the AAC-1000, the settings for each entry are :-

- **BBMD Address.** This is the IP address of the BBMD which will receive broadcast messages.
- **BBMD Port.** The port number of the BBMD.
- **BBMD Distribution mask.** Normally set to 255.255.255.255, but it can be used to filter broadcast messages to the BBMD.

Note : Each BBMD should contain the same number of entries in it’s Broadcast Distribution Table.
### MSTP Port

There are 2 MSTP ports available on the AAC-1000 and there is a separate web configuration screen for each port as shown below.

- **Enable.** This should be left as enabled and it should be deactivated if the BACnet IP to MSTP routing is not required.
- **Status.** This is a read only value that displays the status of the port. If it is working correctly then it will display the text “Alive”.
- **Error Message.** This is normally empty, but if there is an error then it will display a message.
- **MSTP network number (1-65535).** This value represents the BACnet MSTP network number for all devices connected to this MSTP port. This setting defaults to 11 and each MSTP port requires a unique network number. If there are multiple AAC-1000 devices used, then the network number on each MSTP port will need to be changed with a unique number.
- **Port Number.** This should not be changed, it represents the internal com port number used by the AAC-1000 and will be either 2 or 4.
- **Baud Rate.** The AAC-1000 baud rate does not auto select and the value used should correspond to the value required by the MSTP devices.
- **Data Bits.** This value should always be set to 8 bits.
- **Stop Bits.** This value should always be set to 1.
- Parity. This value should always be set to none.
- MAC address. Each MSTP device requires a unique MAC address in the range 0-127. It is common for the MSTP router to have a MAC address of zero, so this value can be left unchanged for most systems.
- Max Info Frames. The default is 50 and it represents the value of the Max_Info_Frames property of the node's Device object. The value of Max_Info_Frames specifies the maximum number of information frames the node may send before it must pass the token. Max_Info_Frames may have different values on different nodes. This may be used to allocate more or less of the available link bandwidth to particular nodes. If Max_Info_Frames is not writable in a node, its value shall be 1.
- Max Master. The default is 100 and it represents the value of the Max_Master property of the node's Device object. The value of Max_Master specifies the highest allowable address for master nodes. The value of Max_Master shall be less than or equal to 127. If Max_Master is not writable in a node, its value shall be 127.
- MAX Apdu Length Accepted. A read only value indicating the maximum APDU length accepted by this port.
- Sole Master. MAX Apdu Length Accepted. A read only value indicating the maximum APDU length accepted by this port. If this value is true, then there are no other MSTP master devices connected to this port.
- Silence Timer. A timer with nominal 5 millisecond resolution used to measure and generate silence on the medium between octets. It is incremented by a timer process and is cleared by the Receive State Machine when activity is detected and by the SendFrame procedure as each octet is transmitted. Since the timer resolution is limited and the timer is not necessarily synchronized to other machine events, a timer value of N will actually denote intervals between N-1 and N. When this value continues to rise then it is an indication that there is a communication failure or no devices connected.
- Token count. The number of tokens received by this node. When this counter reaches the value Npoll, the node polls the address range between TS and NS for additional master nodes. TokenCount is set to one at the end of the polling process.
- Next Station. Is the MAC address of the node to which This Station passes the token. If the Next Station is unknown, NS shall be equal to TS.
- Poll Station. Is the MAC address of the node to which This Station last sent a Poll For Master. This is used during token maintenance.
- This Station. Is the the MAC address of this node. TS is generally read from a hardware DIP switch, or from nonvolatile memory. Valid values for TS are 0 to 254. The value 255 is used to denote broadcast when used as a destination address but is not allowed as a value for TS.
- Event Count. Used to count the number of received octets or errors. This is used in the detection of link activity.
- Frame Count. The number of frames sent by this node during a single token hold. When this counter reaches the value Nmax_info_frames, the node must pass the token.
- Bad Crc Count. Number of CRC errors.
- Invalid Frame Count. Number of invalid frames.
- Received Valid Frame Count. Number of valid frames received.
- Retry Count. A counter of transmission retries used for Token and Poll For Master transmission.

