BACNET/IP INTEGRATION KIT FOR PAC CONTROL GUIDE

Form 2188-170627—June 2017

OPTO 22

Automation made simple.

43044 Business Park Drive • Temecula • CA 92590-3614
Phone: 800-321-OPTO (6786) or 951-695-3000
Fax: 800-832-OPTO (6786) or 951-695-2712
www.opto22.com

Product Support Services
800-TEK-OPTO (835-6786) or 951-695-3080
Fax: 951-695-3017
Email: support@opto22.com
Web: support.opto22.com

BACnet/IP Integration Kit for PAC Control
Form 2188-170627—June 2017

The information in this manual has been checked carefully and is believed to be accurate; however, Opto 22 assumes no responsibility for possible inaccuracies or omissions. Specifications are subject to change without notice.

Opto 22 warrants all of its products to be free from defects in material or workmanship for 30 months from the manufacturing date code. This warranty is limited to the original cost of the unit only and does not cover installation, labor, or any other contingent costs. Opto 22 I/O modules and solid-state relays with date codes of 1/96 or newer are guaranteed for life. This lifetime warranty excludes reed relay, SNAP serial communication modules, SNAP PID modules, and modules that contain mechanical contacts or switches. Opto 22 does not warrant any product, components, or parts not manufactured by Opto 22; for these items, the warranty from the original manufacturer applies. Refer to Opto 22 form 1042 for complete warranty information.

Opto 22
Automation Made Simple.

Table of Contents

1: Using the Integration Kit

IMPORTANT: See page 19 for this integration kit's BACnet PIC statement.

Introduction

BACnet/IP Integration Kit for PAC Control™ (Part # PAC-INT-BAC-IP) enables Opto 22 PAC systems equipped with a PAC controller running a PAC Control™ strategy to communicate with devices on a BACnet/IP network. The integration kit also works with strategies running on a SoftPAC™ software-based controller.

BACnet™ is a communications protocol for building automation and control networks. BACnet/IP is a Master-Slave specification of BACnet that allows the BACnet protocol to use Ethernet networks. With the BACnet/IP Integration Kit, your PAC systems can access BACnet networks and their devices.

The integration kit is a full BACnet/IP master with slave functions. The kit contains the following example PAC Control flowcharts:

- BACnet_IP_Protocol—Broadcasts a WHO-Is at startup, and acts as a slave to other masters. It also opens the communication handle that all charts use.
- BACnet_Master—Includes the master subroutines used to communicate with BACnet devices.
- VAV_Poll_Example—Uses the master subroutines to poll MS/TP VAV boxes using a BACnet/IP-to-BACnet MS/TP router. This chart will help you understand how the subroutines work.

Once configured, you'll be able to use the BACnet/IP protocol in your own PAC Control strategies.

The BACnet/IP Integration Kit is based on BACnet Protocol Standard 135-2008, version 1, revision 9.

For the serial communication version, see the BACnet MS/TP Protocol Integration Kit for SNAP-PAC-S™ (Part # PAC-INT-BAC) in the Downloads section of the Opto 22 website.

What You'll Need

This guide assumes you know how to use PAC Control and the BACnet/IP protocol, and how to configure and use a PAC controller.

In addition to the integration kit, you'll need:

- A PC with PAC Control R9.4a or higher (Basic or Pro)
- An Ethernet-connected PAC controller with PAC firmware R9.4a or higher, or an Ethernet-connected PC running SoftPAC R9.4a or higher

NOTE: To automatically recover communications with I/O units, Opto 22 recommends including IO Enabler logic in your PAC Control strategies. For details, see I/O Enabler for Ethernet and SNAP PAC Brains in the Downloads section of the Opto 22 website.

BACnet_IP_Protocol Chart

At startup, the BACnet_IP_Protocol chart opens a communication handle that all charts use, and then broadcasts a Who-Is request to the network. Masters will respond with an I-Am.

The chart then fills its binding tables with data from the I-Am responses:

Chart Table	Data from Responding Masters
ntBNMasterBindingSRC	IP address
ntBNMasterBindingSADR	Source address
ntBNMasterBindingSNET	Source network number
ntBNMasterBindingInstance	Instance number
ntBNMasterBindingAPDULength	APDU lengths the device can accept
ntBNMasterBindingSegmentation	Whether device supports segmentation
ntBNMasterBindingVendor	Vendor ID

If a master broadcasts a Who-Is, the example BACnet_IP_Protocol flowchart responds to the network with an I-Am.

If a master sends a time sync request, read property, read property multiple, write property, write property multiple, or reinitialize request, the example BACnet_IP_Protocol flowchart responds as a slave.

Exporting the Charts, Adding the Subroutines, and Importing the Charts

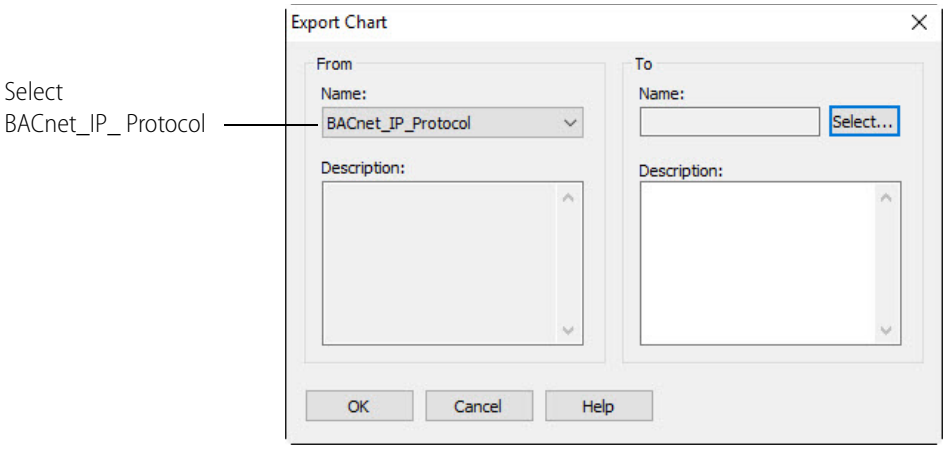
To use the example PAC Control flowcharts (BACnet_IP_Protocol, BACnet_Master, and VAV_Poll_Example), you must first export them from the example strategy (included in the integration kit), add the subroutines to your own strategy, and then import the BACnet charts.

Exporting the Charts

1. Download the integration kit zip file from the Opto 22 website, and then extract the contents to a directory on your hard drive.
2. Start PAC Control, and then open PAC-INT-BACNET-IP.idb (the strategy file you just extracted to your hard drive).