**Sedona Web Browser**

From a web browser it is possible to view the Sedona Components in real time and write to any of the configurable properties. The Sedona Browser works in much the same way as the Sedona Workbench does. The components are all list in the tree on the left and navigation using the tree will display any component within the device. The screenshot below displays the BACnet local device component within the Sedona application.

![Sedona Web Browser Screenshot](image)

Read only properties cannot be changed, however configurable properties can be changed from the web browser. After changing the value the save button must be pressed to store the value in the device.
CBMS Engineering Tool

Connect to the device

Using the Engineering from CBMS Studio, connect to the device using a Sedona connection with the username and password set on the device. After the connection has been made, select the App node.
**Configure BACnet Device Id**

Once connected, navigate to the component labelled local under the path app/drivers/bacnet. This will display the cbmsBACnet::LocalDevice component and within the form there is a property called device Id. By default this will be set to 9999 and it should be changed on site to a unique number across all networks. For example, if 10 BACnet routers are installed, then each router must be given a different device Id.

To change the value of the device ID, simply type a new value into the text box located next to the label for the device Id and then select the save button at the lower right hand side of the engineering tool. This will write the value to the application.

After updating any value it is held in memory by the device. To make it persistent so that it is loaded again after a restart, the application itself requires saving. To do this press the “save” button in the toolbar.
Configure BACnet IP Network Number

Navigate to the component labelled ip under the path app/drivers/bacnet. This will display the cbmsBACnet::PortIp component and within the form there is a property called network. By default this will be set to 1. This represents the network number for the Ethernet network and all devices physically connected to the Ethernet network should have their network number set to the same value. For example, if there are 10 routers connected to the Ethernet network, then each router should have the same IP network number assigned, eg 1.
Configure BACnet MSTP Network Number

There are 2 MSTP ports configured by default on the BACnet router called mstp1 and mstp2.

Navigate to the component labelled ip under the path app/drivers/mstp1. This will display the cbmsBACnet::PortMSTP component and within the form there is a property called network. By default this will be set to 11 and mstp2 will be set to 21. This represents the network number for all of the devices physically connected to the RS485 port.

Network numbers need to be unique, so if 2 routers are installed the router 1 and router 2 cannot use network 21 for the MSTP network.

If the MSTP port is not in use, the Enable property should be set to disabled. This will turn off routing for the RS485 port.
Date and Time

The AAC-1000 has a battery backed real time clock which will keep track of time when the power has been disconnected. To set the time, date and time zone open the Engineering Tool, connect to the AAC-1000 and navigate to Sedona-app-service-date.

From this page, the time zone should be changed to match the time zone that you are in. This is required when the Time Schedules, Trend logs or Alarms are used because the time used is local time.

The Set Local Time button will synchronize the time on the device with the time on your PC.
Sedona Histories

Sedona Histories are a simple way to take timed samples of any component that derives from a Boolean or Numeric point object. Most components derive from these objects making it very easy to add a new history object, for example a Modbus floating point added to the modbus driver can be extended to store historical data. Sedona Histories are not available as BACnet Trendlogs which are configured separately in the next section.

To add a Sedona History, navigate to the component you wish to add the history to and change the property called History to Interval. Once changed, the history will be created and sampling will start.

In the example below, BACnet remote points have been added which are reading from another device on the network. The History property has been set to interval and sampling is active.
To view the history, navigate to the folder `app-sedona-service-history` and select the history object with same path name as the one that was created. The sample interval is set to 15 and can be changed from the property sheet view. It may take some time for samples to be generated, but after some time the result will be a graph of the value over time.

The toolbar contains options for displaying the data in tabular view, graphical view. There is an option to download the data to a csv file.
BACnet Time Schedules

The AAC-1000 supports the BACnet Schedule object which can be used to schedule the operation of equipment based on the time of day. The output of the schedule object can be wired into other logic components from the wiresheet. To create a new time schedule, navigate to Sedona-app-drivers-bacnet-local and select the Add Points button to bring up step 1 of the Add Points Wizard as shown below.