   NOTE: If a "The subroutine files were not found in their original location" warning message is displayed, click Yes to continue.

3. In PAC Control's menu bar, click Chart > Export to open the Export Chart dialog box.
4. In the From area's drop-down list, select the chart to export (BACnet_IP_Protocol).



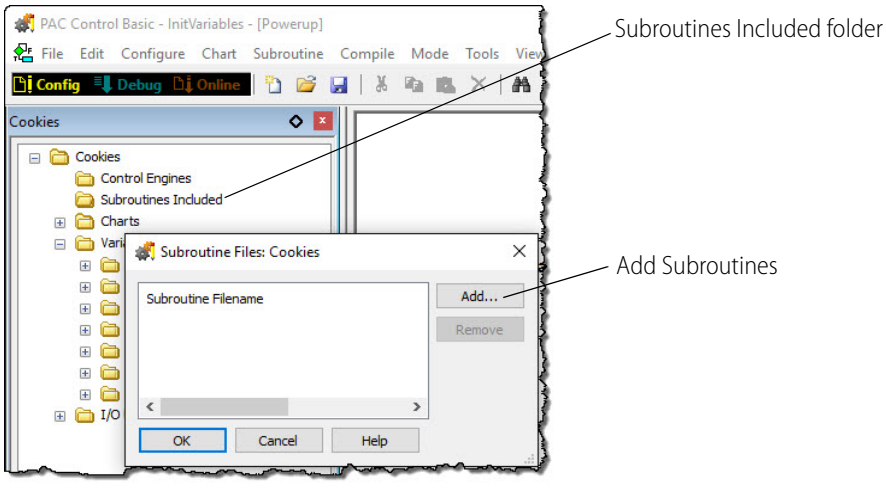
5. In the To area, click Select to open the Select Destination dialog box, and then browse to an appropriate directory (for example, your strategy's directory).
6. In the File name field, type a name for the exported chart (for example, the chart's name followed by `_exported`), and then click Save to close the Select Destination dialog box.
7. Click OK to export the file and close the Export Chart dialog box.
8. Repeat steps 3 through 7 to export the BACnet_Master chart.

Adding the Subroutines

To add the integration kit's subroutines to your own strategy:

1. Open the strategy you want to use with the BACnet protocol.
2. In the strategy navigation tree, double-click the Subroutines Included folder.

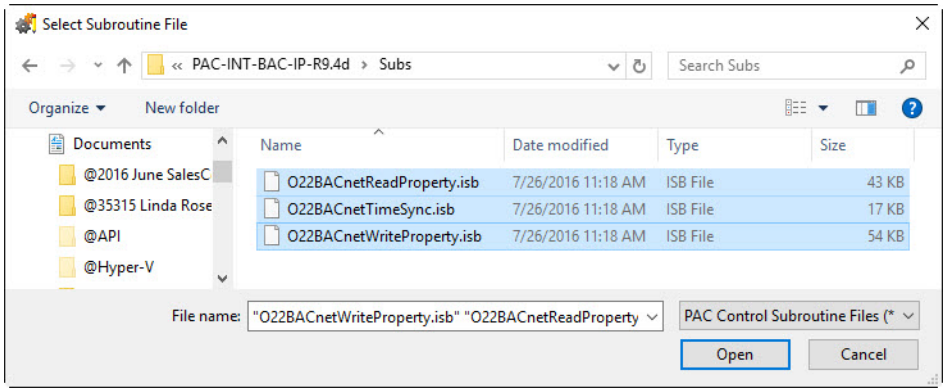
   The Subroutine Files dialog box opens.



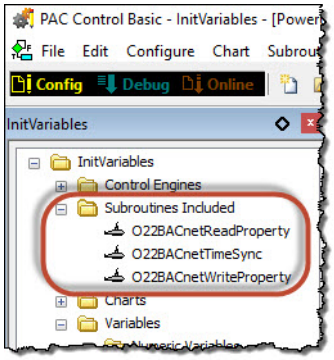
3. Click Add to open the Subroutine Files dialog box.

4. In the Subroutine Files dialog box, browse to the folder where you extracted the BACnet/IP Integration Kit. Select the three subroutines.

   TIP: To select all subroutines at one time, press and hold the Ctrl key while clicking each subroutine.



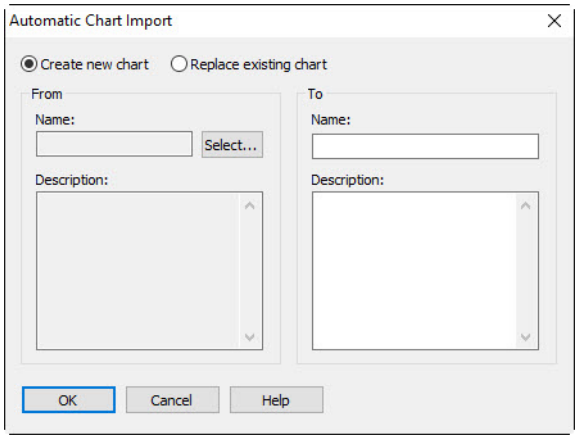
5. In the Select Subroutine File dialog box, click Open, and then in the Subroutine Files dialog box, click OK to add the subroutines to your strategy.



Importing the Charts

1. Make sure your strategy is open in PAC Control, and in the menu bar, click Chart > Import.

   The Automatic Chart Import dialog box opens.



2. With "Create new chart" selected, click Select to open the Select File to Import dialog box.
3. Browse to the directory that contains the export files.
4. Select the chart to import (BACnet_export.cxf), and then click Open.

   The Select File to Import dialog box closes, and the Automatic Chart Import dialog box reappears.
5. In the To Name field, type `BACnet_IP_Protocol` and then click OK.
6. Repeat steps 1 through 5 to import the Master_export.cxf file. In step 5, name the chart `BACnet_Master`.

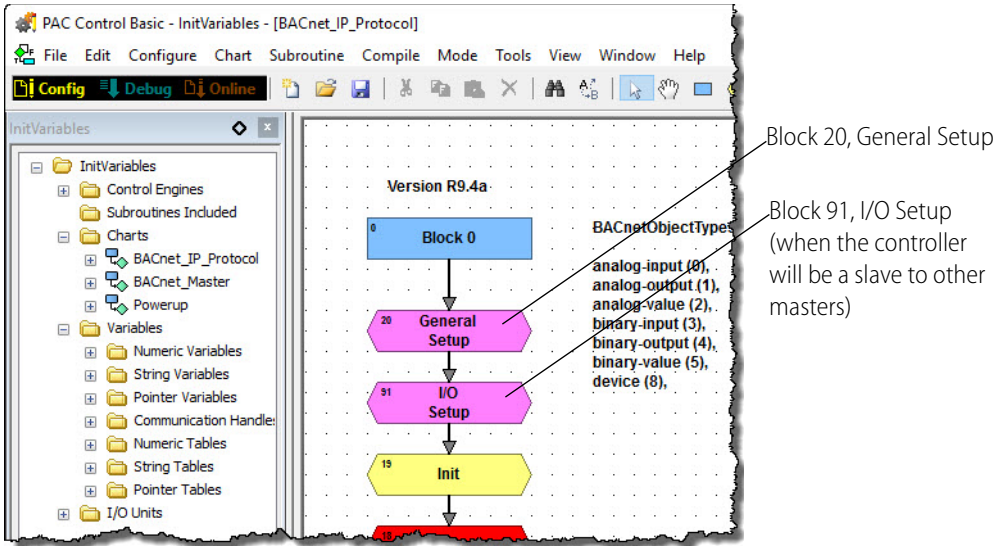
Configuring General Setup

You enter user setup parameters in Block 20 of the BACnet_IP_Protocol chart.

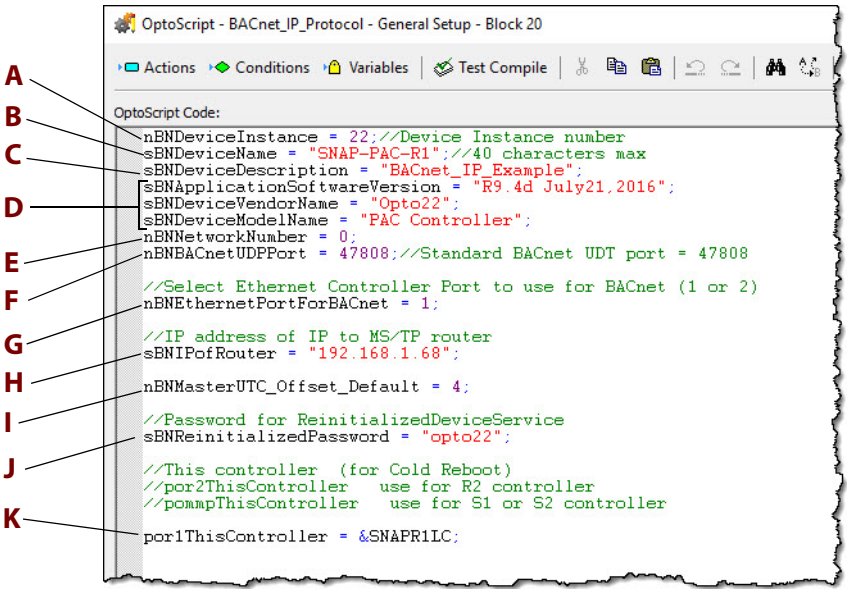
NOTE: I/O setup parameters are entered in Block 91. I/O setup is used only if the controller will be a slave to other masters. If this is not the case, you can delete Block 91. For details, see page 6.

To configure user setup parameters:

1. With your strategy open in PAC Control, open the BACnet_IP_Protocol chart.



2. Double-click Block 20, General Setup, to open it, and then enter the following parameters, according to your device and environment.



A—Device instance number for the PAC controller.

B—User-friendly name for the PAC controller (maximum length: 40 characters).

C—(Optional) Descriptive name for the controller

D—Do not change. The integration kit uses this value.

E—BACnet/IP network.

F—Network port number you plan to use for BACnet communication. (The standard BACnet UDP port is 47808.)

G—Number of the controller's Ethernet port (1 or 2) that you will use for communication.

H—IP address of the BACnet/IP-to-BACnet MS/TP router.

I—Time offset between your local time and UTC.

J—Password for the integration kit to pass in if the controller is reinitialized.

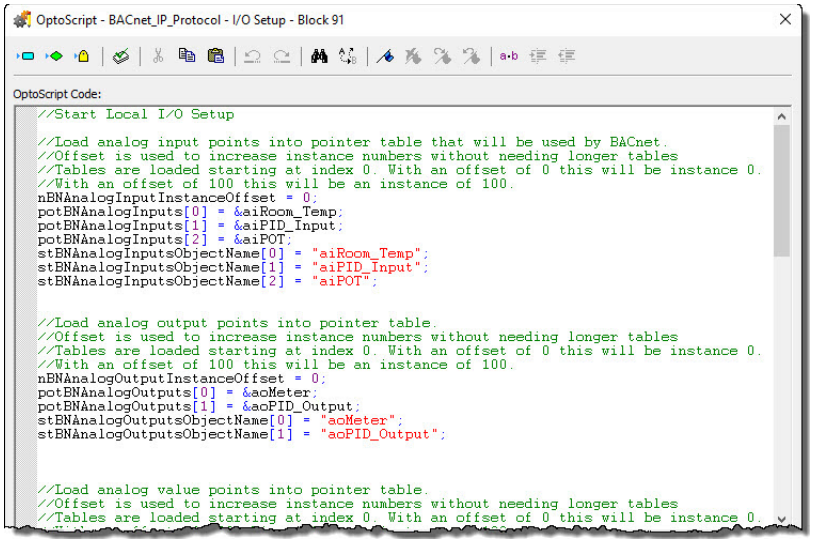
K—Used for Cold Reboot. Move your controller into one of the three pointer variables:

- For an R1-series controller, use: por1ThisController
- For an R2-series controller, use: por2ThisController
- For an S1 or S2-series controller, use: pommpThisController

Configuring I/O Setup

If the controller will be a slave to other masters, you must also configure I/O setup parameters in Block 91, I/O Setup, in the BACnet_IP_Protocol chart. If the controller is not a slave, you can delete Block 91.

For each of the following parameter groups, enter the information for your device:



Opto 22 Table Index-to-Object Identifier Offset Defaults

Offsets are used to increase instance numbers without the need for longer tables.

Tables are loaded starting at index 0. (The default offset = 0.)

- With an offset of 0, index 0 will be an instance of 0.
- With an offset of 100, index 0 will be an instance of 100.

Analog Input

```
//Load analog input points into pointer table that will be used by BACnet.
//Offset is used to increase instance numbers without needing longer tables
//Tables are loaded starting at index 0. With an offset of 0 this will be instance 0.
//With an offset of 100 this will be an instance of 100.
A nBNAnalogInputInstanceOffset = 0;
B potBNAnalogInputs[0] = &aiRoom_Temp;
  potBNAnalogInputs[1] = &aiPID_Input;
  potBNAnalogInputs[2] = &aiPOT;
C stBNAnalogInputsObjectName[0] = "aiRoom_Temp";
  stBNAnalogInputsObjectName[1] = "aiPID_Input";
  stBNAnalogInputsObjectName[2] = "aiPOT";
```

Enter the following information:

A—Index-to-object identifier offset = 0. (For details about default offsets, see page 7.)