From the ObjectType pull down list, change the type to Schedule and change the starting instance and quantity to the desired values.

Click the add button to populate the list of schedules to be added to the device. Once you have finished adding schedules to the list, click the next button and the schedules will be saved to the device.
Navigate to app-drivers-local-points-schedule-schedule 1 where you can view the weekly schedule. New weekly schedules can be added from this page by clicking on the grid. In the screenshot below a switch time has been added to Sunday which will activate the schedule during those times.

The present value of the Time Schedule is an output which can be wired into a component as an input. For example, we have created a Binary Output on the Raspberry Pi linked to GPIO pin 4. To activate the GPIO pin from the timeschedule, navigate to the wiresheet that contains to component you want to link to and then click the "Make Link" tool bar icon.

This will display the Make Link dialog box from which wil can wire the Out slot from the Time Schedule to one of the In slots on the Output.
After adding the link a circle will be displayed to the slot to indicate that the link is connected to a component on another wiresheet. In this case it is In10 which is connected to the timeschedule out slot and because the timeschedule is active, the Binary Output is active which turns the GPIO on.
BACnet Trend Logs
The AAC-1000 supports the BACnet Trendlog object which can be used to store real time values at timed intervals. To create a new trendlog, navigate to Sedona-app-drivers-bacnet-local and select the Add Points button to bring up step 1 of the Add Points Wizard as shown below.

From the ObjectType pull down list, change the type to Trendlog and select an object from the list of available objects. There must be an available BACnet Object of type AI,AO,AV,BI,BO or BV inside the device which will be referenced by the Trendlog Object.

Change the starting instance and quantity to the desired values and select a log interval and buffer size.

Click the add button to populate the list of trend logs to be added to the device. Once you have finished adding schedules to the list, click the next button and the objects will be saved to the device.
Navigate to app-drivers-local-points-trendlog-trendlog 1 where you can view the trendlog. The data will update when new samples are taken which is set by the log interval property of the BACnet trendlog. From the toolbar the data can be exported to CSV or displayed in a table.

To change the BACnet Properties for the trendlog, click on the "BACnet Object" tab which will bring up a list of properties associated with the trendlog. From this screen the buffer size, log interval, etc can all be changed.
**BACnet Alarms**

The AAC-1000 supports the BACnet Event Enrolment object which can be used to raise an BACnet Alarm. The Event Enrolment object references another BACnet object, checks it’s values and if an alarm condition exists to will send it to the recipients in the Notification Class object. The add points wizard will create both the event Enrolment object and the notification class object.

To create a new event enrolment, navigate to Sedona-app-drivers-bacnet-local and select the Add Points button to bring up step 1 of the Add Points Wizard as shown below.

![Add Points Wizard Step 1](image)

From the ObjectType pull down list, change the type to event enrolment and select an object from the list of available objects. There must be an available BACnet Object of type AI,AO,AV,BI,BO or BV inside the device which will be referenced by the event enrolment object.

Change the alarm settings if required and then click the add button to populate the list of objects to be added to the device. Once you have finished adding objects to the list, click the next button and the objects will be saved to the device.
To change the BACnet Properties for the EventEnrolment, navigate to app-drivers-bacnet-local-eventenrolment-event enrolment 1 and click on the "BACnet Object" tab which will bring up a list of properties associated with the object. From this screen the time delay, etc can all be changed.
A Notification class will be added if one does not already exist. The notification class is used to determine where an alarm will be sent. There must be at least one notification class on the device and each alarm generating object will specify which notification class object it will use.

To change the BACnet Properties for the Notification Class, navigate to `app/drivers/bacnet/local/notificationclass/notificationclass1` and click on the "BACnet Object" tab which will bring up a list of properties associated with the object. From this screen the recipient list, etc can all be changed.
Modbus to BACnet Gateway

Introduction

A Modbus to BACnet gateway will read and write data from one or more Modbus slaves and make it available as BACnet Objects. The AAC-1000 has drivers for Modbus TCP/RTU as well as BACnet IP and MSTP and it can be configured to operate as a Modbus to BACnet Gateway.

Any of the following combinations between Modbus and BACnet can be configured.