B—Enter the variable associated with each analog input point.

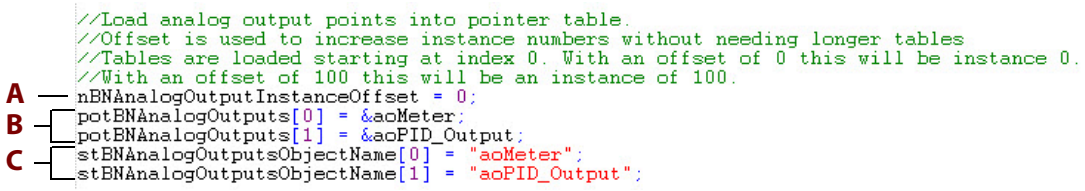
C—Enter a string name for each analog input point.

Analog Input Supported Properties

BACnetObjectType = 0

Properties Supported	Property Data Type	Identifier	Read/Write
Object_Name	CharacterString	77	R
Present_Value	REAL	85	R/W

Analog Output



Enter the following information:

A—Index-to-object identifier offset = 0 (For details about default offsets, see page 7.)

B—Enter the variable associated with each analog output point.

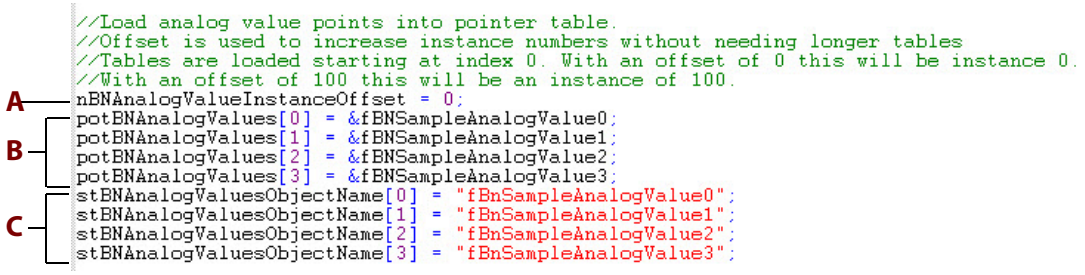
C—Enter a string name for each analog output point.

Analog Output Supported Properties

BACnetObjectType = 1

Properties Supported	Property Data Type	Identifier	Read/Write
Object_Name	CharacterString	77	R
Present_Value	REAL	85	R/W

Analog Values



Enter the following information:

A—Index-to-object identifier offset = 0 (For details about default offsets, see page 7.)

B—Enter the variable associated with each analog value point.

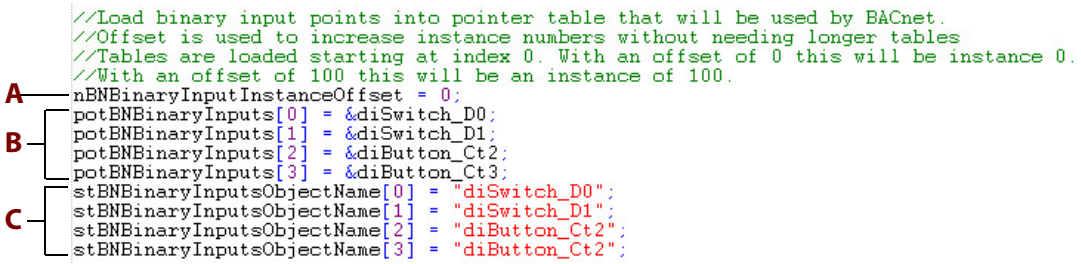
C—Enter a string name for each analog value point.

Analog Value Supported Properties

BACnetObjectType = 2

Properties Supported	Property Data Type	Identifier	Read/Write
Object_Name	CharacterString	77	R
Present_Value	REAL	85	R/W

Binary Input



Enter the following information:

A—Index-to-object identifier offset = 0 (For details about default offsets, see page 7.)

B—Enter the digital point associated with each binary input point.

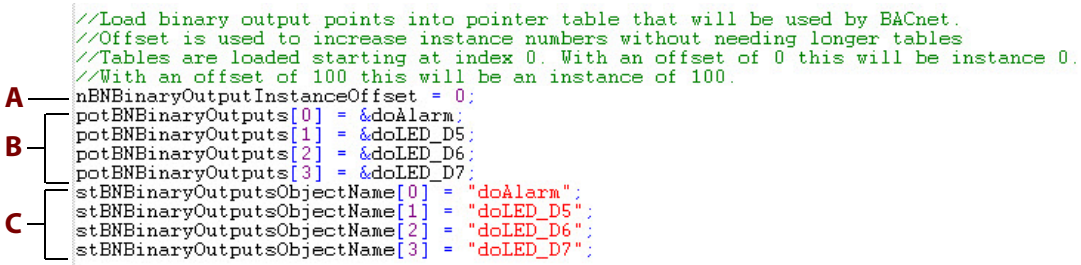
C—Enter a string name for each binary input point.

Binary Input Supported Properties

BACnetObjectType = 3

Properties Supported	Property Data Type	Identifier	Read/Write
Object_Identifier	BACnetObjectIdentifier	75	R
Present_Value	BACnetBinaryPV	85	R/W

Binary Output



Enter the following information:

A—Index-to-object identifier offset = 0 (For details about default offsets, see page 7.)

B—Enter the variable associated with each binary output point.

C—Enter a string name for each binary output point.

Binary Output Supported Properties

BACnetObjectType = 4

Properties Supported	Property Data Type	Identifier	Read/Write
Object_Identifier	BACnetObjectIdentifier	75	R
Present_Value	BACnetBinaryPV	85	R/W

Binary Values

```
//Load binary value points into pointer table.
//Offset is used to increase instance numbers without needing longer tables
//Tables are loaded starting at index 0. With an offset of 0 this will be instance 0.
//With an offset of 100 this will be an instance of 100.
A  nBNBinaryValueInstanceOffset = 0;
   potBNBinaryValues[0] = &nBNSampleBinaryValue0;
B  potBNBinaryValues[1] = &nBNSampleBinaryValue1;
   potBNBinaryValues[2] = &nBNSampleBinaryValue2;
   potBNBinaryValues[3] = &nBNSampleBinaryValue3;
   stBNBinaryValuesObjectName[0] = "nBnSampleBinaryValue0";
C  stBNBinaryValuesObjectName[1] = "nBnSampleBinaryValue1";
   stBNBinaryValuesObjectName[2] = "nBnSampleBinaryValue2";
   stBNBinaryValuesObjectName[3] = "nBnSampleBinaryValue3";
```

Enter the following information:

A—Index-to-object identifier offset = 0 (For details about default offsets, see page 7.)