- Modbus TCP Slave to BACnet IP
- Modbus TCP Slave to BACnet MSTP
- Modbus RTU slave to BACnet IP
- Modbus TCP slave to BACnet MSTP

The BACnet driver supports BACnet virtual devices making it possible to map a Modbus Slave to BACnet Virtual device. This is useful when there are several Modbus Slaves and each slave is mapped into a separate BACnet device. Mapping into separate devices will allow the same numbering convention to be used inside all of the BACnet devices.

Modbus Remote

The ModbusRemote folder is located at Sedona-app-drivers-modbus-remote and it contains the ports, slaves and registers which are used to determine what will be updated by the driver. This folder represents Modbus Slaves which are external or remote to the Gateway. By default there will be no items in the folder, they will be created when points are added to the device using the Add Points Wizard.
Add Points Wizard – Step 1
To start adding points, navigate to the folder Sedona-app-drivers-modbus-remote and select Add Points. The Add Modbus Remote Points wizard will be displayed, and from there select a Modbus Port and Slave device. If the Network dropdown list is empty, then a new network needs to be added. Click on the Add button to add a new Modbus TCP, or RTU network and then click ok to add the new port, a slave will automatically be created at this point.

Network
The network dropdown box contains a list of currently configured network ports. Each port can be either TCP or RTU and the port acts as a master to connect to one or more slave devices. The Add button to the right of the network combo box is used to add a new TCP or RTU port to the gateway. Once a new port is added it will appear in the dropdown box.

Slave Id
The Slave Id contains a list of slaves currently configure for the selected port. Each network can have up to 16 slaves. The Add button to the right of the slave id combo box is used to add a slave to the network. Once a new slave is added it will appear in the dropdown box.
Component Type
When a modbus register is added to the gateway it must be assigned to one of the following component types.

- BooleanPoint – A read only binary value.
- BooleanWritable – A writable binary value.
- NumericaPoint – A read only numeric value.
- NumericWritable – A writable numeric value.

Each component type may have additional parameters associated with it which may need to be specified.

Starting Address
This will correspond to the address of the first register to be added to the slave when the Add button is pressed.

Quantity
This is the quantity of components that will be added when the Add button is pressed. The address is incremented by 1 or 2 depending on the number of registers that a component uses. For example a floating point number requires 2 modbus registers.

BooleanPoint - Data Type
When BooleanPoint is selected as the component type then the Data Type dropdown list will be populated with the following types.

- Input Contact – Often referred to as modbus function code 02.
- Output Coil – Often referred to as modbus function code 01.
- Register Bit – A single bit within an input register often refer to as modbus function code 04.

BooleanPoint – Register Bit
When the Register Bit is selected there will be an option for selecting the first bit and last bit. For each of the bits in the range a BooleanPoint will be added to the slave.

BooleanWritable - Data Type
When BooleanWritable is selected as the component type then the Data Type dropdown will contain only 1 option for an Output Coil, referred to in modbus as function code 01. When writing to the Output Coil function code 5 is used.

BooleanWritable – Write Priority
The write priority dropdown is used when a BACnet extension point is added to this component. The out slot of the modbus BooleanWritable component will be used to write to the BACnet priority array at the priority level specified.
**BooleanWritable – Restore Last In value**
The Restore Last In Value field will be used to retain the last write value after a power failure and write the value to the modbus slave after power has been restored.

**NumericPoint – Data Type**
When NumericPoint is selected as the component type then the Data Type dropdown list will be populated with the following types.

- Float – 32 bit value which requires 2 modbus 16 bit registers.
- Short Integer – 16 bit integer value from a single register.
- Long Integer – 32 bit integer value which requires 2 modbus 16 bit registers.

**NumericPoint – Register Type**
The register type can be either an input register (modbus function code 04) or it can be a holding register (modbus function code 03)

**NumericPoint – Byte Order**
The byte order determines the order in which the gateway will decode the bytes and convert it to either a float or long value. Each of these data types are 4 bytes in size and are read from 2 consecutive modbus registers. The following combinations cover all of the possible byte orders.