B—Enter the variable associated with each binary value point.

C—Enter a string name for each binary value point.

Binary Value Supported Properties

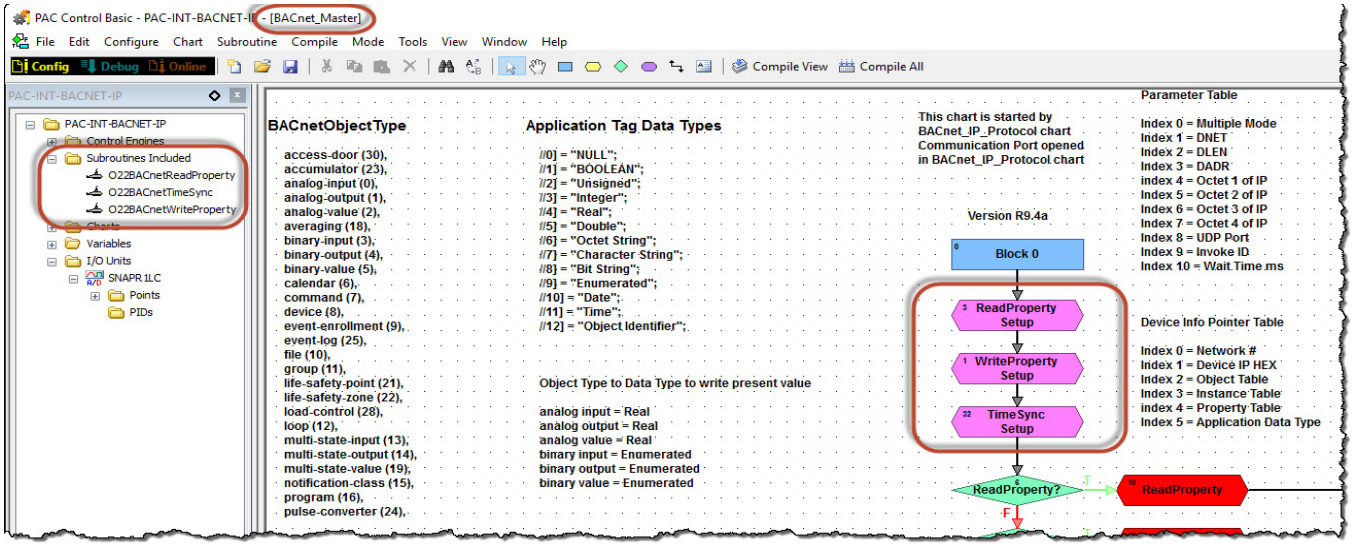
BACnetObjectType = 5

Properties Supported	Property Data Type	Identifier	Read/Write
Object_Identifier	BACnetObjectIdentifier	75	R
Present_Value	BACnetBinaryPV	85	R/W

BACnet_Master Chart Subroutines

This section describes the setup blocks for the subroutines used in the BACnet_Master chart:

Each subroutine has a setup block, which describes the parameters used by the subroutine and provides an example to help you set up the parameters.



O22BACnetReadProperty

Block 3: ReadProperty Setup

Supports BACnet Interoperability Building Blocks (BIBBs) Read Property and Read Property Multiple

In this subroutine, received integer and float data is stored in controller table ftBNRPReturnValues. Received string data is stored in controller table stBNRPReturnValues.

NOTE: You will need to add command instructions to move data from ftBNRPReturnValues and stBNRPReturnValues to the variables configured to store the data for that particular device.

The first value is stored at index 0, and subsequent values (up to nBNNumReturnValues) are stored in consecutive table elements. Read Property uses Index 0.

Example (Read Property Multiple)

Example Read	Stored in Table[Index]	Example Response
Analog Input,0,Present Value	ftBNRPReturnValues[0]	75.2
Analog Value,4,Present Value	ftBNRPReturnValues[1]	256
Binary Value,70,Present Value	ftBNRPReturnValues[2]	1
Device,20,Object Name	stBNRPReturnValues[3]	Pump Room 1

Code sample: Move data from controller tables to device variables

```
ftVAV34[0] = ftBNRPReturnValues[0];
ftVAV34[4] = ftBNRPReturnValues[1];
ftVAV34[10] = ftBNRPReturnValues[2];
sVAV34Name = stBNRPReturnValues [3];
```

O22BACnetWriteProperty

Block 1: WriteProperty Setup

Supports BIBBs Annex K Write Property and Write Property Multiple

In this subroutine, integer and float data to be written is stored in controller table ftBNWriteValues. String data is stored in controller table stBNNumWriteValues. Write Property uses index 0.

NOTE: You will need to add command instructions to move data from the variables configured for that device to the controller's ftBNWriteValues and stBNWriteValues tables.

The first value is stored at index 0, and subsequent values (up to nBNWriteValues) are stored in consecutive table elements. Read Property uses Index 0.

Example (Write Property Multiple)

Object to Write	Subroutine Table	Example Value to Write
Analog Value,4,Present Value	ftBNWriteValues[0]	45.2
Binary Input,3,Present Value	ftBNWriteValues[1]	1
Device,20,Object Name	ftBNWriteValues[2]	25
Analog Value,5,Present Value	stBNWriteValues[3]	1149.8

Code sample: Write data from device variables to controller tables

```
ftBNWriteValues[0] = 45.2;
ftBNWriteValues[1] = 1;
stBNWriteValues[2] = 25;
ftBNWriteValues[3] = 1149.8;
```

O22BACnetTimeSync

Block 32: TimeSync Setup

Supports BIBBS DM-TS-B (Time Synchronization) and DM-UTC-B (UTC Time Synchronization).

This subroutine uses the controller time and date to sync the time and date in the BACnet devices. It can send to one device or all devices in a broadcast.

2: VAV Polling Example Chart

Introduction

Included in the BACnet/IP Integration Kit is the VAV Polling chart (VAV_Polling_Example)—a sample chart to help you learn how to use the integration kit's subroutines.

The chart polls BACnet MS/TP VAV (Variable Air Volume) controllers by using a BACnet/IP-to-BACnet MS/TP router. You can also modify the chart to poll any other BACnet device.

To use the VAV_Polling_Example chart, you must first export it from the BACnet/IP Integration Kit, and then import it into your own strategy. Your strategy must also include the BACnet_IP_Protocol chart and the integration kit's subroutines. For instructions on exporting and importing charts and adding subroutines, see page 2.