- ABCD
- CDAB
- BADC
- DABC

**NumericPoint – Byte Order Ascending**
The byte order for a short integer can be either ascending or descending.

**NumericPoint – Signed Integer**
The Short Integer and Long Integer types can be stored in the modbus device as either unsigned or signed values.

**NumericPoint – Bit Mask**
The Bit Mask field can be used to mask an integer value. The gateway will simply and the value with the bit mask and store the resultant value. This may be required there are less than 16 bits associated with the integer value. Masking the value will remove the masked bits from the value.

**Export Slave Profile**
The point configuration for a slave can be saved to a CSV file using this option. The CSV file can be edited on Excel or saved as a template for future use.

**Import Slave Profile**
A previously saved CSV file can be used as a template for creating the points for a new slave.
Tutorial
To start adding points, navigate to the folder Sedona-app-drivers-modbus-remote and select Add Points. The Add Modbus Remote Points wizard will be displayed, and from there select a Modbus Port and Slave device. If the Network dropdown list is empty, then a new network needs to be added. Click on the Add button to add a new Modbus TCP, or RTU network and then click ok to add the new port, a slave will automatically be created at this point.

Step 3
Modbus points are first mapped to Sedona components of type, NumericPoint, NumericWritable, BooleanPoint or BooleanWritable. The modbus address, data type, register type and byte order needs to be selected for each modbus point that needs to be added. In this example, we have selected 1 component of each type and added it to the list. Once all of the points have been added, select the Next button to continue.
Step 4
You can review your selection from the Wizard Step 2 page and then select the Next button to continue.

Step 5
To map the modbus points to BACnet points we create a BACnet extension for each of the modbus components. Select the radio button for BACnet and then select the device for adding the points to. Virtual devices are supported if you want to create multiple BACnet devices, but for this demonstration we will add them to the local device. You can also select a starting instance number for the BACnet objects. Click next when you have finished making your selections.
Step 6
Review your selection from the Wizard Step 4 page and if everything looks correct select the Next button to continue.

Step 7
The Modbus components will appear under the Modbus Remote Slave device as shown by the screenshot on the right.

While the corresponding BACnet object will be visible under the BACnet local device. The BACnet points can be viewed from any BACnet client connected to the network.
BACnet to Modbus Gateway

Introduction
A BACnet to Modbus gateway will read and write data from one or BACnet Devices and make them available to a Modbus Master as registers within a Modbus Slave. The BACnet driver supports reading from any BACnet device on the network.

The AAC-1000 has drivers for Modbus Master TCP/RTU as well as BACnet IP and MSTP and it can be configured to operate as a Modbus to BACnet Gateway.

Any of the following combinations between Modbus and BACnet can be configured.

- BACnet IP to Modbus RTU Slave
- BACnet MSTP to Modbus TCP Slave
- BACnet IP to Modbus RTU Slave
- BACnet MSTP to Modbus TCP Slave

Tutorial

Step 1
Initiate a connection to the AAC-1000 from the Engineering Tool. Available devices are listed in the tree of the engineering tool by their IP address or computer name. If the device is not in the tree then select File - Open to initiate a connection.
**Step 2**
The BACnet devices and objects can be automatically discovered which reduces the effort required to engineer the gateway. Navigate to the bacnet remote folder and then select the button called Scan Devices. This will display step 1 of the device scan wizard as shown by the figure on the right. The low and high limits can be changed to reduce the result set, and then press the Next button to continue.

**Step 3**
The network will be scanned for any device matching the high and low limit settings, and the Next button will activate when the device scan has finished. If there are no results, check the port and local device settings to make sure that the BACnet driver is running. Existing devices are highlighted in orange. Select the devices that you would like to add to the remote folder and then click the Next button.
**Step 4**
The object list will be read from each selected device and the names of every object will be read. The object list and names will not be read again, if the device configuration changes then another device scan will need to be executed manually. When the wizard completes, the remote folder will be populated with a list of networks and devices.

![Device Scan](image)

**Step 5**
After the device scan has completed and the entire list of devices and objects have been created, the BACnet points which are to be mapped to Modbus points can be added to the device.