At startup, the BACnet_IP_Protocol chart:

- Opens a communications session.
- Starts the VAV_Polling_Example chart.
- Executes a Who-Is broadcast.
- Waits 10-seconds for devices to respond, and then executes a Read Binding Name for each device that responded to the Who_Is broadcast.

The VAV_Polling_Example chart can write to each VAV controller configured in Block 64. The values to write are stored in a float table for each address. If the value to write is not a float, the strategy corrects the data type before sending it to the VAV controller. The example includes a write table for VAV at address 13. You will need to configure a table for each address in your system.

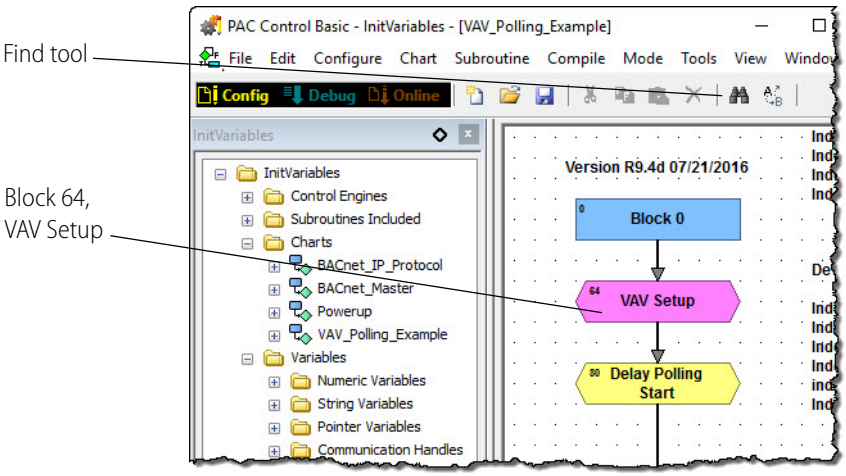
To use the VAV_Polling_Example chart, you must first export it from the BACnet/IP Integration Kit, and then import it into your own strategy. Your strategy must also include the BACnet_IP_Protocol chart and the integration kit's subroutines. For instructions on exporting and importing charts and adding subroutines, see page 2.

In This Chapter:

Enter VAV Setup Parameters

You enter VAV setup parameters in Block 64 of the VAV_Polling_Example chart. After importing the VAV_Polling_Example chart into your strategy:

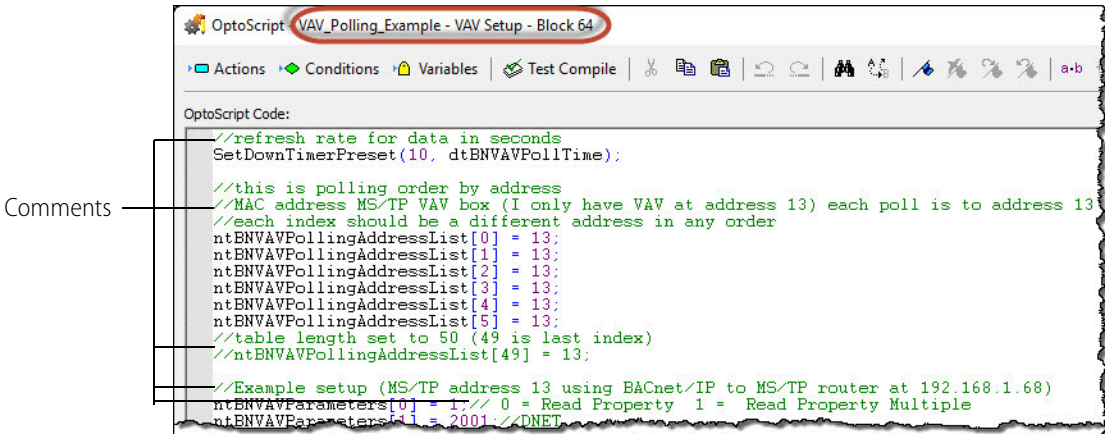
1. Open the VAV_Polling_Example chart.



2. Double-click Block 64, VAV Setup, to open it. Each section in the setup has comments to explain what the parameters do, and what you need to configure them.

   TIP: To quickly find text in an OptoScript window, either use the Find tool, or put your cursor in the OptoScript window, and then press Ctrl +F to open the Find dialog box.

   Note that the setup table length is set to 50. This makes the last index 49. To poll more than 50 devices, you'll need to change the configured table length by using the Strategy Tree.



Complete Poll and Refresh Poll

The VAV_Polling_Example chart has two poll types, each with its own setup parameters in Block 64: Complete Poll and Refresh Poll.

Complete Poll

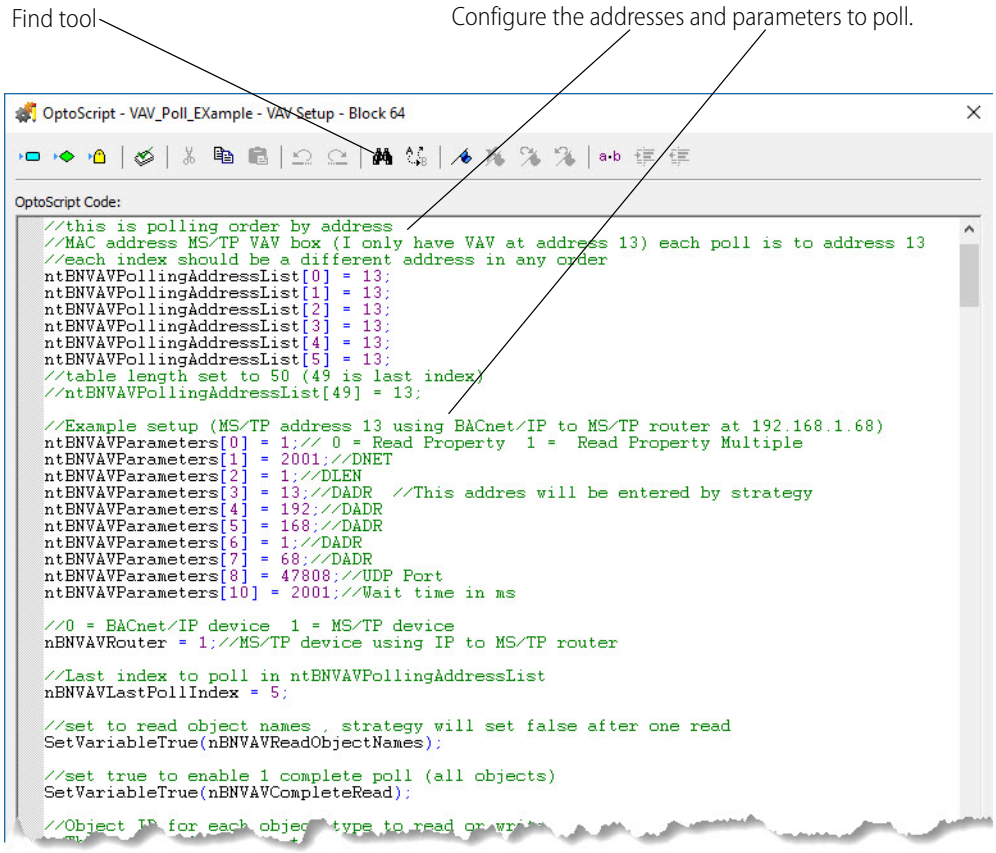
Complete Poll reads the VAV controller polling addresses and parameters you've configured in Block 64, VAV Setup, and then polls each object once.