Select Add Points and select the network, device or individual points that you would like to create mappings for. By default mappings will be created for the present value property of Analog, Binary and Multistate points. Press next when done.

![Add Points](image)
Step 6

Additional filtering based on the object name can be applied on this page by matching the text or using a regular expression.

The component type to which it is mapped can be changed from this page. For example, if an Analog Output object is to be used as a read only value then it can be mapped to a NumericPoint component.

Press Next to continue.

---

Step 7

The entire list of points that are to be mapped will be displayed on this page. By default all points will be selected, if you wish to exclude some points then you can change the selected items manually.

Press Next to continue.
**Step 8**  
Change the extension type to Modbus and then select the Add button next to network combo box. You can then select a Modbus TCP or RTU driver, press Ok to complete the selection. This will create both the Modbus Driver and a Slave.

Press Next to continue.

**Step 9**  
The entire list of points and mappings to Modbus will be displayed. If everything looks correct then the next button can be pressed and all of the points will be created.

Press Next to continue.
The BACnet components will be displayed in the points folder beneath the remote device. The out property contains the value which is being read from the presentValue of the BACnet Analog, Binary or Multistate point.

Each of the components in the points folder will have a child component corresponding to the Modbus point to which it is mapped. In the example shown on the right, BACnet Binary Input 1 is mapped to Modbus Discrete Input contact address 0.
The modbus points can also be viewed from the points folder beneath the Modbus local slave as shown by the image on the right.
Logging
Logging of system generated messages can assist in trouble shooting the device during commissioning. To view the log files and change the log levels, navigate to the folder Sedona-app-service-log. From there a list of log levels is displayed and can be changed as required.

Logging should be reset to None after commissioning has finished.

The log files can be displayed by selecting the Log Files tab and the files can be downloaded by pressing the Download Files button.
Import/Export the Sedona application

Introduction
The AAC-1000 application can be backed up using the Engineering Tool and restored at a later stage if required using the Import/Export utilities.

There are 2 types of file that reside in the device that can be exported or imported.

- The app.sab file contains a list of all the components that reside in the device.
- The kits.scode file is the file that defines the functions that a component can perform.

Tutorial

Connect to device.
Initiate a connection to the AAC-1000 from the Engineering Tool. Available devices are listed in the tree of the engineering tool by their IP address or computer name. If the device is not in the tree then select File - Open to initiate a connection.

Export
Select file - export from the menu to display the export dialog shown to the right.

There are 3 options for saving the app.sab file, kits.scode file or both.

Import
Select file - import from the menu and then select the name of the file you wish to import.
Security

Groups
Components are assigned into one or more security groups. There are four security groups indicated by the least significant nibble in the Component.meta property:

- Group 1 0x01
- Group 2 0x02
- Group 3 0x04
- Group 4 0x08

Components can be in multiple groups. For example: to assign a component to groups 2 and 3, set the bottom nibble of Component.meta to 0x06. If a component is not included in any groups, then that component is not network accessible.

To change the component group level, navigate to any component and select the button next to the meta property to bring up the following dialog where the groups can be changed.

Slots
Every slot in a component is defined as operator level or admin level for security purposes. By default a slot is admin level.

Permissions
There are seven security permissions defined as a bitmask:

- User.or operator read 0x01
- User.ow operator write 0x02
- User.oi operator invoke 0x04
- User.ar admin read 0x08
- User.aw admin write 0x10
- User.ai admin invoke 0x20
- User.ua user admin 0x40
To add a new user navigate to Sedona-app-service-users and select the Add User button.

To edit an existing user account navigate to Sedona-app-service-users and then select one from the list. From this screen the password and access levels can be changed.
Firmware Upgrade

Introduction
The firmware on the AAC-1000 can be upgraded to a more recent version using the Engineering Tool. During the upgrade process the existing application will be upgraded and new files will be written to the device and a reboot will take place once all of the files have been transferred to the AAC-1000.

It is advisable to export the existing app.sab and kits.scode files before performing the firmware upgrade.