- The variables for polling addresses are named ntBNVAVPollingAddressList[]
- The variables for parameters are named ntBNVAVParameters[]

The Complete Poll stores the value data in a float table, and the object name in a string table. After the poll, the chart sets the nBNVAVCompleteRead value to 0 (false). This ensures that the Complete Poll happens only once.

The example chart includes a value data table and an object name table for the VAV controller at address 13. You will need to configure tables for each address in your system.

To locate the Complete Poll setup area in Block 64, use the Find tool to search for "polling order by address."

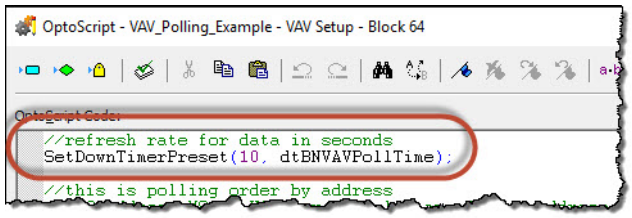


Refresh Poll

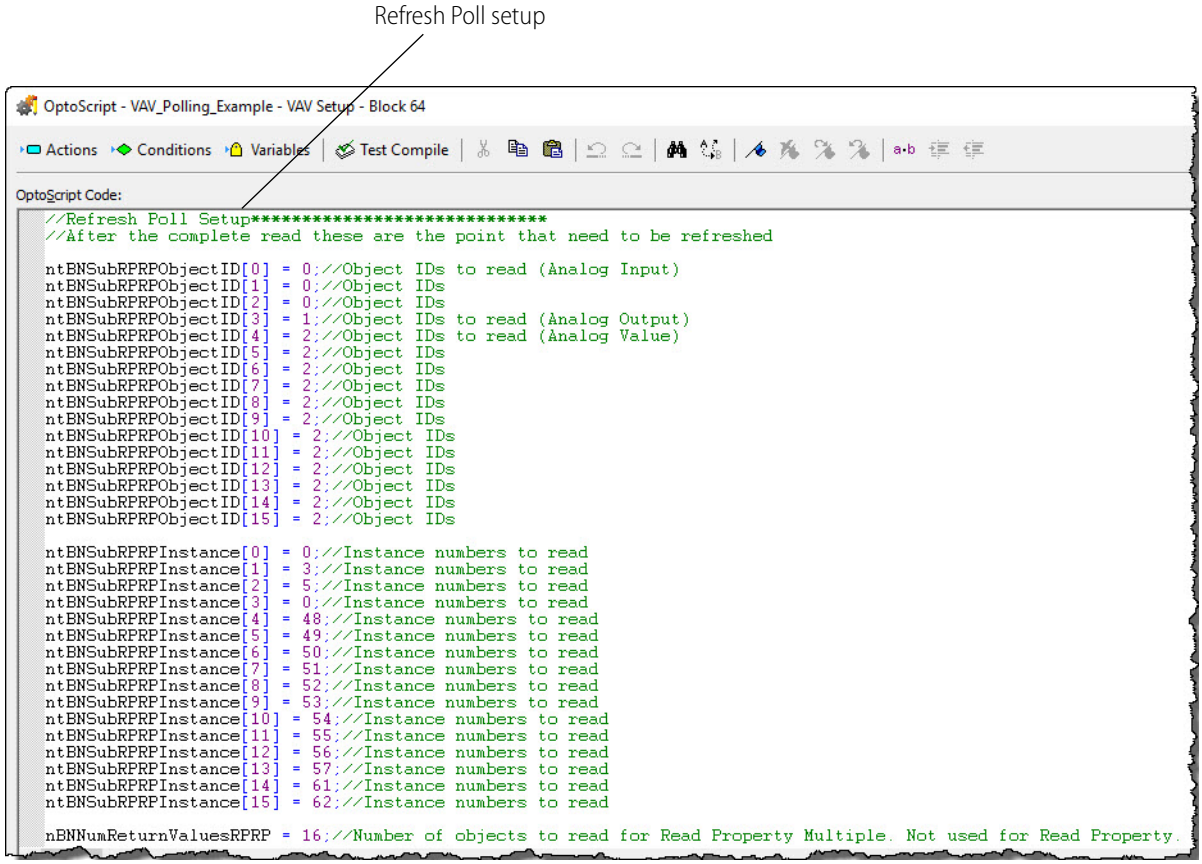
The Refresh Poll section of Block 64, VAV Setup, refreshes and reads the values for a configured list of objects, and then stores the data in a float table.

The refresh interval is configurable—the default value is 10 seconds.

To locate the refresh rate set up, search for "refresh rate".



To locate the Refresh Poll setup area in Block 64, use the Find tool to search for "Refresh Poll Setup."



Write to VAV

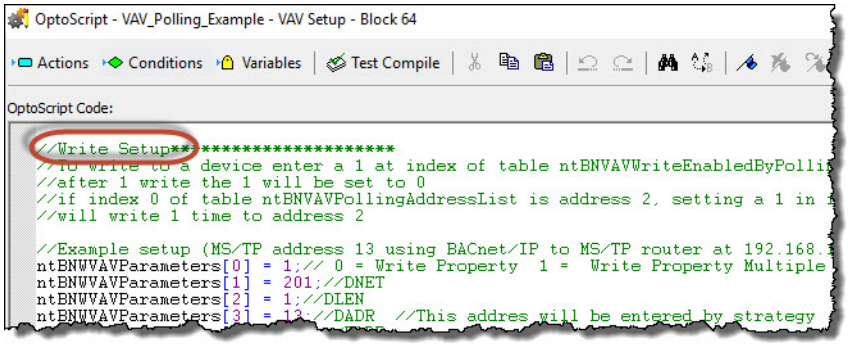
Write is a three-step process. You configure steps 1 and 2 in VAV Setup, Block 64. Each section in the block contains comments and instructions for configuring values.