Tutorial

Step 1.
Open up the Engineering Tool and select file - open to initiate a connection. Once connected navigate to Services - Plat as shown by the image on the right.
Step 2.
Click on the Upgrade SVM at the bottom of the screen and select the firmware version that you wish to use. Press the next button to start the upgrade, you will be asked to confirm your decision after pressing the next button. The SVM should only be upgraded if there is a more recent version available.

Step 3.
The progress bar will display the status of the upload and the device will automatically reboot after the upgrade has completed. You should not interrupt the process while the upgrade is running.
Creating new kits and components

Prerequisites

Install the JAVA SE Development Kit from oracle. We have installed version 8.0.250.18 onto a Windows 8.1 X86 PC and if it is installed correctly you will see it listed in programs and features as shown by the image on the right.

Check to make sure that the JAVA Home directory has been set correctly. On Windows 8.1 the environment variable can be set by selecting System - Advanced system settings which will open up the System Properties dialog.

Select Environment variables and then check to make sure that there is a JAVA_HOME variable listed in the User variables. The path must correspond to the location where the JAVA SE Development Kit was installed.

Download the CBMS custom kits files from www.cbmsstudio.com and unzip the contents onto your PC into a new directory, eg d:\projects\CBMS

This will be the location where you will write the code to create the custom components. Open up a command prompt and change to this directory. There are 3 batch files in this directory which will help to write the code.

- build.bat will build the kit file called cbmsProj and then compile it into a new kits.scode files for each of the CBMS products which can then be downloaded onto the device.
- test.bat will build and run the unit tests which are used during development.
- run.bat will run a windows version of Sedona Virtual machine the development PC which can be used for testing.

During the build cycle, the new kit files will be located in the folder d:\projects\cbms\home in files called aac-xxx-kits.scode. In order to use the newly created kit file on any of the CBMS devices, the kits.scode file corresponding to the device needs to be imported using the Engineering Tool. The import utility will read the app.sab file from the device, update it with the new kit file and then download both the app.sab and kits.scode file back onto the device.

In order to locate the new manifest files, the Engineering Tool Sedona Directory needs to be changed. Selecting Tools - Options from the menu and select d:\projects\cbms as the new directory.

To import the new kits.scode file, select File - Import and then navigate to the project home directory d:/projects/cbms/home and select the aac-xxx.kits.scode file that matches your CBMS hardware.

After the kit file has been imported a new kit, in this case cbmsProj, will appear in the pull down list and the new component called FaultChangeover will appear. The new component can be added to the wiresheet and used like any other component as shown by the diagram below.
OBIX Web Service

OBIX (Open Building Information Xchange) is a Rest Web Service used in Building Automation by many companies around the world to exchange real time data with TCP, HTTP.

Using the OBIX Rest Web Service you can communicate to the CBMS products with an OBIX client or a web browser.

Setting up the OBIX service requires very little work. The driver is installed by default with a component called OBIX which must be a child to the Web Service component as shown in the diagram below.

The OBIX driver will be operational if the Status property is Alive which indicates that both the XML Rest Web Service, and the JSON Rest Web Service are running.

- The url for OBIX XML is a subdirectory called obix, eg http://localhost/obix
- The url for OBIX JSON is a subdirectory called obix, eg http://localhost/json
The CBMS Web server supports basic authentication and compression. When authentication is set to true, the OBIX web service will require a username/password to be entered. When compression is set to true, then the results returned by the web server will be compressed if the client supports compression.

To change these settings, navigate to app/service/web using the Engineering Tool.

To verify that the web service is operational open a web browser and enter the name ip address of the CBMS device followed by either obix or json, eg http://localhost/obix.

If authentication is set to true then you will receive a login dialog. Enter your username and password to display the OBIX rest Web Service. The output for both the XML and JSON Obix Rest Web Service are shown on the right.
The "lobby" is the root object in OBIX and has folders for alarms, histories, watchService and batch. The OBIX protocol specification has a complete description of these folders and how they operate. The Sedona application appears in a folder called app and it is built dynamically from the application using the same tree structure. When you open the app folder, all of the children and slots (properties and actions) will be displayed for the app component.