Step 1: Write Setup

In the Write Setup section of VAV Setup:

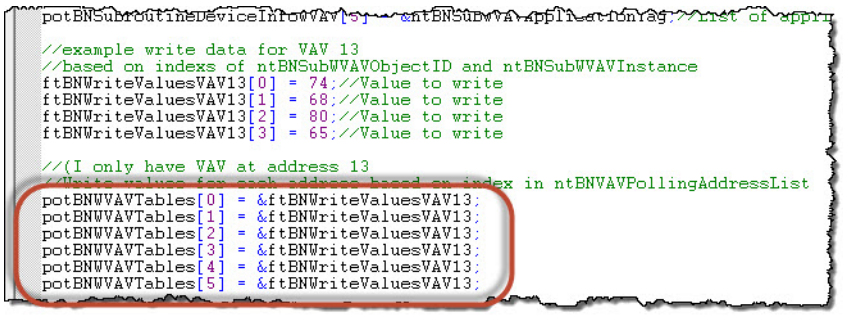
1. Enter network parameters.
2. Enter the Object ID for each object to write.

3. Enter the Object Instance number for each object to write.
4. Enter the property ID for each object to write. (85 = Present Value)
5. Enter the Application ID (Data Type) for each object to write.
6. Set variable nBNNumWVAVWriteValues to the number of objects to write.
7. Load the pointer table with the write data table for each address.



Step 2: Move Write Data

For each VAV address to write, move the data to be written to the write data table. In the example, only the VAV at address 13 is configured.



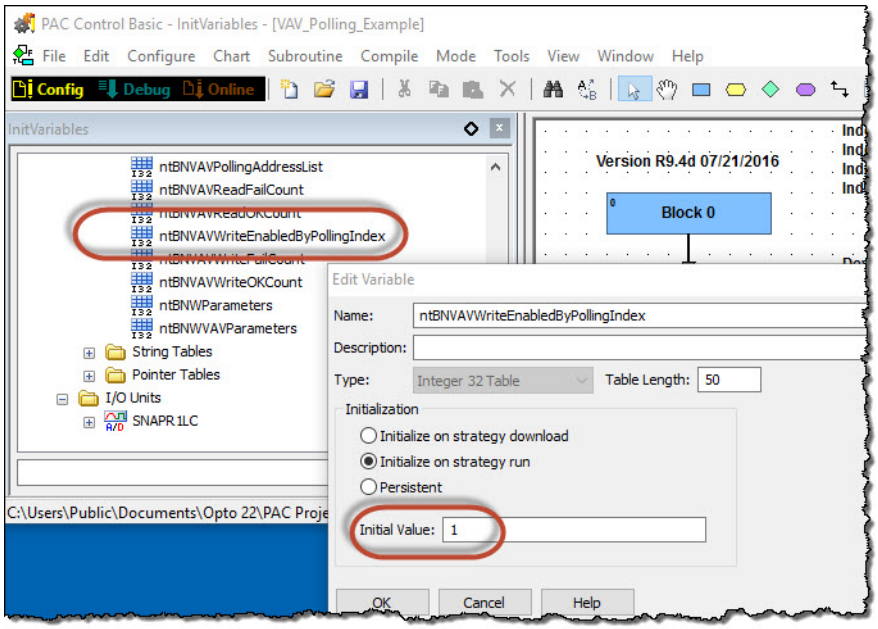
Step 3: Enable Writing

To write to a device, in the ntBNVAVWriteEnabledByPollingIndex table, type a 1 at the index for the address to write. After one write, the chart sets the value to 0 (false).

For example, if index 0 of table ntBNVAVPollingAddressList is address 13, setting a 1 in index 0 of the table ntBNVAVWriteEnabledByPollingIndex will write one time to address 13 of all objects configured in the Write section of Block 64, VAV Setup.

If more than one address is enabled, the strategy writes all configured objects to each address one time.

In your strategy you will need to enable writes to addresses as needed. You can also write your own code to check if the setpoints have changed, and write again as needed.



1. Click OK to save the setting and close the dialog box.

NOTE: After writing to the device, the strategy sets the value to 0.

For example, if index 0 of table ntBNVAVPollingAddressList is address 13, set the value for table ntBNVAVWriteEnabledByPollingIndex to **1** to write once to address 13.

A: BACnet PIC Statement

BACnet Protocol Implementation Conformance Statement

Date	11.20.2015
Vendor name	Opto 22
Product name	BACnet/IP Integration Kit for PAC Control™
Product part number	PAC-INT-BAC-IP
Application software version	R9.4a
Firmware revision	N/A
BACnet protocol version	9

Product Description

BACnet/IP Integration Kit for PAC Control is a software utility that enables Opto 22 PAC controllers to act as BACnet client/servers using BACnet/IP.

BACnet Standardized Device Profile (Annex L)

BACnet Application Specific Controller	(B-ASC)

BACnet Interoperability Building Blocks (BIBBs) Supported (Annex K)

DS-RP-A	Data Sharing – Read Property-A
DS-RP-B	Data Sharing – Read Property-B
DS-RPM-A	Data Sharing – Read Property Multiple-A
DS-RPM-B	Data Sharing – Read Property Multiple-B
DS-WP-A	Data Sharing – Write Property-A
DS-WP-B	Data Sharing – Write Property-B

DS-WPM-B	Data Sharing – Write Property Multiple-B
DS-COV-A	Data Sharing – Change of Value -A
DS-COVU-A	Data Sharing – Change of Value-Unsolicited-A
DM-TM-B	Device Management – Text Message-B
DM-TS-B	Device Management – Time Synchronization-B
DM-UTC-B	Device Management – UTC Time Synchronization-B
DM-RD-B	Device Management – Reinitialize Device-B
DM-DDB-A	Device Management – Dynamic Device Binding-A
DM-DDB-B	Device Management – Dynamic Device Binding-B
DM-DOB-B	Device Management – Dynamic Object Binding-B

Segmentation Capability

None

Standard Object Types Supported

Device Object
Analog Input
Analog Output
Analog Value
Binary Input
Binary Output
Binary Value
Multi-state Input
Multi-state Output
Multi-state Value

Data Link Layer Options

MS/TP master (Clause 9), baud rates	9.6K
	19.2K
	38.4K
	76.8K
	115.2K

Device Binding Methods

Send who-Is, receive I-Am	DM-DDB-A
Receive who-Is, send I-Am	DM-DDB-B
Receive who-Has, send I-Have	DM-DOB-B

Networking Options

None

Character Sets Supported

ANSI X3.4